완전 적응 자기 경로제어 알고리즘을 사용하는 고장 감내 ATM 스위치 - 사이클릭 베니안 네트웍

정회원 박 재 현*

A Fault Tolerant ATM Switch using a Fully Adaptive Self-routing Algorithm -- The Cyclic Banyan Network

Jae-Hyun Park* *Regular Member*

요 약

본 논문에서는 새로운 고장 감내 ATM 스위치와 그 스위치 페브릭 내부에서 고장 감내를 위해서 사용하는 새로운 적응 자기 경로 제어기법을 제안한다. 이 스위치 페브릭은 동일한 단계(Stage)에 있는 스위칭 요소들(Switching Elements: SE)에 링크들을 추가하여 구성하고, 라우팅 알고리즘으로 베니안(Banyan) 네트웍의 자기 경로제어 기법을 확장하여 사용한다. 이렇게 구성된 스위치는 기존의 대부분의 스위치보다 입력단과 출력단 사이에 더많은 다중 경로(Path)들을 제공할 수 있다. 본 경로 제어 기법은 베니안 네트웍처럼 간단하며, 이는 패킷을 동일한 목적지로 보내는 스위칭 요소들간의 위상적 관계들에 기반을 두고 있다. 이 베니안 네트웍의 위상적 특성들은 본 논문에서 보여진다. 본 논문에서는 대수적으로 알고리즘의 정당성을 증명하였으며, 분석적 신뢰성 분석을 통해서 다른 스위치들과 정량적인 비교를 제공하였다. 이는 새로운 스위치 페브릭이 신뢰성 측면에서 베니안 네트웍이나 다른 네트웍보다 비용 대 효과비가 우수함을 보이고 있다.

ABSTRACT

In this paper, we propose a new fault tolerant ATM Switch and a new adaptive self-routing scheme used to make the switch to be fault tolerant. It can provide more multiple paths than the related previous switches between an input/output pair of a switch by adding extra links between switching elements in the same stage and extending the self-routing scheme of the Banyan network. Our routing scheme is as simple as that of the Banyan network, which is based on the topological relationships among the switching elements (SE's) that render a packet to the same destination with the regular self-routing. These topological properties of the Banyan network are discovered in this paper. We present an algebraic proof to show the correctness of this scheme, and an analytic reliability analysis to provide quantitative comparisons with other switches, which shows that the new switch is more cost effective than the Banyan network and other augmented MIN's in terms of the reliability.

I. 서 론

베니안 네트웍은 입출력 단들 간에 병렬 연결을 제공하는 수단으로써, 비용 대 효과비 측면에서 크로스바 스위치의 좋은 대체 스위치 이다. 베니안 네트웍은 이미 상용화 되어왔지만 [13], 네트웍상의 각 입출력 쌍에 대해서 각각 유일한 경로만을 갖기 때문에 고장 감내성이 없다는 문제점을 가지고 있다. 그러므로, 어떠한 하나의 고장도 이러한 종류의 네트웍을 동작불가능 상태로 만든다.

그들이 고장 감내성을 갖게하기 위해서 몇몇 기

* 삼성전자 정보통신본부 데이터 네트웍 시스템 개발팀 시스템 구조 설계 연구실 대형 ATM 교환기 프로젝트
(jaehyun.park@ieee.org)
논문번호 : 99245-0616, 접수일자 : 1999년 6월 16일

법들이 개발되었는데, 이들 작업들은 고장이 생긴 경로들(paths)를 돌아서 가는 여분의 경로들을 제공하는데 중점을 두고 있다[1, 4, 5, 11, 14]. 그러나 이들 작업들은 추가된 링크를 통해서 신뢰성을 높이는데 성공했으나, 기존의 링크가 우회 경로로 사용될 수 있음을 간과 했다.

본 논문에서는 사이클릭-베니안 네트웍(Cyclic Banyan network)이라 불리우는 새로운 추가된 베니안 네트웍 (Augmented Banyan Network)과 그 네트웍에 적합한 적응 자기 경로제어 알고리즘을 제안한다. 이 스위치에서는 고장난 링크가 생겼을 때, [6, 7, 8, 9, 10]의 방법과 같이, 추가된 링크 뿐만 아니라 기존의 링크도 패킷의 경로를 위해서 사용된다. 본 스위치는 간단한 라우팅 알고리즘을 사용하며, 스위칭 요소의 복잡도도 기존의 스위치와 같게 유지한다. 본 논문에서는 대수학적 방법으로 경로제어 알고리즘의 정당성을 증명하고, 분석적 모델을 통해서 신뢰성과 비용-효과비(cost-effectiveness)를 분석한다.

2절에서 본 스위치와 스위치 내에서의 경로제어 알고리즘을 유도하기위해 베니안 네트웍의 특성들을 조사한다. 3절에서는 본 스위치와 경로제어 알고리즘을 정의하고, 경로제어 알고리즘의 정당성을 증명한다. 4절에서 이 스위치의 신뢰성을 분석한다. 마지막으로 5절에서 결론을 맺는다.

Ⅱ. 베니안 네트웍의 특성

베니안 네트웍은 Goke와 Lipovski에 의해서 각 프로세서와 메모리 간에 유일한 경로을 갖는 네트웍으로 정의 되었다. 이 정의는 한 SE로 가는 경로들의 집합은 신장 트리(spaning tree)를 구성하고, 그 SE로부터의 경로들도 또한 신장 트리를 형성한다는 것을 암시하고 있다 [3].

이들 모든 신장 트리들 중에서 한 종류의 신장 트리만이 중요시 되는데, 그 이유는 동일한 신장 트리에 속하는 모든 SE들은 기존의 자기 경로제어 알고리즘을 사용하여 한 패킷을 동일한 출력단으로 내보낼 수 있기 때문이다. 그림 1에서 굵은 선들로 그려진 신장 트리가 이러한 종류의 것이다. 이 트리들은 각 출력단으로 부터 (이를 트리의 뿌리로 하고) 네트웍의 모든 입력단들로의 (이들은 트리의 잎들로 취급하면) 링크들과 SE들의 노드로 이루어진다.

관련연구에서 [4, 5, 11, 14], 이러한 신장 트리는 우회 경로들을 제공하여 베니안 네트웍을 고장 감내가 가능하게 하기 위해서 이용되어 왔다. 그 중 연구[4, 14]에서는 그 트리의 동일한 단계의 SE들을 추가한 (augmented) 링크를 사용해서 묶음(chaining)으로써 우회 경로을 얻는다. 선택된 출력 링크에 고장이 발생했을때 패킷은 그림 1에서 화살표 "A"로 그려진 것과 같은 트리내의 동일한 수준(level)의 한 SE로 보내질 수 있다. 그리고 그 SE로부터 다시 기존의 베니안 네트웍의 자기 경로제어 알고리즘으로 목적지인 출력단으로 올바르게 보낼질 수 있다. 또다른 관련 연구에서는 [11]([5]), 추가되는 후진 (전진) 링크를 자식 (부모) SE에 연결하여 우회 경로을 만든다. 고장이 일어났을 경우, 그림 1의 화살표 "B" ("C")로 그려져 있는 것 같이 트리내의 한단계 낮은 (높은) 수준의 자식 (부모) SE로 패킷을 보낼 수 있다.

