순위차원라우팅을 사용한 완전 이진트리의 3차원 메쉬로의 링크 충돌 없는 임베딩

(Link-Disjoint Embedding of Complete Binary Trees into 3D-Meshes using Dimension-Ordered Routing)

박 상 명 † 이 상 규 †† 문 봉 희 †††

(Sang-Myung Park) (Sang-Kyu Lee) (Bong-Hee Moon)

요 약 본 논문에서는 링크 충돌을 최소화 하는데 중점을 두고 순위차원 라우팅을 사용하여 완전이진트리를 3차원 메쉬로 임베딩하는 방법을 제안한다. 기존의 연구[14]에서는, 완전이진트리를 3차원 메쉬에 임베딩 할 때 순위차원 라우팅을 사용하는 경우 1, 2 차원에서 링크 충돌이 발생하였고, 순위차원 라우팅을 따르지 않는 경우 1차원에서만 링크 충돌이 존재하도록 하는 임베딩 방법을 보였었다. 이와 비교하여 본 논문에서는 순위차원 라우팅을 사용하고 링크 충돌이 존재하지 않는 임베딩 방법을 제안하며, 이 방법에 의해 임베딩을 수행한 결과, 임베딩을 위해 사용된 메쉬의 크기가 최적크기의 1.27 배를 넘지 않음을 증명한다.

Abstract This paper is considered with the problem of embedding complete binary trees into 3-dimensional meshes using dimension-ordered routing with primary concern of minimizing link congestion. The authors showed that a complete binary tree with 2^p-1 nodes can be embedded into a 3-dimensional mesh with optimum size, 2^p nodes, if the link congestion is two[14]. (More precisely, the link congestion of each dimension is two, two, and one if the dimension-ordered routing is used, and two, one, and one if the dimension-ordered routing is not imposed.) In this paper, we present a scheme to find an embedding of a complete binary tree into a 3-dimensional mesh of size no larger than 1.27 times the optimum with link congestion one while using dimension-ordered routing.

1. 서 론

Guest 그래프 G를 호스트 그래프 H로 임베딩 하는 문제는 여러 가지 다른 조건 아래에서 다양하게 연구되어져 왔다. 이러한 임베딩 결과들은 Monien과 Sudborough [1]의 서베이 논문에서 잘 정리해 주고 있다.

임베딩을 할 때에는 임베딩의 효율을 측정할 수 있도록 하는 목적 함수들을 고려해야 하는데 이러한 함수들로는 *dilation, expansion, 링크 충돌* 등이 있다. Dilation은 guest 그래프 G에서 인접했던 두 노드들이 호스트 그래프 H로 임베딩 되면서 갖게 되는 경로의 길이들 중 최대 값을 말하며, expansion은 guest 그래프의 노드 수에 대한 호스트 그래프의 노드수의 비율 (즉, $|V_H|/|V_G|$)을 나타내는 것으로, 임베딩 시에 호스트 그래프에 얼마나 많은 노드가 사용되었는가를 말한다. G에서 H로 임베딩을 할 때, G의 링크는 H에서 링크 또는 경로(path)로 매핑 되는데, H의 각 링크에 매핑 되는 G의 링크수가 1보다 크게 되면 이러한 경우 *링크 충돌*이 발생되었다고 한다.

지금까지 임베딩에 관한 대부분의 연구들은 *store-and-forward* [2, 3, 4, 5, 6, 7] 통신방식에 연유하여 dilation을 최소화하는데 초점을 맞추어 왔다. 최근에는 *circuit-switching*과 웜홀 라우팅[8]이라는 스위칭 기술의 도입으로 멀티컴퓨터에서의 통신 오버헤드를 크게

† 비 회 원 : 숙명여자대학교 자연과학연구소 연구원
smpark@cs.sookmyung.ac.kr
†† 정 회 원 : 숙명여자대학교 전산학과 교수
sanglee@sookmyung.ac.kr
††† 종신회원 : 숙명여자대학교 전산학과 교수
moon@sookmyung.ac.kr
논문접수 : 1999년 1월 28일
심사완료 : 1999년 10월 25일

줄일 수 있었을 뿐 아니라 라우팅 메커니즘의 평가 기준도 변화시켰는데, circuit-switching과 웜홀 라우팅에서의 통신 시간은, 링크 충돌이 없고 메시지의 크기에 비해 패킷의 크기가 훨씬 작다면 두 노드 사이의 거리에 거의 영향을 받지 않는다[8, 9, 10]. 그러므로, circuit-switching 또는 웜홀 라우팅을 지원하는 멀티컴퓨터에서 임베딩을 수행하고자 할 때에는 dilation 보다는 링크 충돌을 최소화하는데 중점을 두어야 한다.

본 논문에서는 웜홀 라우팅을 지원하는 네트워크를 갖는 시스템 상에서, 링크 충돌을 최소화하는데 중점을 두어 완전이진트리를 3차원 메쉬로 임베딩하는 방법을 제안한다.

