Log- hypercube’s Properties and Routing Algorithm

Ye-seol Jung¹, Hyeong-Ok Lee²
1 Dept. of Computer Edu., Sunchon National Univ., Suncheon, Chonnam, 540-950, Republic of KOREA
2 Dept. of Computer Edu., Sunchon National Univ., Suncheon, Chonnam, 540-950, Republic of KOREA (corresponding author)

Abstract - The well-known hypercube interconnection network has an advantage of symmetric, routing and commercial algorithm, imbedding and is realized through commercialization. But it has an disadvantage that when its node numbers increase, its degree also increase proportionally. Because of this problem the hardware price goes up when the system is built based on hypercube. In this research we suggest a new design way that can improve the inter connection network’s degree. In this new design way the degree increases by 1 when the number of symbols that express the node are multiplied by 2. As a result the number of nodes become 2ⁿ, therefore it is a way to improve the inter connection network’s network price. In this research we suggest a new log-hypercube graph applying the suggested design way. The log-hypercube graph’s degree is log₂ n + 1, and its diameter 1.5n, and has recursive expansion, optimum break-down permission. Log-hypercube’s routing uses divide conquer technique and have a simple routing algorithm. The network cost of the log-hypercube is O(n log₂ n), which is better than the network cost of the hypercube O(n²).

Also if we apply the proposed inter relation network to the pre existing inter relation network then the network cost can be improved from inter relation network O(n log₂ n).

Keyword: interconnection network, hypercube, Log-hypercube, routing algorithm

1. Introduction

Flynn’s classification is a method of classifying computers, and it classifies by instruction word termination and data calculation. There are 4 types, SISD, SIMD, MISD, MIMD and SIMD, MIMD are parallel processing computers. SIMD is an array processor which uses one word for several datas at the same time, and MIMD is the most frequently used parallel process system and its several processors process each different data. The parallel process system can be classified as share memory and disperse memory by how it gets to the memory. The share memory is a structure based on bus and several processors use one group memory bus, so it does not require a separate transcending program. So, the processors are connected to memory which is able to connect to entire H/W and S/W therefore it is able to approach the memory at the same time, but there can be collisions and bottleneck phenomenon which downgrades the performance. On the other hand, in the dispersion memory each processors have a memory domain, and when it is sharing data there is a delay because it sends a message using a different processor. Each processors in the parallel processing system have its own memory device, and its processors are made of an inter connection network which is based on graph theory. The definition of an inter connection network is the location between processors and connection structure, and it is the one of the factors that decide the performance of the parallel system. Other than that there are data exchange method, CPU development for the improvement of the performance, and the performance of the inter connection network is influenced by the status therefore to upgrade the entire system’s performance and expansion there must be continuous researching. In this research we suggest a new inter connection network design method which can improve the inter connection network’s network cost. We use the proposed inter connection network design method to suggest and analyze the log- hypercube. The network cost of a hypercube type is O(n²), and the network cost of a log-hypercube is O(n log₂ n).

2. Related research

The network cost of the inter connection network is degree x diameter. The method to improving the network cost is to either decreasing the degree, or
decreasing the diameter. The degree means the price of the hardware and the diameter means the price of the software. Generally if the degree is increased the hardware price increase, but the diameter is decreased. Adversely if the degree is decreased the diameter is increased, and the waiting time for the data in the entire connection network increases therefore the processing quantity gets worse.

n-dimension hypercube \( Q_n \) is comprised of \( 2^n \) nodes and \( n2^{n-1} \) edges. Each nodes’ address is expressed as \( n \)-bit binary, and if the address between two nodes are exactly different by 1, then there is an edge between the two. Therefore hypercube \( Q_n \) is a regular graph with a degree of \( n \) [19]. The definition of a hypercube are as follows.

\[
V = \{ u_n u_{n-1} \cdots u_{i+1} u_i u_{i-1} \cdots u_2 u_1 | 1 \leq i \leq n, u_i = 0 \text{ or } 1 \}
\]

\[
E = \{ (u_n u_{n-1} \cdots u_{i+1} u_i u_{i-1} \cdots u_2 u_1), (u_n u_{n-1} \cdots u_{i+1} u_i u_{i-1} \cdots u_2 u_1) | 1 \leq i \leq n \}
\]

The 3-dimensional, 4-dimensional hypercube are [Figure 1].

