t/k-시스템을 이용한 하이퍼큐브 네트워크의 결함 진단

김장환* · 이충세**

요 약

시스템-레벨 진단 알고리즘은 결함의 개수가 t개를 초과하지 않는다는 t-진단가능 시스템의 특성을 이용한다. 기존의 진단 알고리즘으로 대형 멀티프로세서 시스템에서의 보다 많은 수의 결함을 처리하기에는 한계가 있다. Somani와 Peleg은 진단의 정확 여부를 판단할 수 없는 충분히 작은 개수의 노드가 존재한다는 것을 허용으로써 결함의 갯수가 t개를 초과할 경우에도 시스템을 진단하는 t/k-diagnosable 시스템을 제안하였다. 본 논문에서는 t/k-diagnosable 시스템을 이용한 하이퍼큐브 진단 알고리즘을 제안한다. 결함의 개수가 t개를 초과하는 경우에 대하여, k개의 부정확한 진단을 허용한다. 성능 실험 결과 제안 알고리즘은 HADA알고리즘보다 우수함을 보여 주었다. 또한 제안 알고리즘은 HYP-DIAG알고리즘과의 성능 비교에서도 비슷한 결과를 보여 준다.

Fault Diagnosis Using t/k-Diagnosable System in Hypercube Networks

Jang-Hwan Kim* · Chung-Sei Rhee**

ABSTRACT

System level diagnosis algorithms use the properties of t-diagnosable system where the maximum number of the faults does not exceed t. The existing diagnosis algorithms have limit when dealing with large fault sets in large multiprocessor systems. Somani and Peleg proposed t/k-diagnosable system to diagnose more faults than t (dimension) by allowing upper bounded few number of units to be diagnosed incorrectly. In this paper, we propose hypercube diagnosis algorithm using t/k-diagnosable system. When the number of faults exceeds t, we allow k faults to be diagnosed incorrectly. Simulation shows that the performance of the proposed algorithm is better than Feng's HADA algorithm. The proposed algorithm also gives similar performance compared to HYP-DIAG algorithm.

Key words : t/k-Diagnosable, Hypercube, Diagnosis, Fault

* 성결대학교 공과대학
** 충북대학교 컴퓨터과학과

1. 서 론

병렬처리 시스템의 규모가 커짐에 따라 시스템 내에서 발생되는 결함의 빈도가 높아지고 있다. 결함의 발생으로 인하여 시스템이 다운되고 이를 복구하는데 소요되는 비용은 병렬처리 시스템의 성능을 저하시키는 가장 큰 요인이기 때문에 시스템의 신뢰성과 가용성을 향상시키기 위한 많은 연구가 진행되어 왔다. 다양한 병렬처리 시스템 중에서 하이퍼큐브 형태의 병렬처리 시스템은 정규적이며 계층적인 구조를 갖는다. 이런 하이퍼큐브의 특성은 효율적인 진단 알고리즘을 개발하는데 유리하게 적용될 수 있다.

시스템 내에 t(dimension of hypercubes)개를 초과하는 결함 노드가 존재할 경우, 진단의 가능 여부에 대한 문제를 Over-d문제라고 정의한다[1]. 이런 경우에는 불완전하지만 정확한 진단이 수행되어야 한다. Somani, Agarwal, Avis[2]는 Over-d 결함진단에 대하여 언급하였지만, 그들은 비적응적 진단의 경우만을 고려하였다. Friedman은 확률적인 방법을 이용한 Over-d 문제의 해결을 시도하였다[3].

Preparata, Metze와 Chien은 처음으로 시스템 레벨 진단의 개념을 사용하는 결함 진단 방법을 제안하였다[4]. PMC 모델이 제안된 후 많은 연구가 이 모델의 특성을 사용하여 진행되었고 진단 알고리즘은 많은 발전을 이루었다. Feng은 하이퍼큐브의 구조적인 특징을 이용하는 적응적 진단 알고리즘 HADA(Hypercube Adaptive Diagnosis Algorithm)와 IHADA (Improved HADA)를 제안하였다[1].

