DBpia

論文 98-23-6-15

일반적 네트워크에서의 결함허용 시스템 구성 알고리즘에 관한 연구

正會員 문 윤 호*, 김 병 기**

A Study on Fault-Tolerant System Construction Algorithm in General Network

Yun-Ho Moon* and Byung-Ki Kim** *Regular Members*

요 약

시스템의 신뢰성은 디지탈 컴퓨터 시대가 시작된 이래 가장 중요한 관심사가 되어 왔다. 신뢰성을 증가시키는 가장 최근 방법 중 하나는 결함허용 시스템을 고안하는 것이다. 본 논문은 일반적 그래프 형태에 대해 결함허용 시스템의 구성 방법을 제안한다. 이 시스템은 여러 개의 예비 노드를 가진다. 최근까지 결함허용 시스템 고안은 루프 형태와 트리 형태의 네트워크에서만 적용되어 왔다. 그러나 그것들은 매우 제한적인 경우이다. 본문의 새로운 알고리즘은 그러한 많은 제약을 가지지 않고서 어느 유형이든 적용될 수 있는 융통성을 갖도록 시도되었다. 이 알고리즘은 여러 단계로 구성되는데 최소 직경 스패닝 트리의 검출 단계, 최적 노드 결정 단계, 원래의 연결성 복구 단계 및 최종적으로 중복 그래프의 구성 단계이다.

ABSTRACT

System reliability has been a major concern since the beginning age of the electronic digital computers. One of the modest ways of increasing reliability is to design fault-tolerant system. This paper propose a construction mechanism of fault-tolerant system for the general graph topology. This system has several spare nodes. Up to date, fault-tolerant system design is applied only to loop and tree networks. But they are very limited cases. New algorithm of this paper tried to have a capability which can be applied to any kinds of topologies without such a many restriction. The algorithm consist of several steps : minimal diameter spaning tree extraction step, optimal node decision step, original connectivity restoration step and finally redundancy graph construction step.

I. 서 론

초기에 진공관으로 출발한 컴퓨터 시스템이 점차 발전하면서 시스템의 신뢰도에 대한 관심이 더욱 증가하게 되었고 컴퓨터 산업 전반에 걸쳐 자연히 신뢰성 있는 시스템의 구성을 위한 움직임이 활발히 추진되어 왔다[3]. 이러한 노력은 오늘날까지 계속되고 있으며 그 중에도 최근에 활발히 연구되는 분야 중의 하나가 결함허용에 대한 연구이다. 이 방법은 시스템 내의 어떤 노드에 결함이 발생하는 경우 그 시스템이 자체적으로 결함의 검사와 위치 확인을 행하게 하여 어느 한계까

* 숭실대학교 전산학과
** 숭실대학교 정보과학대학
論文番號 : 97142-0429
接受日字 : 1997年 4月 29日

www.dbpia.co.kr

지는 결함을 감당할 수 있도록 하자는 것이다[2]. 결함허용이란 본래의 알고리즘을 시스템 구성요소들의 고장에 관계없이 정확하게 수행할 수 있는 능력을 의미한다. 따라서 결함허용 시스템 구성의 목적은 결함이 발생했을 때라도 만족스러운 방식으로 수행할 수 있는 컴퓨터 시스템을 개발하여 실용화하는 것이라고 하겠다[6]. 결함허용 컴퓨터는 인공위성, 우주선, 산업기지의 제어,통신장비 등을 비롯한 군사 및 우주산업분야에서 주로 사용되어왔으나 점차 일반사무처리 및 공장자동화 분야로 사용범위가 일반화되고 있으며 또한 시스템 유지 및 보수 비용의 증가현상은 유용성과 신뢰도를 증대시켜 줄 수 있는 결함허용시스템에 대한 요구를 더욱 커지게 하는 요인이 되고 있다[6]. 본 연구에서는 일반적인 그래프의 형태로 구성된 멀티컴퓨터 또는 멀티프로세서 시스템을 위한 결함허용 시스템의 구축방식을 제안하였다. 이 방법의 중심 개념은 중복(redundancy[12])이며 본문에서도 예비노드를 포함하는 중복그래프를 구성하여 결함에 대비하는 방식을 사용했다. 본문의 내용 2장에서는 본 연구를 위해 필요한 기초지식과 결함허용을 위한 제반 개념들을 간단히 소개하며 3장에서는 일반적 그래프 형태의 시스템을 위한 결함허용 시스템 구축 방법을 제안하고 4장에서 그에 대한 결론 및 추후 방향을 논의하게 된다.