이러한 기법들에서는 자기 적응 경로 알고리즘에 의해 의도하지 않았던 출력 링크로 패킷을 일단 내보내면, 원하던 출력단으로의 경로가 존재하지 않기 때문에, 기존의 링크들을 우회 경로로 사용할 수가 없다. 본 연구의 목적은 고장난 링크나 SE를 만난 패킷을 의도했던 출력단에 보내기 위해서 추가된 링크 뿐만 아니라 기존의 링크들도 우회 경로들로 사용하는 추가된 베니안 네트웍과 경로제어 알고리즘을 개발하는 것이다.

이 절에서는 사이클릭 베니안 네트웍과 경로제어 알고리즘을 개발하기 위해서 베니안 네트웍의 SE들 간의 위상적(topological) 관계들에 관해서 살펴본다.

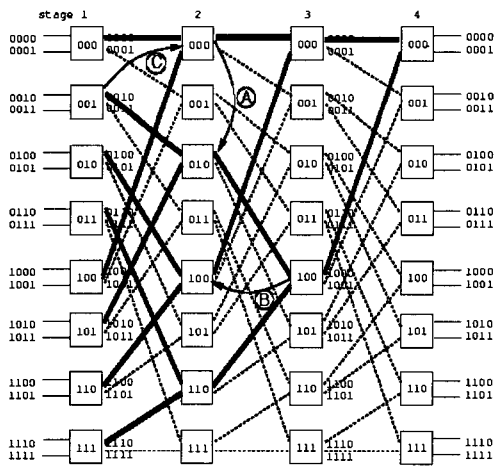

그림 1. 델타 네트워크(N=16)에서의 신장 트리, SE들의 동치부류, 그리고 우회 경로들

많은 MIN이 위상적으로 같다는 것은 널리 알려져 있기때문에 [15], 델타 네트워을 예로 들어 설명한다. 본 논문에서는 [15]의 것과 동일한 형태의 명명법 (naming scheme)을 델타 네트워과 본 사이클릭 베니안 네트워의 형태(configuration)를 기술하기 위해서, 그리고 경로제어 알고리즘의 정당성 증명을 위해 사용한다. 각 단계는 첫단계를 1로 하며, 연속(sequence)으로 log_2N까지 번호를 붙인다. N개의 입출력단들은 log_2N 자리수의 이진수로 번호를 붙인다: $link(p_{log_2N}, p_{(log_2N)-1}, \ldots, p_1)$. 각 단계에서, 한 SE는 $(log_2N)-1$ 자리수의 이진수들로, $(p_{(log_2N)-1}, p_{(log_2N)-2}, \ldots, p_1)$, 표현되는데, 이는 꼭대기부터 시작하여 단계내에서의 위치를 이진수로 표현한 것이며, 이는 수준(level)이라고도 부른다. 입력 패킷의 목적 주소는 $A = (a_1,a_2,\ldots,a_{log_2N})$로 표현 된다. 먼저, 델타 네트워을 다음과 같이 정의하자.

정 의 1 델타 네트워의 각 링크에 대한 위상 기술방법은 다음과 같다.

$\beta_0[(p_l,p_{l-1},\ldots,p_1)_i] = (p_{l-1},p_{l-2},\ldots,p_1,\ 0)_{i+1}$,

$link(p_l,p_{l-1},\ldots,p_1,0)_i$ 로 연결

$\beta_1[(p_l,p_{l-1},\ldots,p_1)_i] = (p_{l-1},p_{l-2},\ldots,p_1,\ 1)_{i+1}$,

$link(p_l,p_{l-1},\ldots,p_1,1)_i$ 로 연결

여기서 $l = (log_2N) - 1,\ 1 \le i \le l$.

β_j는 단계 i의 한 SE로 부터 단계 $i+1$의 두 SE들로의 사상(mapping)으로 상호 연결을 기술한다. 이는 SE들간의 위상적 관계들을 유도하기 위해서, β_0는 SE의 상위 출력 링크를 통해서 도달 가능한 SE를 기술하기 위해서 사용된 것이고, β_1은 하위 출력 링크를 위해서 사용된다. 이제, SE와 링크를 기술하기 위한 약식 기호를 다음과 같이 정의한다.

정 의 2 $n_{i,j}$는 단계 i, 수준 j의 SE이다, $1 \le i \le log_2N,\ 0 \le j < N/2$.

다음은 SE들간에 하나의 관계를 정의하여, 이 관계가 동치관계(equivalence relation) 이고, 델타 네트워의 각 단계에는 동치부류(equivalence class)들이 있음을 보인것 이다.

정 의 3 $R(n_{i,j})$는 델타 네트워에서 자기 경로제어 함수 α_i에 모든 가능한 주소들을 적용 시켰을 때, SE $n_{i,j}$로 부터 도달 가능한 마지막 단계의, 단계 log_2N, 출력 링크들의 집합이다. 여기서

$\alpha_i[(p_l,p_{l-1},\ldots,p_1)_i,(a_1,a_2,\ldots,al+1)] = (p_{l-1},p_{l-2},\ldots,p_1,\ a_i)_{i+1}$, $link(p_l,p_{l-1},\ldots,p_1,a_i)_i$로 연결.

여기서, $(a_1,a_2,\ldots,al+1)$는 각 패킷의 주소.

한 패킷이 단계 i의 한 SE의 입력 링크에 도착했을때, α_i는 단계 $i+1$의 한 SE로 출력 링크를 통해서 그 패킷을 보내기 위해서 적용된다.

정 의 4 단계 i의 SE들의 집합 A_i에서 정의되는 관계 $\sim$ 는 다음과 같은 속성을 갖는 cartesian product $A_i \times A_i$의 부분 집합이다:

$n_{i,j} \sim n_{i,k} \Leftrightarrow R(n_{i,j}) = R(n_{i,k})$.

보조정리 1 집합 A_i에서 정의된 관계 $\sim$ 는 동치관계 (Equivalence Relation) 이다.

이제 적응 자기 경로제어 알고리즘을 유도하는데 유용한 속성들이 있음을 증명할 수 있다.

정 리 1 단계 i의 한 동치부류에 속하는 SE들은 그들의 SE 번호 $(p_{l-i+1},\ldots,p_1,\underline{d_l,d_2,\ldots,d_{i-1}})_i$의 접미사(suffix)가 동일한 이진수 연속 "$\underline{d_l,d_2,\ldots,d_{i-1}}$"을 갖는다. (단, $1 < i \le log_2N.$) $i = 1$ 일때, 즉, 첫 번째 단계에서, 모든 SE들은 하나의 동치부류에 속한다.

증 명: 각 단계에 대한 수학적 귀납법으로 간단히 증명된다 [6, 7, 8].

정리 1에 따라, 다음 것들이 만족된다.

정 의 5 델타 네트워에서 한 단계 내의 한 SE로 부터 다른 SE의 상대적 위치를 표현하기 위한 위상기술 방법은 다음과 같다.

$\gamma_k[n_{i,j}] = n_{i,(j+k)\ mod\ (N/2)}$ 그리고

$\gamma_{-k}[n_{i,j}] = n_{i,(j+N/2-k)\ mod\ (N/2)} = \gamma_{N/2-k}[n_{i,j}]$,

여기서 $-N/2 \le k \le N/2$.

