Journal of Korea Multimedia Society Vol. 18, No. 9, September 2015(pp. 1083-1090)
http://dx.doi.org/10.9717/kmms.2015.18.9.1083

Application of the Hamiltonian circuit Latin square to a Parallel Routing Algorithm on Generalized Recursive Circulant Networks

Dongmin Choi†, Ilyong Chung††

ABSTRACT

A generalized recursive circulant network(GR) is widely used in the design and implementation of local area networks and parallel processing architectures. In this paper, we investigate the routing of a message on this network, that is a key to the performance of this network. We would like to transmit maximum number of packets from a source node to a destination node simultaneously along paths on this network, where the i^{th} packet traverses along the i^{th} path. In order for all packets to arrive at the destination node securely, the i^{th} path must be node-disjoint from all other paths. For construction of these paths, employing the Hamiltonian Circuit Latin Square(HCLS), a special class of (n x n) matrices, we present $O(n^2)$ parallel routing algorithm on generalized recursive circulant networks.

Key words: Node-disjoint Paths, Generalized Recursive Circulant Networks, Hamiltonian Circuit Latin Square

1. INTRODUCTION

A generalized recursive circulant network(GR) [1] is widely used in the design and implementation of local area networks and parallel processing architectures.

This network denoted by $GR(m_h, m_{h-1}, ..., m_1)$ has $\prod_{i=1}^{h} m_i$ vertices where $m_i \geq 2$ for $1 \leq i \leq h$. Index I is a dimension of the labeling system while m_i is the base of dimension i. Each vertex in the network can be expressed as an h-tuple $(x_h, x_{h-1}, ..., x_1)$ with $0 \leq x_i \leq m_i - 1$ for $i = 1, 2, ..., h$. The number of edges at a node is $\sum_{h}^{i=1} k_i$, if $m_i > 2$, then $k_i = 2$, else $k_i = 1$.

The routing of message is thus a key to performance of networks. We look for algorithms that are capable of handling, multiple data items simultaneously transmitted from a starting node to a destination node. There are a few algorithms on the hypercube-like network that allow us to locate n disjoint paths such as the Hamiltonian path Algorithm[2], the Rotation Algorithm using Tree Structure[3], the Disjoint Path Algorithm[3], and the Routing Algorithms[4]. Node-disjoint paths can be categorized into three classes-one-to-one, one-to- many, and many-to-many. The first class considers the disjoint paths from a source node to a destination node, the second from a source node to k destination nodes and the third from k source nodes to k destination nodes. One-to-one disjoint paths were constructed on several networks such as hypercubes[5], k-ary n-cubes[6] and star graphs[7]. One-to-many disjoint paths were designed on hypercubes [8,9] and star graphs [8]. For

※ Corresponding Author : Ilyong Chung, Address: Dept. of Computer Engineering, Chosun University, Seosoeg-Dong, Dong-Gu, Gwangju, 501-759, Korea, TEL : +82-62-230-7712, FAX : +82-62-230-7754, E-mail : iyc@chosun.ac.kr
Receipt date : Apr. 30, 2015, Revision date : July 20, 2015
Approval date : July 28, 2015
† Division of Undeclared Majors, Chosun University (E-mail : jdmcc@chosun.ac.kr)
†† Dept. of Computer Engineering, Chosun University
※ This study was supported by research funds from Chosun University, 2015

the generation of many-to-many disjoint paths some work has been done [10,11].

In this paper, we propose an algebraic approach to routing of messages on $GR(m_h,m_{h-1},...,m_1)$. As described above, maximum number of packets are simultaneously transmitted from a starting node to a destination node, where the i^{th} packet is sent along the i^{th} path. In order for all packets to arrive at a destination node quickly and securely, the i^{th} path must be node-disjoint from all other paths. To accomplish this, we employ the operations of nodes presented in Cayley Graph[12] and the special matrix called as Hamiltonian Circuit Latin Square(HCLS)[5], which was used to find a set of node-disjoint paths on hypercube network[5], circulant networks[13] and recursive cube of rings networks[14].