HADA/IHADA의 방법은 하이퍼큐브의 모든 노드를 하나의 링에 임베딩하는 반면에, Kranakis와 Pelc는 전체 하이퍼큐브 Hn을 $H_j \times H_{n-j}$로 분할하여 진단을 수행하는 알고리즘 HYP-DIAG를 제안하여 진단의 효율을 높였다[5]. 그러나 진단 가능한 결함의 최대 수를 n으로 가정하였기 때문에 대규모의 병렬처리 시스템에 알고리즘을 적용하기는 적합하지 않다. 멀티프로세서 시스템의 진단 가능한 노드 수 d는 통신 그래프의 연결성에 의하여 결정된다. 하이퍼큐브 시스템은 연결도 d가 선형적으로 증가됨에 따라 노드의 개수는 지수적으로 증가하는 특성을 갖는다. 대규모의 하이퍼큐브 멀티프로세서 시스템은 수 천 개의 노드를 포함하고 있다. 이런 복잡한 환경에서 많은 노드에서 결함이 발생될 수 있다. 이런 사실은 시스템 규모가 커짐에 따라 결함의 갯수가 t개 이상으로 증가할 개연성을 증가시킨다. Somani와 Peleg는 전통적인 t-진단가능 시스템의 단점을 감안하여 더 많은 결함을 진단할 수 있는 t/k-진단가능 시스템을 제안하였다[6]. 이 논문에서는 결함의 개수가 t개를 초과하는 경우에 대하여 진단의 정확 여부를 판단할 수 없는 충분히 작은 개수의 노드가 k개 존재한다는 것을 허용함으로써, 진단 가능한 결함의 최대수를 증가시키는 알고리즘을 제안한다.

2. t/k-HYP-DIAG 알고리즘

2.1 하이퍼큐브의 t/k-진단성

n개의 장치로 구성된 시스템 S에서, t개 이하의 모든 결함장치가 명확히 결정될 수 있다면 시스템 S는 t-결함 진단 가능하다(단, $0 \le t \le n$)라고 일반적으로 정의하는데, 대부분의 병렬처리 시스템에서 얻어진 t의 값은 시스템에 있는 프로세서의 수와는 상관없이 매우 작다[2].

정의 1. 시스템은 t-진단가능 하다. 만약 테스트하는 노드에서 얻어진 테스트 결과가 주어졌을 때, 결함노드들의 수가 t개를 넘지 않는다는 조건 하에서 모든 결함노드들을 확인할 수 있다.

프로세서들 사이에 제한된 연결에도 불구하고 시스템의 진단 능력을 증가시키는 방법이 제안되었다. Somani와 Peleg는 다음과 같이 t/k-진단가

능 시스템을 제안하였다[5].

정의 2. n개의 노드들의 시스템 S는 t/k-진단 가능하다. 만약 결함의 집합 F에 대하여 시스템에서 얻어진 임의의 신드롬이 주어졌을 때, 결함노드들의 수가 t개를 넘지 않는다는 조건 하에서 모든 결함노드들은 집합 F′안에 분리될 수 있다. 여기서, 집합 F′안에는 많아야 k개의 노드가 결함이 아닐 수 있다. 즉, $|F'| \le |F| + k$이다. 또한 Somani와 Peleg는 하이퍼큐브 구조의 시스템의 t/k-진단성에 대한 다음과 같은 정리를 증명하였다.

정리 1. $t=(k+1)n-(k+1)(k+2)/2+1$에 대하여 t/k-진단 가능하다($k \le n, n>3$).

2.2 HYP-DIAG 알고리즘

HYP-DIAG알고리즘은 4개의 단계에 걸쳐 진단을 수행한다. 각 단계에서 수행되는 작업을 설명하면 아래와 같다.

단계 1 : H_n의 서브그래프 $H_k \times R_r$을 구성한다. 여기서, $r = \lfloor logn \rfloor +1$, $k=n-r$이고, R_r은 H_r 안에 있는 RGC-링이다. 시계방향으로 R_r의 모든 노드들의 테스트를 수행한다. 안전한 링을 모두 확인한다. 안전한 링의 모든 노드를 결함이 아니라고 진단한다.

단계 2 : 인접한 안전한 링의 모든 노드를 테스터로서 사용하여 모든 보호된 링에 있는 노드를 진단한다.

단계 3 : 만약 보호되지 않은 링이 존재한다면, 외래 이웃들이 모두 결함을 갖는 유일한 노드 x만을 제외하고 유일한 보호되지 않은 링의 모든 노드를 진단한다.

단계 4 : 만약 전 단계까지 진단되지 않은 노드가 있다면, 마지막으로 그 노드를 진단한다.

