서브-링의 신드롬 분석을 이용한 하이퍼큐브 진단 알고리즘

김학원[0] 김동군 최문석 이충세
충북대학교 전자계산학과
{hakwon, tnt2002, bidulgia}@algo.chungbuk.ac.kr ,csrhee@cbucc.chungbuk.ac.kr

Hypercube Diagnosis Algorithm Using Syndrome Analysis of Sub-Ring

Hak-Won Kim[0] Dong-Kun Kim Moon-Seok Choi Chung-Sei Rhee
Dept. of Computer Science, Chungbuk National University

요 약

하이퍼큐브의 정규적이며 계층적인 구조적 특성은 효율적인 진단 알고리즘 개발에 유리하게 적용될 수 있다. Feng et al.의 HADA/IHADA와 Choi와 Rhee의 적응적 큐브 분할 방법은 하이퍼큐브의 전체 노드를 하나의 링으로 임베딩하여 링의 진단 특성을 이용하기 위하여 분할 및 정복 방법을 이용하였다. 또한 Kranakis와 Pelc는 결함을 모두 포함하는 최소의 서브링을 하나의 노드로 하는 하이퍼큐브의 형태로 분할하는 HYP-DIAG 알고리즘을 제안하였다. 또한 최악의 경우에, 테스트 수만을 고려하여 $2^n+3n/2$의 테스트 수를 갖는 FAST-HYP-DIAG 알고리즘과 병렬 시간만을 고려하여 많아야 11테스트 라운드 이내에 진단을 수행하는 EXPRESS-HYP-DIAG 알고리즘을 제안하였다. 본 논문에서는 HYP-DIAG의 첫 번째 단계에서 얻어진 서브링들의 신드롬을 분석하여 테스트의 수와 테스트 라운드를 모두 고려하는 알고리즘을 제안한다.

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. 결함진단 문제로 알려져 있는 이런 분야에서 가장 중요한 문제 중의 하나가 시스템 내에 있는 모든 결함 프로세서의 위치를 정확하게 확인하는 것이다. 결함을 진단하는 전통적인 방법은 Preparata et al.에 의해서 제안되었다[1]. 각 프로세서는 시스템내에 있는 자신과 이웃한 프로세서들의 테스트를 수행하고 진단은 이들의 테스트 결과를 분석하여 수행된다. 결함이 아닌 프로세서들은 항상 정확한 결과를 주는 반면에 결함이 있는 프로세서들의 테스트 결과는 신뢰할 수 없다. 즉, 테스트를 받는 프로세서의 상태와는 상관없이 임의의 테스트 결과를 준다. 결함은 영구적이며 프로세서들의 결함 상태는 테스트를 수행하고 진단하는 동안 변하지 않는다고 가정한다. [1]에서 이용된 최악의 경우 시나리오로써 시스템에는 많아야 t개의 프로세서들이 결함이며 이들은 진단하기 가장 어려운 위치에 있다고 가정한다.

다양한 병렬처리 시스템 중에서 정규적이며 계층적인 구조를 갖는 하이퍼큐브 형태의 병렬처리 시스템은 Intel iPSC[2]와 nCUBE[3] 등에서 이미 상용화 되어있다. 또한, 하이퍼큐브의 정규성과 계층성은 효율적인 진단 알고리즘을 개발하는데 유리하게 적용될 수 있다. Feng et al.에 의해 제안된 HADA(Hypercube Adaptive Diagnosis Algorithm)와 IHADA(Improved HADA)는 하이퍼큐브의 구조적인 특징을 이용하였다[4]. HADA와 IHADA는 RGC(Reflected Gray Code)[5]를 이용하여 하이퍼큐브를 링에 임베딩을 한 후, Vaidya와 Predhan에 의해 증명된 링의 1-FL-(|R|-2)-FD 성질[6]과 분할 및 정복 방법을 이용하여 결함 진단을 수행한다. Choi와 Rhee는 HADA와 IHADA의 방법과 다르게 링을 분할하기 위하여 신드롬을 분석하여 분할하는 차원을 선택하는 적응적 큐브 분할 방법을 이용한 진단 알고리즘을 제안하였다[7]. HADA/IHADA와 적응적 큐브 분할 방법은 하이퍼큐브의 모든 노드를 하나의 링에 임베딩 하는 반면에 Kranakis와 Pelc는 전체 하이퍼큐브 H_n을 $H_k \times H_{n-k}$로 분할하여 진단을 수행하는 알고리즘 HYP-DIAG를 제안하고, 또한 HYP-DIAG을 수정하여 최악의 경우에 n차원 하이퍼큐브에 대하여 $2^n+O(n\log n)$의 테스트 수를 사용하여 $O(\log n)$의 테스팅 라운드 안에 진단하는 알고리즘 FAST-HYP-DIAG와 $2^n+(n+1)^2$의 테스트 수를 사용하여 많아야 11번의 테스팅 라운드 안에 진단하는 알고리즘 EXPRESS-HYP-DIAG를 제안하였다[8]. FAST-HYP-DIAG는 테스트 수만을 고려한 알고리즘이고, EXPRESS-HYP-DIAG는 테스팅 라운드만을 고려한 알고리즘이다.