메쉬 interconnection 네트워크에서 두 노드들 간의 통신에는 여러 개의 경로가 존재한다. 이때 여러 개의 경로 중 하나를 선택하는 방법을 라우팅이라고 하는데, 본 논문에서는 경로 선택 시 가장 일반적인 순위차원 라우팅을 사용하도록 한다. 메쉬에서 순위차원 라우팅은 차원에 순서를 두고 그 순서에 따라 메시지를 전달하는데, 각 차원에서는 목적지 노드와 동일한 차원의 주소를 갖는 중간 노드까지 경로를 설정한 후 다음 순서의 차원으로 라우팅을 계속하여 실제 목적지까지 이르도록 하는 방법이다.

이와 비슷한 개념으로, 임베딩 시에 호스트 그래프로 매핑되는 guest 그래프의 모든 edge (u,v)에 대해서 u로부터 v에 이르는 경로가 순위차원 라우팅을 따르면 이를 **순위차원 임베딩**이라고 하겠다. 이 방법은 Symult 2010과 MIT J-machine등 상용화 되어있는 멀티컴퓨터에서 사용되고 있다[11, 12].

만약, 이진트리를 메쉬로 임베딩 할 때 메쉬의 크기와 차원의 수에 제한이 없다면 링크 충돌이 없도록 하는 임베딩은 항상 가능할 것이다. 예를 들면, 그림 1은 VLSI 설계 시 잘 알려진 H-tree 구성방법 [13]을 사용하여 깊이가 7인 완전이진트리를 15×15 메쉬로 링크 충돌없이 임베딩 하는 방법을 보여주고 있다.

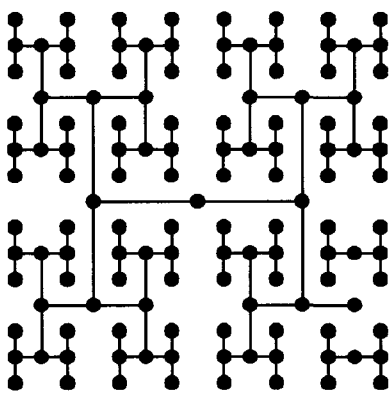


그림 1 T_7의 H-Tree 설계

깊이가 p로 2^p-1개의 노드를 가진 완전이진트리를 T_p라고 하자. T_p 를 H-tree 형태로 임베딩 할 때, p가 $p\geq 5$이면서 홀수이면 T_p는 전 단계에서 만들어지는 네 개의 서브트리 T_{p-2} 를 추가의 열과 행을 사용하여 각 네 개의 서브트리의 루트에 연결하는 방법을 사용하여 순환적으로 구성할 수 있다. (이렇게 구성된 T_p에서 중앙에 있는 노드를 루트노드로 한다.)

T_p의 H-tree 설계를 위해서는 $2^{p+1}-2^{\frac{(p+3)}{2}}+1$개의 노드가 필요하다 (즉, $(2^{\frac{(p+1)}{2}}-1)\times(2^{\frac{(p+1)}{2}}-1)$ 메쉬의 크기). 그런데 T_p를 위한 모든 H-tree 설계에는 적어도 트리의 노드 개수인 2^p-1 만큼의 노드 개수를 갖는 2차원 메쉬가 필요하다. 따라서, T_p의 임베딩에 필요한 메쉬의 최적 크기를 2^p개의 노드 개수라 하면, H-tree 방식으로 임베딩을 한 경우 메쉬의 크기는 최적크기의 약 두 배 정도로 수렴함을 볼 수 있다. 프로세서가 많은 시스템보다는 적은 시스템이 비용 면에서 더 효과적이므로 임베딩을 수행할 때에는 expansion도 고려해야 한다.

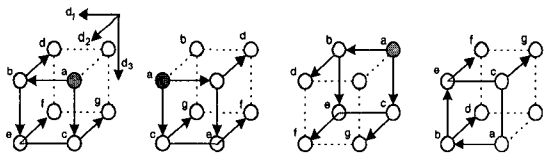
다음 장에서는 링크 충돌이 존재하지 않고, expansion은 최적의 1.27 배를 넘지 않으며 순위차원 임베딩을 사용하여 완전이진트리를 3차원 메쉬로 순위차원 임베딩하는 방법을 보인다. 마지막으로 3장에서는 결론을 맺는다.

2. 임베딩

이 절에서는 순위차원 라우팅을 사용하고 모든 링크들에 대하여 링크 충돌이 발생하지 않도록 하여 완전이진트리를 3차원 메쉬로 임베딩하는 방법을 제안한다. 기존의 연구[14]에서는, 완전이진트리를 3차원 메쉬로 임베딩 할 때 순위차원 라우팅을 사용하는 경우 1, 2차원에서 링크 충돌이 발생하였고, 순위차원 라우팅을 따르지 않는 경우는 1차원에서만 링크 충돌이 발생하도록 하는 임베딩 방법을 보였었다.

