보호 스위칭에 의한 경로 설정에 있어서 서비스 보장을 위한 복구 경로의 소비 대역 분석

이 황 규[†] · 홍 석 원[††]

요 약

빠른 복구 시간과 함께 서비스의 보장은 망의 신뢰성을 보장하는데 있어서 주요한 목표이다. 보호 스위칭 방식에 의하여 특정 서비스를 요구하는 세션의 서비스 품질을 보장하는 한 가지 방법은 작업 경로를 설정할 때 작업 경로의 대역을 보장하는 복구 경로를 함께 설정하는 것이다. 작업 경로의 대역을 보장할 수 있는 복구 경로를 설정할 때 다른 작업 경로에 대한 복구 경로의 대역을 서로 공유하면 대역의 소비를 감소시킬 수 있다. 본 논문에서는 보호 스위칭에 의한 복구 경로 설정에 있어서 작업 경로상에 최대로 할당된 링크의 대역을 기반으로 복구 경로의 공유 대역을 결정하는 방안을 설명한다. 그리고 이러한 단순 공유 방안의 문제점과 원인을 지적하고 이것을 실제망에 적용했을 때의 결과를 시뮬레이션을 통해서 보여준다. 그리고 이러한 문제점을 해결하기 위한 방안으로 링크 데이터베이스를 통한 완전 공유를 구현할 수 있는 방안을 제시하고 그 결과를 보여준다.

Analysis of the Bandwidth Consumed by Restoration Paths for Service Guarantee in the Protection Switching Scheme

Hwang-kyu Lee[†] · Sugwon Hong[††]

ABSTRACT

Fast restoration time and service guarantee are the important goals to achieve the network reliability. In the protection switching scheme, one way to guarantee service fro an application session if a network happens to fail is to establish the restoration path that amounts to the same bandwidth of the working path of the session at the same time. When we setup the restoration path, we can reduce the bandwidth consumption by the restoration path if the path can share the bandwidth required by the other paths. This paper explains the methods how to determine the shared bandwidth of the restoration path in the protection switching scheme, given the maximum bandwidth assigned to a link along the working path. We point out that such sharing algorithm can not reduce the bandwidth consumption by the restoration paths in some cases, which contradict the general conception. We explain why this can happen, and show the simulation results in real network topologies to prove our arguments. We explain the reason of the failure of the sharing effect by the simple sharing algorithm. Finally we propose the way of how we can overcome the failure of the sharing effect, using the complete sharing algorithm based on the link database and showing the results.

키워드 : 트래픽 엔지니어링(Traffic Engineering), 망 보구(Restoration), 대역 공유(Bandwith sharing)

1. 서 론

망의 신뢰성은 수많은 가입자를 서비스해야 하는 망에서 요구되는 주요한 요구사항으로서 여러 계층에서의 망의 보호, 복구 기능에 대한 연구가 진행되고 있다[1, 2]. 최근 인터넷의 경우에도 백본망과 핵심망에서 동기식 전송 방식(SDH/SONET) 혹은 파장 분할 다중화(WDM) 기반의 고속 트렁크 선로의 설치로 인해 링크의 장애가 발생할 경우 수많은 가입자 트래픽에 영향을 주게 되므로 망 복구 기능에 대한 고려가 높아지고 있다. 현재까지의 망 복구 기능은 주로 SDH/SONET 링 구조의 보호 기능에 의존하고 있다. 하지만 최근 일반적인 메쉬 구조의 망에서 서비스 차원의 복구를 위해서 물리 계층 보다 상위의 계층에서 세션 혹은 논리 채널 단위의 복구 보호에 대한 연구가 진행되고 있다[3].

망의 보호 복구 기능을 제공하는데 있어서 주요한 목표는 빠른 복구 시간이다. 노드 혹은 링크의 장애를 검출했을 경우 이것을 빠른 시간 안에 복구 기능을 수행하는 노드에게 통보하여 트래픽을 대체 경로(backup path)로 전달해야 한다. 이 논문에서는 트래픽이 전달되는 원래의 경로를 작업 경로(working path, primary path)로 부르고 장애가 발생했

※ 본 연구는 과학기술부 및 한국과학재단의 ERC 프로그램을 통한 지원으로 이루어졌으며 이에 감사를 드립니다.
† 정 회 원 : 텔코웨어 응용프로토콜1팀
†† 종신회원 : 명지대학교 컴퓨터소프트웨어학과 교수
논문접수 : 2001년 10월 11일, 심사완료 : 2003년 1월 16일

을때 대체되는 경로를 복구 경로(restoration path, backup path)로 부르도록 한다. 이러한 복구 방식은 복구 기능을 수행하는 노드의 위치, 복구 대상의 범위, 그리고 복구 경로의 설정 시점에 따라서 여러 가지로 분류할 수 있다. 복구 경로의 설정 시점에 따라서 크게 보호 스위칭(protection switching)과 사후 재경로 설정(on-demand rerouting)의 두 가지 방식으로 구분할 수 있다. 보호 스위칭의 경우 장애가 발생했을 경우 미리 설정된 복구 경로를 통해서 트래픽을 전송할 수 있다는 점에서 후자의 방식 보다 빠른 복구가 가능한 이점을 갖는다. 보호 스위칭의 경우 복구 경로를 통해 작업 경로의 트래픽을 대체하는 방법에 따라서 1+1, 1 : 1, 1 : n, n : m의 방식이 존재한다[3]. 복구 대상의 범위는 경로의 시점과 종점 사이의 전체 경로에 대해서 복구 경로를 설정하는 방법과 장애가 발생한 링크 혹은 노드에 가장 근접한 두 개의 노드 사이에서 복구 경로를 설정하는 부분 경로 복구 방법이 존재한다[9].