2. DESIGN OF THE SHORTEST PATH

Let A and B be any two nodes on $GR(m_h,m_{h-1},...,m_1)$. The paper's objective is to find algorithms that will facilitate the transmission of data from a source node to a desired node on generalized recursive circulant networks. In order for the data to traverse to a desired node, it must cross, successively, intermediate nodes along a path.

Definition 1. $GR(m_h,m_{h-1},...,m_1)$ is defined as follows:

Each vertex in the network can be expressed as an h-tuple $(x_h,x_{h-1},...,x_1)$ with $0 \le x_i \le m_i-1$ for $i=1,2,...,h$ and the edge set E is $\{ (v,w)|w=(x_h,...,x_i,...,x_1), +i^{\pm}=(x_h,...,x_i \pm 1,...,x_1)$, where $v=(x_h,...,x_i,...,x_1)$, if $x_i+1=m_i$, then $x_i=0, x_{i+1}=x_i+1$ and if $x_i-1=-1$, then $x_i=m_i-1$, $x_{i+1}=x_{i+1}-1$.

The number of edges at a node is $\sum_{h}^{i=1} k_i$, if $m_i>2$, then $k_i=2$, else $k_i=1$. It means that if $m_i>2$, then $(m_h,...,m_i,...,m_1)$ has two edges at dimension $i-(m_h,...,m_{i+1},...,m_1)$ and $(m_h,...,m_{i-1},...,m_1)$, else it has one edge $-(m_h,...,m_{i+1},...,m_1)$. When the operations occur at the leftmost dimension m_h, since m_{h+1} is invisible, addition and subtraction to m_{h+1} are neglected. This means that vertices $(0,x_{h-1},...,x_1)$ and $(m_h-1,x_{h-1},...,x_1)$ are adjacent (see Fig. 1).

Researches on generalized recursive circulant networks(GR) are actively performed in graphic-theoretical areas such as embedding and fault-tolerance. As mentioned earlier, a node on Cayley Graph can traverse to another node by performing a certain operation. We now employ these operations to GR.

Definition 2. The Routing function R for $i^{\pm}$ is as follows:

$R_{i^{\pm}}(A)=A+i^{\pm}(\mathrm{mod}\ n)$, where A is an address of a node

Node A is physically connected to k neighboring nodes and these paths are node-disjoint. Data is transmitted from a source node along the i^{th} path. The path above is selected by the routing function described in Definition 2. To do this, the relative address of two nodes can be obtained below.

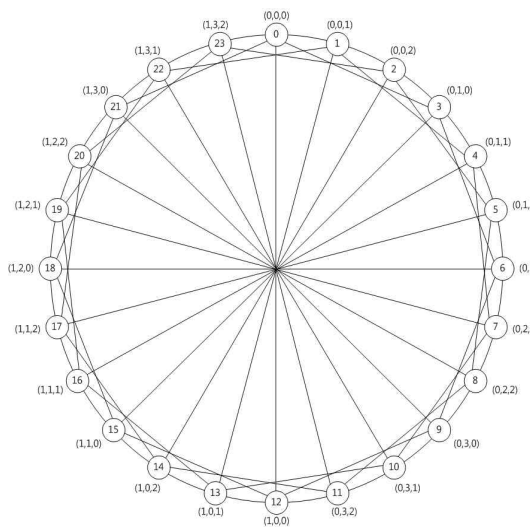

Fig. 1. $GR(2,4,3)$.

Definition 3. Let A and B be $(a_h,a_{h-1},...,a_i,...,a_1)$ and $(b_h,b_{h-1},...,a_i,...,b_1)$, respectively. The relative address r of nodes A and B is computed as the difference between each dimension of two nodes.

$r=(b_h-a_h,b_{h-1}-a_{h-1},...,b_i-a_i,...,b_1-a_1)$

Let A and B be (1,2,1) and (3,2,2) on GR(4, 5, 3). The relative address r is (2,0,1), which can be described as a sequence S of operations $<3^+,3^+,1^+>$ since $r_3=2$ and $r_1=1$. Also, x^+ is interchangeable to x and S is $<3,3,1>$.