Ⅱ. 이론적 배경

2.1 결함관련용어 및 검사 방법의 개념

우선 본문에서의 이해를 돕기 위해 다음과 같은 용어들의 개념을 간단히 살펴보기로 한다.

fault - 시스템을 error 상태로 만들 수 있는 결함으로 failure의 근본적인 원인이 되기도 한다.

error - 자원의 어느부분이 바람직하지 못한 상태에 있을 경우를 말하며 이를 교정하기 위한 행 동을 취하지 않으면 failure가 발생할 수 있다.

failure - 시스템이 본래의 기능대로 동작하지 않는 것이 외부에 인식되었을 때를 뜻하며 사용자의 입장에서는 기대된 서비스가 중단된 상태이다[10, 14].

만일 시스템에서 failure가 발생했을 경우에는 다음과 같은 세단계의 처리 과정을 거치는 것이 일반적이다. 첫째, 결함검사 및 진단단계로 고장난 구성요소의 위치를 찾는다[16]. 둘째, 시스템 보수 및 재구성 단계로 고장 요소로 인한 시스템 전체의 손실을 방지하기 위해 남아 있는 시스템 구성 요소를 재배치하고 고장난 구성 요소를 물리적 혹은 논리적으로 제거한다. 재배치작업은 시스템 구성요소의 물리적 재구성이나 시스템 안에 있는 다른 프로세서의 작업부담에 대한 논리적 재분배로 이루어진다. 셋째, 시스템의 회복 단계로 시스템을 고장 발생 이전의 상태로 복귀시키기 위하여 시스템 내의 작업과 데이터를 회복시켜 준다[15].

본문에서 논의되는 결함(fault)발생 시 그 발생 여부를 검사하기 위한 방법으로 자체검사(self-test)와 이웃검사(neighbor-test)가 있는데 그에 대한 간단한 설명은 다음과 같다. 자체검사(self-test)는 결함진단 요청을 위하여 시스템 구성 요소에 포함되어 있는 검사 프로그램으로 검사결과는 음수(negative)나 양수가 되는데 음수이면 모듈은 정지상태가 되어 결국 응답불능의 상태(비신뢰성)를 의미하고 양수이면 응답가능한 신뢰상태를 의미한다. 이웃검사(neighbor-test)는 자체검사를 수행하기 위한 명령에 의해 자체검사와 함께 시작되는데 시스템의 각 구성요소들이 인접해 있는 노드들을 규칙적으로 살피면서 상호연결에 따른 어떤 결함이 있는지를 조사해 보는 것이다. 이 검사시에는 모든 활동적인 비결함(fault-free) 노드(Xi)가 주기적으로 자신의 N(Xi)(적용네트워크를 그래프로 표현하고 그것을 G라 했을 때 G에서 Xi와 그에 인접한 모든 노드들을 포함하도록 유도된 부분그래프(subgraph))의 상태를 조사하고 기록하며 N(Xi)중의 어떤 노드들(결함시 Xi가 그 노드들의 수행작업(task)들을 다시 할당해 주기 위한 지역 감독자(local supervisor)로서의 역할이 가능하게 하는 노드들)에 대해 백업파일(backup file)을 저장해 두고 주기적으로 갱신한다.

2. 2 정의 및 정리

이 절에서는 앞으로의 논문 전개를 위하여 다음과 같은 몇 개의 용어를 정의한다.

[정의1] : 알고리즘 A의 그래프가 컴퓨팅 시스템 S의 부분그래프일 때 A는 S에서 '수행가능(excu-table)' 하다고 한다.

[정의2]:시스템 S에서 노드 k개(예를 들어 X,X,....,X)가 고장난 상황을 k-fault F라 하며 F={X,X,....,X}로 표

www.dbpia.co.kr

기한다. 이때 고장난 노드와 그 노드에 연결된 간선(edge)을 모두 제거한 그래프는 Sf로 표기한다.

[정의3] : 알고리즘 A가 Sf에서 수행 가능할 때 S는 A와 F에 대해 'fault-tolerant' 하다고 한다.

[정의 4] : 모든 k-fault F에 대해 알고리즘 AA={A,A,....,A}의 모든 원소 A가 Sf에서 수행 가능할 때 S는 알고리즘 AA에 대해 'k-fault tolerant(k-FT)' 라고 한다.

[정의5] : 시스템 내의 프로세서가 노드로, 통신라인은 간선으로 표현되는 그래프를 'facility 그래프' 라 한다.