그런데 앞서 언급했던 내용에서 알 수 있듯이 웜홀 라우팅을 사용하는 멀티프로세서 시스템의 통신에 있어서 링크 충돌은 상당히 중요한 요소가 되므로 본 논문에서는 링크 충돌을 최소화하는데 중점을 두어 임베딩을 수행한다.

링크 충돌이 존재하지 않는 3차원 메쉬의 호스트 그래프를 $n_X(p)\times n_Y(p)\times n_Z(p)$로 (여기서 p는 guest 그래프인 완전이진트리의 깊이, $n_X(p)$, $n_Y(p)$, $n_Z(p)$ 은 각각 X, Y, Z축 방향으로의 크기를 의미한다.) 나타내고, 링크 충돌이 존재하지 않으면서 T_p를 3차원 메쉬에 임베딩 하기 위해 필요한 총 노드의 개수는 $N(p)$로 표기한다. 또한, T_p를 2^p개의 노드를 가진 d-차원 메쉬로 임베딩 하는 것을 E_p라 하고, $1\le i\le d$ 에 대하여 임베딩 $f_i(E_p)$를 i 차원에 대한 E_p의 거울영상(mirror image)라 하자. 예를 들면, T_3를 $2\times2\times2$ 메쉬로 임베딩 하는 것을 E_3라고 할 때 (그림 2(a)), 그림 2(b~d)는 각각 $f_1(E_3)$, $f_2(E_3)$, $f_3(E_3)$를 보여주고 있다. $f_i(f_j(E_p))$와 $f_j(f_i(E_p))$는 같은 결과를 가지며, 만약 E_p가 순위차원 임베딩 이라면 $f_i(E_p)$ 또한 순위차원 임베딩이 됨을 쉽게 볼 수 있다.



(a) E_3 (b) $f_1(E_3)$ (c) $f_2(E_3)$ (d) $f_3(E_3)$

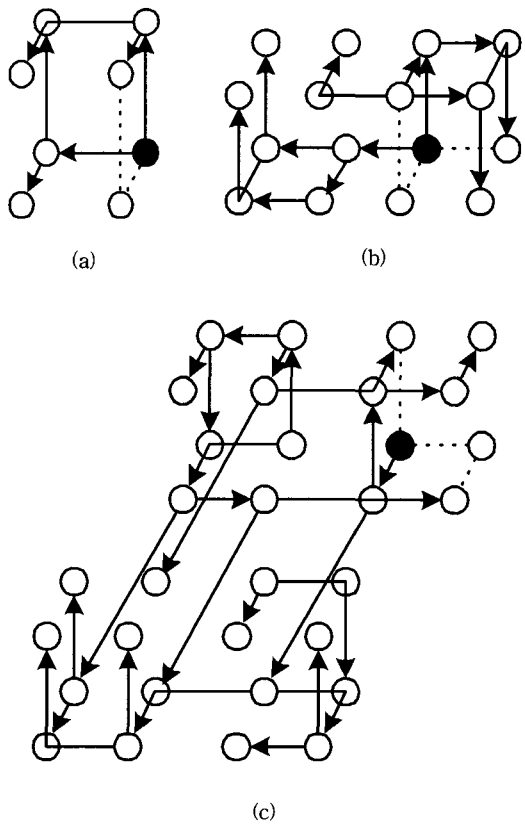
그림 2 T_3의 순위차원 임베딩과 $f_i(E_3)$

정리 1 순위차원 라우팅을 사용하여 깊이가 p인 완전이진트리를 $n_X(p)\times n_Y(p)\times n_Z(p)$ 의 3차원 메쉬로 링크 충돌없이 다음을 만족하도록 임베딩 할 수 있다.

$$N(p)=n_X(p)\cdot n_Y(p)\cdot n_Z(p)\le 1.27\cdot 2^p$$

증명 먼저 완전이진트리의 깊이 p가 $3\le p\le 8$ 일 때의 임베딩 방법을 본다. 그림 3은 깊이가 3, 4, 5인 완전이진트리를 각각 $2\times2\times2$, $4\times2\times2$, $4\times4\times2$의 3차원 메쉬에, 그림 4, 5, 6은 깊이가 6, 7, 8인 완전이진트리를 각각 $4\times4\times4$, $8\times4\times4$, $8\times8\times4$의 3차원 메쉬에 링크 충돌없이 순위차원 라우팅을 따르는 임베딩을 수행한 결과를 보여준다. 그림 (3)~그림 (6)과 같이 임베딩을 수행하면, 완전이진트리의 깊이 p가 $3\le p\le 8$인 모든 경우에 최적크기인 2^p개의 노드를 가진 3차원 메쉬로 임베딩이 가능하므로 p가 $3\le p\le 8$인 경우 정리 1이 참임을 증명하였다.

다음으로 완전이진트리의 깊이 p가 $p\ge 9$인 경우에 대해서도 정리 1이 참임을 보인다.먼저 완전이진트리의 깊이 p가 $p=3k$이고 k는 3이상인 정수일 때를 살펴보겠다. 이러한 경우의 임베딩은 완전이진트리의 깊이 p를 k의 값에 의해 3씩 증가시키며 순환적 방법으로 수행한다.