최근 망의 보호에 있어서 또 다른 목표로서 제시되고 있는 것은 서비스의 보장이다. 서비스를 보장하기 위해서는 장애가 발생했을 때 단순히 트래픽을 대체할 수 있는 경로를 제공하는것 뿐 아니라 원래의 작업 경로의 대역을 복구 경로에서도 보장해 주어야 한다. 물론 모든 세션의 트래픽에 대해서 복구 경로를 설정하거나 대역을 보장할 필요는 없겠지만 주요한 트래픽의 경우 장애가 발생하더라도 서비스를 보장해 줄 필요가 있다.

이러한 복구 경로의 설정의 과제는 최근 트래픽 엔지니어링과 조건 기반 라우팅(Constraint-based Routing)을 지원하는 신호 방식(signaling)이 제공됨으로 인해서 이것을 망 보호에 적용하는 방안이 제안되고 있다[4]. 또한 트래픽 엔지니어링을 지원하기 위한 라우팅 프로토콜을 확장하는 작업이 병행해서 진행되고 있으며, 대역을 보장하는 복구 경로의 설정 과제도 이러한 작업 기반 위에서 진행되고 있다[5-7].

본 논문에서는 보호 스위칭 방식에서 링크에 장애가 발생하였을 때 전체 경로를 복구의 대상으로 할 경우 작업 경로의 대역을 보장하기 위한 복구 경로 설정 방식에 대해서 분석한다. 이 논문은 망의 보호를 위해 망의 자원이 과도하게 소비되는 것을 피하기 위해서 하나의 복구 경로를 여러 작업 경로가 공유해서 사용할 경우 공유 대역을 결정하는 방법과 공유에 의한 대역 감소 효과에 초점을 맞추어 설명한다. 그리고 공유할 수 있는 대역을 결정하기 위해서 필요한 링크 정보의 수준을 기반으로 가능한 공유 방법에 대해서 논의한다. 불완전한 정보에 근거하여 공유 대역을 사용할 경우 일반적으로 생각하는 바와 같이 공유 효과를 얻지 못하는 경우가 발생하는데 이의 문제점과 그 원인을 분석하고 시뮬레이션을 통해서 그 결과를 보여준다. 마지막으로 완전한 공유를 구현하기 위하여 필요한 정보와 계산을 최소화할 수 있는 방법으로 각 노드에서 링크 데이터베이스를 사용한 구현 알고리즘을 제시한다.

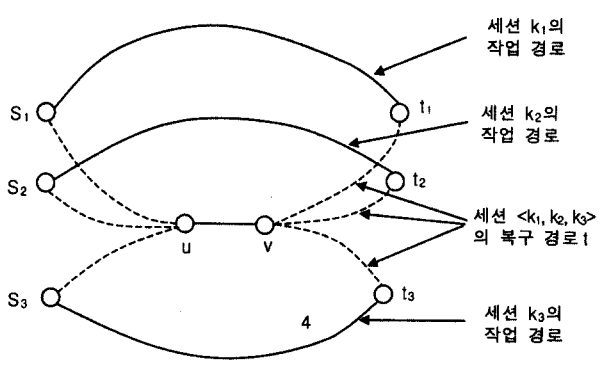
논문의 구성은 다음과 같다. 다음 장에서는 보호 스위칭에 의한 복구 경로 설정 방안에 대해서 설명한다. 3장에서는 제한된 링크 정보에 기반을 두고 공유 대역을 결정할 경우 공유 효과를 얻을 수 없는 경우와 그 원인을 설명하고 실제 망에서 시뮬레이션을 통해 분석 내용을 검증한다. 그리고 4장에서는 완전한 공유를 위한 구현 알고리즘을 설명하고 그 결과를 보여준다. 마지막으로 5장 결론에서는 지금까지의 논의를 기반으로 보호 스위칭 기반으로 복구 경로의 대역을 공유하기 위한 접근 방안을 제안한다.

2. 보호 스위칭에 의한 복구 경로 설정에 있어서 대역 할당 방식

2.1 보호 스위칭에 의한 복구 경로

보호 스위칭에 의한 복구 경로의 설정 방법은 경로상에서 장애가 발생했을 때 데이터 트래픽의 손실을 최소한으로 할 수 있는 방안이다. 이 방법은 작업 경로를 설정할 때 복구 경로를 함께 설정한다. 복구 경로의 시작 노드는 복구를 시행하는 노드(point of repair)가 되며 장애가 발생했을 때 복구 경로의 시작 노드는 트래픽을 작업 경로에서 복구 경로로 대체하여 전송을 하게 된다. 보호 스위칭의 주요한 특징은 복구 경로의 시작 노드는 장애가 발생한 지점의 노드(point of failure)가 아닌 작업 경로상의 노드가 된다는 점이다. 이 노드는 일반적으로 작업 경로의 입구 노드(ingress node)로 정할 수 있다. 또 하나의 특징은 복구 경로의 설정은 장애가 발생하기 전 작업 경로의 설정과 함께 미리 이루어진다는 점이다.

(그림 1)은 보호 스위칭에 의한 복구 경로 설정의 예를 보여주고 있다. 이 그림에서 보는 바와 같이 세션 k_1, k_2, k_3의 트래픽은 원래의 작업 경로상에서 장애가 발생하였을 때 링크(u, v)를 동일한 복구 경로로 사용하여 트래픽을 전달하게 된다.