[정의6] : 모든 노드 X에 그 노드의 상태와 함께 레이블이 붙여진 그래프를 'configuration 그래프' 라 한다.

[정의7] : 레이블이 있는 n개 노드를 갖는 유용성(facility) 그래프를 '기초(basic) 그래프' 라 한다.[정의8] : 기초 그래프와 동질형(isomorphic)인 부분그래프를 갖는 그래프를 '중복(redundancy) 그래프' 라 한다.

일반적으로 결함허용 시스템에 대하여 알려진 기존의 정리중에는 아래의 내용들이 있다[14].

[정리1] : C가 단일루프(single loop) 시스템 Cn의 k-FT realization 중의 하나라면 C에 있는 모든 노드들의 degree는 적어도 'k+2' 이다.

[정리2] : 단일 루프시스템에 있는 간선들의 개수를 m이라 하면 m은 (k+2)(n+k)/2보다 크거나 같다(즉 m >= (k+2)(n+k)/2 이고 n은 노드 수이다).

[정리3] : G가 n+k 노드들을 갖는 k-hamiltonian 그래프라면 G는 C에 대하여 k-FT이다.

2.3 트리 네트워크에 대한 FT 시스템

여기서는 네트워크에 적용되었던 기존의 FT시스템 구축방식을 살펴본다. 앞서 소개한 바 와 같이 기존의 네트워크 적용방식은 루프와 트리형태가 있으나 루프는 보다 특수한 형태로 적용대상이 한정되어 있는 반면 트리는 계층형태를 수용 할 수 있으며 본문에서 다루려는 논점의 일부와 관련이 있는 바 본문에서의 기존적용 사례로는 트리네트워크에 대한 경우만을 보기로 한다.

트리네트워크는 계층적인 특성을 가지고 있으며 모든 간선을 하향으로 생각 할 수 있다. 그러므로 간선의 방향을 양쪽으로 다 고려해야 하는 일반적 그래프에 비해 취급이 용이하다고 할 수 있는데 기존의 트리네트워크에 대한 FT 시스템의 구축을 보면 루트노드 위에 예비노드를 배치하는 방식으로 이루어졌음을 알 수 있다. 다음의 프로시듀어 Tr은 계층성을 이용하여 트리네트워크에 FT 시스템을 위한 중복그래프를 구성하는 방식[8]이다.

< procedure Tr > : Tree with Rendundancy

1. Tg가 Tr의 산출자그래프일 때 Tg를 루트가 있는 트리로 변환한 다음 루트노드를 레벨 k로 하고 레벨 i의 자노드는 레벨 i+1로 하여 터미널노드까지 레벨을 할당한다. 이때 터미널노드의 레벨은 k+r이 된다.
2. 루트 위에 있는 k개의 부가적 레벨 각각에 대해 하나씩의 예비노드를 할당하고 레벨 i와 레벨 i+1의 예비노드를 연결(1<i<k-2)해주며 레벨 k-1의 예비노드를 Tg의 루트노드에 연결한다.
3. Tr을 구성하기 위하여 레벨 i에 있는 모든 노드들을 레벨 j((j<=i+k+1):(0<=i<=k+r))에 있는 모든 후손노드에 연결한다.

[예제1] : 위에서 살펴본 프로시듀어 Tr을 이용하며 트리시스템에서 FT 시스템을 구성하는 것을 간단히 확인해보기 위해 그림 2.1과 같은 트리형태를 가정하여 모델로 삼고 Tr의 각 단계를 적용했을 경우 그림 2.3과 같은 시스템이 만들어지는지를 살펴보자. 여기에서 가정된 트리모델은 본문의 본론에서 다룰 일반적 그래프 모델과 관련이 있는 것으로 구체적으로는 일반적 그래프모델을 계층화시켰을 경우의 트리형태이며 이러한 모델로 예를 들어보면 본론의 경우와 기존의 경우를 자연스럽게 비교 할 수 있는 효과도 얻을 수 있다.