다음의 따름 정리는 단 안에서의 한 동치 클레스 안에 있는 인접한 SE들 간의 위상적 거리를 표현한다.

따름정리 1 단계 i 안에서의 한 동치 클레스 안에 있는 인접한 두 SE들 간의 위상적 거리는 2^{i-1} 이다: $n_{i,j} \sim n_{i,k} \Leftrightarrow |j-k| = 2^{i-1} \times l$,

여기서 $l \in \{0, 1,\ldots,N/2^i\}$.

증 명: 정의 5 과 정리 1에 따라 증명됨. ■

다음의 정리와 따름정리들은 본 스위치와 적응 경로제어 알고리즘의 기본을 이룬다.

따름정리 2 $\beta_k[n_{i,j}] \sim \beta_k[\gamma_{2^{i-1}}[n_{i,j}]] \sim \beta_k[\gamma_{-2^{i-1}}[n_{i,j}]]$

여기서 $k \in \{0, 1\}$.

증 명: 정의 5 와 정리 1에 의해서 증명됨. ■

정 리 2

$\beta_j[\gamma_k[(p_l,p_{l-1},\ldots,p_1)_i]] = \gamma_{k\times 2}[\beta_j[(p_l,p_{l-1},\ldots,p_1)_i]]$

여기서 $1 \le i < log_2N,\ j \in \{0, 1\}$.

증 명: $d = 1$ 일때,

$\beta_1[\gamma_2[(p_l,p_{l-1},\ldots,p_1)_i]] = \gamma_2[\beta_1[(p_l,p_{l-1},\ldots,p_1)_i]]$

(정의 1 과 5 에 의해서)

$d = j$의 경우 다음과 같이 가정하자:

$\beta_1[\gamma_j[(p_l p_{l-1},\ldots,p_1)_i]] = \gamma_{j\times 2}[\beta_1[(p_l p_{l-1},\ldots,p_1)_i]]$

$d = j+1$ 의 경우,

$\beta_1[\gamma_{j+1}[(p_l p_{l-1},\ldots,p_1)_i]] = \beta_1[\gamma_j[\gamma_1[(p_l p_{l-1},\ldots,p_1)_i]]]$

(정의 5에 의해서)

$= \gamma_{j\times 2}[\beta_1[\gamma_1[(p_l p_{l-1},\ldots,p_1)_i]]]$ ($d = j$ 경우에 의해)

$= \gamma_{j\times 2}[\gamma_2[\beta_1[(p_l p_{l-1},\ldots,p_1)_i]]]$ ($d = 1$ 경우에 의해)

$= \gamma_{(j+1)\times 2}[\beta_1[(p_l p_{l-1},\ldots,p_1)_i]]$ (정의 5 에 의해)

따라서 수학적 귀납법에 의해서 증명된다.

β_1를 β_0로 대치하면, 대칭적인 경우의 증명을 얻을 수 있다. ■

다음의 따름정리는 한 출력 링크로 시작되는 경로가 동일한 SE의 다른 출력 링크로 시작되는 경로로 대치될 수 있음을 나타내고 있다.

따름정리 3

$\gamma_1[\beta_0[n_{i,j}]] = \beta_1[n_{i,j}]$ (정의 1과 5에 의해서)

$\gamma_{2^i-1}[\beta_1[n_{i,j}]] = \gamma_{2^i-1}[\gamma_1[\beta_0[n_{i,j}]]]$ (위의 등식)

$\sim \beta_0[n_{i,j}]$(정의 5 와 정리 2)

$\gamma_{-1}[\beta_1[n_{i,j}]] = \beta_0[n_{i,j}]$ (정의 1과 5)

$\gamma_{-(2^i-1)}[\beta_0[n_{i,j}]] = \gamma_{-(2^i-1)}[\gamma_{-1}[\beta_1[n_{i,j}]]]$ (위의 등식)

$\sim \beta_1[n_{i,j}]$ (정의 5와 정리 2에 의해)

여기서 $1 \le i < log_2 N$.

위에서 베니안 네트워크에 새로운 고장 감내 MIN과 그것의 자기 경로제어 알고리즘을 유도하는데 유용한 위상적 특성들이 있음을 보였다. 만약 단계내에서 SE들간에, 즉 한 SE $n_{i,j}$로 부터 다른 한 SE $\gamma_k[n_{i,j}]$ 로의 경로들을 위한 상호연결을 만들면, 앞서 보인 성질들을 이용하여, 각 추가된 링크들 뿐만 아니라 기존의 모든 출력링크들도 적응 경로제어를 위해 사용될 수 있다. 다음 절에서는 비용 대 효과비가 좋은 이러한 상호 연결을 추가한 스위치를 제안한다.

Ⅲ. 사이클릭 베니안 네트워크와 자기 경로제어

이 절에서는 비용 대 효과면에서 좋은 베니안 네트워크의 확장, 사이클릭 베니안 네트워크과 그에 적합한 적응 자기 경로제어 알고리즘을 제안 한다.

1. 스위치 형태

관계 γ_k를 이용하여, 우회 경로들을 얻기위해서, 각 SE에 일반적인 입출력 링크들과 더불어 연쇄-입력(chain-in) 링크와 연쇄-출력 링크를 그림 2에 나타난 것과 같이 덧붙인다. 이것은 4 × 4 크로스바 스위치로 다음 절에서 기술되는 경로제어 알고리즘을 사용하여 동작한다.

$N \times N$ 사이클릭 베니안 네트워크는 $N \times N$ 델타 네트워크의 각 단계의 모든 SE들을 묶는 링크들을 추가 함으로써 얻을 수 있다.

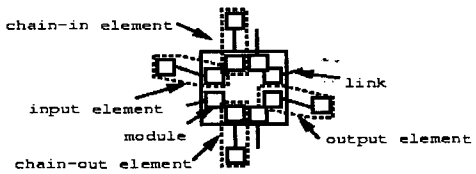

그림 2. 추가된 기본 스위치

정 의 6 사이클릭 베니안 네트워크의 연쇄-입력 링크에 대한 위상 기술방법은 다음과 같다.

$\gamma_1[n_{i,j}] = n_{i,(j+1) \bmod (N/2)}$ 그리고

$\gamma_{-1}[n_{i,j}] = n_{i,(j+N/2-1) \bmod (N/2)} = \gamma_{N/2-1}[n_{i,j}]$.

γ_1과 γ_{-1}는 연쇄-출력 링크에 대해서 한 SE를 동일한 단계내의 다른 SE로 사상한다. 사이클릭 베니안 네트워크은 정의 1과 다음의 정의 6으로써 정의할 수 있다. 일반적인 16 × 16 델타 네트워크으로부터 사이클릭 베니안 네트워크을 구성하는 예가 그림 3에 나타나 있다.

