International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-2S10, September 2019

# An Efficient Exchanged Hyper Cube for Parallel and Distributed Network

**P Venkat Reddy, Sudarson Jena, V Kamakshi Prasa**

***Abstract*: Enormously parallel distribution memory designs are accepting and expanding regard to satisfy the expanding need on processing power. Numerous topologies have been projected for interconnecting the processors of distributed computing systems. The hypercube topology has attracted significant consideration because of a significant number of attractive properties. The engaging properties of the hypercube topology, for example, vertex and edge balance, recursive structure, logarithmic diameter, maximally fault-tolerance, simple routing and broadcasting, and the capacity to recreate other interconnection systems with least overhead have made it a brilliant possibility for some parallel processing applications. Numerous varieties of the hypercube topology have been accounted for the literature, mostly to add the computational power of the hypercube. One of the gorgeous versions of the hypercube was introduced for the improvement of the presented Exchanged hypercube. An Exchanged hypercube has the equivalent structural complexities of the hypercube. It protects the gorgeous properties of the hypercube and diameter the communication time by dropping the diameter by a factor of two. This paper presents the fundamental communication and some of the essential operations normally required in parallel computing on the Exchanged hypercube interconnection networks.***

***Keywords: interconnecting network, routing protocol, Hypercube, exchanged hypercube.***

## I. INTRODUCTION

An interconnection network is normally demonstrated by an associated graph in which vertices signify to processors and edges signify to associate between processors. The connectivity is one of the significant parameters to assess the reliability and fault tolerance of a network.

Though that as the conventional connectivity always underestimates the strength of huge networks the improvement of multiprocessor systems, improving the conventional connectivity is necessary. Motivated by the deficiencies of conventional connectivity, a few generalized connectivity and edge connectivity were considered by numerous creators. The component connectivity and edge connectivity were presented in [2] and [7] independently. The component connectivity can more exactly estimate the reliability and fault tolerance for large scale parallel processing systems in that similar manner. In [5], it examines the component connectivity of hyper cubes.

Give G a chance to be a non-complete associated graph. The g-segment cut of G is a lot of vertices whose erasure brings about a graph with at any rate g parts. The g-segment availability cκg (G) of a graph G is the size of the littlest g-part cut of G. By the meaning of cκg (G), it tends to be seen that cκg+1(G) ≥ cκg (G) for each positive whole number g. The n-dimensional hypercube Qn is an undirected diagram Qn = (V, E) with |V | = 2n and |Qn| = n2 n−1 . Every vertex can be spoken to by a n-bit double string, and each bit position is 0 or 1. There is an edge between two vertices at whatever point there paired string portrayal varies in just one bit position.

As one of the significant variations of the hypercube arrange, the n-dimensional collapsed hypercube F Qn, proposed by El-Amawy [4], is acquired from a n-dimensional hypercube Qn by including an edge between any pair of vertices with integral locations. The collapsed hypercube F Qn is better than Qn in certain properties; see [3, 4, 6]. In this way the collapsed hypercube is an upgrade on the hypercube Qn and F Qn is acquired by including an ideal coordinating M the hypercube, where M = {(u, u)|u ∈ V (Qn)} and u speaks to the supplement of the vertex u, that is, all their double strings are supplement and 0 = 1 and 1 = 0. One can seen that E(F Qn) = E(Qn) S M. For accommodation, F Qn can be communicated as D0 ND1, where D0 and D1 are (n − 1)- dimensional sub cubes incited by the vertices with the I-th facilitate 0 and 1 individually

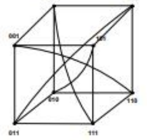


Figure 1. The 3-dimensional Folded hypercube.

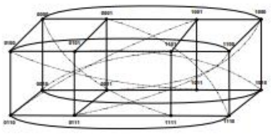


Figure 2. The 4-dimensional Folded hypercube.

**Revised Manuscript Received on September 22, 2019.**

**P Venkat Reddy,** Research Scholar, JNTUH, Hyderabad, India. patlollavenkat83@gmail.com

**Sudarson Jena,** GITAM University, Hyderabad, India.sjena2k5@rediffmail.com

**V Kamakshi Prasad,** JNTUH Hyderabad, India. kamakshiprasad@yahoo.com

*Retrieval Number: B11500982S1019/2019©BEIESP*
*DOI:10.35940/ijrte.B1150.0982S1019*

*Published By:*
*Blue Eyes Intelligence Engineering*
*& Sciences Publication*

let v to be a vertex of a graph G, we use NG(v) to indicate the vertices that are adjoining v. Let u, v ∈ V (G), d(u, v) indicates the separation among u and v. Let A ⊆ V (G), we mean by NG(A) the vertex set S v∈V (A) NG(v) \ V (An) and CG(A) = NG(A) S A. And furthermore, we use θG(g) signifies the base number of vertices that are nearby a vertex set with g vertices in G. For a vertex v ∈ V (G), d(v) indicates the level of the vertex v in G. The private neighbors of a vertex v ∈ V ′ ⊆ V (G), as in [8], meant by P N(v), are those neighbors of v which are not shared by different vertices in V ′ and are not themselves in V ′ , i.e., P N(v) = N(v) − N(V ′ − {v}) − V ′ . We pursue Bondy [1] for phrasings not given here.