Definition 4. Let *T(A,S)* be a logical transmission path starting form node A to a destination node, where r is a multiset and a sequence of operations, via which data can reach at a destination node. *T(A,S)* is determined by the order of elements in *S*.

Given node *A* and multiset *S* on *GR(4,5,3)*, we would like to transmit to a destination node via intermediate nodes. Let node *A* and set *S* be (1,2,1) and <3,3,1>, respectively. Operations in S are applied to the routing function sequentially and then the traversal of data is (1, 2, 1) → (2, 2, 1) → (3, 2, 1) → (3, 2, 2).

Since the paper's objective is to find an algorithm that will facilitate the secure transmission of data from a starting node to a destination node, those operations should be appropriate for this objective. Given S, a size of this sequence must be minimized since a routing distance is equal to a size of a sequence.

For example, supposed that $S=<1^-,1,1,1,3,3,1^->$ on *GR(4,5,3)*. then S can be minimized to $<1^-,2,3,3,1^->$since $x_1+1+1+1(\mathrm{mod}\ 3)=x_1$ and a carry 1 is added to x_2.

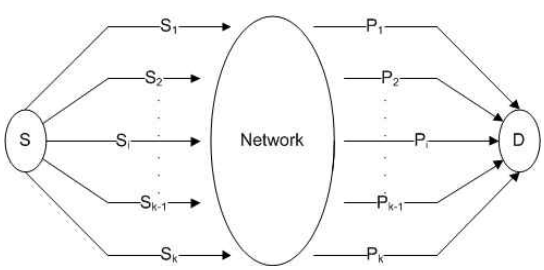

Fig. 2. The operations applied at the first step and the last step.

3. Application of the Hamiltonian Circuit Latin Square to the Parallel Routing Algorithm on Generalized Recursive Circulant Networks

The k packets are transmitted from a source node to a destination node on $GR(m_h,m_{h-1},...,m_1)$. In this section, we would like to construct a set of k node-disjoint paths in order to transmit these packets safely. First, k packets residing at a node are sent to its neighboring nodes along a set of disjoint paths. These paths are generated by employing k different operations at the beginning step and by performing k different operations at the last step in order to arrive at a destination node. The figure below illustrates the operations applied to generate k paths from a source node to a destination node.

The i^{th} packet is transmitted along the i^{th} path, the first intermediate node of which is obtained from applying operation s_i at a starting node and the last intermediate node transmits this packet to a destination node by applying operation p_i. In some cases, s_i and p_i can be the same.

Definition 5. Let O^s be a set of operations occurring at a starting node when k packets are transmitted simultaneously and Let O^d be a set of operations occurring at the last k intermediate nodes in order for k packets to arrive at a destination node. These sets are defined as follows:

$O^s=\{s_1,s_2,...,s_{k-1},s_k\}$

$O^d=\{p_1,p_2,...,p_{k-1},p_k\}$

$O^s=O^d$

We now apply the HCLS(Hamiltonian Circuit Latin Square) to find a set of m shortest and node-disjoint paths. A latin square is a square ma-

trix with m^2 entries of m different elements, none of the elements occurring twice within any row or column of the matrix. The integer m is called the order of latin square. The next definition describes the HCLS.

Definition 6. The HCLS M^1 is constructed as follows: Given distinct m points $a_0, a_1, ..., a_{m-2}, a_{m-1}, a$, a Hamiltonian circuit $a_i \rightarrow a_j \rightarrow ... \rightarrow a_k \rightarrow a_i$ is randomly selected. On the circuit each row of M^1 can be obtained from the Hamiltonian path. starting at any position $ak (0 \le k \le m-1)$, under the condition that no two rows begin at the same position. If a Hamiltonian path is $a_i \rightarrow a_j \rightarrow ... \rightarrow a_k$, then the row obtained from it is $[a_i a_j ... a_k]$.