(그림 1) 세션 k_1, k_2, k_3의 작업 경로와 보호 스위칭 복구 경로

2.2 복구 경로의 대역 공유 방식

복구를 위한 대역의 과도한 소비를 방지하기 위해서는 여러 작업 경로는 동일한 복구 경로상의 링크의 대역을 공유하는 방법이 있다. (그림 1)에서 보여주는 바와 같이 세션 $<k_1, k_2, k_3>$의 작업 경로는 동일한 복구 경로상의 링크의 대역을 같이 사용하고 있다. 본 논문에서는 여러 작업 경로들이 동일한 복구 경로상의 링크의 대역을 공유할 때 다음과 같은 조건을 충족하는 경로를 선택한다고 가정한다.

조건 1 : 작업 경로상의 어떠한 링크의 장애에 대해서도 트래픽의 대역을 보장한다.

조건 2 : 두 개 이상의 작업 경로상에서 장애가 동시에 발생하지 않는다.

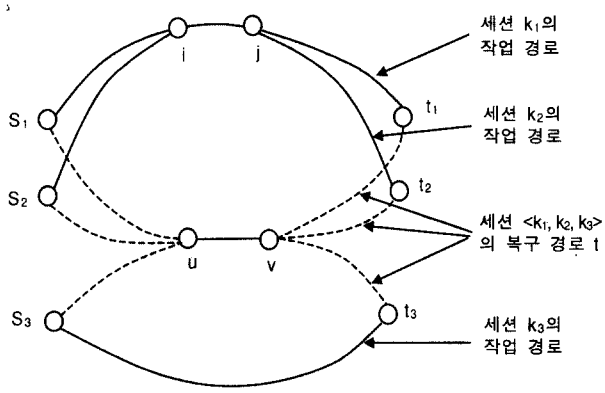
(조건 1)을 만족하기 위해서 복구 경로는 작업 경로와는 어떤 링크도 공유하지 않는, 즉 디스조인트 경로(disjoint path)를 선택한다.

(조건 2)는 현실적인 가정으로서 동시에 두 개 이상의 링크에 장애가 발생할 확률은 극히 작기 때문에 어떠한 링크도 공유하지 않는 두 개 이상의 작업 경로는 동일한 복구 경로의 대역을 같이 사용할 수 있다.

이러한 복구 경로 대역 공유의 한 방법은 (그림 1)에서 복구 경로의 링크(u, v)에서 요구되는 세션 k의 복구 경로의 요구 대역은 이미 링크(u, v)에 설정된 다른 세션의 복구 경로에 할당된 대역과 공유를 하는 방법이다. 이 경우 세션 k는 링크(u, v) 상에 설정된 다른 세션의 복구 경로 대역과 공유가 가능한지를 판단해야 한다. 만약 세션 k의 작업 경로와 동일한 링크를 통과하지 않는 세션이라면 공유가 가능하다. 이것은 서로 공유하는 링크가 없는 작업 경로를 갖는 세션들의 복구 경로는 서로 공유하더라도 (조건 2)에 의해서 서비스의 중단은 발생하지 않는다.

하지만 (그림 2)에서 보여주는 바와 같이 세션 k_1와 세션 k_2의 작업 경로가 적어도 하나의 링크(i, j)를 공유하고 있다고 가정하자. 이 경우 만약 두 세션의 복구 경로가 대역을 공유하고 있다면 두 세션이 공유하는 작업 경로의 링크, 즉 링크(i, j)에 장애가 발생했을 때 두 세션의 서비스를 보장할 수 없다. 따라서 세션 k는 자신의 작업 경로의 링크와 공유하는 다른 세션이 있는 경우에는 대역을 공유하지 않고 현재 설정된 대역에 추가하여 복구 경로의 대역을 설정해야 한다. (그림 2)에서 세션 k_2가 링크(u, v)에서의 필요로 하는 복구 경로의 대역은 세션 k_1과 공유할 수 없으며 별도의 추가 대역을 할당해야 한다.

이러한 공유 방식은 동일한 링크를 지나는 여러 세션 간에 최대한도로 복구 경로의 대역을 공유하면서 장애가 발생시 원래의 트래픽의 전달을 보장해 준다. 하지만 이러한 공유 여부를 결정하기 위해서는 각 링크를 지나는 모든 세션에 대한 작업 경로의 모든 링크 정보를 필요로 한다. 즉 링크(u, v) 상의 세션 k의 복구 경로를 설정하기 위해서 링크(u, v)를 지나는 복구 경로의 세션들의 작업 경로에 대한 링크 정보가 필요하다.



(그림 2) 동일한 링크를 공유하는 두 세션의 작업 경로와 복구 경로

2.3 복구 경로 대역의 단순 공유

복구 경로 대역을 공유하는 문제에 있어서 노드에서 복구 경로의 필요 대역을 판단하기 위해서 요구되는 많은 링크 정보와 복잡한 계산을 피하는 방법으로서, 주어진 링크 정보가 불완전한 정보일지라도 이것을 통하여 최대한 공유할 수 있는 복구 경로의 대역을 판단하는 방법을 적용할 수 있다.

한 가지 방법은 세션의 작업 경로상의 각 링크에서 이미 할당된 작업 경로에 의해 사용중인(혹은 예약된) 최대 대역을 복구 경로의 필요 대역으로 사용하는 것이다. 즉, 현재 설정 중인 작업 경로의 링크 중에서 현재 할당된 대역이 가장 큰 링크의 대역 만큼 복구 경로의 대역으로 할당한다. 따라서 복구 경로를 설정할 때 각 링크에 할당된 대역(현재 사용중인 대역 혹은 현재 예약된 대역) 정보만을 사용하고, 또한 다른 작업 경로와 중복된 링크가 있는지를 비교하는 복잡한 계산은 피할 수 있다.

이 경우 복구 경로는 작업 경로와 디스조인트 경로를 택하므로 (조건 1)을 만족하며, 현재 설정중인 작업 경로와 링크를 공유하는 다른 어떠한 작업 경로와 복구 경로를 같이 사용하더라도 장애가 발생시 작업 경로에서 요구되는 대역을 복구 경로는 보장을 해 줄 수 있으므로 (조건 2)를 만족한다. 이러한 복구 경로의 대역 설정을 이 논문에서는 단순 공유라고 부르기로 한다.