The hypercube has been the subject of numerous recent researches [1]-[8]. Different specialists have done broad work in demonstrating the parallel computational intensity of the hypercube structure in numerous ways. One way, numerous specialists have demonstrated the capacity of the hypercube to mimic different systems, for example, rings, trees, matrices and other interconnection systems with least overhead. Toward another path, scientists have demonstrated the intensity of the hypercube in taking care of numerous computational issues in parallel, for example, arranging, consolidating, parallel prefix, and different issues. A third way, scientists have demonstrated the heartiness and adaptation to non-critical failure of the hypercube, concentrating on the hyper cube’s capacity to reproduce, figure, and reconfigure itself within the sight of flaws. Numerous specialists have proposed adjustments on the hypercube structure to improve its computational power. Bhuyan and Agrawal [9] proposed a summed up hypercube structure that is fit to numerous applications. Preparation and Vuillemin [10] presented the cube connected cycles in which the level of the distance across was diminished to a fixed consistent. El-Amaway and Latifi [11] proposed the collapsed hypercube to diminish the width and the traffic clog with little equipment overhead. You and Narahari [12] proposed the banyan hypercube system to diminish the correspondence overhead. Zheng at el. [13] proposed the star-hypercube cross breed interconnection system to consolidate the worthwhile highlights and properties of the two stars and hyper cubes. Traded hyper cubes demonstrated to contain the appealing properties of the hypercube and better correspondence abilities. In parallel structures, the correspondence cost overwhelms the calculation cost. The overall presentation of the parallel machine depends intensely on the fundamental interconnection organize. In an Exchanged hypercube, the diameter of the system is decreased by a factor of two over that of the hypercube. A considerable lot of the hypercube alluring highlights, for example, parcelling, directing, adaptation to internal failure, and inserting are joined into the Exchanged hypercube and new gains are accomplished in width, normal separation, and implanting effectiveness [14]-[20].

## II. BACKGROUND:

### 2.1 Hypercube

A dimensional Hypercube (HCn) network with N processing elements (PEs) is projected by routing function as:

$$C_i(a_{n-1}\cdots a_1a_0)=a_{n-1}\cdots a_{i+1}\bar{a}_{i-1}\cdots a_1a_0\ ,\ \ for\ i=0,1,2,\cdots n-1$$

In the n cube, each processing component situated at a corner is directly associated with n neighbours. The neighbouring processing components vary in exactly one bit position. There are 2n number of processing components and n.2n'1 number of connections in a n-dimensional hypercube (HCn). A 3-dimensional hypercube network topology is introduced in Fig.

### 2.2 Folded Hypercube

As a variation of the hypercube, the n - dimensional collapsed hypercube (FHn) was first proposed by Amaway and Latifi. It is gotten from the hypercube HCn by including 2n-1 edges, called reciprocal edges. Every one of them in the middle of vertices, X= (x1, x2...................xn) and the measurement of FH n is half of the width of HC n. In this way the folded hypercube is prevalent and is an upgraded variant of HC n. There are n+1 inside disjoint ways of length probably (n/2) +1, between any pair of vertices in FH n. The erasure of not exactly [n/2]-2 vertices or edges doesn't builds the distance across of FH n and the cancellation of up to n vertices or edges increment the width by at generally one. The above properties imply that interconnection systems displayed by FH n are extremely robust.

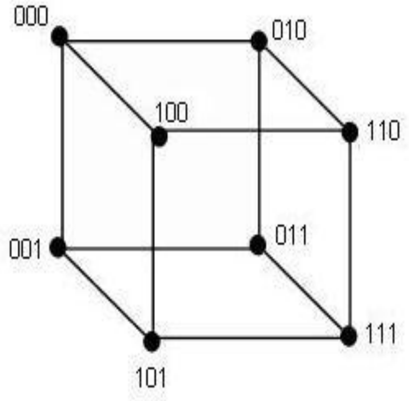


Fig 2.2 end hypercube network for n=3

Lt-tolerant hypercube (FTHn) arrange is built from a standard n-dimensional hypercube by setting up extra interfaces between every hub and its most remote hub. The extra connections going along with one hub and its most distant hub are known as a reciprocal connection or c-interface. The quantity of c-joins is 2n-1 and in this way, the connection repetition is 1/n Nodes in the FTHn are allocated double marks from 0 to 2n-1 with the end goal that names of two neighbouring hubs vary in just one piece. Let < x0, x1..., xn-1> speak to the component of n-3D square. The FTHn can be envisioned as a system with n+1 measurements < x0, x1, xn-1..., xn>, where xn is the measurement comparing to c-links.

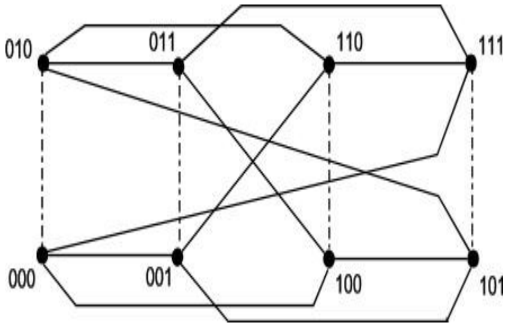


Figure 2.3: Fault-tolerant Hypercube Network *(for n=3)*

## III. MY STUDY

The topological structures in interconnection systems have been a significant subject for research identified with hugely parallel and disseminated PC systems. Unwavering quality and legitimacy examinations give critical criteria during the structure of interconnection systems. A portion of the elements that are pivotal during the transmission of data are accuracy, proficiency, and information volume. These elements can be utilized to break down degrees of execution. In late decades, countless examinations have researched related possibility properties of interconnection systems [5], [6], [7], [11], [23]. An interconnection system can for the most part be spoken to by an undirected diagram G(V,E), where the set V of vertices/hubs speaks to the processors and the set E of edges/joins speaks to the correspondence connects between the processors. All through this paper, the terms systems and graph are compatible.