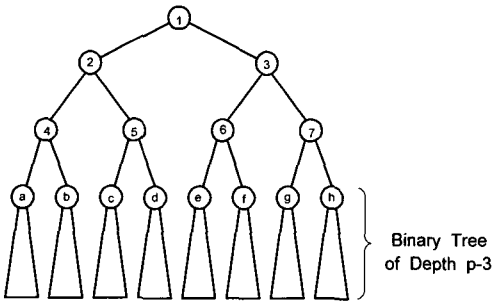


(a) (b) (c)

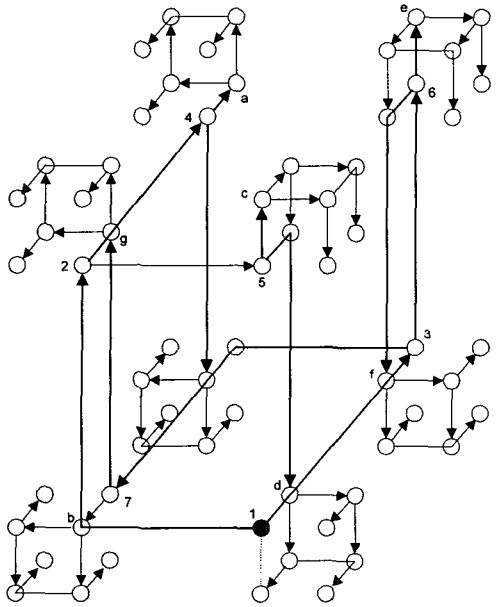
그림 3 깊이 $p=3,4,5$인 완전이진트리의 3차원 메쉬로의 임베딩

깊이 p의 완전이진트리를 3차원 메쉬로 순환적 임베딩하는 방법은 이진트리의 깊이가 $p-3$ 일 때 임베딩으로 생성된 8개의 임베딩 결과들과 2차원 메쉬를 하나 더 사용하여 그림 7과 같이 수행한다. 첫 번째로 순환적 임베딩을 수행할 때, 즉 이진트리의 깊이 p가 9일 때 그림 7(b)에서 8개의 육면체로 나타나 있는 기본(base) 임베딩으로는 깊이가 6인 완전이진트리를 그림 4와 같이 $4\times4\times4$의 3차원 메쉬로 임베딩 한 결과를 사용한다.

그림 7(a)의 깊이가 p인 완전이진트리 T_p를 보면,



(a)



(b)

그림 4 깊이 p=6인 완전이진트리의 4×4×4 메쉬로의 임베딩

T_p는 $a, b, \cdots, h$ 노드를 각각 루트로 갖는 8개의 T_{p-3} 서브트리, 첫 번째 레벨에서부터 상위 세 번째 레벨에 있는 1, 2, …, 7 노드들 그리고 1, 2, …, 7 노드들과 서브루트 $a, b, \cdots, h$ 노드들을 연결하는 14개의 링크들로 나누어 생각할 수 있다. 깊이가 3씩 증가하는 순환적인 임베딩에 의해 T_p를 임베딩 하려면, T_{p-3}를 3차원 메쉬로 임베딩 하는 방법을 이미 찾았다고 가정할 수 있고, $a, b, \cdots, h$ 노드를 루트로 갖는 그림 7(b)의 8개의 육면체가 이러한 T_{p-3} 서브트리들의 임베딩을 나타낸다.