3. Log-hypercube definition and routing algorithm

3.1 Log-hypercube definition and its properties

In the existing inter connection network the mesh, torus, honeycomb graphs has constant degree. But the rest, hypercube and star graph type have a degree of \( O(n) \). In this research we suggest a new method to decrease the hypercube’s degree increase rate and improve \( O(n) \) to \( O(\log n) \). The new method used the recursive dividing property to design the edge when the inter connection network is expanded. The edge design method makes the edge add 1 more edge when the symbol that express the node address becomes double. When the number of symbols which expresses the node address becomes \( n \) to \( 2n \) then the 1 edge is added to the number of edge. The relation between the number of nodes and degree is a form of a log function. Using the edge design method the inter connection network’s degree can be improved to \( O(\log n) \). Log-hypercube \( LH_n \) is a binary with a number of \( n(=2^k) \) is and expressed as \( S = s_0 s_{n-1} \cdots s_{n/2+1} s_{n/2} \cdots s_1 s_0 \). The edge of log-hypercube graph have log-edge \( L_p \) and complement-edge \( C \). Log-edge \( L_p \) is an edge which connects and exchanges number of \( p \) bit string \( \alpha \) and \( \beta \) in the node address \( s_0 s_{n-1} \cdots s_{n/2+1} s_{n/2} \cdots s_1 s_0 \) which is expressed by a binary number. In node \( S \) log-edge’s \( p \) means the number of \( \frac{n}{2^1} \), \( \frac{n}{2^2} \), \( \frac{n}{2^3} \), \( \frac{n}{2^{\log_2 n}} \) part bits, and \( L_p \) and \( L_n \) is the same.(\( 1 \leq p \leq \frac{n}{2^k} \)).

Bitstring \( \alpha \) and \( \beta \), which is exchanging number of \( p \) by log-edge \( L_p \), is moving to the left side from the first bit \( s_1 \) which is located from the very right side of node \( S = s_0 s_{n-1} \cdots s_{n/2+1} s_{n/2} \cdots s_1 s_0 \) and it is sequentially assigning \( \frac{n}{2^k} (1 \leq k \leq \log n) \).
If the exchanging number of \( \frac{n}{2^k} \) ’s part bitstring the same(\( \alpha = \beta \)), the exchanging bitstring \( \alpha \) and \( \beta \)'s bitstring will have \( \alpha \beta \). The exchanging part's bitstring's number is decreasing by \( n/2 \), therefore, log-edge \( L_p \)’s number is \( \log_2 n \times n = 2^k \). 1 \( \leq k \leq \log_2 n \). Complement-edge \( C \) is an edge which connects the node that has a complement from the very right and first side of the bit \( s_1 \) from \( S=s_n s_{n-1} s_{n-2} \ldots s_{n/2} \ldots s_2 s_1 \), \( (n = 2^k) \). Complement-edge is \( C \)-edge and exists in 1 in each nodes. Therefore \( n \) dimensional log-hypercube \( LH_n \)'s node have a number of \( 2^n = (2^k, 1 \leq k \leq \log_2 n) \), and its degree is \( \log_2 n + 1 \). Log-hypercube \( LH_4 \) graph is in [Figure2].

![Figure 2] Log-hypercube \( LH_4 \) graph

In this research adjacent node \( S' \) by node \( S \) and log-edge \( L_p \) is expressed as \( S' = L_p(S) \).