본 논문에서는 알고리즘 HYP-DIAG의 첫 번째 단계에서 테스트된 링들의 결과를 분석하여 테스팅 라운드와 테스트 수를 모두 고려하였다. 2장에서는 관련 용어를 정의하고, 3장에서는 알고리즘 HYP-DIAG과 수정된 알고리즘 FAST-HYP-DIAG와 EXPRESS-HYP-DIAG에 대하여 설명한다. 4장에서는 알고리즘 HYP-DIAG에서 얻어진 링들의 신드롬을 분석하여 분할하는 차원을 선택하는 방법을 이용한 진단 알고리즘 Enhanced-HYP-DIAG을 제안하며, 마지막으로 5장에서는 결론을 맺는다.

2. 관련 용어 정의

병렬처리 시스템은 방향성 그래프 $G(V,E)$로 표현되며, 여기서 V는 시스템 내의 각 프로세서를 나타내고, E는 프로세서간의

테스트 수행 관계를 나타낸다. 테스트를 수행하고 난 후에 각 간선은 0이나 1의 값을 갖게 된다. 즉, 간선 (u,v)가 존재하면 프로세서 u가 프로세서 v를 테스트할 수 있음을 의미한다. 간선의 값이 0일 경우에는 프로세서 v가 프로세서 u에 의해 결함이 없는 노드로 테스트되었음을 의미하고, 간선의 값이 1일 경우는 프로세서 v가 프로세서 u에 의해 결함이 있는 노드로 테스트되었음을 의미한다. 또한 각 간선이 테스트를 수행한 후에 갖는 값들의 집합을 신드롬이라 한다. 결과적으로, 신드롬을 함수 $S : A \rightarrow \{0, 1\}$로 표현할 수 있다. 시스템에 있는 모든 결함 프로세서들의 집합을 결함 집합(*faulty set*)이라 한다.

임의의 정수 n에 대하여, n차원 하이퍼큐브(n-큐브)는 그래프 $H_n = (V_n , E_n)$으로 표현한다. 여기서, V_n은 길이가 n인 이진수의 집합이고, E_n은 $\{(u, v) : u, v \in V$이고, u와 v는 정확히 하나의 위치만 다르다$\}$인 집합이다. n-큐브는 각 노드가 차수 n을 갖는 2^n개의 노드로 구성된다. 결과적으로, n-큐브에서 진단될 수 있는 노드의 수는 최대 n이다. 본 논문에서는 많아야 n개의 결함이 있는 것으로 가정한다. n-큐브의 비적응적 진단은 $2^n n$개의 테스트를 수행하여야 한다. 즉, 양방향으로 n-큐브의 모든 간선에 대하여 테스트가 행되어야 한다[9]. 다른 방법으로, 2^n개의 노드와 많아야 n개의 결함을 갖는 시스템의 모든 적응적 진단은 적어도 2^n+n-1개의 테스트를 수행해야 한다[10]. 그래서, 2^n+n-1는 임의의 n-큐브를 진단하는 테스트 수에 대한 이론적인 하한 값이다.

