Z-cube 네트워크의 직경, 고장직경과 정점간 평균거리

권 경 희† · 이 계 성†

요 약

정점차수가 하이퍼큐브의 3/4 이면서도 하이퍼큐브의 위상적 특성을 대부분 보유하고 있는 새로운 네트워크인 Z-cube가 최근 제안된 바 있는데, 이는 하드웨어로의 구현이라는 측면에서는 하이퍼큐브의 좋은 대체 안이 될 수 있다. 본 논문에서는 Z-cube에서의 정점들 간의 통신 성능을 평가하기 위해 Z-cube의 직경과 두 정점들 간의 평균거리, 그리고 고장직경을 산출하였다. 이를 위해 Z-cube의 재귀적 구조와 두 정점간의 최단경로, 평균거리에 대한 점화관계(Recurrence Relation)가 유도되었으며 정점 무중복 경로 (Node Disjoint Path)가 소개되었다.

일반적으로 네트워크의 정점차수가 감소되면 직경과 두 정점간 평균거리가 증가하여 통신성능도 그만큼 저하되리라 예상되지만, 본 논문은 Z-cube와 하이퍼큐브의 직경이 같고 Z-cube에서의 두 정점들 간의 평균거리가 하이퍼큐브의 1.125배에 지나지 않으며 고장직경은 차수에 따라 대략 하이퍼큐브의 1.4배 내지 1.7 배인 것을 보여 주고 있다.

Diameter, Fault Diameter and Average Distance between Two Nodes in Z-cube Network

Kyung-Hee Kwon† · Gye-Sung Lee†

ABSTRACT

Recently, a new hypercube-like interconnection network, the Z-cube, was proposed. The Z-cube retains most good topological properties, however, its node degree is 3/4 of hypercube's one. Considering hardware implementations, the Z-cube is a good alternative to the hypercube. In this paper, we obtained the diameter, fault diameter and the average distance between two nodes to evaluate the communication performance of the Z-cube. The recursive structure, the shortest path between two nodes in Z-cube and recurrence relation on the average distance were deduced, and node disjoint path was introduced.

Although it is generally expected that the communication performance in an interconnection network with reduced node degree falls as much as that, this paper shows that the Z-cube's diameter is the same as the hypercube's one and the average distance between two nodes in Z-cube is about 1.125 times the average distance between two nodes in the hypercube and the fault diameter of Z-cube ranges approximately from 1.4 times to 1.7 times the fault diameter of the hypercube.

※ 이 논문은 단국대학교 대학연구지원비의 지원으로 연구되었음.
† 정 회 원 : 단국대학교 전자계산학과 교수
논문접수 : 1998년 9월 16일, 심사완료 : 1998년 11월 4일

1. 서 론

병렬처리 컴퓨터의 설계에 있어서 프로세서들 간의 네트워크 구성 방법은 가장 중요한 요소 중의 하나로 인식되어지고 있다. 일반적으로 작은 정점 차수, 짧은 직경, 두 정점 사이의 많은 수의 disjoint path, 단순한 라우팅 요소, 하드웨어로의 구현이 쉬운 네트워크가 병렬처리 컴퓨터의 구조로 적합하다고 할 수 있다. 이러한 요소들은 서로 다른 요소들에 영향을 준다. 예를 들어 정점차수가 커지면 정점간 정점 무중복 경로(Node Disjoint Path)의 수는 더 많아지게 되며 네트워크의 직경은 짧아지게 된다. 그러나 정점차수가 큰 네트워크는 하드웨어로의 구현이 어렵다. 현재의 전자공학적 기술에 의하면 네트워크 설계에서 가장 중요한 요소의 하나는 VLSI로의 구현 가능성이며 소결합(Loosely coupled) 컴퓨터 시스템에 서는 연결선의 배열이 복잡하지 않은 것이 바람직하다 하겠다.

하이퍼큐브 네트워크는 이러한 요소들이 이상적으로 결합되어 있는 네트워크로 잘 알려져 있다. 즉 짧은 직경, 구조의 규칙성, 대용량의 bandwidth, 그리고 많은 다른 네트워크들의 내포(Embedding)등을 들 수 있으며 하이퍼큐브의 이러한 장점을 살려 상용화되어 나온 것으로는 Connection Machine, nCube 등을 들 수 있다. 이러한 하이퍼큐브의 성능을 개선한 네트워크가 80년대 후반과 90년대 초반에 소개되었는데 이러한 것들에는 X-hypercube, Multiply-Twisted Hypercube [1, 2] 등이 있다. 이들은 하이퍼큐브와 똑같은 구조적인 복잡성 즉 같은 수의 정점에 같은 수의 연결선을 가지면서 직경을 거의 반으로 줄인 것이다.

