# A Nearly Optimal One-to-Many Routing Algorithm on k-ary n-cube Networks

Dongmin Choi, Ilyong Chung

**Abstract**

The k-ary n-cube $Q_n^k$ is widely used in the design and implementation of parallel and distributed processing architectures. It consists of $k^n$ identical nodes, each node having degree $2n$ is connected through bidirectional, point-to-point communication channels to different neighbors. On $Q_n^k$ we would like to transmit $2n$ packets from a source node to $2n$ destination nodes simultaneously along paths on this network, the $i^{th}$ packet will be transmitted along the $i^{th}$ path, where $0 \leq i \leq 2n-1$. In order for all packets to arrive at a destination node quickly and securely, we present an O($n^3$) routing algorithm on $Q_n^k$ for generating a set of one-to-many node-disjoint and nearly shortest paths, where each path is either shortest or nearly shortest and the total length of these paths is nearly minimum since the path is mainly determined by employing the Hungarian method.

Keywords : k-ary n-cube network | node-disjoint paths | parallel routing algorithm | Hungarian method

## I. INTRODUCTION

The k-ary n-cube $Q_n^k$ [1,7,8,9] consists of $k^n$ identical processors (nodes). Each processor, provided with its own sizable local memory, is connected through bidirectional, point-to-point communication channels to $2n$ different neighbors. Due to these properties, $Q_n^k$ can be widely used in the design and implementation of parallel and distributed processing architectures.

In this paper, nearly optimal one-to-many parallel routing algorithm on the k-ary n-cubes is designed. $2n$ packets are transmitted from a source node to $2n$ destination nodes simultaneously along paths on $Q_n^k$, the $i^{th}$ packet will be transmitted along the $i^{th}$ path $(0 \leq i < 2n)$. In order for all packets to arrive at their destination nodes quickly and securely, a set of $2n$ node-disjoint paths with nearly minimal total length should be constructed. To accomplish this, the operations of nodes presented in the Cayley Graph[6], the MGNDP (Matrix for generating node-disjoint paths) and the Hungarian method are emplyed [2,5].

This paper is organized as follows. Section II describes the design of the shortest path on $Q_n^k$. Section III is the central contribution of this paper. This section focuses on Hungarian method and its application is to a parallel routing algorithm on $Q_n^k$. This paper concludes with Section IV.

## II. DESIGN OF THE SHORTEST PATH

The k-ary n-cube $Q_n^k (k \geq 2$ and $n \geq 1)$ is a graph consisting of $k^n$ nodes, each of which has the form $u = (u_{n-1}u_{n-2}\ldots u_0)$ or $v = (v_{n-1}v_{n-2}\ldots v_0)$ and $Z_n$ is defined as the set of nonnegative integers less than n where $v_i, u_i \in Z_k$ for $i \in Z_n$. Two nodes $u = (u_{n-1}u_{n-2}\ldots u_0)$ and $v = (v_{n-1}v_{n-2}\ldots v_0)$ on $Q_n^k$ are adjacent if and only if there exists an integer $j$, $j \in Z_n$, such that $u_j = v_j \pm 1$ (mod k) and $u_i = v_i$, for every $i \in \{0,1,\ldots,n-1\}\backslash\{j\}$. Such a link $(u,v)$ is called a $j$-dimensional link

Membe, Chosun University
* This study was supported by research fund from Chosun University, 2016

Manuscript : 2018 03 06
Revised: 2018. 03. 31
Confirmation of Publication: 2018. 05. 21
Corresponding Author :Ilyong Chung, e-mail : iyc@chosun.ac.kr

**Definition 1.** * The routing function of $Q_n^k$ on the $i^{th}$ dimension is defined as follows:
$R_{g_{\pm i}}(u_i) = u_i \pm 1$ (mod k), $0 \le i \le n-1$, where $g_{+i}$ is the positive operation on the $i^{th}$ dimension.

In this paper, we use $g_{+i}$ and $g_i$ interchangeably. Employing positive operations – $g_0$ and $g_2$ on $Q_3^5$, (010) is connected to (011) and (110), respectively. (010) is connected to (014) and (410) employing negative operations – $g_{-0}$ and $g_{-2}$, respectively.

**Definition 2.** Let $T(A,S)$ be the path of data starting from node $A$ to the destination node, where $S$ is a sequence of operations, via which data can reach at a destination node. $T(A,S)$ is determined by the order of operations in $S$.