From the HCLS given in Definition 6, the MHCM(Modified Hamiltonian Circuit Matrix) is constructed below.

Definition 7. Given the HCLS $M^1 = (a_{ij})$, the MHCM M^2 is constructed as follows: $M^2 = (A_{ij}), A_{ij} = (a_{i,0}, a_{i,1}, ..., a_{i,j-1}, a_{i,j}), a_{ij} \in Z_m, 0 \le i \le m-1, 0 \le j \le m-2$.

The following example will provide a better understanding of Definition 6 & 7.

Example 1. If the Hamiltonian path is 2 → 3 → 1 → 4, then the HCLS M^1 and the MHCM M^2 are constructed as follows.

$$M^1 = \begin{bmatrix} 2 & 3 & 1 & 4 \\ 3 & 1 & 4 & 2 \\ 1 & 4 & 2 & 3 \\ 4 & 2 & 3 & 1 \end{bmatrix} \qquad M^2 = \begin{bmatrix} (2) & (2,3) & (2,3,1) \\ (3) & (3,1) & (3,1,4) \\ (1) & (1,4) & (1,4,2) \\ (4) & (4,2) & (4,2,3) \end{bmatrix}$$

Definition 8. Call M^3 the MGNDP(Matrix for Generating Node-Disjoint Paths). No two entries of this matrix are the same. It satisfies the following conditions.

$M^3 = (A_{i,j}), A_{i,j} \subset \{Z_m \cup \{s\}\}$,

$0 \le i \le m-1, 0 \le j \le m-2$ s means "stay at the current node".

(1) $|A_{i,j}| = j+1$

(2) $A_{i,j} \neq A_{k,l}$, if $i \neq k$

(3) $A_{i,j} \subset A_{i,j+1}$

(4) $A_{i,j+1} = A_{i,j} \cup \{s\}$, if $s \in A_{i,j}$

Referring to [5], the MHCM satisfies the conditions of the MGNDP(Matrix for Generating Node-Disjoint Paths), which constructs a set of node-disjoint paths on the hypercube network. Since The HCLS belongs to the latin square, a set of elements in the first column is the same as that of the last column. On generalized recursive circulant networks, an element in the HCLS is represented as an operation. Also, O^s and O^d in Definition 5 are described as all the elements in the first column and all the elements in the last column, respectively. We, intuitively, realize that a set of m node-disjoint paths is generated if the number of distinct sequences of operations for arriving at an arbitrary node in a short time is $m (m \le k)$. The remaining operations excluding these distinct operations from O^s and O^d, should be performed.

Example 2. Let A and B be (1,2,1) and (3,2,2) on $GR(4,5,3)$. According to Definition 4, sequence S is computed as < 3, 3, 1 > and a set of node-disjoint paths is generated as follows. A set of distinct operations in S is < 1, 3 >. Using these operations, (2 × 2) HCLS can be obtained from Definition 6.

$$HCLS = \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}$$

Operations in the i^{th} row of the HCLS generated above are performed for traversal of the i^{th} packet and the remaining operation is also executed at the point except the first and the last points. In order to assure that these paths are node-disjoint, the remaining operation should be executed at the same time. In this example, the running point of the remaining operation is the second. Physical transmission paths from node A to node B are described below.

Path 1 : T(A,< 1, 3, 3 >) : A → (1, 2, 2) → (2, 2, 2) → B

Path 2 : T(A,< 3, 3, 1 >) : A → (2, 2, 1) → (3,

2, 1) → B

From Definitions 1 and 6, O^s and O^d are obtained, that is, $O^s = O^d = \{1, 1^-, 2, 2^-, 3, 3^-\}$. Excluding the operations 1 and 3, which are performed in the first and the last steps, from O^s and O^d, O^s and O^d become $\{1^-, 2, 2^-, 3^-\}$. Recall that we deal with design of six node-disjoint paths, out of which two node-disjoint paths are now constructed. Examining Definition 2, when a packet traverses in a positive direction on the i^{th} dimension and then traverses in a reverse direction on the same dimension, a packet comes back to its original position. This idea is applied to generation of disjoint paths. If $\{i, i^-\}$ exists as a subset of O^s and O^d, these operations are executed at the first and the last steps and the operations obtained from design of the shortest distance at the middle steps. Path 3 and Path 4 are constructed below.