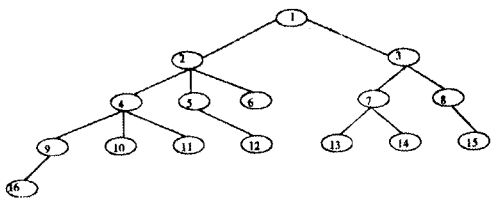


그림 2.1 초기상태 트리

먼저 Tr의 단계1을 적용해보면 주어진 모델이 이미 루트가 있는 트리이므로 루트노드를 레벨1(k=1)로 하고

www.dbpia.co.kr

레벨1의 자노드들을 레벨2로 하여 같은식으로 터미널 노드까지 계속하면 터미널노드의 레벨은 k+r이므로 5가된다. 따라서 노드1은 루트노드로 노드2와 노드3의 부노드이고 노드3은 노드8의 부노드이며 노드15는 노드8의 자노드이자 터미널노드가 된다. 다음으로 Tr의 단계2를 적용하면 k가 1이므로 루트위에 하나의 예비노드를 할당하고 레벨을 0으로 하며 레벨i와 레벨i+1의 예비노드(여기서는 레벨0의 예비노드)를 연결해주며 레벨k-1의 예비노드를 루트와 연결하는데 그 상황은 그림 2.2와 같다. 여기서 예비노드는 s로 표기한다.

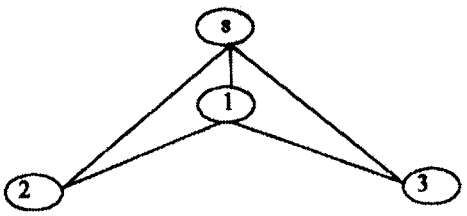


그림 2.2 예비노드의 연결

최종적으로 Tr을 구성하기 위해 단계3을 적용하면서 레벨i에 있는 모든 노드들에 대해 레벨 j((j<=i+k+1):(0<=i<=k+r))에 있는 모든 후손노드들과 연결하면 그림 2.3과 같은 결과를 얻게 되는데 이는 앞서 소개한 바와같이 임의의 노드가 결함을 발생시켰을 경우 중복간선의 효과로 인해 연결된 노드들 중 하나가 그 역할을 대신할 수 있는 FT시스템이 된다.

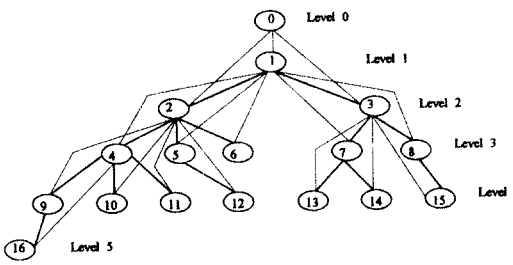


그림 2.3 중복성이 고려된 FT 트리

Ⅲ. 본 론

기존의 루프네트워크와 트리네트워크에 대한 k-FT 시스템 구축방식은 이미 알려져 있지만 그러나 실제의 시스템 중 완전한 루프나 트리 형태를 취하는 것은 거의 없으며 대부분이 복잡한 연결관계를 가지고 있다. 이런 것을 고려할 때 기존의 알고리즘[8]은 현실적으로 적용할 수 있는 범위가 매우 제한적이므로 본 논문에서는 임의의 형태(topology)를 갖는 시스템(일반적인 그래프)에 대하여 k-FT 시스템을 구축하는 방법을 제안한다. 이러한 일반적 그래프에 대한 k-FT 시스템의 구축은 다음 각 절에서 설명되는 바와 같이 여러 단계의 변환을 거쳐 완성되는데 구체적으로는 최소 깊이의 트리형태로 변환하기 위한 스패닝트리 선택단계(minimum depth spanning tree), 스패닝트리를 루트가 있는 트리로 변환하는 단계(centrally rooted tree), 이 트리에 기본그래프의 모든 연결성을 복원시키는 단계(original connectivity), k개의 결함허용을 위한 연결성 구축단계(k-FT system)로 진행된다. 이와 같은 단계들을 순서적으로 처리되도록 하나의 알고리즘화 하면 결국 일반적 그래프에 대한 k-FT 시스템을 구축하는 방법이 된다. 본 연구에 대한 결함허용시스템 구축과정의 예를 보이기 위해 본문에서는 임의의 구조를 갖는 시스템을 실험 모델로 삼았으며 그 시스템 요소 간의 연결성을 그래프화한 것이 그림 3.1의 기초그래프이다.

