MPLS 망에서 칼라 스레드의 루프방지 알고리즘 개선

Improvement of the Colored Thread Algorithm to Prevent Loop in MPLS Network

전 환 식* 김 한 경**
Whan-Sic Chun Han-Kyoung Kim

요 약

MPLS망에서 루프 경로가 형성되는 것을 방지하기 위한 방안으로 Ohba는 칼라 스레드 알고리즘을 제안하였다. LSP를 설정할 때 중간 노드가 칼라와 홉 카운트, TTL 값으로 표현된 스레드 정보를 LDP 메시지를 이용하여 다운스트림 노드에게 전달한다. 그 다음같은 칼라의 스레드가 다시 접수되면, 루프 경로가 형성되었음을 감지하고 다른 경로의 탐색을 시도한다. 메시지가 egress 노드에 도착하면 ingress 노드까지 업스트림으로 확인 메시지를 역전송하여, 루프가 없는 경로를 설정한다. 이를 위해 Ohba가 스레드의 상태를Null, Colored, Transparent의 세 가지로 정의한 것을 Extending, Merging, Stalling, Null, Transparent의 5가지로 확장하여 정의하고, 이에 따라 관련 FSM과 TCB를 재정의하여 스레드 상태의 모호성으로 인한 잘못된 동작과 오버로드의 발생을 개선코자 하였다. 그리고, Stalling 상태에서 스레드 생성을 회피함으로써, 성능의 향상을 유도하였다.

Abstract

Ohba has suggested the Colored Thread Algorithm to prevent looped path when LSP is to be setup. An immediate node sends thread information such as color, hop count, TTL, to downstream node via LDP message for the set-up of LSP. Afterward, decides that the looped path is formed when it receives a message with a same colored thread that was sent to downstream node, and it searches another path. If the message reaches to the egress node, then acknowledged message is sent to upstream node as reverse direction to the ingress node to set up loop-free path. For the algorithm, Ohva has defined three thread states as Null, Colored, Transparent. In this paper, the state of thread is extended to 5 states such as Extending, Merging, Stalling, Null, and Transparent. By the way, related FSM and TCB was redefined to make clear the ambiguity of thread states which causes faulty actions and to remove overhead. And, to imporve performance, it limits to generate a thread in the state of Stalling.

1. 서 론

MPLS(Multi-Protocol Label Switching) 망에서 루프 경로가 설정되면, 루프가 해결될 때까지 패킷이 루프가 형성된 경로의 네트워크 자원을 소모 시켜 네트워크의 성능을 저하시키게 된다. Ohba는 루프 경로가 설정되는 것을 방지하기 위하여 칼라 스레드 알고리즘을 제안하였다[1,2]. 그러나, 이 알고리즘에서의 스레드 상태와 FSM(Finite State Machine), TCB(Thread Control Block)로는 스레드를 정확하게 표현하기가 어렵다. 본 논문은 이들을 재정의하여 스레드 상태의 모호성을 제거하고, 새로운 LSR(Label Switching Router) 아키텍처를 제안하여 스레드 관리와 운용의 효율성을 높임으로써 기존의 칼라 스레드 알고리즘에 대한 개선을 하고자 한다.

기존의 라우팅 프로토콜에서는 경로 설정시의 루프형성은 허용하지만, 패킷 헤더의 TTL(Time-to-Live) 값을 지정하고 이 값을 네트워크의 노드를 통과할 때마다 1씩 감소시켜 값이 0에 이르면 패킷을 제거하는 방법으로 영향을 완화 시키는 방식을 택하였다[1,2,3]. MPLS 망에서 이 문제를 해

* 준회원 : 영우통신(주) 연구소 직원
wschun@ywtc.com
** 정회원 : 국립 창원대학교 공과대학 컴퓨터공학과 교수
hkim@sarim.changwon.ac.kr

결하기 위하여 경로 벡터(path-vector/diffusion) 방법과 칼라 스레드(Colored thread) 방식의 알고리즘을 제안하고 있는데, 전자가 RIP(Routing Information Protocol) 또는 RSVP(Resource Reservation Protocol)에서 전통적으로 적용해 오던 방법임에 비해, 후자는 이전에는 제안된 바가 없으며 그래서 실제 적용 사례가 없는 전혀 새로운 알고리즘이다[3]. 그리고, 본 논문에서 제안한 개선된 알고리즘은 마크드 페트리 넷트(Marked Petri Net)를 이용하여 모델링하고, 시뮬레이터를 이용한 시뮬레이션으로 평가를 하였다[4,5].