The n-dimensional hypercube, which is meant by is one of the most main stream interconnection system structures in parallel PC and correspondence frameworks [8], [13], [14], [20], [25]. This sort of structure has 2n vertices and n • 2n−1 edges. The ubiquity of this sort of structure is incompletely due to the numerous alluring and superb properties of the hypercube, for example, normality, recursive structure, vertex and edge balance, little diameter, most extreme availability, and the successful steering and broadcasting. The traded hypercube EH(s,t) is a variation of the n-dimensional hypercube Qn that was first proposed

By Loh et al. [16]. The traded hypercube is characterized by deliberately expelling a few edges from the hypercube Qn where n = s + t + 1. Studies have been led so as to make EH(s,t) valuable in dependable and basic applications and have delivered some critical outcomes. The traded hypercube keeps up a few significant and attractive properties of the hypercube, for example, little diameter [16], biplane cyclist [17], and super network [18] and therefore, the traded hypercube is desirable over the hypercube. For instance, despite the fact that the quantity of edges of a traded hypercube is just somewhat over portion of that of a hypercube, their distance across are comparative. The distance across of a traded hypercube EH(s,t) is s + t + 2 [16]. Furthermore, the wide and deficiency width of a traded hypercube EH(s,t) is s + t + 3 [22]. In this way, traded hypercube have lower connection costs than hypercube. In addition, connect unpredictability can legitimately influence the expenses of equipment and the execution of huge scale joining (VLSI) frameworks [10], [21]. An astounding distributed interconnection system ought to have low degree, little diameter, high adaptation to non-critical failure, and low dormancy [12], [24]. These qualities lead to abnormal amounts of productivity and execution. Traded hypercube can be utilized as sensible topologies in these situations.

The degree of edge clog in interconnection systems is a significant pointer for cost investigations and execution estimations. Since edges with overwhelming clog may prompt correspondence bottleneck and transmission delay, directing algorithms are required to adjust correspondence stacks on them. Propelled by this perception, the accompanying ideas identified with edge blockage are presented [9]. Assume that the likelihood that data will traded between a given pair of vertices is equivalent to it is for some other pair of vertices. In light of this supposition, we consider all sets for directing when ascertaining the degree of edge clog. For a particular steering calculation in a system, for each edge e, the blockage of e is the quantity of sets of vertices that will be directed through e. The edge clog of a system under a particular steering calculation is the most extreme estimation of the blockage everything being equal. At that point, the edge clog of a system is the base an incentive over all directing algorithms. Clearly, a lower level of edge clog is frequently favoured in a system. Likewise, edge blockage can likewise be utilized to decide the lower bound on the zone and the longest wire length required by VLSI arranges designs. As of late, numerous investigations have concentrated on the theme of edge blockage in explicit sorts of interconnection systems, for example, hypercube [9], crossed 3D squares [3], Exchanged solid shapes [15], and different kinds of systems [1], [2], [4], [9], [19].

## IV. PRELIMINARIES

In this section, we present a portion of the fundamental definitions and documentations that are utilized in the remainder of segments. A chart G is a two-tuple (V,E), where V is a nonempty vertex set and E is a subset of {(u,v) | (u,v) is an unordered pair of V }. We use V (G) and E(G) to indicate the vertex set and the edge set of G, separately. Two vertices, u and v, of a diagram G are contiguous if (u,v) ∈E(G). The areas of a vertex v in diagram G, signified by NG(v), are {x | (v,x) ∈E(G)} and the level of a vertex v, meant by dG(v), is |NG(v)|. A way P of length k from vertex u to vertex v in a chart G is a grouping of particular vertices composed as x0 → x1 → x2 → ••• → xk where x0 = u, xk = v, and (xi,xi+1) ∈E(G) for 0 ≤ I ≤ k − 1 if k ≥ 1. At that point, u and v are alluded to as the source vertex and the goal vertex, individually. The separation between two vertices u and v in diagram G, indicated by dG(u,v), is the length of the most limited way among u and v.

A n-dimensional hypercube Qn is a bipartite diagram with 2n vertices and n•2n−1 edges. Every vertex is marked utilizing a n-bit paired string u = un−1un−2 •••u0, where ui ∈ {0,1} for 0 ≤ I ≤ n − 1. Two vertices are adjoining if and just if their strings contrast precisely in one piece position. Unmistakably, Qn is a n-ordinary bipartite diagram. Let u = un−1un−2 •••u0 and v = vn−1vn−2 •••v0 be two nbit paired strings. The Hamming separation between two vertices u and v, meant by H(u,v), is the quantity of bits that are distinctive in the relating strings for both vertices. That is It pursues that H(u,v) = dQn(u,v). Furthermore, Qn has diameter n [25].

A traded hypercube is characterized as an undirected chart EH(s,t) = G(V,E), where s ≥ 1 and t ≥ 1. A traded hypercube is shaped by expelling edges from a n-dimensional hypercube Qn where n = s+t+1. The meaning of traded hypercube EH(s,t) is displayed as pursues

**Definition 1.** The vertex set V of traded hypercube EH(s,t) (s ≥ 1,t ≥ 1) is the set

{us+t •••ut+1ut •••u1u0 | ui ∈ {0,1} for 0 ≤ I ≤ s + t}.