이렇게 가장 이상적인 네트워크로 평가받는 하이퍼큐브와 이의 성능을 개선한 수정된 하이퍼큐브들도 몇 가지 단점을 갖고 있는데 그 중의 하나가 정점들을 잇는 연결선의 수가 너무 많다는 것이다. 다중프로세서의 VLSI 설계에서는, 프로세서들을 연결하는 배선의 면적이 프로세서들의 면적을 지배하며, 더욱이 각 프로세서의 연결도가 복잡해지면 프로세서 설계가 복잡해진다. 이런 관점에서 볼 때, 하이퍼큐브는 VLSI 구현에는 적합하다고 보기 어렵다.

최근 Z-Cube 라 이름한 새로운 형태의 네트워크가 제안되었는데[3], 이는 하이퍼큐브와 같은 수의 정점을 갖고 있으나 연결선의 수는 하이퍼큐브의 3/4에 지나지 않는다. 또한 Z-cube는 하이퍼큐브가 갖는 대부분의 장점을 그대로 유지하고 있어[3] VLSI 구현의 관점에서 본다면 Z-Cube는 하이퍼큐브의 대체 안이 될 수 있다고 하겠다. 그러나 정점들 간의 연결선의 수가 줄어든 만큼 프로세서들 간의 통신 성능은 Z-Cube 가 하이퍼큐브만큼 우수하지 못하다는 것을 예견 할 수 있다.

따라서 본 논문에서는 Z-Cube의 통신성능에 대해 고찰해 보고자 한다. 다중프로세서에서 프로세서들간의 통신성능을 평가하는 방법이 아직 확립되어 있지 않다. 이는 프로세서들 간의 통신 방식이 응용분야에 따라 사용되어지는 알고리즘에 따라 달라지기 때문이다. 그러나 직경과 정점들 간의 평균거리, 그리고 고장직경은 네트워크의 통신성능을 평가하는 객관적인 기준이 될 수 있다. 즉 직경은 가장 멀리 떨어진 프로세서들 간의 통신 지연을, 그리고 정점들 간의 평균거리는 평균적인 통신 지연을 정량적으로 평가하는 척도가 될 수 있으며 고장직경은 네트워크의 정점에 고장이 발생했을 때의 가장 멀리 떨어진 프로세서들 간의 통신지연에 관한 척도가 된다.

[3]에서는 Z-cube의 몇몇 위상적 특성이 검토되고 각종 통신알고리즘이 제시되었으나 직경과 평균거리에 관해서는 간단한 언급만 있을 뿐 구체적인 설명과 이에 대한 증명이 없으며 고장지름에 대해서는 아직 알려진 바 없다. 본 논문에서는 Z-Cube의 직경이 하이퍼큐브의 직경과 같고 정점들간의 평균거리는 Z-Cube 가 하이퍼큐브의 1.125 배가 되며 고장직경은 차수에 따라 대략 하이퍼큐브의 1.4배 내지 1.7배라는 것을 보여주며 이를 위한 위상적 특성을 분석하고 있다.

2. Z-cube 의 구조

Q_n^Z로 표시되는 n 차원 Z-cube는 2^n개의 정점으로 구성되는 그래프로써, 이때 n은 4의 배수이다. 각 정점은 0에서 2^n-1 사이의 n개의 자릿수로 된 이진수로 표시되어지며 최소유효비트(least significant bit)의 색인 값은 1로 한다. Z-Cube로 연결된 multiprocessor system에서, 각 프로세서 P_b는 이진수 b로 표시된 정

점으로 나타내어지고 b는 P_b의 주소라 하자. 프로세서들 간의 통신을 위한 링크(link)는 그래프의 연결선에 해당된다. Z Cube는 정점들을 이진수로 표시해 주고 이들 간의 연결 방법을 나타냄으로써 나음과 같이 정의될 수 있다.

정의 1 : n이 4의 배수일 때, n 차원 Z cube Q_n^Z은 2^n개의 정점으로 이루어지며 두 정점 $u = u_n u_{n-1} \ldots u_1$와 $v = v_n v_{n-1} \ldots v_1$는, $1 \le m \le n/4$인 m과 $k > 4m$이고 $k < 4(m-1)$인 k에 대해서 $u_k = v_k$일 때, 다음의 다섯 가지 조건 중 한가지가 만족하면 서로 연결선에 의해 연결된다 :

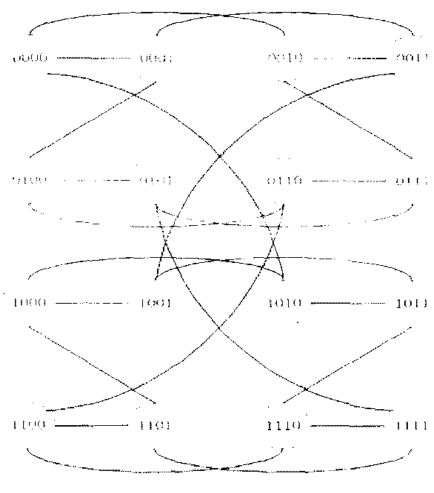
(1) $u_{4m}u_{4m-1} = v_{4m}v_{4m-1}$,이고 $u_{4m-2}u_{4m-3}$과 $v_{4m-2}v_{4m-3}$ 이 한 bit 만 다름;