Let node $A$ and sequence $S$ be (233) and $< g_2, g_2, g_{-0}, g_{-0}, g_{-1}, g_{-1}, g_{-1}, g_{-1} >$ on $Q_3^5$, respectively. Applying the routing function described in Definition 1, $T(A,S)$ is $(233) \to (333) \to (433) \to (432) \to (431) \to (421) \to (411) \to (401) \to (441)$. In this paper, the order of operations defined as follows – $g_i$ is higher than $g_j$, if $i > j$. In this example, the order of operations is from the first operation to the lowest. The size of this sequence must be minimized since a routing distance is equal to the size of a sequence and $S$ is minimized to $< g_2, g_2, g_1, g_{-0}, g_{-0} >$ because $< g_{-1}, g_{-1}, g_{-1}, g_{-1} > = < g_1 >$ and the routing distance is 5. To obtain the minimized routing distance between two nodes, the relative address is computed below.

**Definition 3.** The relative address $r$ of nodes $u$ and $v$ on $Q_n^k$ is denoted by $v_i - u_i = (r_{n-1} r_{n-2} \cdots r_0)$, where if $(r_i > k/2)$ then $r_i = k - r_i$, else if $(r_i < -k/2)$ then $r_i = k + r_i$ (mod k).

Let $u$ and $v$ on $Q_3^5$ be (234) and (410). The relative address $r$ of two nodes is (2-21), which can be described as a sequence S of operations $< g_2, g_2, g_{-1}, g_{-1}, g_0 >$.

## III. A ONE-TO-MANY PARALLEL ROUTING ALOGITHM ON $Q_n^k$

In this section, we would like to construct a set of $2n$ node-disjoint and nearly shortest paths on $Q_n^k$ in order to transmit $2n$ packets securely and quickly. First, these packets residing at a starting node are sent to its $2n$ neighboring nodes by employing $2n$ different operations. Then these packets are transmitted to $2n$ destination nodes along $2n$ node-disjoint paths, where the $i^{th}$ packet is transmitted to the $i^{th}$ destination node.
The MGNDP(Matrix for generating node-disjoint paths) [4] is applied to find a set of node-disjoint paths on hypercube networks. The next definition describes the MGNDP.

**Definition 4:** Call the matrix $M$ as the MGNDP(Matrix for generating node-disjoint paths). No two entries in this matrix thus satisfy the following conditions.
$M = (A_{i,j}), A_{i,j} \in \{g_{\pm i} \cup s\}^+, 0 \le i \le 2n-1, 0 \le j \le [\frac{k}{2}] * n - 1, 0 \le k \le n-1$. $s$ means "stay at the current node".
(1) $|A_{i,j}| = j + 1$
(2) $A_{i,j+1} = A_{i,j} \cup \{g_k \cup s\}, 0 \le |k| \le n-1$
(3) $A_{i,j} \ne A_{k,j}$, if $i \ne k$

In order to design a nearly optimal one-to-many parallel routing algorithm the Hungarian method is applied, which is a combinatorial optimization algorithm solving the assignment problem in polynomial time $O(n^3)$. In this paper, this method models an assignment problem as an $(2n \times 2n)$ communication cost matrix, each element of which represents the cost of transmitting a packet from one node to another node. Here, communication cost means the distance between two nodes on $Q_n^k$.

We now transmit six packets from node (010) to nodes (233), (234), (223), (133), (243) and (333) on $Q_3^5$. First, these packets are sent to node (010)'s 6 neighboring nodes by employing six distinct operations – $g_2, g_1, g_0, g_{-0}, g_{-1}, g_{-2}$ and then reach at nodes (110), (020), (011), (014), (000) and (410). To find a set of six node-disjoint and nearly shortest paths from these intermediate nodes to six destination nodes, the assignment problem will be employed. A $(6 \times 6)$ communication cost matrix $M^0$ is constructed by computing a shortest distance from a neighboring node to a destination node, where $M^k = (m_{ij}^k)$ . By employing the Hungarian method to $M^0$, $M^1$ is generated.

$$\mathbf{M^0} = \begin{bmatrix} 5 & 4 & 4 & 4 & 5 & 6 \\ 5 & 4 & 4 & 4 & 6 & 5 \\ 6 & 6 & 5 & 5 & 6 & 6 \\ 5 & 4 & 4 & 4 & 5 & 5 \\ 6 & 5 & 6 & 5 & 5 & 6 \\ 6 & 5 & 5 & 6 & 6 & 5 \end{bmatrix}$$