Let u = us+tus+t−1 •••u0 and v = vs+tvs+t−1 •••v0 be two vertices in EH(s,t). There is an edge (u,v) in EH(s,t) if and just if (u,v) is in one of the accompanying sets:

E1 ={(u,v) | u0 6= v0,ui = vi for 1 ≤ I ≤ s + t},

E2 ={(u,v) | u0 = v0 = 0,H(u,v) = 1 with ui 6= vi for some t + 1 ≤ I ≤ s + t}, and

E3 ={(u,v) | u0 = v0 = 1,H(u,v) = 1 with ui 6= vi for approximately 1 ≤ I ≤ t},

where H(u,v) means the Hamming separation between two vertices u and v.

From the meaning of EH(s,t), the quantity of vertices is 2s+t+1 and the quantity of edges is (s + t + 2)2s+t−1 where |E1| = 2s+t, |E2| = s•2s+t−1, and |E3| = t•2s+t−1. For a vertex x with x0 = 0, the vertex degree is s + 1, while the vertex degree with x0 = 1 is t + 1. EH(s,t) is a subgraph of the (s + t + 1)- dimensional hypercube Qs+t+1, and therefore it is additionally a bipartite chart. Fig. 1 shows the traded hypercubes EH(1,1), EH(1,2), EH(2,1) and EH(2,2). Dashed connections relate to the edge set E1, strong connections compare to the edge set E2, and striking connections compare to the edge set E3. Furthermore, the quantity of vertices in EH(s,t) is equivalent to the quantity of vertices in Qs+t+1. As it were, |V (EH(s,t))| = |V (Qs+t+1)| = 2s+t+1 and the quantity of edges in EH(s,t) is just somewhat over portion of the quantity of edges in Qs+t+1. To be progressively exact, |E(EH(s,t))| = . Loh et al. [16] and Ma et al. [17] expressed the accompanying properties.

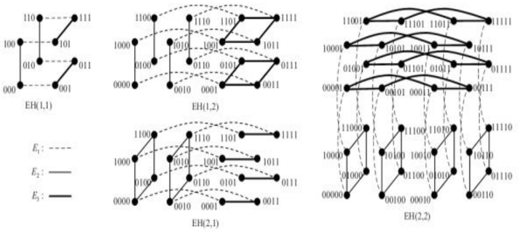
**Property 1.** [16] Let EH(s,t) be a traded hypercube for 1 ≤ s ≤ t.

1) The distance across of EH(s,t) is s + t +2.

2) EH(s,t) is isomorphic to EH(t,s).

3) EH(s,t) can be decayed into two duplicates of EH(s − 1,t) when s − 1 >0 or EH(s,t − 1) when t − 1 >0.

As per the second thing in Property 1, since EH(s, t) is isomorphic to EH(t, s), it pursues without loss of all inclusive statement that s ≤ t in this exploration.



The traded hypercubes EH(1, 1), EH(1, 2), EH(2, 1) and EH(2, 2) That the base level of EH(s,t) is s + 1. By the third thing in Property 1, we can deteriorate EH(s,t) into two subgraghs S0 and S1 that are isomorphic to EH(s− 1,t). Assume that V (S0) = {0us+t−1 •••ut+1ut •••u1u0 | ui ∈ {0,1} for 0 ≤ I ≤ s + t − 1}, and V (S1) =

{1us+t−1 •••ut+1ut •••u1u0 | ui ∈ {0,1} for 0 ≤ I ≤ s+t−1}. The subgraphs incited by the vertex sets V (S0) and V (S1) are signified by S0 and S1, separately. At that point, S0 ∼= EH(s−1,t) and S1 ∼= EH(s−1,t). Furthermore, there are 2s+t−1 edges between V (S0) and V (S1). Property 2. [17]

1) The subgraphs incited by the vertices of the structure st z∗•••∗}|

{utut−1 •••u10 and us+tus+t−1 •••ut+1z∗•••∗}| {1 in EH(s,t) are isomorphic to Qs and Qt, separately, where ∗∈ {0,1}.

2) In the traded hypercube EH(s,t) for 1 ≤ s ≤ t, the vertices in the vertex set Vc = {us+t •••ut+1ut •••u1u0 | u0 = c,ui ∈ {0,1} for 1 ≤ I ≤ s + t}, where c ∈ {0,1}, are vertex-transitive.

We watch the principal thing in Property 2. For utut−1 •••u1, there are 2t various decisions and for us+tus+t−1 •••ut+1, there are 2s unique decisions. Along these lines, there are 2t and 2s unmistakable incited subgraphs that are meant by Qs and Qt, separately. These subgraphs can be spoken to by , where 0 ≤ I ≤ 2t − 1 and by , where 0 ≤ j ≤ 2s − 1.

Let hs(u,v) (separately, ht(u,v)) indicate the quantity of various bits among u and v in measurements t+1 to s+ t (individually, 1 to t). That is, t also, ht(u,v) = Σti=1|ui − vi|. At the point when the setting is clear, hs(u,v) and ht(u,v) can be just composed as hs and ht, separately.

## V. ROUTING

### SHORTEST PATH ROUTING IN EH(s,t)

Most limited way steering is a directing strategy that limits the separation or way cost. In this segment, we propose a system for a most limited way steering calculation from the source vertex u to the goal vertex v in a traded hypercube EH(s,t), where 1 ≤ s ≤ t. We allude to this directing calculation as SPR (EH(s,t),u,v). The mark of a vertex in EH(s,t) is separated into three sections, where the furthest right 0-th bit is measurement 0, the centre t bits are from measurement 1 to t, and the furthest left s bits are from measurement t + 1 to s + t. See the representation in Fig.

A delineation for the name of a vertex in EH(s,t).