[8]은 이러한 공유 방식을 제안하여 공유로 인한 대역의 절감 효과를 비교하고 있다. 이러한 공유 알고리즘에서 복구 경로 설정을 할 경우 링크(i, j)와 (u, v)에서 필요한 대역을 계산하는 절차는 다음과 같이 기술할 수 있다.

a_{ij}^k : 링크(i, j)를 지나는 세션 k를 위한 작업 경로의 필요 대역

b_{uv}^k : 링크(u, v)를 지나는 세션 k를 위한 복구 경로의 필요 대역

m_{ij}^k : 링크(i, j)를 지나는 세션 k의 요구 대역

R_{uv} : 링크(u, v) 상에 남아 있는 대역

G_{uv} : 링크(u, v) 상에 설정되어 있는 복구 경로들 대역의 합

M : 작업 경로상의 링크들 중 최대 작업 경로 대역을 갖는 링크의 작업 경로 대역

F_{ij} : 링크(i, j) 상에서 작업 경로들 대역의 합

$$M = \max_{all(i,j) \in (s,t)} F_{ij}$$

$$a_{ij}^k = \begin{cases} m_{ij}^k & if \ R_{ij}^k > m_{ij}^k \\ \infty & if \ R_{ij}^k > m_{ij}^k \end{cases}$$

$$b_{uv}^k = \begin{cases} 0 & if \ M < G_{uv} \\ M - G_{uv} & if \ M > G_{uv} \text{ and } R_{uv} \geq M - G_{uv} \\ \infty & if \ R_{uv} \leq M - G_{uv} \end{cases} \quad (1)$$

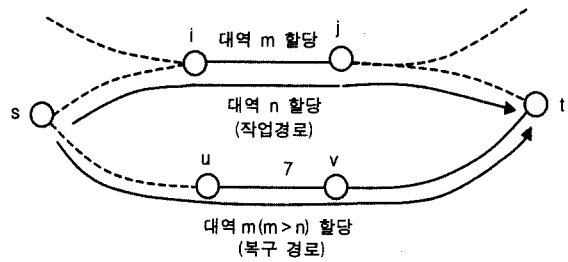
3. 단순 공유 알고리즘에 의한 대역 할당의 분석

3.1 대역 공유의 문제점

단순 공유 알고리즘의 경우 복구 대역의 공유를 통하여 할당된 대역의 소비를 최소화하는 공유의 효과가 항상 나타나는 것은 아니다. 대역 공유의 이점이 발생하기 위해서는 모든 링크간에 대역의 분산이 균형있게 분포된다는 점이 전제가 되어야 한다.

(그림 3)은 노드 s에서 t까지 대역 n을 요구하는 작업 경로를 보여주고 있다. 이 때 식 (1)의 M의 값이 링크(i, j)에 존재하고 이 값이 m(m > n)이라고 하자. 노드 s에서 t까지의 복구 경로는 링크(i, j)의 m값으로 할당을 하게 된다. 그런데 이 복구 경로의 대역 m값을 링크(i, j)를 통과하는 여러 작업 경로들이 항상 공유하는 것은 아니다. 이 경우 m 값이 n보다 훨씬 클 경우 복구 경로를 위해서 원래의 작업 경로보다 훨씬 큰 대역을 할당해야 하는 비합리적인 현상이 발생한다.

이러한 현상은 망에서 노드들의 그룹이 두 개 이상 존재하고 이 그룹들 간을 연결하는 백본 링크가 존재하는 경우 많은 트래픽이 이러한 링크에 집중될 때 심하게 발생하게 된다. 따라서 노드 s가 주어진 링크 정보를 기반으로 복구 경로의 공유 대역을 식 (1)의 M의 값으로 사용할 때 모든 링크에 대한 트래픽의 분산이 공유의 효과를 얻기 위한 전제 조건이 된다.

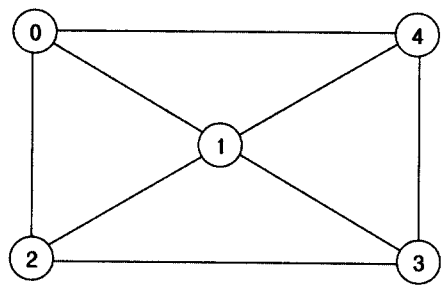


(그림 3) 복구 경로의 공유 대역 결정

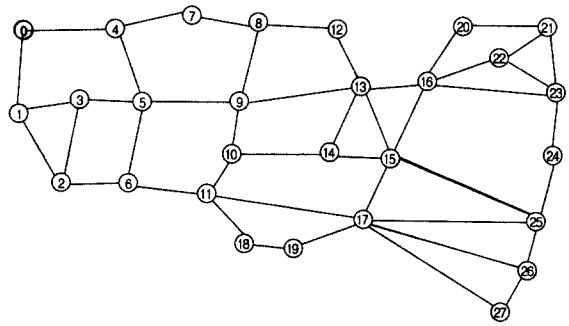
3.2 복구 경로에 할당된 대역의 비교

이러한 단순 공유 알고리즘의 문제점과 원인을 실제 망을 통해서 살펴보도록 한다. 이를 위해 노드 5개와 8개의 양방향 링크로 구성된 (그림 4)의 망과 28개의 노드와 45개의 양방향 링크로 구성된 (그림 5)의 망에 대해서 단순 공유 알고리즘을 적용하여 시뮬레이션을 수행하였다. (그림 5)의 망은 실제 미국의 전달망으로서 [10]에서 사용한 것을 그대로 적용하였다. (그림 4)의 망은 모든 링크에 트래픽의 분산이 이루어지는 경우를 예로 들기 위해서 인위적인 망을 설정하였다.