(2) $u_{4m}u_{4m-2} = v_{4m}v_{4m-2} \in \{00,11\}$, 이고 $u_{4m-1}u_{4m-3} = \overline{v_{4m-1}v_{4m-3}} \in \{01,10\}$;

(3) $u_{4m}u_{4m-2} = v_{4m}v_{4m-2} \in \{01,10\}$, 이고 $u_{4m-1}u_{4m-3} = \overline{v_{4m-1}v_{4m-3}} \in \{00,11\}$;

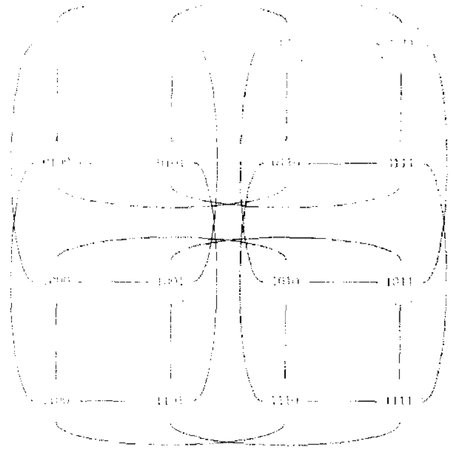
(4) $u_{4m}u_{4m-2} = \overline{v_{4m}v_{4m-2}} \in \{01,10\}$, 이고 $u_{4m-1}u_{4m-3} = v_{4m-1}v_{4m-3} \in \{01,10\}$;

(5) $u_{4m}u_{4m-2} = \overline{v_{4m}v_{4m-2}} \in \{00,11\}$, 이고 $u_{4m-1}u_{4m-3} = v_{4m-1}v_{4m-3} \in \{00,11\}$.



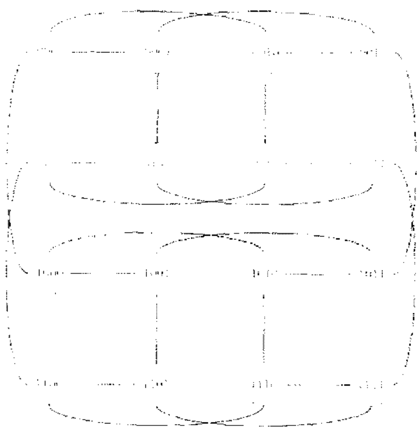
(그림 1) 정의 1에 의한 Q_4^Z

(Fig. 1) Q_4^Z by Definition 1



(그림 2) 4 차원 하이퍼큐브 Q_4

(Fig. 2) 4 dimensional hypercube Q_4



(그림 3) 정의 2에 의한 Q_4^Z

(Fig. 3) Q_4^Z by Definition 2

(그림 1)은 Q_4^Z을 나타내며 (그림 2)는 4차원 하이퍼큐브인 Q_4를 나타낸다. 정의 1에 의해 (그림 1)의 각 정점을 Q_4^Z로 대치하고 연결선은 각각의 상응하는 정점으로 연결하면, Q_8^Z를 구축할 수 있다. 이는 어떤 차원의 Z Cube Q_{4m}^Z도 Q_4^Z의 각 정점을 $Q_{4(m-1)}^Z$로 대치하고 연결선을 각각의 상응하는 정점과 연결하면 재귀적으로 구축할 수 있다는 것이다. Q_{4m}^Z에 있는 16개의 $Q_{4(m-1)}^Z$을 Q_{4m}^Z의 4(m-1) 차원 서브큐브(Subcube)라 하고 $\delta_{4(m-1)}$으로 표시하기로 하자. 물론, 이들 16개의 서브큐브들은 서로 동형(Isomorphic)이다. 기술의 편의상 간단한 이 사실을 **보조정리**로 다음과 같이 요약하자.

보조정리 1 : n차원 Z cube Q_n^Z은 4 차원 Z Cube Q_4^Z의 각 정점을 δ_{n-4}로, 그리고 각 연결선을 16개의 δ_{n-4}의 각각 일치하는 정점을 연결하는 연결선으로 나타냄으로써 재귀적으로 표시할 수 있다.

(그림 2)에서 보여지는 하이퍼큐브의 4개의 정점으로 된 사이클(Cycle)을 (그림 1)의 Z 모양의 경로로 바꾸면 Z-Cube를 구할 수 있는데 이는 Z-Cube로 명명한 이유가 된다.