Calculation: SPR (EH(s,t),u,v)

Let u = us+t •••ut+1ut •••u1u0 and v = vs+t •••vt+1vt •••v1v0 be the source vertex and the goal vertex, separately, in EH(s,t) for 1 ≤ s ≤ t. There are four distinct cases that depend on the estimations of u0 and v0.

Case 1: u0 = 0 and v0 = 0.

Stage 1: Check the furthest left s bits of u and v. On the off chance that u and v are the equivalent in the furthest left s bits, at that point let the current u′ = u and go to Step 2. In the event that u and v vary in the furthest left s bits, at that point we course consecutively individually from measurement t + 1 to s + t while ui 6= vi for t + 1 ≤ I ≤ s + t (from little measurement to huge measurement).

Stage 2: Check the centre t bits of u and v. In the event that u and v are the equivalent in the centre t bits, at that point go to Step 4. In the event that u and v contrast in the centre t bits, at that point we course by changing the furthest right 0-th bit of u′ to 1. Next, we course successively individually from measurement 1 to t while ui 6= vi for 1 ≤ I ≤ t (from little measurement to huge dimension).Say, touching base at u′′.

Stage 3: We course by changing the furthest right 0-th bit of u′′ to 0.

Stage 4: This procedure courses along a briefest way from u to v and STOP.

Case 2: u0 = 0 and v0 = 1.

Stage 1: This progression is equivalent to Step 1 in Case 1.

Stage 2: We course by changing the furthest right 0-th bit of u′ to 1.

Stage 3: Check the centre t bits of u and v. On the off chance that u and v are the equivalent in the centre t bits, at that point go to Step 4. On the off chance that u and v vary in the centre t bits, at that point we course consecutively individually from measurement 1 to t while ui 6= vi for 1 ≤ I ≤ t (from little measurement to enormous measurement).

Stage 4: This procedure courses along a briefest way from u to v and STOP.

Case 3: u0 = 1 and v0 = 0.

Stage 1: Check the center t bits of u and v. In the event that u and v are the equivalent in the center t bits, at that point let the current u′ = u and go to Step 2. In the event that u and v contrast in the center t bits, at that point we course consecutively individually from measurement 1 to t while ui 6= vi for 1 ≤ I ≤ t (from little measurement to huge measurement). State, landing at u′.

Stage 2: We course by changing the furthest right 0-th bit of u′ to 0.

Stage 3: Check the furthest left s bits of u and v. In the event that u and v are the equivalent in the furthest left s bits, at that point go to Step 4. In the event that u and v contrast in the furthest left s bits, at that point we course consecutively individually from measurement t + 1 to s + t while ui 6= vi for t+1 ≤ I ≤ s+t (from little measurement to huge measurement).

Stage 4: This procedure courses along a most limited way from u to v and STOP.

Case 4: u0 = 1 and v0 = 1.

Stage 1: This progression is equivalent to Step 1 in Case 3.

Stage 2: Check the furthest left s bits of u and v. On the off chance that u and v are the equivalent in the furthest left s bits, at that point go to Step 4. In the event that u and v contrast in the furthest left s bits, at that point we course by changing the furthest right 0-th bit of u′ to 0. Next, we course consecutively individually from measurement t + 1 to s + t while ui 6= vi for t + 1 ≤ I ≤ s + t (from little measurement to enormous measurement). State, landing at u′′.

Stage 3: We course by changing the furthest right 0-th bit of u′′ to 1.

Stage 4: This procedure courses along a most brief way from u to v and STOP.

In order to demonstrate algorithm:

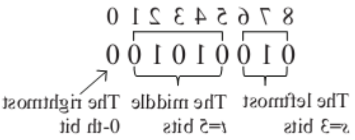
SPR(EH(s,t),u,v), we give the accompanying models.

Model 1. Give u and v a chance to be the source vertex and the goal vertex in EH(3,5), individually. Locate the most limited way from u to v utilizing the briefest way steering Algorithm*:* **SPR**(*EH*(3,5),*u*,*v*).

1) u=010010100, v=011011000.
2) u=001101010, v=100000111.
3) u=100011011, v=010110000.
4) u=110100011, v=011010101.

First, we use one example to recall the notations and terms that are illustrated in Fig.

Algorithm : **SPR**(EH(s,t),u,v) is simply written as Algorithm



The leftmost The middle The rightmost *s*=3 bits *t*=5 bits 0-th bit Fig. 3. The label of the vertex 010010100 in EH(3,5).

Then, we list route for each path in a step by step manner as follows. Note that the number i above the arrow $\rightarrow^{i}$ indicates that the i-th bit of the vertex is being converted. 1) Corresponding to Case 1 of Algorithm.

010010100 $\rightarrow^{6}$ 011010100 $\rightarrow^{0}$ 011010101 $\rightarrow^{2}$ 011010001 $\rightarrow^{3}$ 011011001 $\rightarrow^{0}$ 011011000. *2) Corresponding to Case 2 of Algorithm* .

001101010 $\rightarrow^{6}$ 000101010 $\rightarrow^{8}$ 100101010 $\rightarrow^{0}$ 100101011 $\rightarrow^{2}$ 100101111 $\rightarrow^{3}$ 100100111 $\rightarrow^{5}$ 100000111. *3) Corresponding to Case 3 of Algorithm* .

100011011 $\rightarrow^{1}$ 100011001 $\rightarrow^{3}$ 100010001 $\rightarrow^{5}$ 100110001 $\rightarrow^{0}$ 100110000 $\rightarrow^{7}$ 110110000 $\rightarrow^{8}$ 010110000. *4) Corresponding to Case 4 of Algorithm* .