망의 각 노드들은 인접 노드와 연결된 링크에 대한 정보로서 현재 링크에서 사용되고 남은 대역에 대한 정보 가지고 있다. 시뮬레이션에서 최단 경로를 설정할 때의 링크 비용(cost)은 각 링크마다 남은 대역의 역수(1/남은 대역)로 하였다. 작업 경로의 시작점(source), 종점(destination)은 임의의 노드를 선택하며 요구대역도 임의의 값을 요구하도록 하였다.

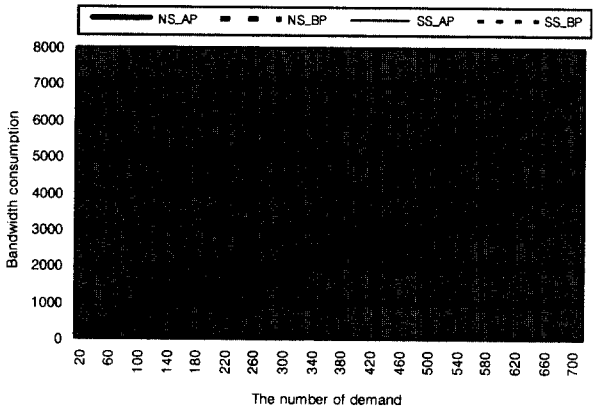


(그림 4) 망 토폴로지 1

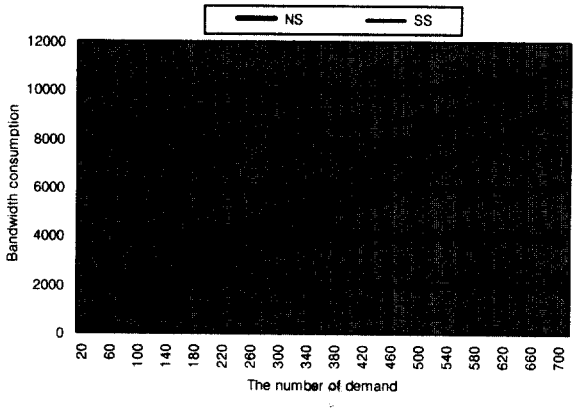


(그림 5) 망 토폴로지 2

(그림 6)과 (그림 7)은 망 토폴로지 1에서 모든 링크의 대역을 무한으로 하여 단순 공유 알고리즘을 적용하여 구한 작업 경로와 복구 경로의 소비 대역을 공유를 전혀 하지 않는 경우와 비교한 것이다. 이 그림에서 공유하지 않은 경우는 NS로, 단순 공유는 SS로, 그리고 작업 경로는 AP로, 복구 경로는 BP로 표시하였다. 이 시뮬레이션에서는 10번 수행한 결과의 평균을 구하였다. 매번 경로 설정에 대해서 1에서 10 사이의 임의의 크기의 대역을 요구하면 200번 설정 요구시 NS의 경우 복구 경로 요구 대역은 2031.5이고 단순 공유의 경우 1465.1이 필요하다. (그림 4)의 망 토폴로지에서는 단순 공유 알고리즘에 의한 대역 공유의 효과가 크게 나타나 공유를 하지 않는 경우 보다 소모 대역을 크게 절약하게 된다.

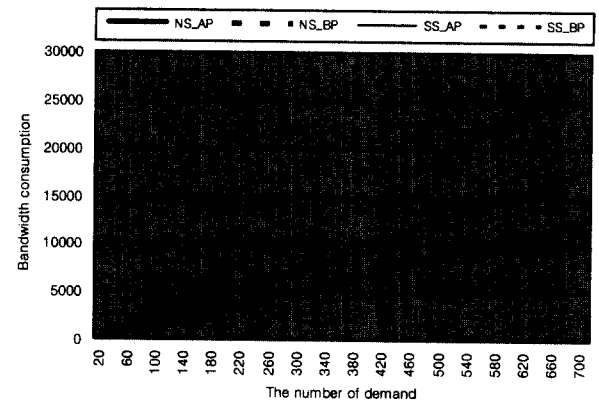


(그림 6) 망 토폴로지 1에서 작업 경로와 대체 경로의 대역 비교

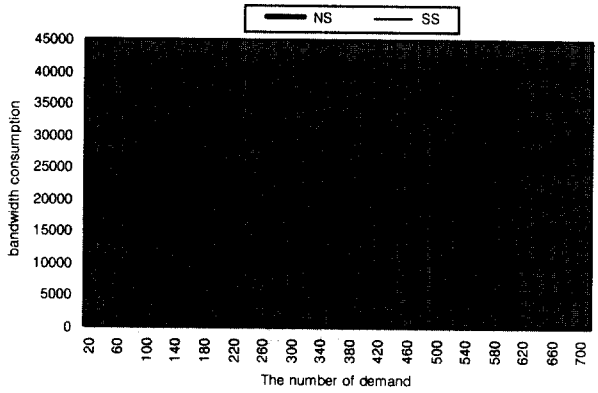


(그림 7) 망 토폴로지 1에 대한 총 대역 비교

망 토폴로지 2에 대한 시뮬레이션 결과는 (그림 8)과 (그림 9)에 나와 있다. 매번 경로 설정에 대해서 1에서 10 사이의 임의의 크기의 대역을 요구하면 200번 설정 요구시 공유를 하지 않을 경우 작업 경로 요구대역이 약 3346.9 그리고 복구 경로 요구대역이 약 4955.9의 대역이 소비되고, 단순 공유의 경우 작업 경로 요구대역이 약 3318.9 그리고 복구 경로 요구대역이 약 6748.8의 대역이 소비된다.