Z 모양의 경로 중 대각선으로 연결된 정점의 위치를 바꾸고 이 정점들을 다시 (그림 1)의 순서로 레이블링하면 (그림 3)을 구할 수 있다. (그림 1)과 (그림 3)에 보여진 두 그래프는 정점을 레이블링하는 방법만 다를 뿐이고 서로가 동형(isomorphic)이다. (그림 3)에서는 (그림 1)에서 찾지 못하는 특성을 쉽게 찾을 수 있으므로 (그림 3)에 의해 Z-cube를 다시 정의하면 다음과 같다.

정의 2 : n이 4의 배수일 때, n차원 Z-cube Q_n^Z은 2^n개의 정점으로 이루어지며 두 정점 $u = u_n u_{n-1} \ldots u_1$와 $v = v_n v_{n-1} \ldots v_1$는, $1 \le m \le n/4$ 인 m과 $k > 4m$이고 $k \le 4(m-1)$인 k에 대해서 $u_k = v_k$일 때, 다음의 세 가지 조건 중 한가지가 만족하면 서로 연결선에 의해 연결된다:

(1) $u_{4m}u_{4m-1} = v_{4m}v_{4m-1}$, 이고 $u_{4m-2}u_{4m-3}$ 과 $v_{4m-2}v_{4m-3}$ 이 한 bit 만 다름;

(2) $u_{4m}u_{4m-1} = \overline{v_{4m}}v_{4m-1}$, 이고 $u_{4m-2}u_{4m-3} = v_{4m-2}v_{4m-3} \in \{00,11\}$;

(3) $u_{4m}u_{4m-1} = v_{4m}\overline{v_{4m-1}}$, 이고 $u_{4m-2}u_{4m-3} = v_{4m-2}v_{4m-3} \in \{01,10\}$.

정의 2에 의한 Q_4^Z 인 (그림 3)과 4차원 하이퍼큐브 Q_4인 (그림 2)를 비교해 보면 Z-cube는 하이퍼큐브의 부 그래프임을 알 수 있으며 또한 Z-cube는 하이퍼큐브에 dilation 2와 congestion 2로 내포된다는 것을 쉽게 짐작할 수 있다.[3] 그리고 Z-cube는 재귀적으로 정의되므로 정의 2에 의해 다음의 사실을 알 수 있으며 이를 기술의 편의상 **보조정리**로 요약하면 다음과 같다:

보조정리 2 : $m>0$일 때, 정점들의 부분집합{ $b_{4m}b_{4m-1} \ldots b_1$ | $k > 4m$이고 $k < 4(m-1)$이며 b_k= 이진상수}에 의해 Q_n^Z에서 유도된 부그래프(Subgraph)는 Q_4^Z와 동형이다.

본 논문에서 빈번히 인용할 Z-cube의 중요한 위상적 특성 중의 하나는 Z-cube의 정점 대칭성이다. [3]에서는 주소가 $u_n u_{n-1} \ldots u_1$인 임의의 정점에 새로운 주소 $u'_n u'_{n-1} \ldots u'_1$를 부여해도 다른 모든 정점의 주소를 어떤 알고리즘에 따라 변환시키면 원래 Q_n^Z의 연결성을 그대로 유지한다는 성질을 나타냄으로써 Z-cube가 정점대칭 네트워크이라는 것을 보여주고 있다. 정점대칭 네트워크에서는 어떠한 정점에서 네트워크가 관찰되더라도 그 네트워크는 원래 주어진 정의에서와 같은 연결성을 유지하고 있기 때문에 위상 분석이나 각종 알고리즘 설계가 매우 용이해 진다. 즉 Q_4^Z 에서 정점 0000에서 관찰된 특성은 어떤 임의의 정점에서 관찰되어도 같은 특성을 지니고 있다. 따라서 본 논문에서 산출될 직경, 평균거리 그리고 고장직경은 먼저 Q_4^Z 의 정점 0000에서부터 관찰되어지고 그리고 **보조정리 1**과 **보조정리 2**에 의해 Q_n^Z 로 일반화된다.

3. Z-cube 의 직경

두 정점간의 거리는 두 정점간의 최단경로의 길이를 말하며, 직경은 그래프 내의 모든 정점들 간의 거리 중 가장 큰 것을 말한다. 작은 직경을 가진 네트워크에서는 가장 멀리 떨어진 프로세서들 간의 통신 지연을 상대적으로 줄일 수 있으며 따라서 전체적인 통신효율을 높일 수 있다. 일반적으로 정점의 차수가 줄어들면 직경은 커지게 되나 Z-cube Q_n^Z의 각 정점들은 하이퍼큐브의 정점보다 차수가 낮은데도 불구하고 직경은 같다.

Q_n^Z에서 두 정점간의 최단 경로를 구하는 문제는 Q_4^Z에서 정점 0000에서 다른 정점까지의 최단경로를 구하는 문제로 간략화할 수 있다. 이는 Q_n^Z의 재귀적인 특성과 정점 대칭성(Vertex Symmetry) 때문이다.[3]

Q_4^Z에서의 정점 0000에서 다른 정점들 간의 최단경로들 중의 하나는 <표 1>에 나타나 있다. Z-cube의 구조와 <표 1>을 이용하여 Z-cube의 직경을 구해 보기로 하자.