Path 3 : $T(A, <2,1,3,3,2^->)$: A → (1, 3, 1) → (1, 3, 2) → (2, 3, 2) → (3, 3, 2) → B

Path 4 : $T(A, <2^-,1,3,3,2>)$: A → (1, 1, 1) → (1, 1, 2) → (2, 1, 2) → (3, 1, 2) → B

Excluding $\{2, 2^-\}$ from O^s and O^d, O^s and O^d become $\{1^-, 3^-\}$. While operations working at the first and the last steps are not the same, operations performing at the remaining steps are the same. As described earlier for Path 3 and Path 4, a packet traverses in a positive direction on the i^{th} dimension and then traverses in a reverse direction on the same dimension. Likewise a packet traverses twice in the same direction on the i^{th} dimension and then traverses twice in the reverse direction on the same dimension, and then this packet comes back to its original position. A sequence of operations for a path is now obtained. Firstly, two operations traversed in the same direction are chosen for the first and the last steps. Secondly, two operations in reverse direction and the operations for the shortest distance would be executed at the middle steps. However, these operations must be minimized in order to be a shortest path and then the minimal operations are executed. Paths 5 and 6 are generated by handling cases of $\{1^-\}$ and $\{3^-\}$, respectively.

Path 5 :

$T(A, <1^-,1,1,1,3,3,1^->) = T(A, <1^-,2,3,3,1^->)$: A → (1, 2, 0) → (1, 3, 0) → (2, 3, 0) → (3, 3, 0) → B

Path 6 :

$T(A, <3^-,3,3,3,3,1,3^->) = T(A, <3^-,1,3^->)$: A → (0, 2, 1) → (0, 2, 2) → B

The process to find a set of node-disjoint paths is described above. We now propose a parallel routing algorithm that generates a set of node-disjoint paths on generalized recursive circulant networks. In this paper, we will use the term "distance" between two nodes to refer to the number of routing steps(also called hopcounts) needed to send a message from one node to another.

GR_Routing_Algorithm

A ← an address of a starting node A

B ← an address of a destination node B

O^s ← a set of operations occurring at a starting node A

O^d ← a set of operations requisite for reaching to a destination node B

begin

(1) Compute the relative address r of nodes A and B

(2) Using the relative address r, a sequence S of operations to arrive at node B in a short time are produced.

(3) In order to design a set of node-disjoint paths, find a sequence S_1 of distinct elements in S. A set of $|S_1|$ shortest and node-disjoint paths are generated. Each path of length is | S |,

(3-1) Using S_1, (n × n) HCLS is constructed, where n =$|S_1|$.

(3-2) Operations in the i^{th} row of the HCLS are performed for traversal of the i^{th} packet and the remaining operations in S should be executed

at the point except the first and the last points.

(3-3) $O^s \leftarrow O^s - S_1$ and $O^d \leftarrow O^d - S_1$

(4) Construct two node-disjoint paths, each path has length $|S|+2$.

(4-1) If $O^s = \phi$, the process is finished.

(4-2) If a set of $\{i,i^-\}$ is found in O^s, then these operations are performed at the first and the last steps, and the operations in S at the middle steps, otherwise go to (5).

(4-3) $O^s \leftarrow O^s - \{i,i^-\}$, $O^d \leftarrow O^d - \{i,i^-\}$ and go to (4-1).

(5) Generate the remaining paths.

(5-1) If $O^s = \phi$, the process is finished.

(5-2) For $i \in O^s (m_i > 2)$, produce S_2 of minimum number of operations by reducing the size of $S \cup \{i^-,i^-\}$, $S_2 = \{i, \min(S \cup \{i^-,i^-\}), i\}$

(5-3) Operation i is performed at the first and the last steps and the remaining operations of S_2 are executed at the middle steps.