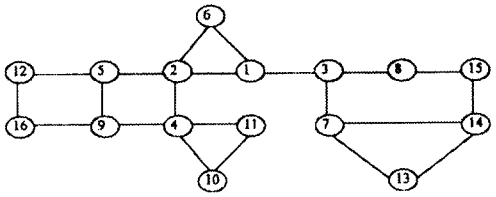


그림 3.1 기초그래프

3.1 최소깊이트리 구성을 위한 스패닝트리 선택

일반적 그래프의 k-FT 시스템을 구축하기 위하여 본 논문은 기존의 트리형태에 적용되었던 방법[5]을 변형시킨 새로운 적용 방법을 제시하는데 그 구현을 위해서는 먼저 기초 그래프에서 스패닝트리를 추출하여 그

www.dbpia.co.kr

것을 루트가 있는 트리로 바꾸어야 한다. 이때 이 트리의 깊이가 크면 부가적으로 필요한 연결의 갯수가 매우 많아지므로 최소깊이스패닝 트리를 구하는 것이 그러한 오버헤드를 줄이는 데 필수적이다. 아래에서 제시되는 프로시듀어 EXTRACT는 비교적 오버헤드가 적으면서 직경(diameter)이 작은 스패닝 트리를 구하기 위한 방법이다.

<procedure EXTRACT >:Minimal Diameter Spanning Tree Extraction

1. E(i)는 노드 i가 갖는 간선들의 전체 개수, N(i)는 노드 i의 인접 노드들의 집합, D(i)는 내림차 순에서의 i번째 노드, FIRST(i)는 큐 FIRST에서의 i번째 위치, SECOND(i)는 큐 SECOND에서의 i번째 위치로 각각 정의하며 E의 초기치는 0으로 한다.
2. 전체 노드를 N(i)에 대해 내림차순으로 정렬하여 D(1)의 E(i)가 D(2)의 E(i)보다 크면 D(1)을 중심노드(center)로 하며, 그렇지 않으면 D(1)과 간선수가 같은 D(i) 중에서 N(Xi)에 터미널(terminal)노드가 없는 것중 최소오더(minimum order[8])의 D(i)가 중심노드가 된다(first-selection). 만일 모든 D(i)가 N(Xi)에 터미널노드를 가지면 D(1)이 중심노드가 된다(art-selection).
3. 중심노드를 FIRST(rear)로 넣어준다.
4. 다음의 일련의 처리를 <조건>이 만족되어 루프를 벗어날 수 있을 때까지 반복 수행한다. N(center)를 E(i)에따라 내림차순으로 정렬한 후 FIRST와 SECOND에 정렬순서대로 입력하고 <SECOND(front)와 SECOND(rear)가 같으면(즉 큐 SECOND가 비어 있으면) 5번으로 분기> 하며 SECOND(front)를 center가 되게 한다.
5. 다음의 일련의 처리를 <조건>이 만족되어 루프를 벗어날 수 있을 때까지 반복 수행한다.

FIRST(front)를 center로 주고 N(center)중 center와 연결해도 사이클(cycle)이 형성되지 않는 노드들을 center와 연결(연결시 E를 1증가)하며 <E가 21이면 루프벗어남>을 실행한다.

위의 알고리즘의 수행으로 기초그래프로부터 얻어진 스패닝트리는 그림 3.2와 같다.

3.2 중심노드를 루트로 갖는 트리로의 변환

여기서는 앞절에서 선택된 스패닝트리를 깊이가 최소인 트리(rooted tree)로 변환하기 위한 방법을 제시하여 일반적인 그래프 형태를 트리형태와 비슷한 방법으로 적용할 수 있도록 유도한다. 이때 루트를 결정함에 있어 중요한 것은 스패닝트리의 전체 노드 중 중심부에 있는 노드를 루트로 선택하는 것이며 그것이 곧 최소의 깊이를 갖는 트리를 만들 수 있는 요인이 된다. 프로시듀어 SELECT는 이러한 방법을 묘사하고 있다.

〈 procedure SELECT 〉 : Centralized Rooted Tree Selection

1. DP(i)는 노드 i가 그래프에서 도달 가능한 최대 거리, I(i)는 오름차순에서의 I번째 노드로 각각 정의하며 각 간선의 길이는 모두 1로 간주한다.
2. 각 노드의 D(i)를 비교하여 오름차순으로 정렬하여 I(1)의 D(i)가 I(2)의 D(i)보다 크면 I(1)을 루트로 하고 그렇지 않으면 N(Xi)에 터미널노드가 없는 노드 중 정렬 순서가 가장 빠른 노드가 루트가 된다. 만일 모든 I(i)가 N(Xi)에 터미널노드를 가진다면 I(1)이 루트가 된다.
3. 계층구조의 구성을 위해 루트를 레벨 k로 하고 레벨 i의 자노드(children)는 레벨을 i+1로 정의 하면서 터