$n>1$에 대한 n-큐브는 해밀턴 그래프가 존재한다. 즉, 이것은 모든 노드들을 포함하는 원소들의 사이클이며, RGC(Reflected Gray Code)를 사용하여 효율적으로 구성될 수 있다[5]. 이런 사이클을 RGC-링이라 한다. n-큐브는 $0<k<n$에 대하여 두 그래프의 곱 형태인 $H_k \times H_{n-k}$로 표현될 수 있으며, 그래프 $H_k \times R_{n-k}$로 임베딩 할 수 있다. 여기서, R_{n-k}는 H_{n-k}에 있는 노드로 구성된 RGC-링이다. 이와 같은 임베딩 방법은 다음의 알고리즘에서 중요한 역할을 한다. 즉, n-큐브 H_n은 그래프 R_{n-k}를 하나의 노드로 하는 2^k개의 서로 다른 그래프로 이루어진 k-큐브 H_k를 구성할 수 있다. H_k와 인접한 노드에 대응하는 링들을 *adjacent*라고 한다. 링 R에 있는 임의의 노드 v에 대하여, R에 인접하는 서로 다른 링 속에 있는 v에 대한 k개의 이웃들을 v의 *foreign neighbors*라 한다. 알고리즘은 R_{n-k}에 있는 노드를 테스트 할 때, 한쪽 방향(시계방향)으로 모든 노드들의 테스트를 수행한다. 이것은 한쪽 방향의 테스트 결과만으로 각각이 링이 결함 노드를 포함하고 있는지 아닌지를 확인할 수 있음을 의미한다. 얻어진 신드롬 속에 1을 하나도 포함하고 있지 않은 링들을 *healthy*라 하고 그렇지 않으면 *unhealthy*로 부를 것이다. 만약 *unhealthy* 링이 *healthy* 링과 인접한다면 *guarded*이고 그렇지 않으면 *unguarded*이다.

3. HYP-DIAG 알고리즘

먼저 알고리즘의 일반적인 개념을 설명한다. $n \geq 9$이고 $r=\lfloor \log n \rfloor +1$이라 하자. 그러므로, $2^r > n$이다. $k=n-r$라 놓고, H_n의 서브그래프 $H_k \times R_r$을 생각해 보자. 여기서 R_r은 H_r의 RGC-링이다.

보조정리 3.1. 많아야 하나의 *unguarded* 링이 존재한다.

보조정리 3.2. 만약 *unguarded* 링이 존재한다면, 이런 링 안에 있는 노드들 중에서 인접한 *foreign neighbors*이 모두 결함을 갖는 노드 x는 많아야 하나 존재한다.

알고리즘 HYP-DIAG

단계 1. H_n의 서브그래프 $H_k \times R_r$을 구성한다. 여기서, $r=\lfloor \log n \rfloor +1$, $k=n-r$이고, R_r은 H_r 안에 있는 RGC-링이다. 시계방향으로 R_r의 모든 노드들의 테스트를 수행한다. *healthy* 링을 모두 확인한다. *healthy* 링의 모든 노드를 *fault-free*로 진단한다.

단계 2. 인접한 *healthy* 링의 모든 노드를 테스터로서 사용하여 모든 *guarded* 링에 있는 노드를 진단하다.

단계 3. 만약 *unguarded* 링이 존재한다면, *foreign neighbors*가 모두 결함을 갖는 유일한 노드 x만을 제외하고 유일한 *unguarded* 링의 모든 노드를 진단한다.

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

알고리즘 HYP-DIAG의 단계2와 3은 [11]의 *few tests*를 이용하여 진단을 수행한다.

정리 1. 알고리즘 HYP-DIAG는 $n \geq 9$에 대하여 최악의 경우에 많아야 $2^n+3n/2$의 테스트를 사용하여 n-큐브의 모든 노드들을 진단한다.

보조정리 3.3 1) n-큐브에 있는 *guarded* 링은 많아야 $s(\lceil \log n \rceil +2)$개의 테스트를 사용하여 많아야 1라운드 이내에 진단할 수 있다. 여기서, s는 링 안에 있는 결함의 수이다. 2) n-큐브에 있는 *guarded* 링은 많아야 $s(n+2)$개의 테스트를 사용하여 1라운드에 진단될 수 있다. 여기서, s는 링에 있는 결함의 수이다.