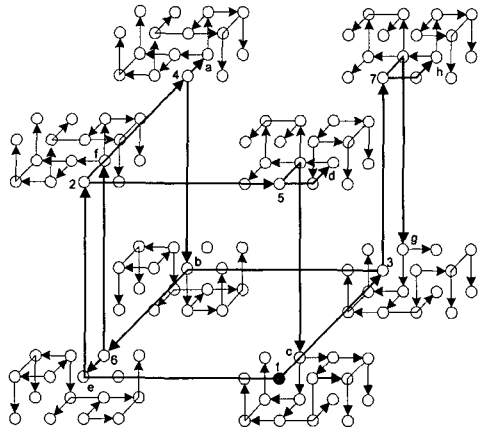


그림 5 깊이 p=7인 완전이진트리의 8×4×4 메쉬로의 임베딩

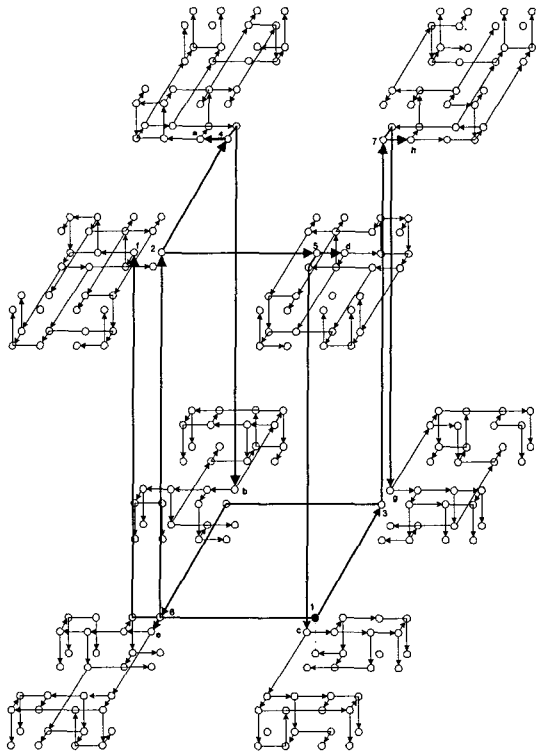


그림 6 깊이 p=8인 완전이진트리의 8×8×4 메쉬로의 임베딩

이제 상위 레벨에 있는 노드들 1, 2, …, 7과 14개의 링크들을 새로이 임베딩 해야 하는데, 1, 2, …, 7의 7개 노드들은 새로 추가되는 2차원 메쉬에 그림 7(b)와 같이 임베딩 하고, 14개의 링크들은 그림 7(b)에 나타나 있는 방법으로 T_{p-3} 서브트리들의 루트노드와 1, 2, …, 7노드들을 연결한다. 이때 14개의 링크들 중 1~7 노드를 연결하는 경로들은 새로 추가된 2차원 평면상에서 이루어지므로 충돌이 발생하지 않으며, 4~7 노드들과 a~h 노드들을 연결하는 나머지 링크들이 임베딩 되는 8개의 경로들은 그림 4의 점선이 나타내듯이 루트노드로부터 3차원의 아래 방향으로 나가는 경로가 비어 있으므로 역시 링크 충돌이 일어나지 않음을 볼 수 있다. 따라서 임베딩 시 모든 경로에 링크 충돌은 존재하지 않는다.

이러한 순환적 임베딩의 방법으로 완전이진트리를 3차원 메쉬로 임베딩 할 때, 깊이 p가 커짐에 따라 한쪽 방향 만으로의 확장됨을 피하기 위해서, 추가로 사용되는 2차원 메쉬는 그림 7, 8, 9에서 보는 것처럼 X, Y, Z 축에 교대로 사용한다. 즉, X축과 평행한 2차원 메쉬를 사용하여 깊이 p의 완전이진트리 T_p를 3차원 메쉬로 임베딩 하면, 다음으로 T_{p+3}의 완전이진트리를 임베딩 하기 위해서는 그림 8과 같이 Y축에 평행한 2차원 메쉬를 사용하여 전 단계에서 수행한 임베딩의 결과로 얻은 8개의 서브트리의 루트노드들과 연결하여 임베딩을 수행한다. 이때 그림 8과 같이 8개의 서브트리의 루트노드들과 노드 4, 5, 6, 7들을 연결하게 되는데, 그림 7(b)에서 보는 바와 같이 루트노드 1에서 2차원 방향으로의 링크들이 모두 비어 있으므로 링크 충돌은 전혀 발생하지 않는다. 완전이진트리 T_{p+6}에 대해서도 Z축과 평행한 2차원 메쉬를 사용하여 마찬가지 방법으로 그림 9와 같이 수행하며임베딩한 결과 링크 충돌은 발생하지 않으며, 이렇게 수행한 순환적 방법 역시 순위차원 임베딩을 만족하고 있음은 쉽게 알 수 있다.