$$\mathbf{M^1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

For example, $m_{43}^0$ = 4 means that the distance of the path from the $4^{th}$ neighboring node to the $3^{rd}$ destination node is 4. The relative address of these nodes is $(21-1)$ and the sequence of operations is $< g_2, g_2, g_1, g_{-0} >$. So, the path is "node (014) → node (114) → node (214) → node (224) → node (223)". From $M^1$ we select the zeroes from column numbers 1,2,3,4,5 and 6, respectively. This means that packets are transmitted from nodes $(110), (020), (011), (014), (000)$ and $(410)$ to nodes $(233), (234), (223), (133), (243)$ and $(333)$, respectively. To reach from node (010) to the destination nodes, operations should be performed. The first and remaining operations for path 0, path 1, path 2, path 3, path 4 and path 5 are $< g_2 > + < g_2, g_1, g_1, g_{-0}, g_{-0} >$,
$< g_1 > + < g_1, g_{-0}, g_2, g_2 >$,
$< g_0 > + < g_0, g_0, g_2, g_2, g_1 >$,
$< g_{-0} > + < g_{-0}, g_2, g_1, g_1 >$,
$< g_{-1} > + < g_{-1}, g_2, g_2, g_{-0}, g_{-0} >$, and
$< g_{-2} > + < g_{-2}, g_1, g_1, g_{-0}, g_{-0} >$, respectively. If the first operation is $g_i$, then the next operation is from $g_i$ to the lowest and then from the highest to $g_i$.

To be a set of disjoint paths, the two exceptional cases should be solved. For the first case, any two sequences of operations satisfy the conditions described in Definition 4. Depending on which operation is performed at last, the collision of two paths may happen. So, the last operation in each sequence should be chosen carefully. These operations not to be selected for Path 0, Path 1, Path 2, Path 3, Path 4 and Path 5 are $\{g_2, g_1, g_{-0}, g_{-1}, g_{-2}\}$, $\{g_0\}$, $\{g_{-1}\}$, $\{g_2\}$, $\{g_1\}$ and $\{g_{-2}\}$, respectively. If $g_{-0}$ is chosen for Path 0, then Path 0 and Path 1 collide at node (234). However, any operation in the sequence for Path 0 can not be selected as the last operation. To be the suitable sequence for path 0, $< g_{-0}, g_{-0} >$ is changed to $< g_0, g_0, g_0 >$ since $g_0 \notin \{g_2, g_1, g_{-0}, g_{-1}, g_{-2}\}$ and $T(A, < g_{-0}, g_{-0} >)$ is the same as $T(A, < g_0, g_0, g_0 >)$ on $Q_3^5$. So, the sequence for Path 0 should be $< g_2 > + < g_2, g_1, g_1, g_0, g_0, g_0 >$. For the second case, Path 2 and Path 3 collide at node (013) since $T(A, < g_0, g_0, g_0 >) = T(A, < g_{-0}, g_{-0} >)$ on $Q_3^5$. To avoid this collision one $g_0$ moves to the last position in the second sequence. The sequence for Path 2 should be $< g_0 > + < g_0, g_2, g_2, g_1, g_0 >$ . So, the MGNDP M is constructed as follows.

$$\begin{bmatrix} g_2 & g_2, g_2 & \cdots & g_2, g_2, g_1, g_1, g_0, g_0 & g_2, g_2, g_1, g_1, g_0, g_0, g_0 \\ g_1 & g_1, g_1 & \cdots & g_1, g_1, g_{-0}, g_2, g_2, s & g_1, g_1, g_{-0}, g_2, g_2, s, s \\ g_0 & g_0, g_0 & \cdots & g_0, g_0, g_2, g_2, g_1, g_0 & g_0, g_0, g_2, g_2, g_1, g_0, s \\ g_{-0} & g_{-0}, g_{-0} & \cdots & g_{-0}, g_{-0}, g_2, g_1, g_1, s & g_{-0}, g_{-0}, g_2, g_1, g_1, s, s \\ g_{-1} & g_{-1}, g_{-1} & \cdots & g_{-1}, g_{-1}, g_2, g_2, g_{-0}, g_{-0} & g_{-1}, g_{-1}, g_2, g_2, g_{-0}, g_{-0}, s \\ g_{-2} & g_{-2}, g_{-2} & \cdots & g_{-2}, g_{-2}, g_1, g_1, g_{-0}, g_{-0} & g_{-2}, g_{-2}, g_1, g_1, g_{-0}, g_{-0}, s \end{bmatrix}$$

Given $A_{i7} (0 \le i \le 6)$ of operations, a set of node-disjoint and nearly shortest paths is generated as below.

Path 0: node (010) → node (110) → node (210) → node (220) → node (230) → node (231) → node (232) → node (233).

Path 1: node (010) → node (020) → node (030) → node (034) → node (134) → node (234).