(그림 8) 망 토폴로지 2에서의 작업 경로와 대체 경로의 대역 비교



(그림 9) 망 토폴로지 2에서의 총 대역 비교

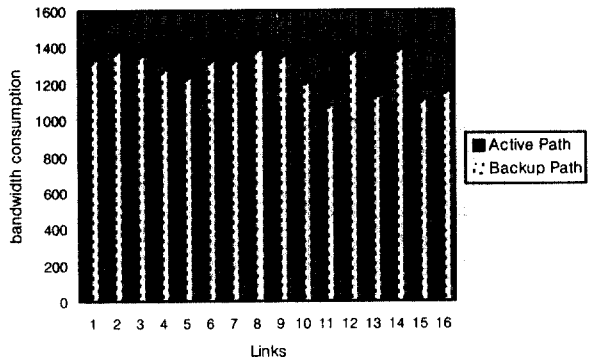
작업 경로의 대역은 두 가지 알고리즘이 같은 방법으로 설정 되기 때문에 랜덤 발생의 요인을 제외하면 동일한 대역을 요구한다. 그러나 복구 경로의 경우 단순 공유가 훨씬 많은 대역을 소비하는 것을 (그림 8)과 (그림 9)를 통해서 볼 수 있다. 이것은 3.1절에서 지적한 바와 같은 단순 공유 방식의 부적절한 할당의 경우가 심하게 나타나기 때문이다. 만약 망 토폴로지 1에서와 같이 각 링크의 작업 경로 대역이 거의 일정하게 분포한다면 특정 링크가 너무 커서 발생하는 대역의 낭비는 없을 것이고, 공유로 인한 이득을 최대한 얻을 수 있다. 하지만 각 링크에 작업 경로 대역이 고르지 않고 특정 링크들에 할당된 작업 경로 대역이 지나치게 많다면, 단순 공유 알고리즘의 경우 오히려 많은 대역이 필요하게 된다.

이러한 단순 공유 알고리즘의 공유 효과가 실패하는 원인을 알아보기 위해서 각 링크에 할당된 대역을 구하여 (그림 10)과 (그림 11)에 나타내 보았다. (그림 10)은 망 토폴로지 1의 경우 각 링크들에 할당된 작업 경로와 복구 경로의 대역을 보여주고 있다. 이 그림에서 보는 바와 같이 망 토폴로지 1에서는 각 링크에 할당된 작업 경로 대역의 차이가 그리 크지 않음을 볼 수 있다. 따라서 단순공유 알고리즘을 적용할 경우 (그림 6)과 (그림 7)과 같이 공유의 효과를 볼 수 있다.

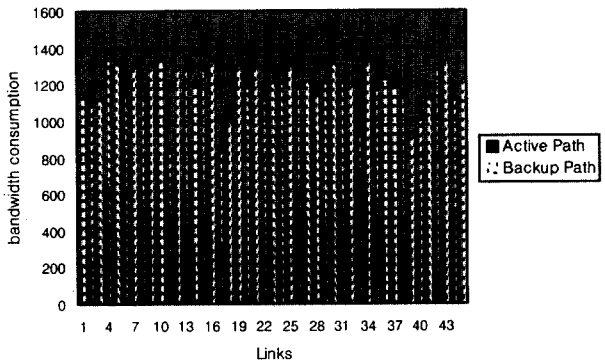
(그림 11)는 망 토폴로지 2의 결과를 보여주고 있다. 이 경

우 각 링크들에 할당된 작업 경로 대역을 보면 몇몇 링크들은 대부분의 다른 링크들에 비해 두 배 이상의 대역을 차지하고 있는 것을 볼 수 있다. 이러한 링크들은 두 개의 노드 그룹을 연결하는 백본 링크에 해당한다. 만약 어떤 작업 경로가 이러한 링크를 지나게 된다면, 복구 경로상의 모든 링크에는 그 크기 만큼의 대역을 할당해야 한다. 이 때문에 오히려 공유를 하지 않은 경우보다 더 많은 대역을 소비하게 된다.

이 결과에서 볼 수 있듯이 작업 경로상의 각 링크에 대역이 고르게 분포되어 있다면 단순 공유 알고리즘의 공유 효과 때문에 대역 소모를 감소할 수 있지만 만약 작업 경로의 대역이 링크에 불균형하게 분포될 경우 3.1절에서 지적한 바와 같은 이유로 인해 공유 효과는 오히려 대역 소모를 증가시키는 역효과를 초래하게 된다.



(그림 10) 망 토폴로지 1에서 각 링크에 할당된 대역



(그림 11) 망 토폴로지 2에서 각 링크에 할당된 대역

4. 완전 공유 방식의 구현

단순 공유 방식의 문제점은 불완전한 정보에 근거하여 복구 경로가 공유할 수 있는 대역의 크기를 결정하는 것이다. 임의의 링크(i,j)를 지나는 복구 경로의 필요 대역을 정확하게 결정하기 위해서는 노드 i는 다음과 같은 정보를 소유하고 있어야 한다.

$$\{\{k_1, k_2, \cdots, k_n\}, (L^{k1}, L^{k2}, \cdots, L^{kn}), (a^{k1}, a^{k2}, \cdots, a^{kn})\}$$

여기서 $K=\{k_1, k_2, \cdots, k_n\}$은 링크$(i,j)$를 지나가는 현재 설정된 복구 경로에 대한 세션들의 집합이다. L^k는 세션 k의 작업 경로가 통과하는 링크들의 집합이며 a^k는 세션 k의 요구 대역이다.