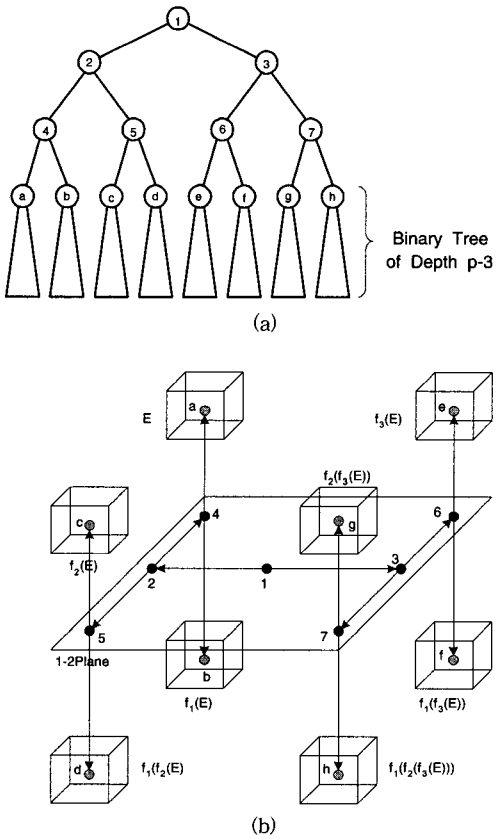


그림 7 깊이 $p=3k(k\geq 3)$인 완전이진트리의 3차원 메쉬로의 임베딩

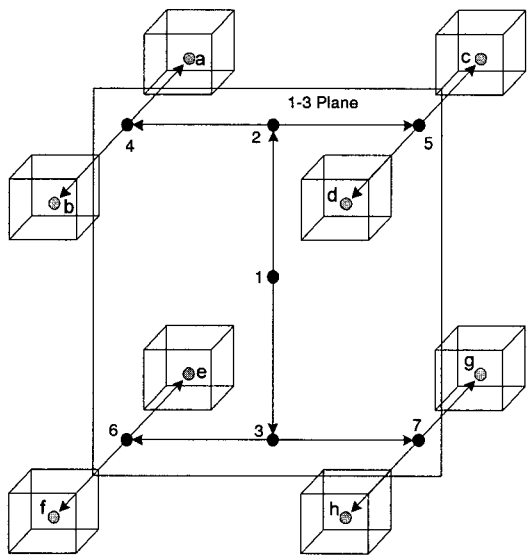


그림 8 깊이 $p=3k+1(k\geq 3)$인 완전이진트리의 3차원 메쉬로의 임베딩

위에서 소개한 절차에 따라 깊이가 p인 완전이진트리를 3차원 메쉬로 임베딩 했을 때, 임베딩에 사용된 3차원 메쉬의 노드수 $N(p)$ 는 다음과 같이 계산할 수 있다.

첫 번째로 $p=3k$ 인 경우,

$$N(6)=4\cdot 4\cdot 4,$$

$$N(9)=(2\cdot 4+1)\cdot(2\cdot 4)\cdot(2\cdot 4),$$

$N(12)=(2^2\cdot 4+2)\cdot(2^2\cdot 4+1)\cdot(2^2\cdot 4)$,

$N(15)=(2^3\cdot 4+2^2)\cdot(2^3\cdot 4+2)\cdot(2^3\cdot 4+1)\cdots$ 이 되고,

X, Y, Z의 노드의 개수를 각각 $n_X(3\cdot(k-1))$, $n_Y(3\cdot(k-1))$, $n_Z(3\cdot(k-1))$ 라 할 때 $N(3\cdot(k-1)) = n_X(3\cdot(k-1))\cdot n_Y(3\cdot(k-1))\cdot n_Z(3\cdot(k-1))$ 라 하자. 그러면,

$$N(3k)=\begin{cases} 2n_X(3\cdot(k-1)+1)\cdot 2n_Y(3\cdot(k-1))\cdot 2n_Z(3\cdot(k-1)) \\ \quad when\ k=3j \\ 2n_X(3\cdot(k-1))\cdot 2n_Y(3\cdot(k-1)+1)\cdot 2n_Z(3\cdot(k-1)) \\ \quad when\ k=3j+1 \\ 2n_X(3\cdot(k-1))\cdot 2n_Y(3\cdot(k-1))\cdot 2n_Z(3\cdot(k-1)) \\ \quad when\ k=3j+2 \end{cases}$$

$N(6)=4\cdot 4\cdot 4$를 사용하여 이 식을 풀면,

$D(k)=\lfloor \frac{k}{3} \rfloor$ 이고, $M(k)=k \bmod 3$ 일 때,

$$N(p)=(2^k+\sum_{i=0}^{D(k)-1}2^{3i+M(k)})\cdot(2^k+\sum_{i=0}^{D(k-1)-1}2^{3i+M(k-1)})\cdot(2^k+\sum_{i=0}^{D(k-2)-1}2^{3i+M(k-2)}) \quad (1)$$

먼저, $M(k)$가 0일 때, $k=3j$ 라 하면 수식 (1)은

$$N(p)=(2^{3j}+\sum_{i=0}^{j-1}2^{3i})\times(2^{3j}+\sum_{i=0}^{j-2}2^{3i+2})\times(2^{3j}+\sum_{i=0}^{j-2}2^{3i+1}) \quad (2)$$

수식(2)는 다시,

$$N(p)=(2^{3j}+\frac{2^{3j}-1}{2^3-1})\times(2^{3j}+\frac{2^{3j-1}-2^2}{2^3-1})\times(2^{3j}+\frac{2^{3j-2}-2}{2^3-1}) \quad (3)$$

수식(3)으로 부터

$$N(p)\le(1+\frac{1}{4}+\frac{1}{7\cdot 8}+\frac{1}{7^3\cdot 8})\cdot 2^{9j}=1.268221574\cdot 2^{9j} \quad (4)$$

다음으로 $M(k)$가 1일 때, $k=3j+1$라 하면, 수식 (1)은

$$N(p)=(2^{3j+1}+\sum_{i=0}^{j-1}2^{3i+1})\times(2^{3j+1}+\sum_{i=0}^{j-1}2^{3i})\times(2^{3j+1}+\sum_{i=0}^{j-2}2^{3i+2}) \quad (5)$$