Path 2: node (010) → node (011) → node (012) → node (112) → node (212) → node (222) → node (223).

Path 3: node (010) → node (014) → node (013) → node (113) → node (123) → node (133).

Path 4: node (010) → node (000) → node (040) → node (140) → node (240) → node (244) → node (243).

Path 5: node (010) → node (410) → node (310) → node (320) → node (330) → node (334) → node (333).

The process to find a set of node-disjoint and nearly shortest paths is described above. We now propose a one-to-many parallel routing algorithm on $Q_n^k$. In this paper, we will use the term "distance" between two nodes to refer to the number of routing steps (also called the hopcount) needed to send a message from one node to another.

$OTM-KNC-Routing$

$A$ ← a starting node

$N_i$ ← the $i^{th}$ neighboring node of node $A$ $(0 \leq i < 2n)$

$D_i$ ← the $i^{th}$ destination node $(0 \leq i < 2n)$

begin

(1) $2n$ packets are sent from $A$ to their $2n$ neighboring nodes by performing $2n$ distinct operations

(2) A $(2n \times 2n)$ communication cost matrix $M$ can be constructed, where $M = (m_{ij})$, $m_{ij}$ is the shortest distance of the path required for transmitting the $i^{th}$ packet from the $i^{th}$ neighboring node to the $j^{th}$ destination node

(3) In order to design nearly shortest paths, the Hungarian method is applied to the communication cost matrix. From the cost matrix computed, we obtain the length of Path $i$ between $N_i$ to a destination node, which is the number of operations in the sequence, the order of which is from $g_i$ to the lowest and then from the highest to $g_i$, where $T(A, < g_i >)$ is the path from $A$ to $N_i$.

(4) The two exceptional cases should be solved.

(4-1) Find operations not to be selected as the last operation for each path. In order for each sequence to satisfy the conditions described in Definition 4, change the last operation in the sequence, if needed.

(4-2) If $[k/2]$ $g_i$s in a sequence exist, then one $g_i$ moves to last position in the sequence.

(5) $2n$ packets are transmitted from a starting node to $2n$ destination nodes via the corresponding neighboring nodes by performing $2n$ sequences of operations

end

Execution of $OMT-KNC-Routing$ is thus fairly straightforward. The time involved in performing Steps (1), (4-2) and (5) is small compared to the remaining steps. The first, the second, and the sixth steps of this algorithm do not, therefore, contribute to an objectionable overhead.

**Theorem 1.** $OMT-KNC-Routing$ can be performed in $O(n^3)$.

**Proof.** There are three important steps for determining the time complexity requisite for the Algorithm. Step (2) constructs a communication cost matrix, which requires $O(n^3)$. Step (3) executes the Hungarian method, which can be computed in $O(n^3)$. Step (4-1) finds operations not to be selected as the last operation for each path. It needs $O(n^3)$. Therefore, the time complexity of the Algorithm is $O(n^3)$.

The paper' s objective is to design a set of $2n$ node-disjoint paths from a single source node to $2n$ destination nodes. The major topological characteristics of $Q_n^k$ are considered and the requisite properties of $2n$ paths obtained from the Algorithm are proven below.

**Theorem 2.** The $2n$ transmission paths produced by $OTM-KNC-Routing$ are node-disjoint and nearly shortest.

**Proof.** Let $S_i$ and $S_j$ be two sequences of operations for sending two packets from a starting node A to two destination nodes, where $S_i = < g_{i1}, g_{i2}, \ldots, g_{it}, g_{i(t+1)}, \ldots, g_{ix} >, S_j = < g_{j1}, g_{j2}, \ldots, g_{jt}, g_{j(t+1)}, \ldots, g_{jy} >, i1 > j1$. Let $SR_i$ and $SR_j$ be two sequences of operations not to be selected as the last operation, where $SR_i = < g_{ri1}, g_{ri2}, \ldots, g_{rip} >, SR_j = < g_{rj1}, g_{rj2}, \ldots, g_{rjq} >$, respectively. Each sequence is ordered from the first operation to the lowest and then from the highest to the first. Suppose that two packets arrive at the same node. In order for this case to occur, we should have the equality that $T(A, S_{it}) = T(A, S_{jw})$, where $A$ is a starting node, $S_{it}$ and $S_{jw}$ are the subsequences of $S_i$ and $S_j$, $S_{it} = < g_{i1}, g_{i2}, \ldots, g_{i(t-1)}, g_{it} >, S_{jw} = < g_{j1}, g_{j2}, \ldots, g_{j(w-1)}, g_{jw} >, t \leq x, w \leq y$. However, these sequences do not appear. To prove it, we consider three cases.