As a similar method, adjacent by node \( S \) and complement edge \( C \) is expressed as \( S' = C(S) \). For example if node \( S = 10001001 \), then the node adjacent to log-edge \( L_2 \) is \( S' = L_2(10001001) = 10000110 \). For example in log-hypercube \( LH_8 \) if node \( S = 10001001 \), let us name the adjacent node as \( S' \). Log-edge which is adjacent to node \( S \) , is \( L_1, L_2, L_4 \) therefore 3 of them, and it has 1 complement-edge. In node \( S = 10001001 \) the bit string which will be exchanged by log-edge \( L_4 \) is \( \alpha = 1000, \beta = 1001 \). \( \alpha \) and \( \beta \) are different to each other, therefore \( S' = L_4(10001001) = 10011000 \). The bitstring which will be exchanged by log-edge \( L_2 \) is \( \alpha = 10, \beta = 01 \). \( \alpha \) and \( \beta \) are different to each other, therefore \( S' = L_2(10001001) = 10001110 \). The bit string which will be exchanged by log-edge \( L_1 \) is \( \alpha = 0, \beta = 1 \). \( \alpha \) and \( \beta \) are different to each other, therefore \( S' = L_1(10001001) = 10001010 \). The node which is adjacent by \( C \)-edge in \( S = 10001001 \) is adjacent to the node which has a complement node \( S' \)'s very first right and first bit, therefore \( S' = C(10001001) = 10001000 \). Therefore \( S = 10001001 \) and its 4 adjacent nodes are \{10011000, 10000110, 10001010, 10001000\}.

In \( LH_n \) when the node’s binary bit string all have the same value name the node \( S = 11111111 \). There are 4 adjacent node \( S' \) to \( S = 11111111 \). node \( S = 11111111 \) and log-edge \( L_i \) will be exchanged with \( \alpha = 1111, \beta = 1111 \), and \( \alpha \) and \( \beta \) are the same. Node \( S' \) is connected to a node that has \( \alpha \) and \( \beta \)'s compliment(\( \alpha \beta \)). therefore node \( S' = L_i(11111111) = 00000000 \). Bitstring \( \alpha = 11, \beta = 11 \), which will be exchanged by node \( S = 11111111 \) and log-edge \( L_2 \)v. \( \alpha \) and \( \beta \) are the same, therefore node \( S' \) is connected to a node that has a compliment for \( \alpha \) and \( \beta \), therefore node \( S' = L_2(11111111) = 11110000 \). Bitstring \( \alpha = 1, \beta = 1 \) will exchange with Node \( S = 11111111 \) by log-edge \( L_1 \), and \( \alpha \ \beta \) are the same. Node \( S' \) is connected to a node which has \( \alpha \) and \( \beta \)'s compliment, therefore node \( S' = L_1(11111111) = 11111100 \). In node \( S = 11111111 \) the node which is adjacent by the compliment-edge is a node in which node \( S' \)'s very first and right side bit has the compliment. Therefore \( S' = C(11111111) = 11111110 \). The 4 adjacent nodes to Node \( S = 11111111 \) are \{00000000, 11110000, 11111100, 11111110\}. [properties 1] Log-hypercube \( LH_n \) is a connected graph(\( n = 2^k \), 1 \( \leq k \leq \log_2 n \)). [properties 2] Log-hypercube \( LH_n \) have a recursive expansion(\( n = 2^k \)). [properties 3] If log-hypercube \( LH_n \)'s node \( S = s_n s_{n-1} s_{n-2} \ldots s_{n/2} \ldots s_2 s_1 \), then \( S_A = s_n s_{n-1} s_{n-2} \ldots s_{n/2} \ldots s_2 s_1 , S_B = s_{n/2} \ldots s_2 s_1 \). Cluster \( LH_n(S_A) \) is a partial node graph. The graph has a binary bit string \( S_A \). In a log-hypercube \( LH \) graph there are cluster \( LH_n(S_A) \), and there \( 2^{n/2} \) of them. [properties 4] log-hypercube \( LH_n \)'s one
cluster $LH_{n}(S_A)$ have $2^{k/2}$ nodes.

[Properties 5] When log-hypercube $LH_n$’s
cluster $LH_{n}(S_A)$ is expressed as one super
node, the log-hypercube $LH_n$ is a
completed graph.

In the interconnection network the
connectivity is one of the criteria when
evaluating the fault tolerance. When dividing
the inter connection network into numbers
that are bigger than 2 the node connectivity
means the least number of nodes that must
be eliminated without getting the node
overlapped. In the inter connection network
when the inter connection network is still
connected; and even though maximum $k-1$
number of nodes are eliminated, and when $k$
number of nodes are eliminated if the inter
connection network is divided into more than
2 then that connection network’s node
connection is $k$. In an inter connection
network if the node connection rate and
degree has the same value then that
connection network have the maximally fault
tolerance. Inter connection network G’s node
connectivity, edge connectivity, and its degree
are $k(G)$, $\lambda(G)$, $\delta(G)$ each, then it is $k(G) \leq$
$\lambda(G) \leq \delta(G)$.