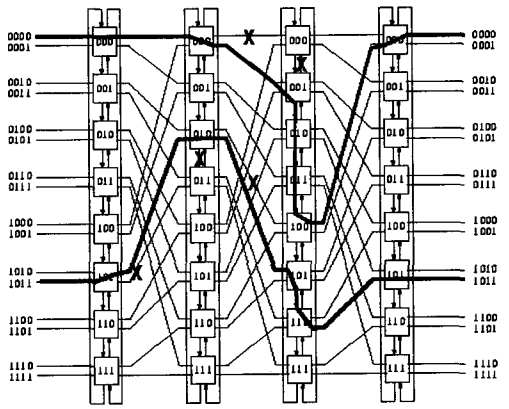

그림 3. 사이클릭 베니안 네트워크의 적응 자기 경로제어 알고리즘

2. 적응 자기 경로제어 알고리즘

다른 많은 MIN들에서와 같이, 사이클릭 베니안 네트워크의 경로제어는 목적지 꼬리표(tag)를 가지고 제어된다. 여기에 덧붙여서, 패킷에 덧붙여지는 고정된 크기의 $log_2 N$ 이탈 꼬리표(deviation tag) k가 각 단계에서 원래 목적이 되는 SE나 이것의 동치

SE로 부터의 위상적 거리를 나타내기 위해서 사용된다. k 값은 그것의 현재값과 선택된 링크 그리고 원래 목적이 되는 링크를 가지고 계산할 수 있다. 따라서, 행선지 주소는 (A, k) 쌍으로 표현된다. 단계 1의 입력 링크에 한 패킷이 도착 했을 때, k 값은 0으로 설정된다: 즉 행선지 주소는 $(A, 0)$ 이다. 본 적응 경로제어 알고리즘은 다음과 같다. (여기서 i는 해당 SE의 현재 단계 번호이다.)

정 의 7 사이클릭 베니안 네트웍의 SE에 있어서 적응 자기 경로제어 알고리즘은 다음과 같다.
(여기서 $s \in \{0, 1\}$.)
각 입력 패킷에 대하여,

1. $k = 0$ 인 경우,
   (a) $\beta_s[\cdot]$ $(= \alpha_i[\cdot])$로 보낸다.
   (b) 만약 실패하면, $\beta_s[\cdot]$ $(\neq \alpha_i[\cdot])$로 보낸다.
   (c) 만약 실패하면, $\gamma_{\delta(j)}[\cdot]$로 보낸다,
   (d) 만약 실패하면, $\gamma_{\delta(j+1)}[\cdot]$로 보낸다.
2. k $\neq$ 0 인 경우,
   (1) $\gamma_{\delta(\theta(0))}[\cdot]$ 로 보낸다,
   (2) 만약 실패하면, $\gamma_{\delta(\theta(1))}[\cdot]$ 로 보낸다 ,
   (3) 만약 실패하면, $\beta_{\theta(0)}[\cdot]$ 로 보낸다 ,
   (4) 만약 실패하면, $\beta_{\theta(1)}[\cdot]$ 로 보낸다 .

k를 수정하는 규칙은 다음과 같다:

1. $k = 0$ 일때,
   (1) 만약 $\beta_0[\cdot] = \alpha_i[\cdot]$ 그리고 $\beta_1[\cdot]$가 선택되었다면, $k \leftarrow 2^{i-1}$.
   (2) 만약 $\beta_1[\cdot] = \alpha_i[\cdot]$ 그리고 $\beta_0[\cdot]$가 선택되었다면, $k \leftarrow 1$.
   (3) 만약 $\gamma_1[\cdot]$가 선택되었다면, $k \leftarrow 2^{i-1}-1$.
   (4) 만약 $\gamma_{-1}[\cdot]$가 선택되었다면, $k \leftarrow 1$.
2. $k \neq 0$ 일때,
   (1) $\beta_0[\cdot] = \alpha_i[\cdot]$ 그리고 $\beta_1[\cdot]$가 선택되었다면, $k \leftarrow k \times 2 - 1$.
   (2) 만약 $\beta_1[\cdot] = \alpha_i[\cdot]$ 그리고 $\beta_0[\cdot]$가 선택되었다면, $k \leftarrow k \times 2 + 1$.
   (3) 만약 $\beta_j[\cdot] = \alpha_i[\cdot]$ 그리고 $\beta_j[\cdot]$가 선택되었다 면, $k \leftarrow k \times 2$, 여기서, $j \in \{0, 1\}$.
   (4) 만약 $\gamma_1[\cdot]$가 선택되었다면, $k \leftarrow k - 1$.
   (5) 만약 $\gamma_{-1}[\cdot]$가 선택되었다면, $k \leftarrow (k + 1) \bmod 2^{i-1}$.

여기서 $\delta(j) = \begin{cases} 1, j \bmod 2 = 1 \text{경우} \\ -1, \text{나머지경우} \end{cases}$ 이고

$\theta(m) = \begin{cases} (m+1) \bmod 2, k \leq 2^{i-2} \text{경우} \\ (m) \bmod 2, \text{나머지경우} \end{cases}$ 이다.

적응 자기 경로제어 알고리즘의 두가지 예를 그림 3에서 볼 수 있다. 여기서 고장은 X로 그려져 있다. 한 패킷이 입력단 (0000)에서 출력단 (0000)으로 고장난 링크를 가진 경로로 나아가려할 때, 원래 목적된 출력 링크가 고장난 SE는 그것을 다른 출력 링크로 보낸다. 그것은 본 경로제어 알고리즘에 의해서 연쇄 링크를 통해서 원래 의도된 SE의 동치 SE중 하나로 보내 질 것이고, 여기서부터 목적지 출력단으로 올바르게 보내질 수 있을 것이다. 정상적인 경로에 3개의 고장들을 가지고 있는 (1011)로 부터 (1011)의 경로제어 경우에 대해, 그림 3에서 보는 것과 같이 본 스위치와 경로제어 알고리즘으로 올바르게 경로를 결정할 수 있음을 알 수 있다.

정 리 3 입력된 패킷을 마지막 단계의 목적된 출력단에 올바르게 내보내기 위해서, 마지막 단계의 SE들을 제외한 사이클릭 베니안 네트웍의 모든 SE들은 기존의 출력 링크들이나 연쇄-출력 링크중 어느 곳으로도 그것을 보낼 수 있다.

증 명: 따름정리 3에 의해, 목적된 출력단으로 올바르게 가기위한 우회 경로로써 의도된 출력 링크 대신에 다른 출력 링크를 사용할 수 있음을 알 수 있다. 따름정리 2에 의해, 우회 경로로써 의도된 출력 링크 대신에 다른 연쇄-출력 링크를 사용할 수 있음을 알 수 있다. 정리 2에 의해, 의도된 연쇄-출력 링크 대신에 다른 출력 링크를 사용할 수 있음을 알 수 있다. ■

이제 사이클릭 베니안 네트웍을 위한 본 적응 자기 경로제어 알고리즘의 정당성을 증명할 수 있다.

정 리 4 사이클릭 베니안 네트웍의 경로제어 알고리즘은 임의의 입력 패킷을 그의 행선지에 올바르게 보낸다.

증 명: 먼저, 본 경로제어 알고리즘은 SE의 가용한 한 링크를 할당한다. 이 링크의 할당의 정당성은 정리 3에 의해서 증명된다.
두번째로, k의 수정 규칙은 다음과 같이 증명된다:

1. 규칙 1-(a)와 1-(b)는 따름정리 3에 의해서 증명된다. 규칙 1-(c)와 1-(d)는 따름정리 2에 의해서 증명된다.
2. 규칙 2-(a)와 2-(b)은 따름정리 3와 정리 2에 의해서 증명된다. 규칙 2-(c)는 정리 2에 의해서 증명된다.
3. k는 현 단계에서 연쇄 링크(들)을 통해서 보정 경로제어가 요구되는 횟수를 의미한다. 따라서

규칙 2-(d)는 증명된다. 또한 따름정리 1과 이 사실에 의해서 규칙 2-(e)가 증명된다.