<표 1> Q_4^Z 에서의 최단 경로
<Table 1> The Shortest Path in Q_4^Z

Destination	First Move	Second Move	Third Move	Fourth Move
0001	0001			
0010	0010			
0011	0001	0011		
0100	0001	0101	0100	
0101	0001	0101		
0110	0010	0110		
0111	0001	0101	0111	
1000	1000			
1001	1000	1001		
1010	1000	1010		
1011	0001	0011	1011	
1100	0001	0101	0100	1100
1101	1000	1001	1101	
1110	1000	1010	1110	
1111	1000	1010	1110	1111

정리 1 : n 차원 Z-cube Q_n^Z의 직경은 n이다.

증명) Z-cube의 차원에 대한 수학적 귀납법으로 증명해보자. 차원 n은 4의 배수이므로, 먼저 n이 4일 때부터 위의 명제가 참임을 보이자. 이것은 <표 1>에 의해 자명한 사실이다. 위 명제가 Q_k^Z에서 참이라 가정하고 Q_{k+4}^Z에서도 참임을 보이면 된다. Q_k^Z의 직경이 k 라면 δ_k의 직경도 역시 k이다. 따라서 **보조정리 1**과 **보조정리** 2에 의해 Q_{k+4}^Z의 직경은 $k+4$이다.

4. Z-cube 의 정점간 평균 거리

Q_n^Z의 부 그래프 G의 두 정점 u와 v 대해, $D(u,v,G)$를 u와 v 사이에서의 거리라 하자. Q_n^Z에서의 두 정점들 간의 평균 거리는 모든 가능한 두 정점간의 $D(u,v,G)$의 합을 그 가능한 정점간의 개수로 나눔으로써 구할 수 있다. Z-cube는 정점대칭이므로 Q_n^Z에서의 정점간 평균거리는 $00...0$을 제외한 모든 n 비트열 b 에 대한 $D(00...0,b,Q_n^Z)$의 합을 b의 개수로 나누어 줌으로써 구할 수 있다. $T(n)$을 $\sum_{b\neq 00...0} D(00...0,b,Q_n^Z)$라 하면, **보조정리 1**에 의해, Q_n^Z은 16개의 $(n-4)$ 차원 서브큐브 δ_{n-4}로 분할되고 δ_{n-4}은 또한 16개의 δ_{n-8}로 분할됨을 알 수 있다. 따라서 <표 1>에 의해,

$$T(n) = 16 \sum_{b\in\delta_{n-4}} D(00...0,b,Q_{n-4}^Z) + 36*2^{n-4}$$

$$T(4) = 36$$

로 표시될 수 있다.

이 점화관계(recurrence relation)를 풀면

$$T(n) = 16 * T(n-4) + 36 * 2^{n-4}$$

$$= \frac{9}{16} * 2^n * n$$

이 되며 모든 가능한 두 정점간의 $D(u,v,G)$의 합인 $T(n)$은 다음의 **보조정리**로 요약할 수 있다.

보조정리 3 : Q_n^Z 에서

$n = 4$일 때, $T(n) = 36$;

$n > 4$일 때, $T(n) = \frac{9}{2^4} * 2^n * n$

D_{ave}을 Q_n^Z에서의 정점간 평균거리라고 하면 $D_{ave} = T(n)/2^n - 1$이 되므로 다음 **정리**를 구할 수 있다.

정리 2 : n차원 Z-cube Q_n^Z에서, D_{ave} 대략 $\frac{9}{16} * n$이다.

5. Z-cube의 고장직경

고장직경이란 네트워크의 정점차수 미만의 정점에 고장이 발생했을 때의 네트워크의 직경을 말한다. 즉 네트워크가 두 개 이상의 그래프로 나누어지지 않을 만큼의 정점이 고장이라면 정점차수 미만의 정점에 고장이 있음을 의미하며, 고장직경은 바로 이때의 가장 멀리 떨어진 두 정점간의 최단경로의 길이를 말한다.

본 장에서는 Z cube에서의 두 정점간의 경로들에 관해 분석해 보고 이를 이용해 고장직경을 산출해 보기로 한다.