[arrange 6] $k(LH_{n})=\log n + 1(n = 2^k$
$1 \leq k \leq \log n)$

3.2 log-hypercube’s routing algorithm

Routing is a route for sending a message
from a random node $U$ which consists of a
graph, to $V$. Generally a route means not
overlapping the peak, but a short route. Node
$U$ and $V$ consists of the address of the node
by a binary $n$ number of bit. In $U$’s node,
routing to $V$’s node is like deciding the edge
rule’s applying sequence to make node $U$’s
bit string accordingly to $V$’s bit string. The
route that follows the routing algorithm is
$R(U \Rightarrow V)$. In a $n$-dimensional
log-hypercube $LH_n$, the starting node $U$ and
arrival node $V$ are expressed as
$U = u_n u_{n-1} \ldots u_1 u_{i-1} \ldots u_2 u_1$
$V = v_n v_{n-1} \ldots v_1 v_{i-1} \ldots v_2 v_1$. Node $U$’s
bit string $\alpha$ is $U_\alpha$, and $\beta$ is $U_\beta$, and it is
expressed as $U = U_{\alpha} U_{\beta}$.

The routing algorithm outline that used the
dividing conquer method are as follows.
(divide) Divide the starting node $U$, the arrival
node $V$’s number of n bits.

$U = U_{\alpha} U_{\beta}$, $V = V_{\alpha} V_{\beta}$
(conquer) Match node $U$’s $U_\beta$ into $V$’s $V_\alpha$.
Calculate log calculation $L_n/2$
(conquer) Match node $U$’s $U_\alpha$ into $V$’s $V_\beta$.

Conquering makes the divided $U_{\alpha}$ and $U_{\beta}$
i nto a starting node and it is a progress that
matches $U_\beta$ into $V_\alpha$, $U_\alpha$ into $V_\beta$. It express it
as $U_{\beta} \leftrightarrow V_{\alpha}$ or $U_{\alpha} \leftrightarrow V_{\beta}$. When the number
of bitstring becomes 2, then it stops dividing and
executes $2^1$-algorithm below.

$2^1$-routing algorithm: In a 2 dimensional
log-hypercube $LH_2$, it is a routing algorithm.
$LH_2$’s node address is expressed as 2bit
$U = u_2 u_1$, $V = v_2 v_1$.

$2^1$-routing algorithm { 
\textbf{if}((U = 00 \text{ and } V = 11) \text{ or } (U = 11 \text{ and } V = 00))
\text{ then perform } \frac{3}{2} \text{ calculation } L_n 
\text{ ;}
\text{ else}
\text{ step 1. if}(u_1 = v_1 \text{ and } u_2 = v_2) \text{ then exit ;}
\text{ step 2. if}(u_1 \neq v_2 \text{ or } u_2 \neq v_1) \text{ then change } u_1 \text{ into a complement ;}
\text{ step 3. if}(U = \overline{V}) \text{ then perform log calculation } L_n 
\text{ ;}
}

[Certifying] log-hypercube $LH_n$ is divided into
part bit string $\alpha$, $\beta$ and its standard is $n/2$.

When $|\alpha|$ and $|\beta|$ are 2 the $2^1$-routing
algorithm is performed. $2^1$- routing algorithm is
divided into 2 cases. When the node’s each
bit that consists of $\alpha$, $\beta$ is the same and when
its starting node and objective node are
complement, then it is expressed as:
$u_2 u_1 = v_2 v_1 (u_2 = u_1, v_2 = v_1)$. So, when
$R(00 \Rightarrow 11)$ or $R(11 \Rightarrow 00)$, it uses a route which is connected
to log-edge $L_4$. When $(U = 11)$ or $(V = 00)$ it is
included in the else.

$n(=2^k)$-Routing algorithm: When $n$-dimensional
log-hypercube $LH_n$’s bit string is $n/2$, then the
routing are as follows.