(5-4) $O^s \leftarrow O^s - \{i\}$, $O^d \leftarrow O^d - \{i\}$ and go to (5-1).

end.

GR_Routing_Algorithm is thus fairly straightforward. The time involved in performing Steps (1), (2) and (4) is small compared to the remaining steps. The first, second and fourth steps of this algorithm does not, therefore, contribute to an objectionable overhead.

Theorem 1. The construction of a set of k node-disjoint paths can be performed in $O(n^2)$ time.

Proof: Applying the Algorithm above to generate k node-disjoint paths. Important steps for determining time complexity requisite for the Algorithm are two things. One is to design the HCLS, which requires $O(n)$. The other is to run Step (5) of *GR_Routing_Algorithm*. In Step (5), in order to run i in O^s at the first and the last steps in transmission, a sequence of operations is determined to $S \cup \{i^-,i^-\}$, and is reduced. Since the reduction process requires O(n) time and the number of elements in O^s is less than n, Step (5) can be computed in $O(n^2)$. Therefore, a set of k node-disjoint paths can be created in $O(n^2)$ time.

The paper's objective is to find a set of k node-disjoint paths between two nodes. The major topological characteristics of the generalized recursive circulant network is considered and the property of k paths obtained from the Algorithm is proven below.

Theorem 2. The k transmission paths produced by *GR_Routing_Algorithm* are node-disjoint.

Proof: The k packets residing at node A are now transmitted at time t_0. These packets reach to its k neighboring nodes at time t_1. Then, each packet traverses to a neighboring node along a shortest path. Suppose that two packets arrive at the same node except a destination node during transmission. In order for this case to occur, the following condition should be satisfied. Let S_i and S_j be two sequences of operations for sending two packets from a starting node to two arbitrary nodes at time t_{ij}, where t_{ij} means that one packet arrives at time t_i and the other arrives at time t_j. Then, S_i and S_j should be the same. In other words, if these sequences do not appear, a set of node-disjoint paths can be constructed. According to the Algorithm described above, three classes of paths are generated on this network. We now consider three cases.

Case 1: Let S_{1i} and S_{1j} be two sequences of operations obtained from design of the shortest distance. Then S_{1i} and S_{1j} must not be the same due to the properties of the MGNDP.

Case 2: Let S_{2i} and S_{2j} be two sequences of operations acquired by running Algorithm-(4). Then S_{2i} and S_{2j} must not be the same because the first elements of S_{2i} and S_{2j} are S_k and S_k^- , respectively

and the rests of them are the same. In order for paths generated through case 1 and case 2 to be node-disjoint, we prove that S_{1i} and S_{2j} must not be the same. Considering the first element S_k of S_{2j}, it is not an element of S_{1i}. Therefore, these sequences are different all the times.

Case 3: Let S_{3i} and S_{3j} be two sequences of operations generated by performing Algorithm-(5), and T be a sequence of operations, which creates a shortest path from a source node to a desired node. Then, the first elements of S_{3i} and S_{3j} do not belong to T, S_{3i} does not contain the first element of S_{3j}, and S_{3j} does not contain the first element of S_{3i}. So, S_{3i} and S_{3j} should not be the same.

To prove that the paths created by all the cases are node-disjoint, S_{3i} and S_{2j} must not be the same. The method of proof is the same as the case 2. Looking into the first element of S_{2j}, the element is not a part of S_{3i}. Therefore, these sequences are different all the time.

4. CONCLUSION

In this paper, we present the algorithm that generates a set of k node-disjoint paths on $GR(m_h, m_{h-1}, \ldots, m_1)$, employing the Hamiltonian Circuit Latin Square(HCLS). Important steps for determining time complexity requisite for the algorithm are two things. One is to design the HCLS, which needs $O(n)$. The other is to execute Step (5) of *GR_Routing_Algorithm*, which requires $O(n^2)$. Therefore, we can create $O(n^2)$ parallel routing algorithm for constructing a set of k node-disjoint paths.