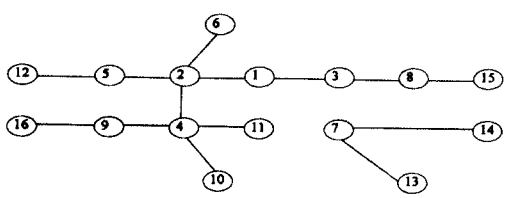


그림 3.2 추출된 스패닝그래프

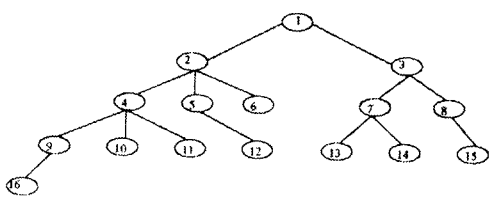


그림 3.3 변환된 rooted 트리

www.dbpia.co.kr

미널노드까지 레벨을 할당한다(k<=i<=k+r-1). 이때 최하위 터미널노드의 레벨은 k+r이 되며 이미 연결되어 있는 노드들은 제외한다.

위의 알고리즘의 수행으로 3.1절의 그림 3.2로부터 산출되는 계층적 트리가 아래 그림 3.3에 나타나 있다.

3.3 기초그래프의 모든 연결성 유지를 위한 간선복원

이제는 3.2절에서 구성된 트리에 기초그래프에서 존재했던 모든 연결성을 부가함으로써 새로이 구성되는 계층적인 그래프가 초기 상태 시스템의 통신능력을 완전히 만족할 수 있도록 해야 한다. 이를 위하여 그림 3.3의 트리에 기본그래프의 간선 중 빠진 것을 모두 부가한다. 그 결과 생성되는 그래프는 초기의 기초그래프와 동질형(isomorphic)의 구조가 되는데 단지 레벨이 할당되었다는 점만이 다르다. 따라서 변환된 계층구조의 각 노드에 할당된 레벨 값을 이용하면 트리네트워크와 비슷한 방법으로 취급할 수 있게 된다. 그림 3.4는 기초그래프의 연결성을 고려하여 재구성한 싸이클(cycle)이 존재하는 계층적 형태의 그래프이다.

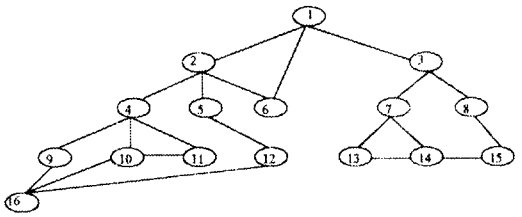


그림 3.4 연결성이 복원된 기초그래프의 계층구조

3.4 결함허용을 위한 연결성 구축

앞에서 구한 싸이클(cycle)이 존재하는 계층구조의 그래프에 대해 결함발생 시 자체 처리가 가능한 이른바 결함허용시스템을 구성하기 위해서는 그에 필요한 연결성이 있어야 하는데 이러한 연결성은 인접노드의 결함발생을 최우선으로 고려하면서 부가하였다. 즉 인접거리에 따른 단계적 처리를 전제하여 연결성을 구성하도록 한 것이 다음의 프로시듀어 CONNECT이다. 여기에서는 싸이클 상의 노드들에 대한 연결성의 중복과 노드의 레벨 등을 연결요인으로 반영하였다.

<procedure CONNECT> : Node and Spare Connection

1. 루트노드를 레벨 k로 시작하여 프로시듀어 Tr[8]과 같은 방법으로 각 노드에 레벨을 부과하고 0부터 k-1까지의 k개 레벨 각각에 대하여 하나씩의 예비(spare)노드를 할당한다.
2. I(0<=i<=k+r-2)에 대하여 J(j<=i+k+1)까지의 모든 후손노드들(decendants)과 연결한다.
3. 싸이클을 형성하고 있는 노드들에 대하여 다음을 수행한다. 첫째, 내부에 싸이클이 존재하지 않는 최소의 싸이클(unit cycle)상의 노드들에 대해 연결의 중복성이 나타날 때까지 현재노드로 부터 거리 k+1이내에 존재하는 노드들 중 현재노드와 동일 부노드(parent)를 갖지 않는 노드들을 현재노드와 연결한다. 둘째, 자노드를 공유하는 부노드들을 서로 연결한다(동일 부노드를 공유하는 자노드들의 연결포함). 셋째, 노드 i(i는 "상위레벨에서 하위레벨로, 왼쪽에서 오른쪽으로" 진행)로부터 거리 k만큼의 하위레벨에 있는 노드에 직접 연결된 노드가 i로부터 거리 k+1 보다도 멀리있는 하위레벨에 존재하는 경우는 노드 i와 직접 연결한다.