$n(=2^k)$- Routing algorithm { 
\textbf{step 1. execute} if$(n=2)$ then $2^1$
\text{-routing algorithm ;}
\textbf{step 2. execute} if$(U_\beta \neq V_\alpha)$ then
$(2^{k-1})$-routing algorithm ;
}
2\textsuperscript{2}-Routing algorithm: 4-dimensional log-
In hypercube \( LH_4 \) it is an algorithm.
When the bit string is 4 then the routing
are as follows.

\begin{itemize}
  \item \textbf{step 1.} If \((n = 2)\), \( 2^1 \)- Call the routing
algorithm

  \item \textbf{step 2.} If \((U_b \neq V_o)\), \( 2^1 \)-Call the
routing algorithm

  \item \textbf{step 3.} Log calculation \( L_2 \)

  \item \textbf{step 4.} If \((U_o \neq V_b)\), \( 2^1 \)-Call the
routing algorithm

\end{itemize}

\( n(= 2^k) \)-In a \( n \)-dimensional log-hypercube \( LH_n \) the routing algorithm
is an routing algorithm\((n = 2^k)\). The summary are as follows.
In \( Routing(U, V, n) \) it means that the
starting node \( U \), arrival node \( V \), bit string
\( n \).

\begin{verbatim}
Routing(U, V, n) {
  if (n == 2) {
    2^1 - routing algorithm;
    return;
  }
  Routing(U_b, V_o, \frac{n}{2})
  Execute \( L_{\frac{n}{2}} \)
  Routing(U_o, V_b, \frac{n}{2})
}
\end{verbatim}

As an example if \( U = 1010101 \) and \( V = 0011000 \), then the routing route \( R \) is
\( R(1010101\Rightarrow 0011000) \). In node \( U \),
\( U_o = 1010 \), \( U_b = 1011 \), and in node \( V \),
\( V_o = 0011 \), and \( V_b = 0110 \). In a routing
route \( 2^1 \)-routing algorithm is expressed as
\( (2^1)\Rightarrow \) and log-edge calculation is marked as
\( (L_{\frac{n}{2}})\Rightarrow \).

\begin{align*}
U & = 10101011 \quad (2^1) \rightarrow 10101000 \quad (L_o) \rightarrow \\
   & = 10100010 \quad (2^1) \rightarrow 10100011 \quad (L_i) \rightarrow \\
   & = 00110101 \quad (2^1) \rightarrow 00111001 \quad (L_2) \rightarrow \\
   & = 00110110 \quad (= V)
\end{align*}

When expressing the edge rule closely in
a \( 2^1 \)-routing algorithm it is as follows.

\begin{align*}
U & = 10101011 \quad (2^1 - L_1) \rightarrow 10101000 \quad (L_o) \rightarrow \\
   & = 10100010 \quad (2^1 - C) \rightarrow 10100011 \quad (L_i) \rightarrow \\
   & = 00110101 \quad (2^1 - L_1) \rightarrow 00111001 \quad (L_2) \rightarrow \\
   & = 00110110 \quad (= V)
\end{align*}

The distance between two random node is
the route distance by routing algorithm and
the graph's diameter is the maximum value of
the distance of all the nodes. Therefore the
diameter is the routing route's lowest limit.

[Summary 7] The diameter of the
log-hypercube is \( 1.5n - 1 \).

[Evidence] In a connection network, the diameter is the maximum value of the
shortest route between two random nodes that
are the farthest from each other, and the diameter of \( LH_n \) is called \( k(LH_n) \). When
\( n \geq 2 \) it repeats the dividing process, and
when \( n = 2 \), it executes the \( 2^1 \)- routing
algorithm.

\( 2^1 \)-routing algorithm’s worst time complex
rate is 2, when step 2 and step 3 are all
executed. Therefore \( k(LH_2) \) is 2. \( 4(= 2^2) \)
-routing algorithm’s worst time complex rate is
when it adds step 2, 3, and 4 all together.
step 2,4 is \( k(LH_2) \), and step 3 is 1. Therefore
\( k(LH_2) \) is 5. \( 16(=2^4) \)- routing algorithm’s
worst time rate is when it adds step 2, 3, and
4 all together. Step 2 and 4 are \( k(LH_2) \) and
step 3 is 1. Therefore \( k(LH_8) \) is 11. The summary are as follows.