REFERENCE

[1] S. Tang, Y. Wang, and C. Li, "Generalized Recursive Circulant Graphs," *IEEE Transactions on Parallel Distributed Systems*, Vol. 23, No. 1, pp. 87-93, 2012.

[2] I. Chung, "Design of a Set of One-to-Many Node-Disjoint and Nearly Shortest Paths on Recursive Circulant Networks," *Journal of Korea Multimedia Society*, Vol. 16, No. 7, pp. 897-904, 2013.

[3] S.L. Johnson and C. Ho, "Optimum Broadcasting and Personalized Communication in Hypercube," *IEEE Transactions on Computer*, Vol. 38, No. 9, pp. 1249-1268, 1989.

[4] M.O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance," *Journal of Association for Computing Machinery*, Vol. 36, No. 2, pp. 335-348, 1989.

[5] I. Chung, "Application of the Special Latin Squares to the Parallel Routing Algorithm on Hypercube," *Journal of Korea Information Science Society*, Vol. 19, No. 5, pp. 569-578, 1992.

[6] Y. Shih and S. Kao, "One-to-one Disjoint Path Covers on K-ary N-cubes," *Theoretical Computer Science*, Vol. 412, No. 35, pp. 4513-4530, 2011.

[7] K. Day and A. Tripathi, "Comparative Study of Topological Properties of Hypercubes and Star Graphs," *IEEE Transactions on Parallel Distributed Systems*, Vol. 8, No. 1, pp. 1196-1202, 1994.

[8] S. Gao, B. Novick, and K. Qiu, "From Hall Matching Theorem to Optimal Routing on Hypercubes," *Journal of Combinatorial Theory, Series B*, Vol. 74, Issue 2, pp. 291-301, 1998.

[9] C. Lai, "Two Conditions for Reducing the Maximal Length of Node-Disjoint Paths in Hypercubes," *Theoretical Computer Science*, Vol. 418, pp. 82-91, 2012.

[10] Q. Gu and S. Peng, "Cluster Fault-Tolerant Routing in Star Graph," *Networks*, Vol. 35, No. 1, pp. 83-90, 2000.

[11] S. Madhavapeddy and I. H. Sudborough, "A Topogical Property of Hypercubes: Node

Disjoint Paths," *Proceedings of the Second IEEE Symposium on Parallel Distributed Processing*, pp. 532- 539, 1990.

[12] J. Zhou, Z. Wu, S. Yang, and K. Yuan, "Symmetric Property and Reliability of Balanced Hypercube," *IEEE Transactions on Computers*, Vol. 64, No. 3, pp. 876-881, 2015.

[13] Y. Cho and I. Chung, "A Parallel Routing Algorithm on Circulant Networks Employing the Hamiltonian Circuit Latin Square," *Information Sciences*, Vol. 176, No. 21, pp. 3132-3142, 2006.

[14] D. Choi, O. Lee, and I. Chung, "A Parallel Routing Algorithm on Recursive Cube of Rings Networks Employing Hamiltonian Circuit Latin Square," *Information Sciences*, Vol. 178, No. 6, pp. 1533-1541, 2008.

Dongmin Choi

He received his B.E. degree from the Kyunghee University in 2003 and M.S. and Ph.D. degrees in computer Science from Chosun University in 2007 and 2011, respectively. He is working as a researcher in Computer Science at Chosun University. His research interests are in information security, sensor network systems, mobile ad-hoc systems, smart grid home network systems and internet ethics.

Ilyong Chung

He received the B.E. degree from Hanyang University, Seoul, Korea, in 1983 and the M.S. and Ph.D. degrees in Computer Science from City University of New York, in 1987 and 1991, respectively. From 1991 to 1994, he was a senior technical staff of Electronic and Telecommunication Research Institute (ETRI), Dajeon, Korea. Since 1994, he has been a Professor in Department of Computer Science, Gwangju, Korea. His research interests are in computer networking, security systems and coding theory.