수식 (5)는 다시,

$$N(p)=(2^{3j+1}+\frac{2^{3j+1}-2}{2^3-1})\times(2^{3j+1}+\frac{2^{3j}-1}{2^3-1})\times(2^{3j+1}+\frac{2^{3j-1}-2^2}{2^{3-1}}) \quad (6)$$

수식 (6)으로부터,

$$N(p)\le(1+\frac{1}{4}+\frac{1}{7\cdot 8}+\frac{1}{7^3\cdot 8})\cdot 2^{9j+3}=1.268221574\cdot 2^{9j+3} \quad (7)$$

마지막으로 $M(k)$가 2일 때, $k=3j+2$라 하면, 수식 (1)은

$$N(p)=(2^{3j+2}+\sum_{i=0}^{j-1}2^{3i+2})\times(2^{3j+2}+\sum_{i=0}^{j-1}2^{3i+1})\times(2^{3j+2}+\sum_{i=0}^{j}s^{3i}). \quad (8)$$

수식(8)은 다시

$$N(p)=(2^{3j+2}+\frac{2^{3j+2}-2^2}{2^3-1})\times(2^{3j+2}+\frac{2^{3j+1}-2}{2^3-1})\times(2^{3j+2}+\frac{2^{3j}-1}{2^{3-1}}) \quad (9)$$

수식(9)로부터

$$N(p)\le(1+\frac{1}{4}+\frac{1}{7\cdot 8}+\frac{1}{7^3\cdot 8})\cdot 2^{9j+6}=1.268221574\cdot 2^{9j+6} \quad (10)$$

그러므로, $M(k)$가 0, 1, 2에 대하여 수식 (4), (7), (10)으로 부터,

$$N(p)=n_X(p)\times n_Y(p)\times n_Z(p)\le 1.27\cdot 2^p$$

따라서 이것으로써 $p=3k$(k는 3이상인 정수)일 때 정리 1을 증명하였다.

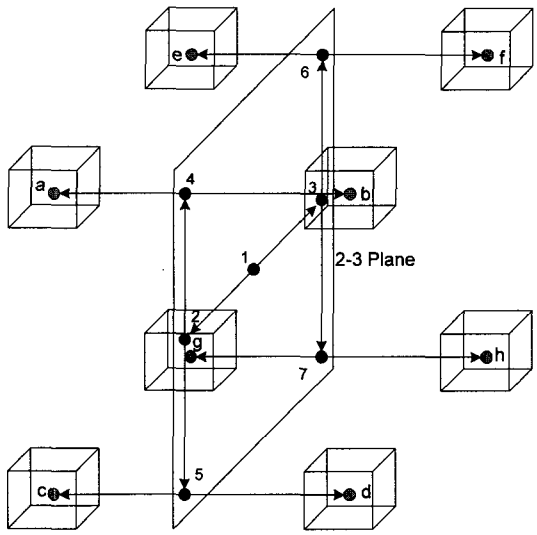


그림 9 깊이 $p=3k+2(k\ge 3)$인 완전이진트리의 3차원 메쉬로의 임베딩

다음으로 완전이진트리의 깊이 p가 $p=3k+1$, $p=3k+2$ (k는 3 이상인 정수)인 경우를 생각해 본다. 이 경우도 앞에서 제안한 $p=3k$일 때의, 깊이가 3씩 증가해 가는 순환적 임베딩 방법을 그대로 적용시킬 수 있다. 단, 이 때의 기본임베딩(깊이 10, 11의 완전이진트리를 3차원 메쉬에 임베딩 할 때 사용되는 그림 7(b)에서의 8개의 서브트리들)으로는 깊이가 7, 8인 완전이진트리의 $8\times4\times4$, $8\times8\times4$ 3차원 메쉬로의 임베딩을 사용하며, 이들은 각각 그림 5와 그림 6에서 보여주고 있다. 깊이 $p=3k$ 일 때와 동일한 순환적 방법으로 임베딩을 수행하므로 $p=3k+1$, $p=3k+2$ 일 때 역시 링크 충돌은 발생하지 않으며 순위차원 임베딩을 지키고 있다.

깊이 $p=3k+1$과 $p=3k+2$의 완전이진트리를 3차원 메쉬로 임베딩을 하는데 있어서의 기본임베딩들은 깊이

$p=3k$의 완전이진트리를 3차원 메쉬로 임베딩 할 때의 기본임베딩보다 메쉬의 크기가 최적의 크기를 가지면서 더 크고 깊이 $p=3k$ 일때와 똑같은 순환적 방법을 사용하므로 임베딩에 필요한 결과 3차원 메쉬의 크기 $N(p)$는 $p=3k$인 경우보다 더 작은 값으로 수렴할 것이다.

따라서, 깊이 p를 가지는 모든 완전이진트리는 $N(p)\leq 1.27\times 2^p$ 의 크기를 갖는 3차원 메쉬에 링크 충돌 없이 순위차원 라우팅을 사용하여 임베딩이 가능하며 이로써 정리1이 성립함을 증명하였다. ∎

3. 결 론

본 논문에서는 순위차원 라우팅을 사용하고 링크 충돌이 존재하지 않도록 최소화하면서 2^{p-1}개의 노드를 가진 완전이진트리 T_p를 최적의 1.27 배를 넘지 않는 크기의 3차원 메쉬로 임베딩 하는 방법을 제안하였다.