정점 u와 정점 v가 직접 연결되어 있다면 이진수로 표시된 u와 v의 주소는 정확히 한 비트만 다르게 되며 이때 n번째 비트가 다르다면 u와 v를 연결하는 연결선을 k차원 연결선이라고 하고 k-link로 표시하기로 하자. 예를 들면 (그림 3)에서 정점 0000과 정점 0001을 연결하는 연결선은 1차원 연결선이며 1 link로 표시된다. n차원 Z-cube Q_n^Z에서의 두 정점 u= $u_n u_{n-1} \ldots u_1$와 $v = v_n v_{n-1} \ldots v_1$사이의 경로는 **보조정리 1**에 의해 우선 $u_n u_{n-1} u_{n-2} u_{n-3}$과 $v_n v_{n-1} v_{n-2} v_{n-3}$사이의 경로를 해당 δ_4에서 찾고 다시 $u_{n-4} u_{n-5} u_{n-6} u_{n-7}$과 $v_{n-4} v_{n-5} v_{n-6} v_{n-7}$ 사이의 경로를 다시 해당 δ_4에서 찾는 과정을 반복함으로써 구할 수 있으며 이는 **보조정리 2**에 의해 Q_4^Z에서의 경로를 $\frac{n}{4}$번 반복하여 찾는 것과 같다. 따라서 본 논문에서는 Q_4^Z에서의 경로에 대해 살펴보기로 한다.

Q_4^Z에서의 정점들을 연결선에 따라서 나누어 보면 1 link, 2-link, 4-link로 연결된 정점들과 1-link, 2-link, 3-link로 연결된 정점들로 크게 나눌 수 있다. 전자를 A형 정점이라 하고 후자를 B형 정점이라고 하면 A형 정점들과 B형 정점들의 개수는 같다. 임의의 정점이 A형인지 B형인지는 그 정점의 이진주소의 마지막 2 비트로서 식별된다. 마지막 2 비트에서 1의 개수가 짝수이면 A형이고 홀수이면 B형이 된다. 먼저 정점들 간의 최단경로에 관해 알아보자.

Z-cube에서의 최단경로는 <표 1>에 나타나 있고 이를 routing table로 사용하면 매우 간단히 정점들 간의 최단 경로를 구할 수 있지만[10] 송신하는 정점의 주소와 수신하는 정점의 주소만으로 다음과 같이 간단한 알고리즘으로 구할 수 있다. (아래의 알고리즘에서 기호 ⊗은 exclusive-or 연산임) 3장에서의 직경 산출과 같은 이유로 Q_4^Z 에서의 정점 0000에서 다른 정점 사이의 최단거리를 조사해 보면 되며 정점 0000은 A형이므로 다음의 2가지 경우만 보면 된다.

첫째, A형 정점(u)에서 A형 정점(v)으로의 최단경로

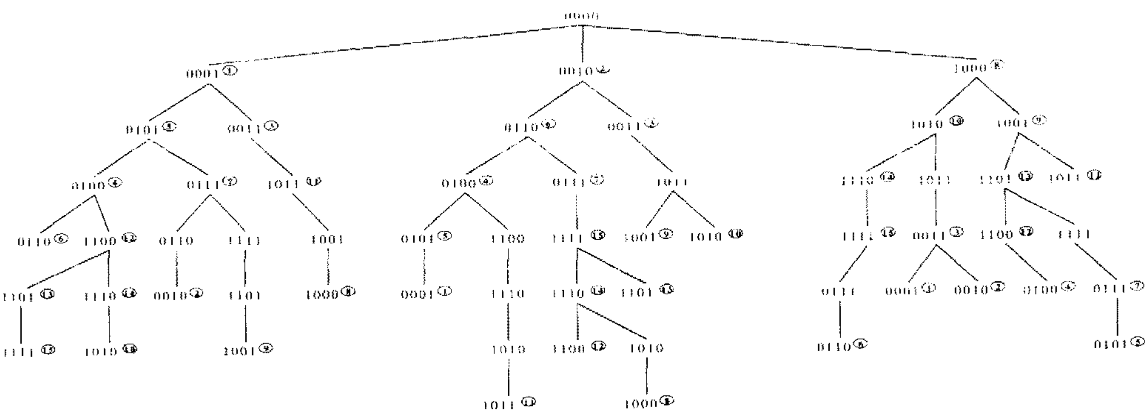
A형 정점 사이에는 3-link가 없으므로 u⊗v에서 3번째 비트가 1이라면 1-link나 2-link를 따라 B형 정점으로 이동하여 B형 정점의 3-link로 움직인 후, 다시 1-link나 2-link를 따라 이동해야 한다. 하지만 u⊗v의 3번째 비트가 1이 아니라면 하이퍼큐브에서와 같이 u⊗v에서 1이 되는 비트의 자리수-link들을 따라 이동한다.

둘째, A(B)형 정점(u)에서 B(A)형 정점(v)으로의 최단경로

B형 정점에는 4-link가 없으므로 u⊗v에서 4번째 비트가 1이라면 처음에는 A형 정점의 4-link를 따라 이동한 후 u⊗v에서 1이 되는 하위 자리수의 연결선을 따라 움직인다. u⊗b에서 4번째 비트가 1이 아니라면 u⊗v에서 1이 되는 하위 자리수의 연결선을 따라 움직인다.

Z-cube에서의 최단 경로를 구하는 위 과정에서 연결선의 순서가 미리 정해져 있지 않아 임의로 선택할 수 있을 경우 하위의 연결선을 우선적으로 선정하여 간다면, 정점 0000에서 다른 모든 정점까지의 최단거리는 <표 1>에서 나타난 바와 같으므로 위의 알고리즘은 정확하게 최단거리를 찾아낸다고 볼 수 있다.