단계 1에서 서브그래프의 구성은 각 노드의 n비트 인덱스 중 k개의 인덱스 비트와 나머지 r개의 인덱스 비트로 분류함으로써 자동적으로 수행된다. 이때 선택된 r개의 비트들로 이루어진 서브큐브 H_r에는 RGC를 이용한 링 임베딩 방법을 이용하여 링의 임베딩이 이루어진다. 이때 전체 시스템에는 2^{n-r}개의 링이 생성되게 된다. 그 다음에 각각의 링에서는 임의의 방향으로 진단을 수행한다. 이 단계에서는 서브-링에 존재하는 결함노드의 정확한 위치를 알고자하는 것이 아니므로 한 방향으로만 진단을 수행하여, 링 내부에 결함노드의 유무만을 판별하고, 결함노드의 정확한 진단은 다음 단계에서 이루어진다. 가정에서 전체 시스템에는 최대 n개까지의 결함이 존재한다고 했고, 서브-링의 노드의 수 $2^{n-k}>n$이므로 한 방향으로의 진단 수행결과가 안전한 링으로 진단되면 서브-링 내부에는 결함노드가 존재하지 않는다고 단정할 수 있다. 따라서, 안전한 링으로 진단된 서브-링들에 대한 진단은 첫 번째 단계에서 완료되고, 추후에는 불안전한 링들에 대한 진단만을 수행한다. 즉, 첫 번째 단계에서 대부분의 노드들에 대한 진단은 완료된다. 단계 1에서 진단을 수행하는 데 걸리는 시간을 계산해 보면 홀수 인덱스의 노드들이 짝수 인덱스의 노드들을 테스트한 후 짝수 인덱스의 노드들이 홀수 인덱스의 노드들을 테스트함으로써 서브-링에 대한 진단이 완료되므로 테스트라운드 2가 소요되며, 전체의 노드들이 모두 진단에 참여하므로 2^n 테스트 횟수가 소요된다.

단계 2에서는 단계 1에서 불안전한 링으로 판명된 링들에 대한 진단이 단계 2에서 이루어지는데, 불안전한 링은 한 개 이상의 결함노드를 포함하고 있으므로 서브-링 스스로는 정확한 진단을 할 수 없다. 이때에 $k=n-r$이므로 이웃한 서브-링들의 수는 k개가 존재하게 되는데, k개의 서브-링들 중에 안전하다고 진단된 링이 존재한다면 즉, 보호받은 링이라면 각각의 노드들에 대하여 대응되는 이웃한 안전한 링의 노드들로 테스트를 수행한다. 단계 2에서는 보호받은 링에 대한 진단이 완료된다.

보조정리 2.1. 많아야 하나의 보호 받지 않은 링이 존재한다.

(증명) 두 개의 보호받지 않은 링이 존재한다고 가정하자. 각 링은 k개의 이웃한 보호 받지 않은 링과 인접해 있게 된다. 따라서 전체 시스템에는 $2k$개의 불안전한 링이 존재하게 되고, 전체 결함 노드의 개수는 $2k$를 초과하게 된다.
$2k = 2(n-r) = 2n-2\lfloor logn \rfloor -2>n$이 되므로 최대 n개의 결함노드가 존재한다는 조건에 위배된다. 따라서 많아야 하나의 보호 받지 않은 링이 존재한다.

단계 3에서는 보호 받지 않은 링에 대한 진단이 수행된다. 보호 받지 않은 링은 이웃한 링들 중에 안전한 링이 한 개도 없다. 그러나 단계 2까지 완료되고 난 후에는 보호 받지 않은 링의 이웃한 링들은 모두 진단이 완료된 상태가 된다. 따라서 보호 받지 않은 링의 각 노드들의 외래 이웃들 중에 결함이 아니라고 진단된 노드들을 가지고 각 노드들을 테스트한다.

보조정리 2.2. 만약 보호 받지 않은 링이 존재한다면, 이러한 링 안에 있는 노드들 중에서 인접한 외래 이웃들이 모두 결함을 갖는 단일 노드 x는 많아야 하나 존재한다.

(증명) 보조정리 2.1 증명과 같은 방법으로 증명한다. 두 개의 단일 노드가 존재한다고 가정을 하면 각 노드의 외래 이웃들은 모두 결함인 노드들이 된다. 전체 시스템에 $2k$개의 결함노드가 존재하기에 $2k = 2(n-r) = 2n-2\lfloor logn \rfloor -2>n$이 되므로 최대 n개의 결함노드가 존재한다는 조건에 위배된다. 따라서 많아야 하나의 단일 노드가 존재한다.

단계 4에서 단일 노드가 존재할 때 두 가지의 경우가 발생하게 된다. 첫째, 단계 3이 완료된 후 n개의 결함노드가 발견이 된 경우, 최대 n개의 결함노드만 존재한다는 조건에 의해서 단일 노드는 자동적으로 결함이 아니라고 진단된다. 두 번째, 많아야 n-1개의 결함노드가 발견이 된 경우는 n차원 하이퍼큐브의 각 노드는 n개의 이웃한 노드를 가지므로 단일 노드의 인접노드들 중에는 반드시 결함이 아닌 노드가 존재한다. 그 노드로 단일 노드에 대한 테스트를 수행한다.