기존 연구 [14]에서는, 순위차원 라우팅을 따르지 않고 링크 충돌이 존재하지 않도록 완전이진트리를 최적 크기의 4차원 메쉬로 임베딩하는 방법을 보였었다.

앞으로 위 두 경우를 바탕으로 완전이진 트리 T_p를 링크 충돌이 없도록 하여 최적 크기의 3차원 메쉬로 임베딩하는 방법이나, 최적 크기의 4차원 메쉬로 순위차원 임베딩을 하는 방법도 생각해 볼 수 있을 것이다. 뿐만 아니라, 링크 충돌을 최소화하는 데 중점을 두어 다른 형태의 호스트 또는 guest 그래프에서의 임베딩도 생각해 볼 수 있다.

참 고 문 헌

[1] B. Monien and H. Sudborough, "Embedding One Interconnection Network in Another," Computing Suppl. 7, pp. 257-282, 1990.

[2] S. Bhatt, F. Chung, T. Leighton, and A. Rosenberg, "Optimal Simulations of Tree Machines," *Proc. 27th Annual IEEE Symp. Foundations of Computer Science*, October 1986, pp. 274 - 282.

[3] S. N. Bhatt and S. S. Cosmadakis, "The Complexity of Minimizing Wire Lengths for VLSI Layouts," *Information Processing Letters*, Vol. 25, 1987.

[4] J. E. Brandenburg and D. S. Scott, "Embeddings of Communication Trees and Grids into Hypercubes," *Technical Report*, Intel Scientific Computers, 1985.

[5] A. Rosenberg, "Data Encoding and Their Costs," *Acta Informatica*, No. 9, 1978.

[6] A. Rosenberg and L. Snyder, "Bounds on the Costs of Data Encodings," *Mathematical Systems Theory*, Vol. 12, 1978.

[7] Y. Saad and M. Schultz, "Data Communication in Hypercubes," *Research Report* 428, Department of Computer Science, Yale University, 1985.

[8] L. M. Ni and P. K. McKinley, "A Survey of Routing Techniques in Wormhole Networks," *IEEE Computers*, pp. 62 - 76, Feb. 1993.

[9] S. H. Bokhari, "A Network Flow Model for Load Balancing in Circuit-Switched Multicomputers," ICASE Report 90-38, May 1990.

[10] S. H. Bokhari, "Communication Over heads on the Intel iPSC-2 Hypercube," ICASE Interim Report 10, May 1990.

[11] C. L. Seitz, W. C. Ahhas, C. M. Flaig, A. J. Martin, J. Seizovic, C. S. Steele, and W. -K. Su, "The architecture ad programming of the Ametek Series 2010 multicomputer," in *Priceedings of the Third Conference on Hypercube Concurrent Computers and Applications*, Vol. I, (Pasadena,CA), pp. 33-36, Association for Computing Machinery, Jan. 1988.

[12] W. J. Dally, "The J-machine: System support for Actors," in *Actors: Knowledge-based Concurrent Computing* (Hewitt and Agha, eds.), MIT Press, 1989.

[13] J. D. Ullman, *Computational Aspects of VLSI*, Computer Science Press, 11, Taft Court, Rockville, Maryland 20850, USA 1984.

[14] S.-K. Lee and H.-A. Choi, "Link -Disjoint Embedding of Complete Binary Trees in Meshes," Journal Networks, Vol. 30, No. 4, pp. 283-292, 1997.

박 상 명

1997년 숙명여자대학교 전산학과 졸업. 1999년 숙명여자대학교 대학원 전산학과 졸업. 1999년 ~ 현재 숙명여자대학교 자연과학연구소 연구원. 관심분야는 컴퓨터구조, 병렬처리, 웹어플리케이션

이 상 규

1989년 University of Southern Califonia Dept. of Computer Science (공학사). 1991년 George Washington University Dept. of Electrical Engineering and Computer Science(공학석사). 1995년 George Washington University Dept. of Electrical Engineering and Computer Science(공학박사). 1995년 ~ 1996년 George Washington University Dept. of Electrical Engineering and Computer Science 박사후 과정. 1997년 ~ 현재 숙명여자대학교 전산학과 교수. 관심분야는 병렬/분산 처리 시스템, 컴퓨터이론, 컴퓨터 통신, 인터넷 프로그래밍.

문 봉 희

1981년 서울대학교 자연과학대학 계산통계학과(이학사). 1983년 서울대학교 대학원 계산통계학과(이학석사). 1992년 서울대학교 대학원 전산과학과(이학박사). 1984년 울산대학교 전자계산학과 전임강사. 1985년 ~ 현재 숙명여자대학교 전산학과 교수. 관심분야는 컴퓨터구조, 웹프로그래밍