따라서, 본 수정 규칙들에 의해서 k를 유지하는, 제시된 경로제어 알고리즘은 패킷을 마지막 단계의 의도된 출력단에 올바르게 내보낸다. 이 사실은 따름정리 2, 3, 그리고 정리 2에 의해서 증명된다. 따라서, 이 경로제어 알고리즘의 정당성이 증명된다. ■

2. 취약한 부분들의 보강

사이클릭 베니안 네트웍의 취약한 부분들은 단계 1의 입력 링크들과 마지막 단계, 단계 log_2N,의 출력 링크들과 연쇄-출력 링크들이다. 만약 단계 1의 입력 링크들과 마지막 단계, 단계 log_2N,의 출력 링크들중 하나라도 고장난다면, 어떤 고장감내 기법이 스위치안에 사용되어도, 스위치는 쓸모없게 될 것이다. 이 문제를 해결하기 위해서, [2]의 것과 비슷한 방법을 사용한다. 따라서 한 입력 정합 모듈은 하나의 입력단 이상에 접근할 수 있고, 한 출력 정합 모듈은 그림 4에서와 같이 하나 이상의 출력단에 의해서 접근이 가능하다. 마지막 단계의 하나의 고장은 시스템을 고장낼 것이기 때문에, 마지막 단계의 연쇄 링크를 그림 4에 보인것 같이 중복시킨다.

각 SE쌍의 경로 $\beta_k[n_i(log_2N)-1,j]$에 의해서 도달 가능한 각 SE를 $nlog_2N,j,k$로 (여기서 $k \in \{0, 1\}$) 이름 붙이자. 마지막 단계에서, 한 SE쌍내의 각 SE, $nlog_2N,j,k$,는 경로 $\beta_1[nlog_2N,j,k]$를 통해서 서로에게 접근 가능하다. 이 SE쌍은 그림 5에 그려져 있다. 그림 4에서 보이는 것과 같은 SE 쌍의 연결로써, 다음과 같은 $\beta_1[nlog_2N,j,k]$로 시작하는 두개의 우회 경로를 얻는다.

$\beta_0[nlog_2N,j,k] \sim \beta_0[\beta_1[nlog_2N,j,k]], \gamma_i[nlog_2N,j,k] \sim \gamma_i[\beta_1[nlog_2N,j,k]]$.

이러한 경로들을 이용하기 위해서, 다음과 같이 마지막 단계의 SE의 경로제어 알고리즘을 간단히 만들 필요가 있다: (i) 패킷이 $\gamma_k[\cdot]$를 통해서 내보내져서 k값이 k-1로 수정되는 경우를 제외하고는, k값은 변하지 않는다. (ii) SE는 다음의 순서로 패킷에 링크를 할당하려 시도한다: k = 0 일때, SE는 먼저 $\beta_0[\cdot]$로 내보내기를 시도하고, $\beta_1[\cdot]$를 다음으로 시도한다. 이러한 기법은 한 SE 쌍이 연쇄 링크들에 대해 하나의 고장을 감내하는 것을 가능하게 한다. 이러한 구조가 다음 절에서 신뢰성 분석을 위해서 가정된다.

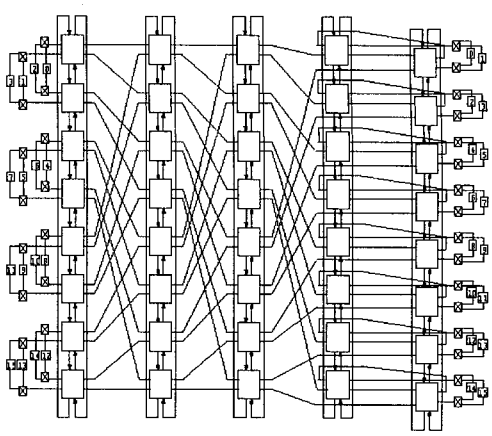

그림 4. 사이클릭 베니안 네트워크(N=16), 여기서 ⊠은 멀티플렉서 또는 디멀티플렉서를 나타낸다

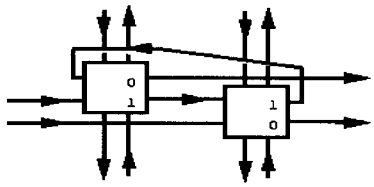

그림 5. 마지막 단계의 SE의 연쇄-출력 요소의 보강

Ⅳ. 사이클릭 베니안 네트웍의 신뢰성 분석

이 절에서는 사이클릭 베니안 네트웍과 적응 경로제어 기법의 신뢰성을 분석한다.

1. 고장 모형과 절대 한계들

신뢰성 분석을 위해서, [12]와 [14]의 것들과 비슷한 다음과 같은 고장 모형을 사용하겠다. 한 SE 내에 들어오고 나가는 각 링크는 그림 2에서와 같이 그와 결합된 하나의 모듈을 갖는다.

정 의 8 모듈은 그의 연결 링크가 입력(출력) 링크이면 입력(출력) 모듈이라고 불리운다. 또한, 모듈이 그의 연결 링크가 연쇄-입력(연쇄-출력) 링크이면 연쇄-입력(연쇄-출력) 모듈이라고 불리운다.

한 모듈은 연결하는 링크를 통해 한 패킷을 내보내는데 필요한 모든 제어 메커니즘을 가진다.

정 의 9 단계 i, $1 \le i < \log\_2N$,의 한 요소(element)는 단계 i의 한 출력 모듈과 그것을 연결하는 링크 그리고 이것이 연결되는 i+1 단계의 또

다른 SE의 입력 모듈로 구성된다. 한 SE의 연쇄-출력 모듈과 그의 연결하는 링크 그리고 그에 연결된 다른 SE의 연쇄-입력 링크는 또한 비슷한 방법으로 하나의 요소를 구성한다.

요소는 그림 2에 그려져 있다. 첫 단계의 하나의 요소는 하나의 스위치와 하나의 입력 모듈 그리고 그들을 연결하는 링크를 그림 4에서 나타낸 것같이 가진다. 유사하게, 마지막 단계에서의 하나의 출력 모듈과 연결하는 링크 그리고 거기에 연결된 스위치는 또한 한 요소를 구성한다. 하나의 입력(출력) 요소는 해당 SE의 입력(출력) 모듈을 담고있는 한 요소이다. 한 SE의 하나의 연쇄-입력(연쇄-출력) 요소도 비슷하게 이름지을 수 있다.

본 신뢰성 분석에서 요소를 기본 단위로 간주할 것이다. 왜냐하면 한 고장난 요소는 해당 요소를 사용하는 몇몇 입출력단들 사이의 경로들이 사용 불가능하게 됨을 의미하기 때문이다. 다음의 가정들은 본 분석작업에서 사용될 것이다.

- 한 요소는 만약 그의 구성요소들, 즉 출력모듈과 그의 연결하는 링크 그리고 그에 연결된 다른 SE의 입력 모듈 중 어느 하나라도 고장이면 고장으로 간주된다.
- 한 요소가 고장나게 되는 사건은 독립 사건이며, 이는 임의로 일어나게 된다.
- 스위치는 고장난 요소들의 위치나 갯수상 스위치의 어떤 임의의 한 입출력 쌍간의 연결을 막을때 고장난 것으로 간주된다.