정리 2. 알고리즘 HYP-DIAG는 $n \geq 9$에 대하여 최악의 경우에 많아야 $2^n + 3n/2$의 테스트를 사용하여 n-큐브의 모든 노드들을 진단한다.

보조정리 2.3 1) n-큐브에 있는 보호받은 링은 많아야 $s(\lceil logn \rceil +2)$개의 테스트를 사용하여 많아야 1라운드 이내에 진단할 수 있다. 여기서, s는 링 안에 있는 결함의 수이다. 2) n-큐브에 있는 보호받은 링은 많아야 $s(n+2)$개의 테스트를 사용하여 1라운드에 진단될 수 있다. 여기서, s는 링에 있는 결함의 수이다.

2.3 t/k-HYP-DIAG 알고리즘

HYP-DIAG 알고리즘은 하이퍼큐브의 t-진단가능 하다는 특성을 바탕으로 하여 진단할 하이퍼큐브를 모든 결함노드를 포함할 수 있는 최소 크기의 RGC링으로 분할을 한다. HYP-DIAG 알고리즘에서는 진단 가능한 최대의 결함 수가 n이라고 가정하였으므로 하이퍼큐브를 분할하는 RGC링의 크기는 $r = \lfloor \log n \rfloor +1$이다. n개를 초과하는 결함을 진단하기 위해서, t/k-시스템의 하이퍼큐프를 분할할 때 RGC링의 크기는 t개의 결함노드를 포함할 수 있게 분할해야 한다. 즉 RGC링의 크기를 $r = \lfloor logt \rfloor +1$로 변환하여야 한다. 이렇게 변환한 후 원래의 정리가 성립되지 않기 때문에 변환에 따른 새로운 정리와 알고리즘의 정확성이 증명되어야 한다.

t/k-진단가능 시스템에서 k가 커지면 불확실 진단의 개수가 늘어나면서 진단의 정확성이 떨어지게 되므로 k가 작으면서도 진단 가능한 노드의 최대수가 많이 증가되기를 원한다. 그러므로 이 논

문에서는 $k=1$로 가정하여 진단 가능한 노드 수가 $t=2(n-1)$일 경우를 연구하였다. 정리 1에서 $k=1$인 경우에 $t=2(n-1)$이므로 하이퍼큐브는 $2(n-1)/1$ 진단 가능 시스템이다.

t/k-HYP-DIAG 알고리즘은 아래 5개의 단계에 걸쳐 진단을 수행한다.

단계 1 : H_n의 서브그래프 $H_j \times R_r$을 구성한다. 여기서, $r=\lfloor logt \rfloor+1$, $j=n-r$이고, R_r은 H_r 안에 있는 RGC-링이다. 시계방향으로 R_r의 모든 노드들의 테스트를 수행한다. 안전한 링을 모두 확인한다. 안전한 링의 모든 노드를 결함이 아니라고 진단한다.

단계 2 : 인접한 안전한 링의 모든 노드를 테스터로서 사용하여 모든 보호받은 링에 있는 노드를 진단한다.

단계 3 : 만약 보호 받지 않은 링이 존재한다면, 외래 이웃들이 모두 결함인 노드를 제외한 모든 노드를 진단한다.

단계 4 : 아직 진단되지 않은 노드가 있다면 모든 이웃이 결함인 단일한 노드를 제외한 모든 노드들을 진단한다.

단계 5 : 전 단계에서 테스트 되지 않은 노드가 있다면, 마지막으로 그 노드를 진단한다.

단계 1에서 $r=\lfloor logt \rfloor+1$의 크기로 하이퍼큐브를 분할한다. 노드의 n비트 인덱스 중 j개의 인덱스 비트와 나머지 r개의 인덱스 비트로 분류함으로써 자동적으로 수행된다. 이때 선택된 r개의 비트들로 이루어진 서브큐브 H_r에는 RGC를 이용한 링 임베딩 방법을 이용하여 링의 임베딩이 이루어진다. 그 다음에 각각의 링에서는 임의의 방향으로 진단을 수행한다. 이 단계에서는 서브-링에 존재하는 결함노드의 정확한 위치를 알려고 하는 것이 아니므로 한 방향으로만 진단을 수행하여, 링 내부에 결함 노드의 유무만을 판별하고, 결함 노드의 정확한 진단은 다음 단계에서 이루어진다. 전체 시스템에는 최대 n개까지의 결함이 존재한다고 가정하였고, 서브-링의 노드의 수 $2^r>t$이므로 한 방향으로의 진단 수행 결과가 안전한 링으로 진단되면 서브-링 내부에는 결함 노드가 존재하지 않는다고 단정할 수 있다. 따라서, 안전한 링으로 진단된 서브-링들에 대한 진단은 첫 번째 단계에서 완료되고, 추후에는 불안전한 링들에 대한 진단만을 수행한다. 즉, 첫 번째 단계에서 대부분의 노드들에 대한 진단은 완료된다. 단계 1에서 진단을 수행하는데 걸리는 시간을 계산해 보면 홀수 인덱스의 노드들이 짝수 인덱스의 노드들을 테스트한 후 짝수 인덱스의 노드들이 홀수 인덱스의 노드들을 테스트함으로써 서브-링에 대한 진단이 완료되므로 테스트 라운드 2가 소요되며, 전체의 노드들이 모두 진단에 참여하므로 2^n 테스트 횟수가 소요된다.