2. 관련 연구

2.1 LSP의 설정

MPLS 노드에 의해 LSP(Label Switched Path)를 설정하는 것은 소스 노드에 의해 경로 정보를 명세화 하는 명시적 라우팅(explicit routing)과 계층3의 네트워크 라우팅 기능을 이용하여 다음 홉을 찾아가는 hop-by-hop 방법이 있다[6]. 특히, 후자는 LDP(label Distribution Protocol) 메시지를 사용하여 경로상 각 노드에서의 해당 FEC(Forwarding Equivalence Class)에 대한 레이블 바인딩 정보를 인접 노드와 교환하는데, 결과로 각 노드들은 LIB(Label Information Base)를 유지하게 된다. LIB를 구성할 때, 입력 링크의 state에 대하여 대응되는 출력 링크의 state를 정적으로 명시하는 방법과, 여러 입력 링크의 각 state들을 하나의 outgoing link state로 병합하는 방법이 있는데, VC-병합 기능을 활용하여 scalability를 향상 시키는 후자의 방법이 상대적으로 복잡하며, 특히 동시에 multiple state merging의 경우 특별한 주의가 요구된다[6,7,8].

MPLS 망의 LSR은 FEC를 기준으로 경로를 설정하여, 패킷을 포워딩한다. 각 LSR은 LSP를 설정하기 위하여 FEC와 바인딩 정보를 이용하게 되는데, 자신이 가진 정보를 이용하는 로컬 바인딩과 이웃 LSR의 정보를 이용하는 리모트 바인딩을 통하여, 포워딩 테이블을 완성한다. 바인딩 정보는 데이터 패킷이나, 라우팅 프로토콜 및 LDP를 통해 분배가 가능하지만, 칼라 스레드 알고리즘은 LDP를 이용하여 downstream-on-demand 방식으로 분배한다[3]. 각 노드들은 Hello 메시지를 통해 LDP 상의 이웃 노드들을 관리하고, LDP 세션을 설정하여 LDP 메시지를 교환한다. MPLS 망으로의 진입 노드인 ingress 노드는 외부 망으로부터 트래픽이 발생하면 LSP 설정을 시도한다. LSP가 설정되고 패킷이 유입되면, 해당 FEC에 해당되는 레이블을 바인딩하여 다음 노드로 전달함으로써 레이블 스위칭이 수행될 수 있도록 한다. 이 과정이 반복되면서 패킷이 MPLS 망에서 다른 망으로 진출하는 egress 노드에 도착하면, 레이블을 제거하고 다음 망으로 전달한다[7]. 이때, MPLS 망에서 각 LSR이 지닌 정보를 관리하는 과정에서 일시적으로 "Routing transient" 상태가 발생하여 루프 경로가 형성될 수 있는데, 이럴 경우에는 패킷들이 목적지에 도착하지 못하고, 무한히 망 사이를 떠돌며 라우터의 버퍼, CPU, 대역폭과 같은 네트워크 자원을 소모 시켜, 자원 부족현상을 일으켜, 네트워크 성능을 저하시킨다[5,6].

2.2 Ohba의 칼라 스레드 알고리즘

경로를 구성하는 각 링크의 순서열을 스레드라고 정의한다. 스레드를 상호 구별하기 위하여 칼라(Color)를 부여하는데, 칼라는 발신 노드의 IP 주소값과 순서번호값을 이용하여, 시간적으로나 공간적으로 유일한 값을 갖도록 하며, 각 노드는 입·출력 링크별로 스레드의 칼라를 기억해야 한다. 홉 수(Hop Count)는 리프(leaf) 노드에서 메시지를 생성할 때 1의 값으로 배정되어 전송하면, 경로 상의 각 중간 노드들은 1씩 증가 시킨다. 이 값은 각 노드에서 스레드의 동작을 결정하는 중요한 요소로 참조되는 값이다. 그리고, 스레드 TTL은 메시지의 루핑을 방지하기 위하여 스레드 생성시에 최대 TTL 값을 설정하고, 메시지가 전

Color		
Hop count	TTL	Reserved

(그림 1) 스레드 객체

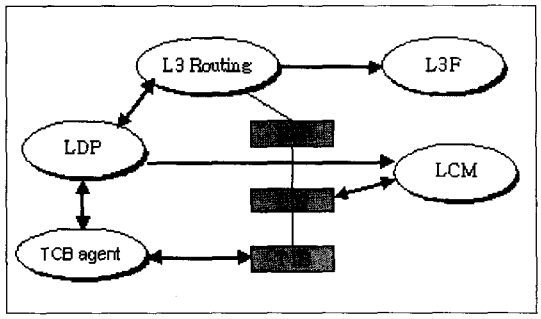
달되는 과정에서 각 노드는 1씩 감소시킨다. TTL 값이 0에 이르면, 루프가 형성되었다고 가정하고 메시지를 폐기한다. 칼라 스레드는 Label Request/ Mapping 메시지와 같은 LDP 메시지의 옵션 부분을 이용하여 구현할 수 있으며, 스레드 객체의 형태는 그림 1과 같다.