링크(i,j)를 지나는 세션 k의 복구 경로가 링크(i,j)에서 필요로 하는 대역 b^k_{ij}을 계산하기 위해서는 세션 k의 작업 경로가 모든 세션 K의 작업 경로 L과 중복되는 링크가 있는지를 조사하여야 한다. 만약 중복되는 링크가 존재할 경우에는 중복되는 링크를 갖는 세션 k_i의 요구 대역 a^{ki}의 값을 사용하여 b^k_{ij}를 계산해야 한다.

이러한 계산을 하기 위해서는 많은 양의 링크 정보와 계산을 필요로 한다. 예를 들어 링크(i,j)에 현재 50개의 복구 경로가 지나가고 있고 작업 경로의 평균 홉수가 10이라면, 500개의 링크에 대한 데이터베이스가 필요하다. 또한 홉 수가 10인 작업 경로가 하나 추가될 때 마다 5000번의 계산이 요구된다.

본 논문에서는 완전 공유를 위한 복구 경로의 필요 대역을 결정하기 위해서 이러한 많은 양의 링크 정보와 계산을 피하면서 이를 구현할 수 있는 방안을 제안한다. 완전 공유를 위해서 최종적으로 필요한 정보는 작업 경로상에서 공유할 수 없는 가장 큰 대역이다. 이것은 모든 노드가 링크 데이터베이스를 갖고 각 노드에 연결된 링크를 지나가는 복구 경로에 대한 작업 경로의 대역을 보관하고 있으면 구할 수 있다.

(그림 12)는 8개의 노드와 10개의 링크로 구성된 망을 보여 주고 있다. 이 망에서 2개의 세션에 대한 작업 경로와 복구 경로가 세션 1, 2의 순서로 설정된다고 하자. 세션 1은 대역 5를, 세션 2는 대역 7을 요구한다고 하자. 이때 링크(2, 3)은 두 세션의 작업 경로를 링크(6, 7)에서 공유하고 있기 때문에 링크(2, 3)을 지나는 두 복구 경로는 상호 대역을 공유할 수 없다. 노드 2는 이 망의 모든 링크에 대한 데이터베이스를 유지하고 있다. 첫 번째 세션 1에 대한 설정 요구가 발생하고, 이 세션의 복구 경로가 링크(2, 3)을 통과할 경우, 노드 2는 링크 데이터베이스에 이 세션의 작업 경로를 구성하는 링크들에 대해서 요구대역을 기록한다. 이 예에서 링크(5, 6), (6, 7), (7, 3)에 요구 대역 5를 기록한다. 그리고 이 링크들 중 최대 크기의 대역을 복구 경로의 필요 대역으로 결정한다.

두 번째 세션 2의 설정 요구가 발생하고 이 세션의 복구 경로는 노드 6-2-3-4를 통과한다. 노드 2는 이 세션의 작업 경로를 구성하는 링크(6, 7), (7, 8), (8, 4)에 요구대역 7을 더한다. 이때 링크(6, 7)은 12가 된다. 그리고 노드 2는 수정된 링크 데이터베이스에 기록된 현재 세션의 작업 경로의 링크 중 최대값 12를 링크(2, 3)에서의 b^k_{ij}로 결정한다.

이와 같은 링크 데이터베이스를 사용한 완전 공유 구현을 위한 알고리즘을 다음과 같이 기술할 수 있다. 이 알고리즘은 노드 i가 링크 데이터베이스 $B(u,v)$를 사용하여 링크(i,j)에서의 복구 경로의 필요 대역 b^k_{ij}을 결정하는 것이다.

a^k : 세션 k의 요구대역

b_{uv}^k : 링크(u, v)를 지나는 세션 k를 위한 복구 경로의 필요대역

$B(u, v)$: 링크(u, v) 상에 현재 설정되어 있는 작업 경로들 대역의 합

$G(u, v)$: 링크(u, v) 상에 설정되어 있는 복구 경로들 대역의 합

L : 망을 구성하는 모든 링크들의 집합

L^k : 세션 k의 작업 경로를 구성하는 링크들의 집합

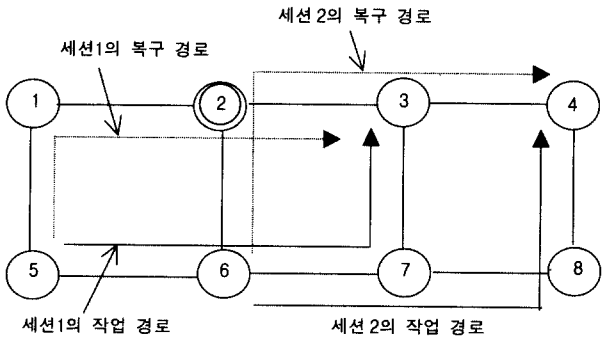
1 : 초기화 : $B(u, v) = 0 \quad for\ all (u, v) \in L$

세션 k의 복구 경로가 링크(i, j)를 지나갈 경우 :

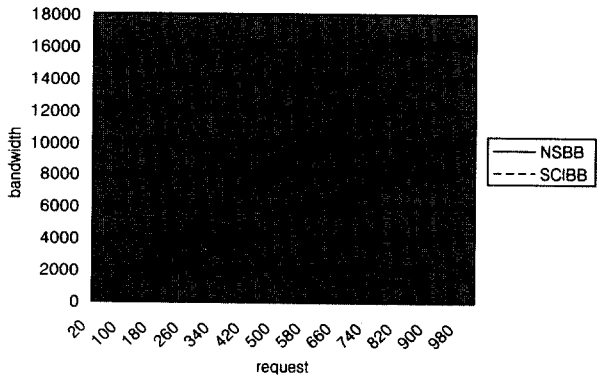
2 : $B(u, v) = B(u, v) + a^k \quad for\ all (u, v) \in L^k$

3 : $b_{ij}^k = \max_{(u, v) \in L^k} B(u, v) - G(i, j)$

이 알고리즘을 사용하여 (그림 5)의 토폴로지2의 망에 대해서 시뮬레이션을 통하여 소비 대역을 구한 결과가 (그림 13)에 나와 있다. 이 결과는 완전 공유를 사용할 경우 공유를 하지 않을 경우와 비교하여 소비 대역이 감소되는 효과를 보여주고 있다. 또한 (그림 9)와 비교하면 불완전한 정보에 의한 단순 공유를 사용할 때 비롯되는 비정상적인 소비 효과를 피하면서 복구 경로에 의한 대역의 공유 효과를 달성할 수 있음을 보여주고 있다.