Case 1: If $m$ $g_i s$ and $(k-m)$ $g_{-i} s$ appear in the beginning part of $S_{it}$ and $S_{jw}$, respectively, then $T(A, S_{im}) = T(A, S_{j(k-m)})$. However, this case does not happen. According to Algorithm (4-2), one $g_i$ moves to the last position of the corresponding sequence.

Case 2: Suppose that $|S_i| = |S_j|$ . Then $T(A, S_i) \neq T(A, S_j)$ since two destination nodes are different. These paths must be node-disjoint because an operation in each sequence is performed in the same way - from the first operation to the lowest and then from the highest to the first operation.

Case 3: Suppose that $S_{it} = S_{jw}(t \leq x, w \leq y)$. In order for this case to happen, $S_{jw}$ should be $< g_{j1}, g_{j2}, \ldots, g_{it}, \ldots, g_{jw} >, jw \leq i1$. However, this case does not occur. In case of $|S_i| < |S_j|$, $g_{jw}$ should be relocated or replaced.

Swaps $g_{j(y-m)}$ in $S_j$, $0 \leq m$, which deserve to be the last position (see Algorithm (4-1)).

The total length of these paths is minimal at most cases since the number of operations is obtained by employing the Hungarian method. However, depending on selecting which elements in the modified cost matrix(see $M^1$ in Section 3), two arbitrary paths may cross at the same node. It causes these paths not to be node-disjoint. So, one of these paths should be detoured to avoid this occurrence, which makes the total length of them longer. Therefore, the Algorithm constructs a set of $2n$ node-disjoint and nearly shortest paths.

## IV. CONCLUSION

In this paper, an algorithm that generates a set of $2n$ nearly shortest and node-disjoint paths on $Q_n^k$ from a source node to $2n$ destination nodes employing the Hungarian method is presented. Three important steps determine the time complexity requisite for the Algorithm. The first constructs a communication cost matrix, which requires $O(n^3)$. The second is to execute the Hungarian method, which can be computed in $O(n^3)$. The final designs a set of $2n$ node-disjoint paths, which requires $O(n^3)$. Therefore, an $O(n^3)$ parallel routing algorithm is created for constructing a set of $2n$ node-disjoint and nearly shortest paths. For further research, this algorithm will be extended to design a set of one-to-many node-disjoint paths on other networks and on fault-tolerant $Q_n^k$.

## REFERENCES


[1] Yali Lv, Jianxi Fan, D. Frank Hsu, Cheng-Kuan, " Structure connectivity and substructure connectivity of k-ary n-cube networks," Information Sciences, vol. 433, pp. 115-124, 2018.

[2] Y. Chen, S. Promparmote, F. Maire, "MDSM: Microarray database schema matching using the Hungarian method," Information Sciences, vol. 176, pp. 2771-2790, 2006.

[3] D. Choi, O. Lee, I. Chung, "A parallel routing algorithm on recursive cube of rings networks employing Hamiltonian circuit Latin square," Information Sciences, vol. 178, pp. 1533-154, 2008.1

[4] I. Chung, "Application of the Special Latin Squares to the Parallel Routing Algorithm on Hypercube," J. of Korea Information Sciences Society, vol. 19, pp. 569-578, 1992.

[5] H. Kuhn, "The Hungarian Method for the Assignment Problem," Naval Res. Logist. Quart., vol. 2, pp. 83-97, 1995.

[6] H.S. Stone, Discrete Mathematical Structures and Their Applications, SRA, Chicago, 1973.

[7] Hee Kyoung Seo, "Development of component architecture to support IoT management," Smart Media Journal, Vol. 6, No. 2, pp.42-49, 2017

[8] Shiying Wang, Weiping Han, "The g-good-neighbor conditional diagnosability of n-dimensional hypercubes under the MM□ model," Information Processing Letters, vol. 116, pp. 574-577, 2016.

[9] S. Zhang, X. Zhang, "Fault-free Hamiltonian cycles passing through prescribed edges in k-ary n-cubes with faulty edges," IEEE Trans. Parallel and Distributed Systems, vol. 26, pp. 434-443, 2015.

## Authors

Il-Yong 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, Chosun University, Gwangju, Korea. His research interests are in computer networking, security systems and Paralle Algorithm, Mobile Ad-hoc Network.

Dong-Min 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. Since 2014, he has been a Professor in College of General Education, Chosun University, Gwangju, Korea. His research interests are in information security, sensor network systems, mobile ad-hoc systems, smart grid home network systems, mobile sensor applications, and internet ethics