그림 3.4의 계층구조 그래프를 대상으로 프로시듀어 CONNECT를 적용하여 구성한 결함허용시스템(k-FT)이 그림 3. 5이다.

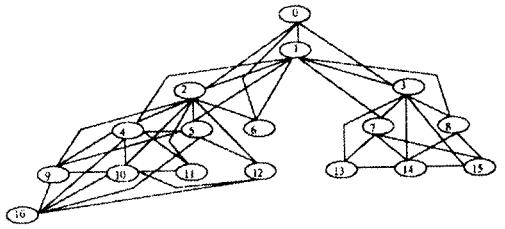


그림 3.5 기초그래프에대한 k-FT 시스템

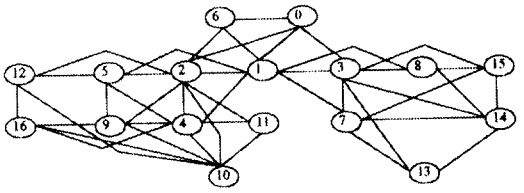


그림 3.6 재구성된 k-FT 시스템

www.dbpia.co.kr

그림 3.5의 k-FT 시스템을 기초그래프의 시스템 요소 배치상황을 고려하여 그것의 망유형을 복원하고 연결성은 k-FT를 유지하도록 재구성한 것이 그림 3.6이다.

3.5 일반적 그래프에 대한 k-FT 시스템 구성을 위한 통합 알고리즘

지금까지 본문에서 다루어왔던 내용을 프로시듀어의 처리 순서를 고려하며 통합하여 주된 처리의 흐름을 하나의 알고리즘으로 재구성하면 일반적인 그래프에 대한 k-FT 시스템을 구성할 수 있는 하나의 주프로시듀어가 되는데 이 프로시듀어는 필요한 부분에서 앞에 제시된 여러 프로시듀어들을 부프로시듀어로 호출하면서 수행되어야 하며 본절에서는 개괄적인 주요흐름만을 고려하여 간결하게 procedure k-FT로 나타내었다.

<procedure k-FT>:k-FT System Construction

1. 일반적인 그래프에서 최소직경 스패닝트리를 추출한다.
2. 스패닝트리에 루트노드를 지정하여 계층적 트리로 변환한다.
3. 기초그래프의 연결성에 맞도록 계층적 트리에 연결성을 추가한다.
4. 루트노드 위의 k개 각 레벨에 하나씩의 예비노드를 할당한다.
5. 레벨 i의 노드(0<=i<=k+r-2)에 대해 레벨 j(j<=i+k-1)까지의 모든 후손노드(decendants)와 연 결성을 구성한다.
6. 싸이클을 형성하고 있는 노드들에 대해 연결의 중복을 이용하며 레벨의 거리에 관계없이 일련의 연결성을 구성한다.

Ⅳ. 결론 및 추후연구방향

시스템의 구성요소가 손상되면 그로 인해 작업량도 악영향을 받게 되며 아울러 전체적인 시스템의 효율적 운영에도 지장을 초래하게 된다. 따라서 구성요소가 손상되는 경우 그에 대한 신속한 조치를 취할 수 있어야 시스템이 약간의 효율적 저하만을 감수하면서 점차 회복할 수 있게 하는 방법(graceful degradation[13])이 적용 가능하게 되며 나아가 시스템이 고장으로 사용자에게 인식되기 이전에 손상된 구성요소의 역할을 다시 활성화시킬 수 있게된다. 이러한 시스템 회복이 가능케 하기 위한 토대가 되는 결함허용시스템 구축을 위해 본 문에서는 몇가지 과정과 그에 필요한 알고리즘을 제안하였다. 이 방법은 기존의 루프나 트리형태의 시스템에 한정시킨 것이 아닌 일반적 유형의 시스템을 대상으로 삼았으며 따라서 시스템 형태를 그래프형태로 표현할 수 있음을 감안하면 일반적 그래프에 응용가능한 k-FT 시스템 구성방식이라 할 수 있다. 따라서 본 연구가 FT 시스템 구축대상의 적용범위를 일반화하는데 기여할 수 있는 기초가 되리라 기대한다.