칼라 스레드 알고리즘은 스레드의 기본 동작으로 5가지를 정의한다. ingress 노드 및 리프 노드들은 경로설정 요청을 접수하면, 스레드 Extending 동작에 의해 스레드를 확장해 줄 것을 다음 노드에게 전송하고, 응답을 기다린다. 이때, 스레드의 칼라 홉 카운트는 현재 노드의 출력 링크와 수신 노드의 입력 링크에 반드시 기억되어야 한다. 스레드 Merging 동작은 노드가 새로운 스레드를 수신하였을 때, 이미 동일한 목적지로 가는 다른 칼라의 스레드가 존재하는 경우, 이 스레드를 기존 스레드에 병합시키고, 별도의 메시지를 발생시키지 않음으로써 스레드 트래픽을 감소시킨다. 스레드 Stalling 동작은 칼라 스레드가 어떤 노드에서 루프를 형성하게 될 때, 루프 처리를 위하여 수신된 스레드를 수신 노드의 링크에 잠시 머물게 하는 동작이다. 스레드 Rewinding 동작은 스레드가 egress 노드에 도착하면 루프가 형성되지 않은 것으로 간주하여, 역경로로 메시지 생성자에게 Label Mapping 메시지를 이용하여 응답 메시지를 돌려주는 동작이다. 마지막으로 스레드 withdrawing 동작은 다음 홉 분실이나, 경로의 변경, 루프 형성 등의 이유로 다음 홉에게 경로 해체(teardown) 메시지를 보내어 다운스트림 경로의 일부를 해체하는 동작을 말한다. 이때, 어떤 노드가 경로 설정을 위해 extending한 스레드는 철수하게 된다.

3. 스레드 알고리즘의 개선

3.1 LSR 모델 제안

LSR 아키텍처를 그림 2와 같이 TCB 에이전트와 TIB(Thread Information Base)를 새롭게 추가하는 것으로 제안한다. L3 Routing은 일반적으로 Dijkstra의 SPF(shortest-path-first) 알고리즘을 사용하는데, LSP 설정을 위한 인터페이스 역할을 하지만, 시스템에 상당한 부담을 주게 된다. FIB(Forwarding Information Base)는 라우팅 프로토콜이 새로운 엔트리의 추가, 삭제, 갱신 등의 망의 변화를 알리면, 새 LSP를 설정, 제거, 재설정과 같은 이벤트를 생성시킨다. LCM(Label Control Block)은 인터페이스에서의 레이블 바인딩 정보를 유지하는 LIB (Label Information Base)를 관리하며, 설정된 연결을 유지한다. L3F 블록은 LSP 설정을 요청하는 트래픽을 처리한다. LDP 모듈은 LDP 메시지의 옵션 부분에 구현되는 스레드를 분류한다. 기존 LSR에 새롭게 추가되는 TCB 에이전트는 각 스레드의 TCB 상태를 관리하며, 이벤트나 수신한 메시지에 대해 적절한 동작을 취한다. TCB 에이전트가 참조하는 TIB는 LIB로부터 필요한 값을 참조하고, 각 스레드가 운반해 온 정보를 취합하여 각 링크가 가지는 스레드 오브젝트를 생성한다. TCB 에이전트와 TIB를 분리하여 LDP 모듈의 부담을 줄이며, 처리 속도를 개선할 수 있다[9,10].



(그림 2) LSR 아키텍처

3.2 FSM 과 TCB의 개선

3.2.1 TCB 정의

TCB는 FEC를 바탕으로 운용되는 스레드의 정보들로 구성된다. 스레드의 상태는 Null, Extending, Merging, Stalling, Transparent의 5가지가 있으며, 업스트림/다운스트림 이웃 노드의 주소, 입·출력 링크의 칼라 값, 홉 수, 레이블 정보 등이 포함된다. S-flag는 링크의 stalled 여부를 나타내고, C-flag는 다음 홉으로의 링크 존재 여부를 나타낸다. 칼라 스레드를 지원하는 LSP 상의 각 노드의 링크는 그림 3과 같은 TCB을 가지며, TCB agent는 이를 통해 스레드를 운용한다.