단계 2에서는 단계 1에서 불안전하다고 진단된 링들에 대한 진단이 단계 2에서 이루어지는데, 불안전한 링은 한 개 이상의 결함노드를 포함하고 있으므로 서브-링 스스로는 정확한 진단을 할 수 없다. 이때에 $j=n-r$이므로 이웃한 서브-링들의 수는 j개가 존재하게 되는데, k개의 서브-링들 중에 안전하다고 진단된 링이 존재한다면 즉, 보호받은 링이라고 한다면 각각의 노드들에 대하여 대응되는 이웃한 안전한 링의 노드들로 테스트를 수행한다. 단계 2에서는 보호 받은 링에 대한 진단이 완료된다.

보조정리 2.4 $n \geq 12$에 대하여, 많아야 3개의 보호 받지 않은 RGC링이 존재한다.

(증명) 4개의 보호 받지 않은 RGC링이 존재한다고 가정하면, 보호 받지 않은 RGC링의 인접 링들이 불안전한 링이기 때문에 $4j$개의 불안전한 링이 존재한다. 두 노드는 많아야 2개의 공동의 이웃을 가지므로 그 중에서 중복 계산된 불안전한 링의 개수가 많아야 8개이다. 그러므로 불안전한 링은 적어도 $(4j-8)$개 존재한다. $4j-8=4(n-r)-8=4n-4(\lfloor logt \rfloor+1)-8>t$, $n \geqq 12$이되므로 최대 t개의 결

함노드가 존재한다는 조건에 위배된다. 따라서 많아야 3개의 보호 받지 않은 RGC링이 존재한다.

단계 3에서는 보호 받지 않은 링에 대한 진단이 수행된다. 보호 받지 않은 링은 이웃한 링들 중에 안전한 링이 한 개도 없다. 그러나 단계 2까지 완료되고 난 후에는 보호 받지 않은 링의 이웃한 링들은 모두 진단이 완료된 상태가 된다. 따라서 보호 받지 않은 링의 각 노드들의 외래 이웃들 중에 결함이 아니라고 진단된 노드들을 가지고 각 노드들을 테스트한다.

보조정리 2.5 만약 보호받지 않은 링이 존재한다면, 이러한 링 안에 있는 노드들 중에서 인접한 외래 이웃들이 모두 결함을 갖는 노드 x는 많아야 3개 존재한다.

(증명) 만약 인접한 외래 이웃들이 모두 결함을 갖는 노드가 4개 존재한다고 가정하면, 각각의 노드는 j개의 결함인 외래 이웃을 갖기 때문에 모두 $4j$개의 결함노드가 존재한다. 임의의 두 노드의 공동 이웃은 많아야 2개이다. 그러므로 4개의 노드 중 둘씩 서로 2개의 공동 이웃을 가진다 하더라도 중복 계산된 결함노드의 개수가 많아야 8개 이다. 그러므로 전체 결함노드 수는 적어도 $(4j-4)$이다. $4j-4 = 4(n-r)-4 = 4n-4(\lfloor \log t \rfloor +1)-4 > t,\ n \geq 12$이 되므로 인접한 외래 이웃들이 모두 결함을 갖는 노드가 4개 존재한다는 가정은 위의 보조 정리와 마찬가지로 최대 t개의 결함노드가 존재한다는 조건에 위배된다. 따라서 많아야 3개의 노드가 인접한 외래 이웃들이 모두 결함이다.

단계 4에서는 외래 이웃들이 모두 결함노드인 노드들에 대한 진단이다. 이런 노드들은 외래 이웃들이 모두 결함이여서 그들로부터 진단 받을 수 없다. 그러나 단계 3까지 완료되고 난 후에는 같은 RGC링에 있는 이웃들은 모두 진단이 완료된 상태가 된다. 따라서 같은 RGC링에 있는 이웃들 중에 결함이 아니라고 진단된 노드들을 가지고 각 노드들을 테스트 한다.