110100011 $\rightarrow^{1}$ 110100001 $\rightarrow^{2}$ 110100101 $\rightarrow^{4}$ 110110101

$\rightarrow^{5}$ 110010101 $\rightarrow^{0}$ 110010100 $\rightarrow^{6}$ 111010100 $\rightarrow^{8}$ 011010100 $\rightarrow^{0}$ 011010101.

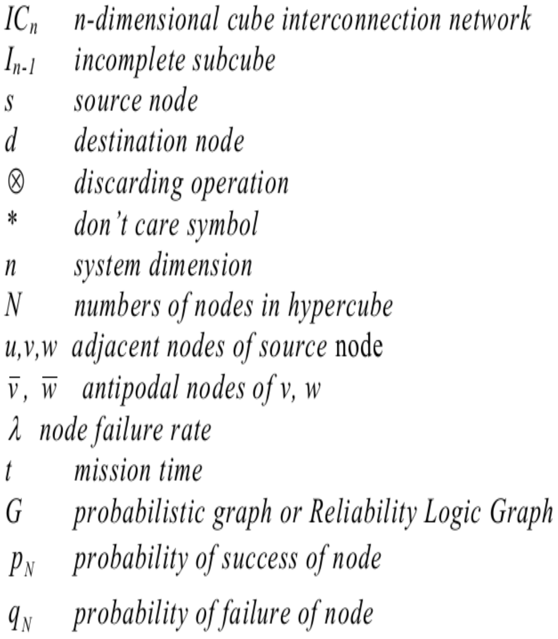
By the definition of EH(s,t), there are three disjoint edge sets, E1, E2, and E3. Each edge (x,y) in EH(s,t) varies by precisely one piece among x and y. For (x,y) ∈E1, the furthest right 0-th bit of x is not the same as y. For (x,y) ∈E2, one of the furthest left s bits of x is not the same as y and the furthest right 0-th bit of those two vertices must be 0. For (x,y) ∈E3, one of the center t bits of x is not quite the same as y and the furthest right 0-th bit of those two vertices must be 1.

## VI. PROPOSED APPROACH FOR RELIABILITY EVALUATION

The accompanying documentation and presumption s are made all through this pap er for unwavering quality assessment of 3D shape - based interconnection systems.

6.1 Notation and Assumptions

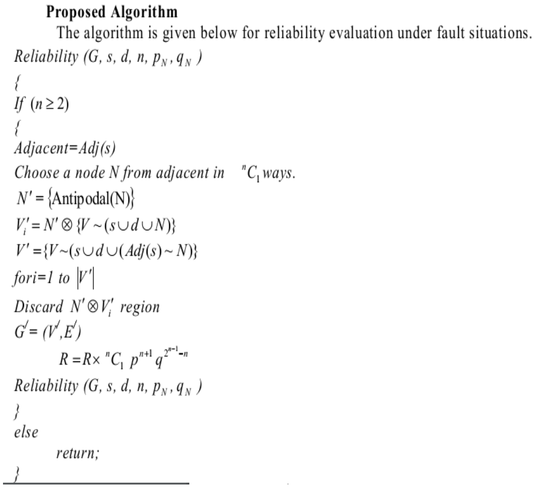
Documentation:



| | |
|---|---|
| $IC_n$ | *n-dimensional cube interconnection network* |
| $I_{n-1}$ | *incomplete subcube* |
| $s$ | *source node* |
| $d$ | *destination node* |
| $\otimes$ | *discarding operation* |
| * | *don't care symbol* |
| $n$ | *system dimension* |
| $N$ | *numbers of nodes in hypercube* |
| $u,v,w$ | *adjacent nodes of source* node |
| $\bar{v}$, $\bar{w}$ | *antipodal nodes of v, w* |
| $\lambda$ | *node failure rate* |
| $t$ | *mission time* |
| $G$ | *probabilistic graph or Reliability Logic Graph* |
| $p_N$ | *probability of success of node* |
| $q_N$ | *probability of failure of node* |

Assumptions:

1. Nodes failures are statistically independent.
2. No repair facility is available



**Proposed Algorithm**

The algorithm is given below for reliability evaluation under fault situations.

```
Reliability (G, s, d, n, $p_N$, $q_N$ )
{
If $(n \geq 2)$
{
Adjacent=Adj(s)
Choose a node N from adjacent in  $^{n}C_1$ ways.
$N' = \{\text{Antipodal(N)}\}$
$V_i' = N' \otimes \{V \sim (s \cup d \cup N)\}$
$V' = \{V \sim (s \cup d \cup (Adj(s) \sim N)\}$
fori=1 to $|V'|$
Discard $N' \otimes V_i'$ region
$G' = (V', E')$
        $R = R \times {}^{n}C_1\, p^{n+1} q^{2^{n-1}-n}$
Reliability (G, s, d, n, $p_N$, $q_N$ )
}
else
        return;
}
```