보조정리 3.4 $n \geq 12$, $r=\lfloor \log n \rfloor +1$이고, $k=n-r$라 놓고, H_n의 서브그래프를 고려하자 여기서 R_r은 H_r의 RGC 서브-링이다. 모든 *healthy* 링이 많아야 7개의 *unhealthy* 링에 할당되는 경우 하나를 제외하고 H_n에 있는 모든 *unhealthy* 링에 인접한 *healthy* 링을 할당하는 것이 가능하다.

보조정리 3.4로부터 예외적인 *unhealthy* 링이 존재한다면, 유일한 *unguarded* 링이거나 *unguarded* 링이 없다면 *guarded* 링이라는 것을 의미한다. 이런 링을 *special*이라 하고 다른 *unhealthy* 링을 *normal*이라 한다.

4. Enhaced-HYP-DIAG 알고리즘 제안

n-큐브 안에 있는 두 노드들 사이의 해밍 거리를 *distance*라 하며, 노드 v_i와 v_j사이의 *distance*는 $d(v_i, v_j)$로 표현한다. 만약 $d(v_i, v_j)=1$이면, 두 노드는 *neighbor* 또는 *adjacent*라고 한다. 또한 $i=1, 2, ..., m$에 대하여 $d(v_i, v_j)=1$이면, 노드 v_j는 노드 v_1, ..., $v_m (m>1)$의 *common neighbor*이라고 한다[12].

보조정리 4.1 $G(V, E)$는 n-큐브의 그래프이고, $v_i, v_j \in V$라 놓으면, 다음을 만족한다.

1) 만약 $d(v_i, v_j) \neq 2$이면, 두 노드 사이에는 *common neighbor*

이 존재하지 않는다.
2) 만약 $d(v_i, v_j)=2$이면, 두 노드 사이에는 정확하게 2개의 *common neighbor*가 존재한다.

보조정리 4.2 $G(V, E)$는 n-큐브의 그래프이고, $d(v_i, v_j)=d(v_j, v_k)=d(v_i, v_k)=2$를 만족하는 $v_i, v_j, v_k \in V$를 선택한다. 이런 3개의 노드는 적어도 두 노드에 *common neighbor*가 되는 단지 4개의 노드를 갖는다.

보조정리 4.3. $G(V, E)$는 n-큐브의 그래프이고, $|V'|=2$를 갖는 $V' \subset V$를 선택한다. 그러면, $|V'| \geq 2n-2$이다.

보조정리 4.4. $n \geq 10$, $r=\lfloor \log n \rfloor+1$이고, $k=n-r$라 놓고, H_n의 서브그래프를 고려하자 여기서 R_r은 H_r의 RGC 서브-링이다. H_n은 많아야 $g=n+4$개의 RGC-링이 결함을 포함한다.

보조정리 4.5. $G(V, E)$는 n-큐브의 그래프이고, $|F_1|=|F_2|=|V_1|=|V_2|=2^{n-2}$를 만족하는 F_1, F_2, V_1, $V_2 \subset V$를 선택하면 다음을 만족한다. 여기서, F_i, $i=1$, 2는 적어도 $2^{n-2}-n$개의 *fault-free* 노드를 포함하고 있는 *fault set*이다. f_1과 f_2는 각각 F_1과 F_2에 있는 결함의 수이고, g는 적어도 하나의 결함을 포함하고 있는 링의 수이다.
1) 만약 $f_1=n(f_2=n)$이면 V_1, V_2와 $F_1(F_2)$은 모두 *fault-free set*이다.
2) 만약 $f_1+f_2=n$이고 $d(F_1, F_2)=2$(또는 $d(F_1, F_2)=1$)이면 V_1과 V_2는 모두 *fault-free set*이다.
3) 만약 $f_1+f_2=g$, $1 \leq g<n$이면 $|V_1|+|V_2|=n-g$이다.

알고리즘 Enhaced-HYP-DIAG
단계 1. HYP-DIAG의 단계 1과 동일하며, 또한 얻어진 링들의 신드롬을 분석하여 보조정리 4.5를 만족하는 네 개의 집합으로 나눈다.
단계 2. *fault-free* 링들의 집합을 테스터로 사용하여 *fault-free*가 아닌 이웃한 링들의 집합을 진단한다.
단계 3. 아직 진단이 안된 링이 존재한다면 각각의 집합에 있는 *fault-free* 링으로 진단한다.
단계 4. HYP-DIAG의 단계 4와 동일하다.