보조정리 2.6 이웃들이 모두 결함인 노드 즉 판단할 수 없는 단일 노드는 많아야 한 개이다. 즉 판단할 수 없는 노드는 많아야 한 개가 존재한다.

단계 5에서 위의 단계에서 아직 진단되지 않은 노드가 존재할 때 두 가지의 경우가 발생하게 된다. 첫째, 단계 4가 완료 후 t개의 결함노드가 발견이 된 경우, 최대 t개의 결함노드만 존재한다는 조건에 의해서 이 노드는 자동적으로 결함이 아니라고 진단된다.

두 번째, 많아야 t-1개의 결함노드가 발견이 된 경우는 1개의 진단 오류 즉 1개의 결함이 아닌 노드를 결함이라고 진단하는 것을 용납하기 때문에 이 노드를 결함이라고 판단한다. 또한 결함인지 아닌지 진단할 수 없을 경우, 결함이 아닌 노드를 결함이라고 진단하면 이 노드를 대응하는 새 노드(프로세서)로 교환하기 위하여 드는 비용은 많지 않지만 결함인 노드를 결함이 아니라고 진단하면 시스템 전체의 정확성이 보증되지 않으므로 단계 5의 진단은 효율적이다.

3. 알고리즘의 성능 분석

이 장에서는 제안된 알고리즘의 성능을 분석하기 위하여 테스트 횟수와 테스트 라운드를 계산하였다. HYP-DIAG 알고리즘과 t/k-HYP-DIAG 알고리즘의 각 단계에서 소요되는 테스트 라운드와 테스트횟수를 각각 계산한 후 합하여 전체의 성능을 분석해 보았다.

3.1 HYP-DIAG 알고리즘 성능 분석

단계 1에서는 분할된 서브큐브에 임베딩된 모든

노드들이 테스트에 참여하므로 2^n의 테스트 횟수가 소요된다. $2^n/2$의 노드들이 테스트를 수행 한 후 다음 단계에서 나머지 $2^n/2$의 노드들이 테스트를 수행하므로 2번의 테스트 라운드가 소요된다.

단계 2에서는 최악의 경우 한 개의 안전한 링이 n-r개의 보호 받은 링을 순차적으로 진단해야하므로 n-r 테스트 라운드가 소요된다. 진단에 참여하는 RGC링이 많아야 n개이므로 $2^r n$의 테스트 횟수가 소요된다.

단계 3에서는 최대 1개가 존재하는 보호 받지 않은 링에 대한 진단을 수행하므로 2^r개의 테스트 횟수가 소요된다. 보호 받지 않은 링의 노드들은 외래 이웃들에 의해서 각각 동시에 테스트되어지므로 1번의 테스트라운드가 소요된다.

단계 4에서는 단일 노드 x에 대한 테스트가 수행된다. 단계 3까지 발견된 결함의 수에 의해서 자동적으로 상태가 판별이 되거나 동일 서브-링 내의 이웃한 노드에 의해서 진단이 이루어지므로 최대 한 번의 테스트 횟수와 한 번의 테스트 라운드가 소요된다.

네 개의 단계에서 소요되는 모든 테스트 횟수와 테스트 라운드의 총합을 구하면 아래와 같다.

테스트 횟수 =

$$2^n+2^r n+2^r+1 = 2^n+(n+1)2^r+1 \quad (1)$$

테스트 라운드 =

$$2+(n\text{-}r)+1+1 = n\text{-}r+4 \quad (2)$$

3.2 t/k-HYP-DIAG 알고리즘 성능 분석

단계 1에서는 HYP-DIAG와 마찬가지로 분할된 서브큐브에 임베딩된 모든 노드들이 테스트에 참여하므로 2^n의 테스트 횟수가 소요된다. $2^n/2$의 노드들이 테스트를 수행한 후 다음 단계에서 나머지 $2^n/2$의 노드들이 테스트를 수행하므로 2번의 테스트 라운드가 소요된다.

단계 2에서는 최악의 경우 한 개의 안전한 링이 n-r개의 보호 받은 링을 순차적으로 진단해야 하므로 n-r 테스트 라운드가 소요된다. 진단에 참여하는 RGC링이 많아야 t개 이므로 $2^r t$의 테스트 횟수가 소요된다.

단계 3에서는 최대 3개가 존재하는 보호 받지 않은 링에 대한 진단을 수행하므로 $3 \cdot 2^r$개의 테스트횟수가 소요된다. 보호 받지 않은 링의 노드들은 외래 이웃들에 의해서 각각 동시에 테스트되어지므로 한번의 테스트 라운드가 소요된다.