이제 스위치 고장을 초래하는 고장 갯수의 하한과 상한을 찾을 것이다. 단계 i의 한 출력 (연쇄-출력) 요소는 단계 i+1 (단계 i)의 한 입력 (연쇄-입력) 요소로 간주되기 때문에, 두번 고려되는 것을 피하기 위해서, 다음에서 단계 1의 입력 요소들을 제외하고는 각 단계에서 단지 출력 (연쇄-출력) 요소만을 고려할 것이다.

정 의 10 한 스위치가 k-고장 감내일 필요충분조건은 스위치에 k 또는 그보다 적은 고장난 요소들이 있으면서도 모든 입출력 쌍간의 연결이 제공되는 것이다.

정 리 5 $N \times N$ 사이클릭 베니안 네트웍은 3 - 고장감내 이다.

증 명: 정리 3에 의해서, 스위치가 세개의 고장을 감내가능하다는 것은 명백하다. ■

정 리 6 $N \times N$ 사이클릭 베니안 네트웍은 최대 $3/2(Nlog_2N + N)$의 고장난 요소들을 감내할 수 있다.

증 명: 사이클릭 베니안 네트웍에서, $N/2$개의 SE들을 가진 단계 i, $1 \le i < \log\_2N$,는 N개의 출력 요소들과 N 개의 연쇄-출력 요소들을 갖을 것이다. 이 스위치가 감내하는 최악의 경우는 단계 i의 각 SE의 두개의 출력 요소들과 두개의 연쇄 출력 요소 중 세개가 고장난 경우이다. 그러므로, 단계 i는 최대 $N/2 \times 3 = 3/2\ N$ 고장 요소들을 감내할 수 있다, 따라서 스위치가 감내할 수 있는 고장들의 최대 갯수는 다음과 같다:

$$\sum_{i=0}^{(\log_2 N)-1} \frac{3}{2}(Nlog_2N - N).$$

마지막 단계에서, 스위치는 각 SE 쌍에서 (그림 5 참조) 하나의 출력 요소와 세개의 연쇄-출력 요소들이 고장인 것을 감내할 수 있다. 각 시스템 구성요소는 그림 4에서 보는 것과 같이 단계 1의 두개의 입력 요소들에 접근할 수 있다. 따라서, 한 스위치는 입력 요소들 중 최대 반이 고장나는 것을 감내할 수 있다. 전체적으로 보면, 감내할 수 있는 고장 요소들의 수는 $3/2\ (Nlog_2N - N) + N/2 + 2N$
$= 3/2\ (Nlog_2N - N)$ 이다. ■

더 나아가서, 스위치 내의 부품의 고장은 실제로는 임의로 일어난다. 다음으로는 스위치 고장을 초래하는 고장들의 기대치를 분석 하겠다.

2. 스위치 고장을 초래하는 고장들의 기대치

다음의 분석에서, 각 고장 패턴은 동일한 발생 확률을 갖는다고 가정한다. 분석의 편리함을 위해서, 스위치의 요소들을 세 부분들로 다음과 같이 나눈다. (1) 첫번째 단계의 입력요소들, (2) 중간 단계들로 불리우는, 단계 1로부터 (log_2N) - 1까지에 있는 출력 요소들과 연쇄-출력 요소들, 그리고 (3) 마지막 단계의 출력 요소들과 연쇄-출력 요소들이 그것이다.

먼저, 중간 단계 s, $1 \le s \le (log_2N)$ - 1,에 관해서 생각해 보자. 단계 s가 i개의 고장난 요소들을 가지면서도 다음 단계로의 연결을 제공할 수 있는 확률 $q_s(i)$ 는 다음과 같다.

$$q_s(i) = \frac{T_s(i)}{\binom{2N}{i}}, \qquad (1)$$

여기서 $T_s(i) = \sum_{j=0}^{\lfloor i/2 \rfloor} \sum_{k=0}^{\lfloor (i-2j)/2 \rfloor} \binom{N/2}{j} \binom{N/2-j}{k}$

$$\cdot \binom{N/2-j-k}{i-3j-2k} 4^{j} 6^{k} 4^{i-3j-2k} - X_s(j,k)$$

그리고 $X_s(j,k) = \begin{cases} 2^j, j = N/2\text{경우} \\ 0, \text{나머지경우} \end{cases}$

이 식의 의미는 다음과 같이 설명 된다. 한 중간 단계 s가 i개의 고장난 요소들을 그안에 가지면서도 고장나지 않는 가능한 고장 패턴들의 총수를 계산하면: 각 중간단계에는 $N/2$개의 SE가 있고, 각 중간단계의 요소들의 총수는 $2N$ 이다. 중간 단계 s에 j개의 SE들이 세개의 고장난 요소들을 가지고 있고, k개의 SE들이 두개의 고장난 요소들을 가지고 있으나, 아직 동작한다고 가정하자. 모두 i개의 고장들이 있다면, 각 SE에 하나의 고장이 난 $i-3j-2k$개의 SE들을 가진다. 한 SE가 가질 수 있는 모든 가능한 하나, 둘, 그리고 세 개의 고장이 갖는 패턴들은 그 SE를 고장낼 수 없다. 가능한 고장 패턴들의 전체 수는 $T_s(i)$의 첫번째 항과 같다. 그런데, 동일한 단계내의 모든 SE들의 모든 출력 요소들이 고장나는 경우는 감내할 수 없다. 따라서, $T_s(i)$의 두번째 항, $X_s(j,k)$ 만큼의 고장 패턴은 배제하여야만 한다.

모두 $\binom{2N}{i}$개의 가능한 고장 패턴이 중간단계 s에서 있기 때문에, 등식 (1)에 보이는 것 같은 $q_s(i)$를 갖는다.

이 $q_s(i)$들을 가지고, 반복적 방법에 의해서 k개의 고장난 요소를 있는 경우의 중간 단계들의 생존 확률(survival probability)을 유도할 수 있다. 부 네트웍(subnetwork) S_j, $1 \le j < log_2N$, 을 다음과 같이 정의하자: 그것은 단계 j의 한 중간 단계와 뒤이어 오는 중간 단계들로 구성되어 있는 S_{j+1}로 구성되어 있다. 유의할 것은 하나의 $S_{(log_2N)-1}$는 곧 단계 $(log_2N) - 1$ 만을 담는다는 것이다. 다음에 오는 정리는 k개의 고장난 요소들을 가진 부 네트웍 S_j의 생존 확률, $Q_j(k)$,을 제공한다.

정 리 7

$$Q_j(k) = \sum_{I_j^k \in U_j^k} D_j(I_j^k)\, q_s(i_{j1})\, Q_{j+1}(i_{j2}),$$

여기서 $U_j^k = \{ I_j^k \mid I_j^k = \langle i_{j1}, i_{j2} \rangle,\ i_{j1} + i_{j2} = k \}$,

$Q_{(\log_2 N)-1}(i) = q_s(i) (s = (\log N) - 1\text{일때})$

그리고 $D_j(I_j^k) = \dfrac{\binom{L_j^1}{i_{j1}} \binom{L_j^2}{i_{j2}}}{\binom{L_j}{k}}$ 이다.