단계 2와 3은 [11]의 *few tests*를 이용하여 진단을 수행한다. 만약, 보조정리 4.5의 1)과 2)를 만족한다면 $n\lceil n/2 \rceil$를 이용하여 1라운드만을 수행하고, 3)의 경우라면 $n(\lceil n/2 \rceil+1)$의 테스트 수를 사용하여 많아야 4라운드 이내에 진단을 수행한다.

5. 결론

HADA/IHADA와 적응적 큐브 분할 방법은 하이퍼큐브의 모든 노드를 하나의 링에 임베딩
한 후에 링의 진단 특성을 이용하여 최소한 한 개 이하의 결함을 갖는 서브링으로 분할을 한다. 반면에 Kranakis와 Pelc는 전체 하이퍼큐브 H_n을 결함을 모두 포함할 수 있는 서브링을 하나의 노드로 하는 새로운 하이퍼큐브 $H_k \times R_{n-k}$로 분할하여 진단을 수행하는 알고리즘 HYP-DIAG를 제안하였다. 또한 HYP-DIAG을 수정하여 최악의 경우에 n차원 하이퍼큐브에 대하여 $2^n+O(n\log n)$의 테스트 수를 사용하여 $O(\log n)$의 테스팅 라운드 안에 진단하는 알고리즘 FAST-HYP-DIAG와 $2^n+(n+1)^2$의 테스트 수를 사용하여 많아야 11번의 테스팅 라운드 안에 진단하는 알고리즘 EXPRESS-HYP-DIAG를 제안하였다[8].
본 논문에서는 알고리즘 HYP-DIAG의 첫 번째 단계에서 테스트된 링들의 신드롬을 분석하여 테스팅 라운드와 테스트 수를 모두 고려하는 Enhanced-HYP-DIAG를 제안하였다. 제안하는 알고리즘은 최악의 경우 $2^n+n(\lceil n/2 \rceil+1)$의 테스트 수를 사용하여 많아야 7번의 테스팅 라운드를 사용한다. HYP-DIAG 알고리즘에서 첫 번째 단계에서 얻어진 링들의 신드롬을 분석하여 결함이 있는 링들의 집합과 결함이 없는 링들의 집합으로 적절히 나눈다면 테스트의 수와 테스팅 라운드를 모두 줄일 수 있다.

참고문헌

[1] 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.
[2] J. Rattner, "Concurrent Processing: A New Direction in Scientific Computing", *AFIPS Conf. Proc.*, pp. 157-166, 1985.
[3] "nCUBE 2 Processor Manual", nCUBE Corp., 1990.
[4] 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.
[5] D.P. Bertsekas and J.N. Tsilsiklis, *Parallel and Distributed Computation, Numerical Methods*, Prentice-Hall Int'l, 1989.
[6] N.H. Vaidya and D.K. Pradhan, "Safe System Level Diagnosis", *IEEE Trans. Computers*, vol. 43. no. 3, pp. 367-370, Mar. 1994.
[7] 최문옥, 이충세,"적응적 큐브 분할을 이용한 하이퍼큐브 진단 알고리즘", *정보과학회논문지*, 제 27권, 제 4호, pp. 431-439, 2000년 4월.
[8] E. Kranakis and A. Pelc, "Better Adaptive Diagnosis of Hypercubes", *IEEE Trans. Computers*", vol. 49, no. 10, pp. 1013-1020, Oct. 2000.
[9] J. Armstrong and F. Gray, "Fault Diagnosis in a Boolean n-Cube Array of Microprocessors", *IEEE Trans. Computers*, vol. 30, pp. 587-590, 1981.
[10] P.M. Blecher, "On a Logical Problem", *Discrete Math.*, vol. 43, pp. 107-110, 1983.
[11] A. Pelc, and E. Upfal, "Reliable Fault Diagnosis with Few Tests", *Combinatorics, Probability & Computing*, vol. 7, pp. 323-333, 1998.
[12] 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.