Algorithm:-

```
function fact(num)
{
   var rval=1;
   for (var i = 2; i <= num; i++)
       rval = rval * i;
   return rval;
}

function ProcessNode(id){
```

```
if(isNaN(document.forms[0].node.value))
   {
       alert(" Please enter Valid Number of nodes");
    return false;
   }
if(isNaN(document.forms[0].pl.value) ||
document.forms[0].pl.value>1 )
   {
       alert(" Please enter valid value for probability");
       return false;
   }

if(isNaN(document.forms[0].pn.value) ||
document.forms[0].pn.value>1)
   {
       alert(" Please enter valid value for node probability");
       return false;
   }
   var firstvalue = ";
  var secondvalue = ";
  if (document.forms[0].node.value.length == 0)
  {
       alert("Please enter Number of nodes");
       return false;
  }
  if (document.forms[0].pl.value.length == 0)
  {
       alert("Please enter probability of link failure");
       return false;
  }
  if (document.forms[0].pn.value.length == 0)
  {
       alert("Please enter probability of node failure");
       return false;
  }

 if(document.forms[0].pl.value == 1 &&
document.forms[0].pn.value == 1)
 {
    document.forms[0].out1.value = R = 0;
    document.forms[0].out2.value = NRH = 0;
    document.forms[0].out5.value = NRC = 0;
    document.forms[0].analysis.value = " Failure rates of
links and nodes are 1";
    return false;
}
else if(document.forms[0].node.value==1)
{
    document.forms[0].out1.value = R = 1-
document.forms[0].pn.value;
    document.forms[0].out2.value = NRH = 1-
document.forms[0].pn.value;
    document.forms[0].out5.value = NRC = 1-
document.forms[0].pn.value;
    document.forms[0].analysis.value = " For a single node
the reliability only depends of node failure rate";
    return false;
}

else
{
```

```
   //Torus odd reliability calculation
   N = document.forms[0].node.value;
   fl = parseFloat(document.forms[0].pl.value);
   fn = parseFloat(document.forms[0].pn.value);
   var R,fl,fn,f,rn,rn1,rn2,i,v,v1,v2,d,d1,d2,n,N;

   f = fl +fn*(1-fl);
// alert(" f is " +f);
   var res,res1,res2,res3,res4;
   var value,value1,value2,value3,value4,value5,value6;
   var out,out1,out2,out3,net;
   net = 0;
   net = N%2;

if(net==1)
{
   n = (N-1)/2;

   if(n == 2)
   {
      d1 = 1;
   }
   else
   {
      d1 = 0;
   }
   if(n==1)
   {
      d = 1;
   }
   else
   {
      d = 0;
   }
   if(n==3)
   {
      d2 = 1;
   }
   else
   {
      d2 = 0;
   }

   if( n >= 4)
   {
      v2 = 1;
      v1 = 1;
      v = 1;
   }
   else if (n>=3)
   {
      v1 = 1;
      v =1 ;
      v2 = 0;
   }
   else if (n>=2)
   {
      v2 = 0;
      v1 = 0;
      v = 1;
   }
```

```
    else
    {
       v = v1=v2 = 0;

    }
    rn2 = 0;
    rn1 = 0;

    for(i = 1; i <= n-3; i++)
    {   res = Math.pow(((1-f*f)/2),i);
        res1  = Math.pow(1-f,n-i-1) * v2 ;
        res2 = [(1-f*f) *d1 ];
        res3 = Math.pow(((1-f*f)/2),n-2);
        res4 = (2-f -f*f*f) * v;
        rn2 = (res *res1) + res2 + (res3*res4);

    }

    value5 = 0;
    for( i = 0; i <= n-2;i++)
    {
       value=Math.pow((1-f*f),n-1);
       value1=Math.pow(1+f*f-f*f*f,i+1) *rn2 * v1;
       value2=(value * value1)/Math.pow(2,(i+1));
       value3=Math.pow((((1-f*f)*(1-f)*(1+f*f-f*f*f))/2),n-1)
;
       value4 = f*f;

       var a = f;

       value5 = 1.00+ a;
       value6 = value5 - value4;
       rn1 = value2 + (value3 * value5) + rn1;

    }

    rn = rn1 * v + (1-fl-f*fl)*d;

    R = (1-f*f) *rn * (1+ f*f - f*f*f*f - f*f*f*f*f*f +
f*f*f*f*f*f*f)*(1-fl);
    var R1 = R.toFixed(6)
     document.forms[0].out1.value = R1;

    //end of torus odd
}
//Hypercube reliability
function IsInteger(x)
{
    var temp = parseInt(x);
    if (temp == x)
    {
     return true;
    }
     return false;
 }
 n = Math.log(N)/Math.log(2);

if(IsInteger(n))
{
    var y,result,result1,result2;
    var NR = new Array();
    NR[2] = Math.pow(1-fl,4) + 4*fl*Math.pow(1-fl,3);

    //n = Math.log(N)/Math.log(2);

    for(i = 2;i<=n-1;i++)
    {
        y = Math.pow(2,i-1);
        result = NR[i]* NR[i] * (1 - Math.pow(fl ,y));
        result1 = 2 * NR[i] * (1-NR[i]) * Math.pow((1-fl) ,y);
        result2 =  2 * NR[i] * (1-NR[i]) * Math.pow(2,i-1) * fl *
Math.pow((1-fl) ,(y-1)) * (1 - Math.pow(fl,i-1));
        NR[i+1] = result + result1+ result2;

    }
    y = Math.pow(2,n);
    NRH = Math.pow((1-fn), y) * NR[n];
    if( N == 2)
    {
    document.forms[0].out2.value = (1-fl);
    NRH = 1-fl;
    }
    else
    {
    var NR1 = NRH.toFixed(6)
     document.forms[0].out2.value = NR1;

    }
}
else
{

    document.forms[0].out2.value = "NA";
    NRH = "NA";
}

//end of hypercube
```

## VII. CONCLUSION

This paper has projected some of the fundamental activities that are required in structuring parallel algorithms on an Exchanged hypercube. These operations normally come into view as sub problems in solving other major problems in parallel computing. The initial investigators demonstrate that the Exchanged hypercube has gorgeous features; it safeguards the good features of the hypercube and decreases the measurement by a factor of two. In this paper, we introduced optimal routing and broadcasting algorithms. Additionally, we develop efficient algorithms for a portion of the fundamental parallel operations, for example, cooperative and prefix activities that show up normally as sup problems in other serious problems. A superior problem will be to reveal a greater amount of the engaging properties of the Exchanged hypercube. Another interesting issue is to demonstrate the capacity of this structure to process, reproduce other interconnection networks, and reconfigure itself within the presence of faults.