단계 4에서는 외래 이웃들이 모두 결함인 노드들에 대한 테스트가 수행된다. 이런 노드가 많아야 세 개이기에 최대한 세 개의 테스트 횟수와 한번의 테스트 라운드가 소요된다.

단계 5에서는 이웃들이 모두 결함인 노드인 유일한 노드에 대한 진단인데 앞 단계에서의 테스트 결과로 판단하므로 테스트 횟수와 테스트 라운드를 소요하지 않는다.

다섯 개의 단계에서 소요되는 모든 테스트 횟수와 테스트 라운드의 총합을 구하면 아래와 같다.

테스트 횟수 =

$$2^n+2^r t+3\cdot 2^r+3 = 2^n+(t+3)2^r+3 \quad (3)$$

테스트 라운드 =

$$2+(n\text{-}r)+1+1 = n\text{-}r+4 \quad (4)$$

두 알고리즘의 성능을 비교하면 다음과 같다.

테스트 라운드의 차 = (3)-(1) = **0**

테스트 횟수의 차 = (4)-(2) = $(t\text{-}n+2)2^r+2$

위에서 얻어진 결과를 바탕으로 두 알고리즘간의 성능을 비교하면 식 (1)과 식 (3)은 같다는 것을 알 수 있고 각 알고리즘의 테스트 횟수 차는 $(t-n+2)\,2^r+2$이다. 그러므로 t/k-HYP-DIAG 알고리즘은 HYP-DIAG 알고리즘의 우수한 성능을 유지하면서 더 많은 결함을 진단할 수 있다.

〈표 3.1〉 n =10과 n =20의 하이퍼큐브 t/k-진단성

k	$n=10$	$n=20$
Diag-k	t	t
1	18	38
2	25	55
3	31	71
4	36	86
5	40	100
6	43	113
7	45	125
8	46	136
9	46	146
10	45	155
11	-	163
12	-	170
13	-	176
14	-	181
15	-	185
16	-	188
17	-	190
18	-	191
19	-	191
20	-	190

$n=10$일 경우와 $n=20$일 경우의 하이퍼큐브의 t/k-진단성은 다음과 같다. <표 3.1>로부터 t/k-진단시스템을 사용하면 최대 진단 가능 결함의 수가 많이 증가됨을 알 수 있다. $n=10$일 경우를 보면 원래의 최대 진단 가능한 결함의 개수가 10으로부터 4배 정도 증가 되었고, $n=20$일 경우는 9배 정도 증가 되었다.

그러나 무조건 n이 커진다고 좋은 것은 아니다. 최대 진단 가능 결함의 수를 증가함에 따라 허용 부정확 진단 노드 수도 증가 되므로 시스템 비용의 낭비를 초래할 수 있다. k가 일정한 수로 증가하면 최대 진단 가능 결함의 수 t가 따라서 증가하지 않거나 k에 비하여 너무 적게 증가 한다. 이럴 경우에는 시스템 비용을 효율적으로 사용하기 위해서 k를 더 증가 시키지 않는 것이 좋다. 이 논문에서 제안한 알고리즘은 HYP-DIAG 알고리즘의 우수한 시간복잡도를 유지하면서 t/k-진단시스템을 하이퍼큐브에서 적용되게 하였다.

3.3 HADA 알고리즘과의 비교

t/k-HYP-DIAG 알고리즘과 HADA 알고리즘의 성능을 비교하면 <표 3.2>와 같다. 하이퍼큐브의 차원이 커질수록 테스트 라운드 수와 테스트 횟수를 비교하면 t/k-HYP-DIAG 알고리즘이 훨씬 적다.

〈표 3.2〉 HADA와 t/k-HYP-DIAG의 성능 비교

Algorithm	Dimension	3	4	5	6	7	8	9	10
t/k-HYP	#rounds	3	4	6	8	8	10	9	10
HADA	#rounds	7	8	9	10	11	12	13	14
t/k-HYP	#test	12	24	48	96	192	384	612	1145
HADA	#test	24	64	128	256	512	1280	2560	5120

Algorithm	Dimension	11	12	13	14	15	16
t/k-HYP	#rounds	11	10	11	9	9	10
HADA	#rounds	15	16	17	18	19	20
t/k-HYP	#test	2192	4265	8388	16609	33024	65825
HADA	#test	10240	20480	40960	81920	163840	393216

4. 결 론

시스템-레벨 진단알고리즘은 병렬시스템에서 중요한 연구 분야이다. 또한 하이퍼큐브 구조는 계층적이며 정규적인 특성을 가지고 있기 때문에 병렬처리 시스템에 많이 이용되고 있다.