Q_4^Z는 정점차수가 3이므로 Q_4^Z에서 임의의 두 정점 사이에 정점 무중복 경로(Node Disjoint Path)의 수는 3이다. 이때 이들 중 가장 긴 경로의 거리는 정점 0000에서 정점 1000과 정점 0000에서 정점 1011사이에서 나온다. 예를 들어 정점 0000과 정점 1000사이의 정점 무중복 경로에 대해 살펴보자. (그림 3)에서 보는 바와 같이 정점 0000에서 정점 1000사이의 정점 무중복 경로의 개수는 3이므로 그중 한 개의 경로는 반드시 정점 0100이나 정점 0111에 연결 된 4-link를 사용해야 한다. 따라서 위 알고리즘의 첫째 경우와 같은 경로는 밟을 수가 없고 정점 0000에서 B형의 정점으로 이동한 후에도 둘째 경우와 같은 경로도 밟을 수가 없으며 정점의 중복을 피하여 정점 0100이나 정점 0111로 이동한 후 다시 정점 1100이나 정점 1111에서 정점 1000을 찾아가야 하며 이때의 경로의 길이는 최악의 경우인 7이다. Q_4^Z에서의 정점 무중복 경로들의 길이를 알기 위해 모든 경우를 나열하기 보다 (그림 4)의 정점 무중복 경로를 나타내는 츄리(Tree)로서 간단히 설명해 보자. (그림 4)에서 위 첨자로 표시가 된 정점은 최종 목적지를 나타내고 그렇지 않은 것은 경로 중에



(그림 4) Q_4^Z 에서의 정점 무중복 경로
(Fig. 4) Node Disjoint Path in Q_4^Z

있는 중간 정점을 말한다. 정점 0000에서 3개의 부츄리(Sub-tree)가 있고 이들에는 각각 15개의 최종 목적지가 있으며 각 부츄리에서 같은 위 첨자가 사용된 정점은 같은 목적지를 나타내고 있다. 즉 Q_4^Z의 정점 0000에서 다른 모든 정점으로의 경로를 각 부츄리가 포함하고 있으며 3 개의 부츄리에 나타난 임의의 정점으로의 경로에는 중복되는 정점이 없다는 것을 확인할 수 있다. 또한 가장 긴 경로의 길이는 7이다. 기술의 편의상 이를 **보조정리**로 요약하면 다음과 같다.

보조정리 4 : Q_3^Z 에서의 임의 두 정점 사이에는 경로의 길이가 7이하인 정점 무중복 경로가 3개 존재한다.

정리 1에서와 같이 **보조정리 1**과 **보조정리 2**를 사용하여 **보조정리 4**를 Q_n^Z으로 일반화하면 다음과 같다.

보조정리 5 : Q_n^Z 에서의 임의의 두 정점 사이에는 경로의 길이가 $n+\frac{3n}{4}$ 이하인 정점 무중복 경로가 n개 존재한다.

정리 3 : n 차원 Z-cube Q_n^Z 에서의 고장직경은 $n+\frac{3n}{4}$ 이다.

증명) 고장직경은 네트워크의 정점차수 미만의 정점에 고장이 발생했을 때의 네트워크의 직경을 말한다. **보조정리 5**에서 Q_n^Z의 임의의 두 정점 사이에는 경로의 길이가 $n+\frac{3n}{4}$ 이하인 정점 무중복 경로가 n개 존재한다고 했으므로 n-1개의 정점에 고장이 발생하더라도 적어도 1개의 경로가 존재하고 그 길이는 $n+\frac{3n}{4}$ 이하임을 의미한다. 따라서 Q_n^Z에서의 고장직경은 $n+\frac{3n}{4}$ 이다. ■

5. 결 론

본 논문에서는 Z-cube 연결 네트워크에서 프로세서들 간의 통신 지연에 관해 고찰해 보았다. Z-cube가 하이퍼큐브보다 작은 정점차수를 갖기 때문에 하드웨어로의 구현은 쉬운 반면 직경이나 정점간의 평균거리는 길어 질 것으로 예상되나 **정리 1**에서 보여진 것처럼 직경은 하이퍼큐브의 것과 동일했으며 **정리 2**에서 보여진 대로 평균거리는 하이퍼큐브의 1.125배에 지나지 않았다. n차원 하이퍼큐브의 고장직경이 n+1이므로 Z-cube의 고장직경은 **정리 3**에서 보여진 바와 같이 차수에 따라 대략 하이퍼큐브의 1.4배 내지는 1.7배 정도 됨을 알 수 있다.