## REFERENCES

[1]. J. Al-Sadi, K. Day, and M. Ould-Khaoua, "Unsafety Vectors: A New Fault-Tolerant Routing for Binary n-Cubes," Journal of Systems Architecture, vol. 47, no. 9, pp-783-793, 2002.

[2]. G. Chiu and K. Chon, "Efficient Fault-Tolerant Multicast Scheme for Hypercube Multicomputers," IEEE Transactions on Parallel and Distributed Systems, vol. 9, no. 10, pp. 952-962, 1998.

[3]. K. Day and A. Al-Ayyoub, "The Cross Product of Interconnection Networks," IEEE Transactions on Parallel and Distributed Systems, vol. 8, no. 2, pp. 109-118, 1997.

[4]. R. Klasing, "Improved Compressions of CubeConnected Cycles Networks," IEEE Transactions on Parallel and Distributed Systems, vol. 19, no. 8, pp. 803-812, 1998.

[5]. T. Leighton, Introduction to Parallel Algorithms and Architectures: Arrays, Trees, and Hypercubes, Morgan Kaufmann, 1992.

[6]. M. Quinn, Parallel Computing: Theory and Practice, McGraw Hill, 1994.

[7]. Y. Saad and M. Schultz, "Topological Properties of the Hypercube," IEEE Transactions on Computers, vol. C-37, no. 7, pp. 867-872, 1988.

[8]. V. Sharma and E. Varvarigos, "Circuit Switching with Input Queuing: An Analysis for the dDimensional Wraparound Mesh and the Hypercube," IEEE Transactions on Parallel and Distributed Systems, vol. 8, no. 4, pp. 349-366, 1997.

[9]. L. Bhuyan and D. Agrawal, "Generalized Hypercube and Hyperbus Structures for a Computer Network," IEEE Transactions on Computers, vol. C-33, no. 4, pp. 323-333, 1984.

[10]. F. Preparata and J. Vuillemin, "The Cube-Connected Cycles: A Versatile Network for Parallel Computation," Communications of the ACM, vol. 24, no. 5, pp. 3000-309, 1981.

[11]. El-Amaway and S. Latifi, "Properties and Performance of Folded Hypercubes," IEEE Transactions on Parallel and Distributed Systems, vol. 2, no. 1, pp. 31-42, 19991.

[12]. Youssef and B. Narahari, "The BanyanHypercube Networks," IEEE Transactions on Parallel and Distributed Systems, vol. 1, no. 2, pp. 160-169, 1990.

[13]. S. Zheng, B. Cong, and S. Bettayeb, "The StarHypercube Hybrid Interconnection Networks," Proceedings of the ISCA International Conference on Computer Application in Design, Simulation, and Analysis, pp. 98-101, 1993.

[14]. E. Abuelrub, "Parallel Computation on Twisted Hypercubes," Al-Manarah Journal, vol. 5, no. 1, pp. 1-10, 2002.

[15]. E. Abuelrub, "Embedding Quad Trees into Twisted Hypercubes," Proceedings of the 2nd IASTED International Conference on Parallel and Distributed Systems, pp. 155-160, 1998.

[16]. E. Abuelrub and S. Bettayeb, "Embedding Rings into Faulty Twisted Hypercubes," Computers and Artificial Intelligence, vol. 16, no. 4, pp. 425-441, 1997.

[17]. K. Efe, "The Crossed Cube Architecture for Parallel Computation," IEEE Transactions on Parallel and Distributed Systems, vol. 3, no. 5, pp. 513-524, 1992.

[18]. W. Huang, J. Tan, C. Hung, and L. Hsu, "FaultTolerant Hamiltonicity of Twisted Cubes," Journal of Parallel and Distributed Computing, vol. 62, pp. 591-604, 2002.

[19]. P. Kulasinghe and S. Bettayeb, "Embedding Binary Trees into Crossed Cubes," IEEE Transactions on Computers, vol. 44, no. 7, pp. 923-929, 1995.

[20]. P. Kulasinghe and S. Bettayeb, "The MultiplyTwisted Hypercube with 5 or more Dimensions is not Edge-Transitive," Information Processing Letters, vol. 53, pp. 33-36, 1995.

[21]. Awwad and J. Al-Sadi, "On the Routing of the OTIS-Cube Network in the presence of Faults," The International Arab Journal of Information Technology, vol. 2, no. 1, pp. 17-23, January 2005.

[22]. Decayeux and D. Seme, "3D Hexagonal Network: Modeling, Topological Properties, Addressing Schemes, and Optimal Routing Algorithm," IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 9, pp. 875-884, September 2005.

[23]. J. Fu and G Chen, "Hamiltonicity of the Hierarchical Cubic Network," Theory of Computer Systems, vol. 35, pp. 59-79, 2002. [24] R. Lander abd M. Fischer, "Parallel Prefix Computation," Journal of the ACM, vol. 27, pp. 831- 838, 1980