증 명:

i_{j1}와 i_{j2}가 S_j의 앞에서 언급한 두 부분들 각각에서 의 고장난 요소들의 수라고 가정하자. I_j^k가 $i_{j1} + i_{j2} = k$인 $\langle i_{j1}, i_{j2} \rangle$쌍이라고 하자. L_j가 부 네트웍 S_j내의 요소들의 총수라 하자, 그리고 L_j^1와 L_j^2는 S_j안의 두 부분들 각각의 요소들 각각의 총 수라고 하자. 여기서 $L_j^1 + L_j^2 = L_j$ 이다. $L_j^1 = 3/2N$이고 $L_j^2 = L_{j+1}$인것은 명백하다. 이때 경계 조건은 $L_{(log_2N)-1} = 3/2N$ 이다. 따라서 부 네트웍 S_j안에서 하나의 고장 패턴 L_j^k이 일어날 확률은 위의 $D_j(I_j^k)$ 이다.
그러므로, 부 네트웍 S_j가 k개의 고장난 요소들을 가진 후에도 계속 동작할 수 있을 확률은 위의 $Q_j(k)$ 이다. ■

따라서 단계 (log_2N) - 1로 부터 시작해서 단계 1로 돌아가는 방향으로 계산해서 등식 (4)를 계산할 수 있다. (log_2N) - 1 번의 반복적 계산들 뒤에, k개의 고장난 요소들을 갖는 중간 단계들의 생존 확률 $Q_1(k)$를 얻을 수 있다.

다음에서, 단계 1의 입력 요소들의 생존 확률에 관해서 살펴보자. 한 시스템 요소는 그림 4에 보이는 것 같이 단계 1의 두 입력 요소들에 접근할 수 있을 것이다. 단계 1의 k개의 입력 요소들이 고장이지만 그들이 스위치의 고장을 초래하지 않을 확률은 다음과 같이 주어진다. (이는 [14]의 것과 비슷한데, 본 논문의 완전성을 위해서 반복된다.)

$$Q_f(k) = \frac{\binom{N/2}{k} 2^k}{\binom{N}{k}}$$

이 식의 의미는 다음과 같이 설명 된다. 각 시스템 요소는 단계 1의 두 입력요소들에 접근한다. 이러한 두 입력 요소들을 하나의 **집합**으로 부를 것이다. 단계 1에는 $N/2$개의 그러한 집합들이 있을 것이다. 한 스위치는 만약 그러한 집합들 중 아무도 그안에 하나이상의 고장난 요소가 없다면 접근되어 질 수 있을 것이다. 그러한 집합안에 k개의, $0 \le k \le N/2$, 고장난 요소들이 있으나, 모든 시스템 구성 요소들이 스위치에 접근 가능하다고 가정하면, 가능한 고장 패턴들의 총수는 위 식의 분자와 같다. 마지막 단계가, 단계 log_2N 이, k개의 고장난 요소들을 가진후에 살아있을 확률은 다음과 같이 주어진다.

$$Q_l(k) = \frac{T_l(k)}{\binom{4N}{k}}, \text{여기서}$$

$$T_l(k) = \sum_{i=0}^{\lfloor k/5 \rfloor} \sum_{j=0}^{\lfloor (k-5i)/4 \rfloor} \sum_{m=0}^{\lfloor (k-5i-4j)/3 \rfloor} \cdot$$

$\sum_{n=0}^{\lfloor (k-5i-4j-3m)/2 \rfloor} C(k,i,j,m,n)\, 2^i\, 13^j\, 26^m \cdot 8^{k-5i-4j-3m-2n}$, 그리고 $C(k,i,j,m,n) =$

$$\binom{N/2}{i}\binom{N/2-i}{j}\binom{N/2-i-j}{m}\cdot\binom{N/2-i-j-m}{n}\binom{N/2-i-j-m-n}{k-5i-4j-3m-2n}.$$

이 식의 의미는 다음과 같이 설명 된다. 각 출력단은 마지막 단계의 두 출력 요소들로 부터 접근될 수 있다. 모든 SE 쌍은 시스템 구성요소로의, 그리고 다음 수준(level)의 SE 쌍의 한 SE로의 경로을 감내하는한 하나, 둘, 셋, 넷, 다섯개의 고장(들)을 감내할 수 있다. 그림 4에서 보인 것과 같이 시스템 구성요소를 연결하기 위해서 별도의 스위치를 사용한다. 출력단으로 가는 하나의 출력 요소와 다른 SE쌍으로 가는 하나의 연쇄-출력 요소가 각 SE에게 가용하면, 출력단들이 네트웍으로 부터 접근되어 질 수 있다고 가정한다. 마지막 단계에 대한 모형은 한 SE쌍의 SE들간에 존재할 수 있는 간접 경로들을 고려하지 않았기 때문에 보수적(conservative) 이다. 각 SE 쌍은 다섯개, 네개, 세개, 두개, 그리고 한개의 고장(들)에 대해 각각 두개, 열세개, 스물여섯개, 스물두개, 그리고 여덟 개의 고장 패턴들을 감내할 수 있다. 다섯 개의 고장들을 갖는 SE 쌍들이 i개, 그리고 네개의 고장들을 갖는 SE 쌍들이 j개 세개의 고장들을 갖는 SE 쌍들이 m개, 그리고 두개의 고장들을 갖는 SE 쌍들이 n개 있다고 가정하면, 하나의 고장을 갖는 SE 쌍들의 수는 k-5i-4j-3m-2n가 될 것이다. 따라서 k개의 고장들이 갖는 가능한 고장 패턴들의 총수는 등식 (7)의 분자와 같다. 따라서 등식 (7)에서 주어진 것과 같은 $Q_l(k)$를 갖는다. (별도의 스위치에 일어나는 고장은 출력 링크의 고장으로 간주될 것이다.)

구해진 $Q_f(k)$와 $Q_1(k)$ 그리고 $Q_l(k)$를 가지고, k개의 고장 요소들을 가진 후에도 전체 네트워이 살아있을 확률 $Q(k)$를 구할 수 있다. $I^k = \langle i_1, i_2, i_3 \rangle$를 i_1와 i_2 그리고 i_3이 각각 단계 1의 입력 요소들, 중간단계들의 출력과 연쇄-출력 요소들, 마지막 단계의 것들에서의 고장의 갯수라 할 때 $i_1 + i_2 + i_3 = k$인 세 쌍(triplet)이라고 하자.

정 리 8

$$Q_k = \sum_{I^k \in U^k} D(I^k)\, Q_f(i_1)\, Q_1(i_2)\cdot Q_l(i_3)$$

여기서 $U^k = \{ I^k \mid I^k = \langle i_1, i_2, i_3 \rangle , i_1 + i_2 + i_3 = k \}$ 그리고

$$D(I^k) = \frac{\binom{N}{i_1}\binom{L_1}{i_2}\binom{3N}{i_3}}{\binom{4N+L_1}{k}}$$

이다. 이 정리의 증명은 정리 7의 증명과 매우 유사하므로 생략한다.