기존의 하이퍼큐브 진단 알고리즘은 진단 가능한 노드의 수가 n보다 작거나 같다는 가정을 하고 있다. Kranakis와 Pelc는 전체 하이퍼큐브 H_n을 결함을 모두 포함할 수 있는 서브 링을 하나의 노드로 하는 새로운 하이퍼큐브 $H_j \times R_{n-j}$로 분할하여 진단을 수행하는 알고리즘 HYP-DIAG를 제안하였다. 그러나 이 알고리즘 역시 진단 가능한 노드의 갯수가 최대로 하이퍼큐브 차원수 n이라는 단점을 갖고 있다. Somani와 Peleg는 진단의 정확여부를 판단할 수 없는 노드의 존재를 허용함으로서 진단 가능한 결함의 최대 수를 증가하는 t/k-진단

가능 시스템을 제안하고 하이퍼큐브가 *t/k*-진단가능 시스템이라는 사실을 증명하였다. 이 논문에서는 HYP-DIAG 알고리즘을 바탕으로 *t/k*-진단가능 시스템의 개념을 적용하여 결함노드 수 $t>n$ 일 경우의 Over-d 문제를 해결하는 *t/k*-HYP_DIAG 알고리즘을 제안하였다. 제안하는 알고리즘은 진단 가능한 최대의 노드 수를 증가하는 대신에 소량의 부정확 진단을 허용함으로써 HYP-DIAG 알고리즘의 효율성을 이용하고 결함의 수가 많을 수 있는 대형 멀티프로세서에서도 이용 가능하게 하였다. $k=1$일 경우 $2n-2$개의 결함을 진단할 수 있게 함으로써 적은 양의 테스트로 비교적 우수한 진단능력을 보여준다. 향후 연구 방향은 시스템-레벨의 가정의 제약점을 보다 완화하고 분산적인 환경에서 수행되는 네트워크 신뢰성 향상에 대한 연구이다.

참 고 문 헌

[1] C. Feng, L. N. Bhuyan, and F. Lombardi, "Adaptive System-Level Diagnosis for Hypercube Multiprocessors", IEEE Trans. Computers, Vol. 45, No. 10, pp. 1157-1170, Oct. 1996.

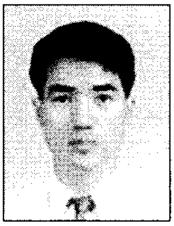
[2] A. K. Somani, V. K. Agarwal, and D. Avis, "A Generalized Theory For System Level Diagnosis", IEEE Trans. Computers, Vol. 36 No. 5, pp. 538-546, May 1987.

[3] A. D. Friedman, "A New Measure of Digital System Diagnosis", Proc. Fifth Int'l Symp. Fault-Tolerant Computing, pp. 167-170, 1975.

[4] F. P. Preparata, G. Metze, and R. T. Chien, "On the Connection Assignment Problem of Diagnosable Systems", IEEE Trans. Electronic Computers, No. 12, pp.848-854, Dec. 1967.

[5] E. Kranakis and A. Pelc, "Better Adaptive Diagnosis of Hypercubes", IEEE Trans. Computers, Vol. 49, No. 10, pp. 1013-1020, Oct. 2000.

[6] A. K. Somani and O. Peleg, "On Diagnosability of Large Fault Sets in Regular Topology-Based Computer Systems", IEEE Trans. Computers, Vol. 45. No. 8, pp. 892-903, Aug. 1996.



김 장 환

1980년 서울대학교 경제학학사
1997년 한국과학기술원 전산학 석사
2003년 충북대학교 전산학박사
1984년~1988년 쌍용정보통신 연구원
1988년~1993년 Qnix Data System 연구원
1993년~1998년 SK Telecom 중앙연구원 연구원
1998년~2005년 대덕대 교수
2005년~현재 성결대 공대 교수
관심분야 : Information Security, Mobile & Wireless Communication, Performance Analysis of Networks, Database System, Mobile Multimedia, Mobility Managements, Mobile Embedded System, Ubiquitous Computing, 알고리즘 및 계산이론, 결함허용, 정보통신 경제 예측

이 충 세

1979년 Univ. of South Carolina 컴퓨터과학과 석사
1990년 Univ. of South Carolina 컴퓨터 과학과 박사
Univ. of North Dakota 전산학과 조교수
1991년~현재 충북대학교 컴퓨터과학과 교수
관심분야 : 결함허용, 알고리즘, 전문가시스템, 정보보안