\begin{align*}
k(LH_2) & = 2 \\
k(LH_4) & = k(LH_2) + 1 + k(LH_2) = 2 \\
(k(LH_2)) + 1 & = 5 \\
k(LH_8) & = k(LH_4) + 1 + k(LH_4) = 2 \\
(k(LH_4)) + 1 & = 11 \\
k(LH_{16}) & = k(LH_8) + 1 + k(LH_8) = 2 \\
(k(LH_8)) + 1 & = 23 \\
\end{align*}
\[ k(LH_n) = k(LH_{n/2}) + 1 + k(LH_{n/2}) = 2 \]
\[ (k(LH_{n/2}))) + 1 \leq 1.5n - 1 \]

Therefore the diameter of \( LH_n \) is 1.5n - 1 and its network cost is \((\log_2 n + 1)\) \(\times\) diameter(1.5n - 1) \(\approx\) 1.5\(n\log_2 n\) + 1.5n - \(\log_2 n\) - 1. By the notation of Big O, the network cost is \(O(n\log_2 n)\).
When the hypercube type graphs have the same number of nodes \(n = 2^k, (1 \leq k \leq \log_2 n)\), we used the evaluation measure in [table 1]

<table>
<thead>
<tr>
<th>Number of nodes</th>
<th>Degree</th>
<th>Diameter</th>
<th>Network cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypercube</td>
<td>(2^n)</td>
<td>(n)</td>
<td>(n^2)</td>
</tr>
<tr>
<td>Folded Hypercube</td>
<td>(2^n)</td>
<td>(n + 1)</td>
<td>(\lfloor \frac{n}{2} \rfloor)</td>
</tr>
<tr>
<td>HCN(n, n)</td>
<td>(2^n)</td>
<td>(n + 1)</td>
<td>(n + \lfloor \frac{n}{2} \rfloor + 1)</td>
</tr>
<tr>
<td>Log-hypercube</td>
<td>(2^{\log_2 n})</td>
<td>(\log_2 n + 1)</td>
<td>1.5n - 1</td>
</tr>
</tbody>
</table>

summary 1. The diameter of \(BBN_n\) is maximum 2n - 1.
Evidence. For the evidence we will name the primitive node and objective node as \(v = v_1v_2\ldots v_n\) \(w = w_1w_2\ldots w_n\). Each distance between two nodes, v and w is the total sum of the r-edge and c-edge, and it consists the route that was set by the simple routing algorithm. When the \(\text{dist}(v, w)\) is at its maximum value, it is when the bit strings consisting v and w are all the same. In this case the route which was set by the simple routing algorithm consists the route using the r-edge n times, c-edge n times. Therefore \(\text{dist}(v, w) = 2n\), in this case \(v = w\). So the route does not exist. Therefore the distance calculated by the suggested simple routing algorithm is smaller than 2n, therefore \(BBN_n\)’s diameter is when at maximum 2n - 1.

4. Conclusion
The existing interconnection network’s network cost is high because the degree increase rate is usually proportional to the increase rate of the number of nodes.

In this research we suggested a new interconnection network method which its degree increase rate have a constant value at the increase rate of the number of nodes. Also the new design method have a recursive property, therefore it has the advantage of being able to apply the divide conquer method in a routing, commercial algorithm. In this research we applied the new design method and suggested the log-hypercube. The log-hypercube \(LH_n\) have \(2^n\) nodes, \(\log_2 n + 1\) degrees, and its network cost is \(O(n\log n)\). The log-hypercube \(LH_n\) have advantages which are as follows: A recursive structure and a simple routing algorithm, and optimum default allowance rate. The hypercube’s degree and diameter is n each, and the network cost is \(O(n^2)\). Because of the result in [table 1], which compared the existing inter connection network with the log-hypercube, it can be concluded that the log- hypercube is better on the perspective of network cost efficiency. Also in this research we can improve the existing inter connection network’s network cost from \(O(n^2)\) to \(O(n\log n)\), if we use the new inter connection network design method as suggested in this research. From now on research on Commercials that can use log-hypercube, imbedding, the optimum routing algorithm is needed.

Acknowledgment
This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2017R1D1A3B03032173)

Reference