$Q(k)$를 가지고, k개 또는 그보다 적은 고장들이 전체 네트웍 고장을 초래하는 $P(k)$로 표현되는 확률은 $P(k) = 1 - Q(k)$ 이다. 또한 다음과 같이, $p(i)$를 i 번째 고장이 전체 네드워크의 고장을 초래할 확률이라고 하자.

$$P(k) = \sum_{i=3}^{k} p(i)$$

그러면 $P(k)$에 관한 위의 두 개의 등식들로 부터, 모든 $p(i)$들 각각을 얻을 수 있다. 이제, κ를 전체 네트웍 고장을 초래하는 고장 요소들의 기대값이라고 하자. 그러면,

$$\kappa = \sum_{i=3}^{B} i p(i)$$

(여기서 $B = N\log_2 N + N + 1$.) 이다. 유의할 것은 정리 6에 의해서 $i=B$ 경우 전체 네트웍은 완전히 고장날 것이기 때문에 $i > B$ 경우 $p(i) = 0$ 라는 것이다. 다양한 네트웍 크기, N에 대한 κ는 표 1에서 주어진다. N이 증가함에 따라, κ가 또한 증가하는 것은, 네트웍이 커질수록, 사이클릭 베니안 네트웍은 더 많은 우회 경로들을 제공하므로, 더 많은 고장난 요소들에 대해 살아있을 가능성이 높다는 것을 의미한다. 이는 각 입출력단 쌍 사이에 단지 하나의 유일한 경로를 가지는 네트웍, 예를 들면 베니안 네트웍과 크게 비교되는 점이다. 또한, 사이클릭 베니안 네트웍이 [14]의 추가된 베이스라인 네트웍 (Augmented baseline network)보다 좋음에 유의하자.

3. 사이클릭 베니안의 고장까지의 평균시간

다음에서, 사이클릭 베니안 네트웍의 평균 고장나는 시간(*MTTF*)을 다음의 하나의 가정아래 구할 것이다: 고장은 상수인 비율 λ로 각 요소들에게 독립적으로 일어날 것이다. $R(t)$를 사이클릭 베니안 네트웍의 신뢰성 함수라고 하면,

$$R(t)=\sum_{i=0}^{E} Q(i)\binom{E}{i}\left(e^{-\lambda t}\right)^{E-i}\left(1-e^{-\lambda t}\right)^{i}$$

여기서 $E=N+3(Nlog_2N+1)/2$는 네트웍 내의 요소들의 총 수이다. 네트웍의 $MTTF$는 다음과 같이 얻을 수 있다.

정 리 9

$$MTTF=\frac{1}{\lambda}\sum_{i=0}^{E}\frac{Q(i)}{E-i}$$

이 정리의 증명은 [14]에 있다.

네트웍크들의 공정한 신뢰성 비교를 위해서, 먼저 고장율을 결정하는 하드웨어 복잡도에 관해 알아보자. 하드웨어의 복잡도가 로직 게이트의 수에 비례한다고 가정하면, 각 네트웍의 스위치 페브릭 SE의 하드웨어 복잡도는 표 1과 같다.

표 1. 스위치 페브릭의 하드웨어 복잡도

Component	Banyan	Tzeng's Switch	Cyclic Banyan
Input Cell Buffers	7168	10752	14336
Complete Interconnection	454	1020	1812
Output Cell Latches	224	336	448
Contention Control	54	132	196
Subtotal	7900	12240	16792
Deviation-tag Processing	0	0	1948
Additional Contention Control	0	112	448
Total	7900	12576	19188

하나의 단위 스위치 페브릭의 게이트 수를 계산하기 위해서, 우리는 다음과 같이 가정한다. NAND 게이트를 위해서 하나의 게이트가 필요하고, Exclusive-OR를 위해서 3개의 게이트가, Flip-Flop을 위해서 7개의 게이트가 필요하다. 또한 각 단위 스위치가 입력 셀 버퍼들과 출력 셀레취들, 그리고 이 입력셀 버퍼로부터 출력셀렛취로 셀을 전송하는 완전 연결(Complete Interconnection)들, 그리고 일반적인 충돌 제어 로직으로 구성됨을 가정한다.

입력셀 버퍼는 64 바이트들을 저장하기 위해서 512개의 플립플롭들로 구성된다. 출력셀 렛취는 2 바이트들을 저장하기위한 16 플립플롭들로 구성된다. 완전 연결을 구성하기위한 플립플롭들의 수는 (입력 셀버퍼의 수)×(출력 셀렛취들의 수)×(2 바이트 대역폭)으로 계산된다. 여기에 추가적으로 이탈 꼬리표k를 처리하기 위한 로직과 정의 7에서 정의된 충돌 해결을 위한 로직이 필요하다.

위의 표 1에 보인 바와 같이, 스위칭 네트웍의 하드웨어 복잡도는 셀버퍼와 완전상호연결에 의해 결정된다. 표 1에 나타난 바와 같이, 본 네트웍에서의 하나의 추가된 2×2 SE는 근본적으로 하나의 4×4 크로스바 스위치이기 때문에, 한 2×2 추가된 SE의 한 요소의 하드웨어 복잡도는 한 일반적인 2×2 SE에서의 요소의 그것의 대략 2 배라고 할 수 있다. 또한 [14]의 네트웍의 추가된 2×2 SE는 근본적으로 하나의 3×3 크로스바 스위치이기 때문에, 한 2×2 추가된 SE의 한 요소의 하드웨어 복잡도는 한 일반적인 2×2 SE에서의 요소의 그것의 대략 1.5 배라고 할 수 있다.

따라서 공정한 신뢰성 비교를 위해, 각 네트웍의 요소들의 고장율들을 하드웨어 복잡도를 고려하여, 각각 다음과 같이 가정한다. 추가된 2×2 SE 요소의 고장율이 λ라고 가정할 때, 일반적인 2×2 SE의 한 요소가 고장날 가능성을 $(1/2)\lambda$ 라고 가정한다. 또한 [14]의 네트웍의 추가된 2×2 SE의 경우 $(2/3)\lambda$ 라고 가정한다. 일반적인 델타 네트웍의 $MTTF_0$는 다음과 같다 [14].

$$MTTF_0=\int_0^{\infty} e^{-(1/2)\lambda E_0 t}dt=\left(\frac{1}{2}\lambda E_0\right)^{-1}$$

여기서 $E_0=N+Nlog_2N$.

일반적인 델타 네트웍에 대해 사이클릭 베니안 네트웍의 신뢰성의 증가를 분석하기 위해서, [14]의 표를 확장 할 것이다. 많은 MIN들이 위상적으로 동치이기 때문에 [14, 15]의 표를 간단히 확장할 수 있다.

표 2. 전체 네트워크들의 κ, MTTF

N	4	16	64
κ_T	4.5	11.5	27.2
κ_T	4.7	21.8	58.3
$\overline{MTTF_0}$	$(6.0\lambda)^{-1}$	$(40.0\lambda)^{-1}$	$(224.0\lambda)^{-1}$
$\overline{MTTF_T}$	$(2.3\lambda)^{-1}$	$(6.9\lambda)^{-1}$	$(17.2\lambda)^{-1}$
$\overline{MTTF}$	$(2.3\lambda)^{-1}$	$(6.0\lambda)^{-1}$	$(13.7\lambda)^{-1}$

표 2는 일반적인 델타 네트웍과 Tzeng [14]의 추가된 네트웍, 그리고 본 사이클릭 베니안 네트웍의