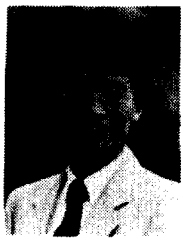
현재 본문과 관련하여 구체적 호출에 따른 세부알고리즘을 적용시킨 통합알고리즘의 간소화 및 FT 시스템을 위한 중복연결성을 줄일 수 있는 방법적 연구가 진행중이며 그것은 곧 본문의 구성시스템 구조를 실용적 측면에서 보다 합리적이고 효율적으로 단순화하는 직접적 요인이 될것이다. 아울러 예비노드의 최적배치를 위한 연구도 병행되어야 할 것으로 생각된다. 즉 본문에서 소개한 루트노드 위에 예비노드를 배치하는 방식 외에도 수평배치나 레벨별 분산배치방식도 고려 해볼만한 것으로 생각된다. 또한 중앙집중형과 같은 특수형 망유형에 대해서도 실용적 측면에서 두 개 이상의 결함을 분산처리로 감당할 수 있는 효율적 연결방법등도 연구되어야 할것이다.

참 고 문 헌

1. Robert Geist and Kishor Trivedi, "Reliability Estimation of Fault-Tolerant Systems:Tools and Techniques," IEEE Trans. on Conputer, pp 52-56, 1990.
2. Victor P.Nelson, "Fault-Tolerant Computing-:Fundamental Concepts," IEEE Trans. on Computer, pp. 19-25, 1990.
3. Danial P.Siewiorek, "Architecture of Fault-tolerant Computers:An Historical Perspec-tive" Proceedings of the IEEE, vol. 79, no. 12, december 1991.
4. Ernst J.Schmitter and Peter Baues, "The Basic Fault-Tolerant System", IEEE Micro, pp. 66-74, 1984.
5. David A. Rennels, "Fault-Tolerant Com-puting - Concepts and Examples," IEEE Trans. on Computer,

www.dbpia.co.kr

vol.c-33, no.12, december 1984.
6. William C. Carter, "Fault-Tolerant Compu-ting:An Introduction and a View-point," IEEE Trans. on Computer, Vol.c-22, No.3, March 1973.
7. Danial P.Siewiorek, "Architecture of Fault-Tolerant Computers," IEEE Computer, August 1984.
8. Raif M.Yanney and John P. Hayes, "Distributed Recovery in Fault-Tolerant Multi-processor Networks," IEEE Trans. on Computer, Vol.C-35, No.10, October 1986.
9. Tim Olson, "Fault-Tolerant Chips Increase System Reliability," Computer Design, March 15, 1986.
10. Alsirdas Avizienis and John P.J.Kelly, "Fault-Tolerance by Design Diversity: Concepts and Experiments," IEEE Computer, August 1984.
11. A.S.Tenenbaum,"Computer Networks," Englewood Cliffs, NJ:Prentice-Hall, 1981.
12. John P.Hayes, "A Graph Model for Fault-Tolerant Computing Systems," IEEE Trans. on Computers, Vol. C-25, No. 9, September 1976.
13. Israel Koren and Melvin A.Breuer, "On Area and Yield Consideration for Fault-Tolerant VLSI Processor Arrays," IEEE Trans. on Computers, Vol. C-33, No.1, January 1984.
14. Thomas Anderson and John C.Knight, "A Framework for Software Fault-Tolerance in Real-Time Systems," IEEE Trans. on Software Engineering, Vol. SE-9, No. 3, May 1983.
15. F.Harary, "Graph Theory," Addison-Wesely, 1986.
16. Samir Kamal and Carl V.Page, "Intermittent Faults:A Model and a Detection Procedure," IEEE Trans. on Computers, Vol. C-23, No. 7, July 1974.



문 윤 호 (Yun-Ho Moon) 정회원
1985 전북대학교 전산통계학과 학사
1987 숭실대학교 전산학과 석사
1998 숭실대학교 전산학과 박사 과정
1998 전주기전여자대학 컴퓨터과 조교수
<연구분야> 분산처리, 결함허용시스템, 컴퓨터구조

김 병 기 (Byung-Ki Kim) 정회원
1977 서울대학교 전자공학과 공학사
1979 KAIST 전산학과 이학석사
1979 경북대학교 전산학과 전임강사
1997 KAIST 전산학과 박사
1998 숭실대학교 정보과학대학 교수
<연구분야> ATM, 이동데이타통신, 컴퓨터구조

www.dbpia.co.kr