프로세서들을 연결하는 배선의 수는 하이퍼큐브의 3/4에 지나지 않으면서도 다른 유사 하이퍼큐브류(Hypercube-like Network)[1][2] 보다 매우 단순한 routing 구조를 가지고 있다. 정상적인 작동 시에는 Z-cube의 통신 성능이 하이퍼큐브와 거의 같다고 말할 수 있으며 고장 발생시에도 하이퍼큐브보다는 최악의 경우 다소 통신지연이 커질 수 있으나 이 역시 하드웨어로의 구현이 주안점이 될 때에는 무시 될만한 수준이므로 Z-cube는 하이퍼큐브를 대체할 수 있다고 하겠다.

또한 다른 유사 하이퍼큐브류(Hypercube-like Network)[1][2]의 특성과 Z-cube의 특성을 혼합한다면 하

이퍼큐브보다 적은 연결차수와 짧은 직경을 갖는 네트워크의 설계가 가능할 것으로 사료되며 앞으로의 연구과제로 제안하는 바이다.

참 고 문 헌

[1] K. Efe, P. Blackwell, T. Shiau, and W. Slough, "A reduced diameter interconnection network," Dept. of Computer Science, Univ. of Missouri, Columbia, 1988.

[2] Y.Y. Sung, "X-hypercube : A better interconnection network," Proc. 26th Annual Southeast ACM Conf., pp.557-561, 1988.

[3] K.H. Kwon, S. Latifi, E.K. Park, S.Q. Zheng, "Hypercube-like networks with reduced interconnection degrees," Journal of Software Engineering & Computer, Vol.2, No.1, pp.111-133, 1994.

[4] K.H. Kwon, S. Latifi, and S.Q. Zheng, "Average distance of data communication in twisted hypercubes," Proc. Int. Conf. Finite Fields, Coding Theory, and Advances in Communication and Computing(Lecture Notes in Pure and Applied Math.) New York : Marcel Dekker, pp.329-340, 1991.

[5] S.Q. Zheng, K.H .Kwon, and J.H. Chen, "Finding a shortest path in twisted hypercubes," Congressus Numerantium, Vol.83, 1991, pp.75-90.

[6] K. Efe, "The crossed cube architecture for parallel computation," IEEE Trans. Parallel and Distributed Systems, Vol.3, No.5, pp.513-524, 1992.

[7] C.T. Ho, "Full bandwidth communications on folded hypercubes," Proc. Int. Conf. Parallel Processing, Vol.1, pp.276-280, 1990.

[8] F.T. Leighton, "Introduction to Parallel Algorithms and Architectures," Morgan Kaufmann Publishers, Inc, 1992.

[9] M.J. Quinn, "Parallel Computing," McGraw-Hill 1994.

[10] William Stallings. "Data and Computer Communications," 5th Edition, Prentice Hall, 1997.

[11] M.S. Krishnamoorthy, B. Krishnamurthy "Fault Diameter of Interconnection Networks," Computer and Math. Applic. Vol.13, pp.577-582, 1987.

[12] Sharam Latifi, "On the Fault Diameter of the Star Graph," Information Processing Letters, Vol.46, pp.663-665, 1994.

[13] S. Lennart Johnsson, Ching-Tien Ho, "Optimum Broadcasting and Personalized Communication in Hypercubes," IEEE Transactions on Computers, Vol.38, No.9, pp.1249-1268, 1989.

[14] Sheldon B. Akers, Balakrishnan Krishnamurthy, "A Group-Theoretic Model for Symmetric Interconnection Networks," IEEE Transactions on Computers, Vol.38, No.4, pp.555-566, 1989.

[15] 권 경희, "통신성능의 관점에서 본 Z-cube 네트워크", 한국정보과학회 병렬처리시스템 학술발표회 논문지, Vol.6, No.2, pp.59-64, 1995.

권 경 희

e-mail : khkwon@cs.dankook.ac.kr

1976년 고려대학교 물리학과(학사)

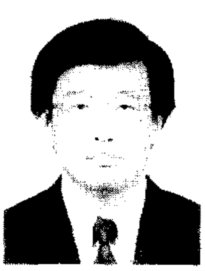
1986년 Old Dominion Univ. Computer Science Dept. M.S.

1991년 Louisiana State Univ. Computer Science Dept. Ph.D.

1979년 ~ 1984년 산업연구원 연구원

1993년 ~ 현재 단국대학교 전자계산학과 조교수

관심분야 : 병렬처리, 네트워크, 알고리즘



이 계 성

e-mail : gslee@cs.dankook.ac.kr

1980년 서강대학교 전자공학과(학사)

1982년 한국과학기술원 전산학과(석사)

1994년 Vanderbilt Univ. Computer Science Dept. Ph.D.

1982년 ~ 1985년 경제기획원 조사통계국 전산처리관

1994년 ~ 1996년 대구대학교 전산정보학과 전임강사

1996년 ~ 현재 단국대학교 전자계산학과 조교수

관심분야 : 전문가시스템, 기계학습, ITS, Data Mining