(그림12) 세션 1과 세션 2의 작업 경로와 복구 경로 설정



(그림 13) 완전 공유 알고리즘을 사용한 망 토폴로지 2에서의 소비 대역 비교

5. 결 론

본 논문에서는 보호 스위칭 방식을 사용하여 복구 경로를 설정할 때 망 보호를 위해 과도한 대역의 소비를 줄이기 위해 복구 대역을 공유할 필요성에 대해서 설명하였다. 그리고 작업 경로상의 링크의 최대 사용 대역 정보를 기반으로 한 단순 공유 방법의 경우 일반적으로 인정하고 있는 바와 다르게 대역 공유에 있어서 함정이 있음을 지적하고 이것을 시뮬레이션을 통하여 실제 망에 적용한 결과를 보여 주었다.

그리고 단순 공유 알고리즘의 함정의 원인을 지적하고 이 함정을 피하면서 대역 공유의 효과를 얻기 위해서는 링크의 대역 분산(load balancing)이 필수적이며 이것이 전제되지 않을 경우 오히려 공유하지 않는 경우 보다 못한 결과를 초래할 수 있음을 보여 주었다. 마지막으로 많은 링크 정보와 복잡한 계산을 피하면서 완전한 공유를 구현하기 위하여 각 노드 마다 링크 데이터베이스를 사용한 알고리즘을 제안하고 그 결과를 보여주었다.

이 논문의 연구 결과로서 보호 스위칭에서 서비스의 보장을 위한 복구 경로 설정 방안으로 다음과 같은 방안을 제시한다. 먼저 보호를 위한 작업 경로의 수가 많지 않을 경우 대역의 공유 없이 복구 경로를 설정하도록 한다. 하지만 이것은 경로의 수가 증가됨에 따른 소비 대역이 격증하는 문제가 있게 된다. 경로의 수가 증가할 경우 망 보호를 위한 자원의 낭비를 최소한으로 하기 위해 단순 공유에 의한 경로를 할당할 경우 이 논문에서 지적한 바와 같이 링크의 대역 분산을 반드시 고려할 필요가 있다. 그리고 또 다른 해결책으로 논문에서 제안한 바와 같이 각 노드 마다 링크 데이터베이스 정보를 기반으로 작업 경로 간에 완벽한 공유를 할 수 있는 방안을 사용하는 것이다. 이 경우 각 노드에서는 링크 데이터베이스를 별도로 관리해야 하는 과제가 요구된다.

참 고 문 헌

[1] K. Owens et al., "Network Survivability Considerations for Traffic Engineered IP Networks," Internet Draft <draft-owens-te-network-survivability-01.txt>, July, 2001.

[2] W. S. Lai, et al, "Network Hierarchy and Multilayer Survivability," Internet Draft <draft-team-tewg-restore-hierarchy-00.txt>, July, 2001.

[3] V. Sharma et al, "Framework for MPLS-based Recovery," Internet Draft <draft-ietf-mpls-recovery-frmwrk-03.txt>, July, 2001.

[4] D. Awduche et al, "Requirements for Traffic Engineering Over MPLS," RFC 2702, August, 1999.

[5] K. Kompella et al, "OSPF Extensions in Support of Gen-

eralized MPLS," <draft-ietf-ccamp-ospf-gmpls-extensions-07.txt>, 2002.

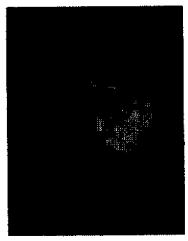
[6] T. Yagyu et al, "Extensions to OSPF-TE for supporting shared mesh restoration," <draft-yagyu-gmpls-shared-restoration-routing-00.txt>, June, 2002.

[7] D. Katz et al, "Traffic Engineering Extensions to OSPF," <draft-katz-yeung-ospf-traffic-06.txt>, 2002.

[8] M. Kodialam and T. V. Lakshman, "Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration," INFOCOM 2000.

[9] M. Kodialan and T. V. Laskshman, "Dynamic Routing of Locally Restorable Bandwidth Guaranteed Tunnels using Aggregated Link Usage Information," INFOCOM 2001.

[10] K. Murakami, "Optical Capacity and Flow Assignment for Self-Healing ATM Networks Based on Line and End-to-End Restoration," IEEE/ACM Trans On Networking, 6(2), April, 1998.



이 황 규

e-mail : morion@telcoware.co.kr

2000년 명지대학교 정보통신공학과(학사)

2002년 명지대학교 정보통신공학과(석사)

2002년 현재 텔코웨어 응용프로토콜1팀

관심분야 : MPLS, UMTS

홍 석 원

e-mail : swhong@mju.ac.kr

1979년 서울대학교 물리학과(학사)

1988년 North Carolina State University 전산학과(석사)

1992년 North Carolina State University 전산학과(박사)

1993년~1995년 한국전자통신연구원 선임연구원

1995년~현재 명지대학교 컴퓨터소프트웨어학과 부교수

관심분야 : 망 구조 및 프로토콜 설계, 성능 분석