FEC	State	Incoming links					Outgoing links				
		neighbor address	color	hop count	label	S-flag	neighbor address	color	hop count	label	C-flag

(그림 3) TCB 구조

3.2.2 FSM의 개선

개선된 칼라 스레드 알고리즘은 7가지 이벤트를 정의한다. Next hop acquisition 이벤트는 주어진 FEC에 대해 경로 탐색 과정을 통하여 다음 홉을 찾는 동작을 요구하는 이벤트이며, Label Mapping 메시지에 의해 이벤트를 인식한다. Next hop loss 이벤트는 다음 홉을 인식하지 못하게 된 경우에 발생하는 이벤트로 다음 홉의 시스템이 다운되었거나, 링크가 끊어진 경우, 또는 루프 경로가 형성되었음을 감지할 때 생성하는 withdraw 메시지 등에 의해 발생된다. Next hop change 이벤트는 다음 홉을 변경하고자 하는 이벤트로, 보다 최적의 경로를 새로 탐지하였거나, 새로운 경로 설정이 필요한 경우에 발생하는 이벤트이며, Next hop loss 이벤트와 Next hop acquisition 이벤트의 결합된 형태이다. 경로 탐색이 진행되면서 스레드가 망의 egress LSR에 도착하면, 경로 탐색이 성공적으로 수행되었음을 나타내는 응답 메시지를 업스트림으로 발생시키게 되는데, 각 중간 노드는 이 메시지를 수신하여 Rewound 이벤트를 감지한다. Withdrawn 이벤트는 홉 간의 링크가 끊어진 경우나, 새로운 경로 탐지로 인해 경로해제 요청을 수신할 때 인지되는 이벤트이다. Thread 수신 이벤트는 노드의 입력 링크를 통해 입수된 여러 가지 스레드에 의해 발생하는 이벤트이다. 마지막으로 Reset to unknown 이벤트는 스레드의 홉 카운트 값을 unknown으로 설정하라는 이벤트로, stalling 상태에서 칼라 스레드를 수신하게 될 경우 발생하는 이벤트이다.

각 노드의 링크는 입력 링크와 출력 링크로 구분되는데, 입력 링크는 TCB의 S-flag를 이용하여 stalling 여부를 나타내고, 출력 링크는 TCB의 C-flag를 이용하여 현재의 출력 링크가 연결되어 있는지 여부를 나타낸다.

3.2.3 상태 정의

Ohba의 칼라 스레드 알고리즘은 스레드를 Null, Colored, Transparent의 세가지 상태로 정의하고 있다. 기존 알고리즘은 칼라 스레드가 extending동작을 취한 경우와 두개 이상의 스레드가 병합된 경우, 그리고 루프 상태에 있는 스레드를 모두 Colored 상태로 정의한다. 이와 같은 상태 구분은 알고리즘의 성능에 영향을 미칠 수 있으므로 명확히 구분되어야 하고, 각 상태에 따라 이벤트나 동작들도 명확히 정의되어야 한다. 따라서, 스레드 상태를 Null, Extending, Merging, Stalling, Transparent의 5가지로 확장하여 정의한다.

표 1의 FSM에서 Hmax는 최대 수의 링크 홉 카운트, Hout은 현재 다음 홉에 대한 출력 링크 홉 카운트, Ni는 Unstalled 입력 링크 홉 카운트, Hrec는 수신 스레드의 홉 카운트를 나타낸다.

① Null 상태

Null 상태는 출력 링크와 Unstalled 입력 링크가 없는 상태로 이벤트에 대한 동작은 표 1과 같이 정의한다.

② Extending 상태

새로운 스레드를 생성하여, 다음 홉에게 전달한 후, 응답 메시지를 기다리는 상태로 이벤트에 대한 동작은 표 2와 같다.

③ Merging 상태

Extending 상태에서 동일한 목적지를 가진 스레드를 수신하게 되면, 새로운 스레드를 생성하지 않고, 병합시킨다. Merging 상태의 동작은 표 3과 같다.

④ Stalling 상태

한 노드가 출력 링크로 전송했던 extending tm 레드가 입력 링크로 수신될 때, stalling 상태로 정의하며, 동작은 표 4와 같이 정의한다.

⑤ Transparent 상태

Extending 스레드에 대하여 rewound 메시지를 접수한 노드들의 입·출력 링크의 상태를 Transparent로 설정하며, 이때의 동작은 표 5와 같이 정의한다.

(표 1) Null 상태의 동작

스레드 수신	칼라값을 가졌고, egress 노드	스레드 ewinding 하고, 링크를 투명으로 설정	transparent
	칼라값을 가졌고, 리프 노드	칼라 값 변화없이, 스레드 extending	extending
Next hop acquisition	리프 노드	새 칼라 스레드 생성하고 extending	extending
그 외		수신 이벤트 무시	변화 없음

(표 2) Extending 상태의 동작

이벤트	조건	동 작	상태 변경
스레드 수신	투명이고, Hmax+1<Hout<unknown	홉수 갱신을 위하여 새 칼라 스레드를 생성하여,extending 한다.	변화 없음
	칼라값을 가졌고, Hmax<Hout	스레드 Merging 한다.	merging
	칼라값을 가졌고, Ni=0, 루프 형성, 리프 부적	수신 스레드 stalling, 스레드 철수를 초기화한다.	stalling
	칼라값을 가졌고, Ni>0, Hrec<unknown	Reset to unknown 이벤트 스케줄한다.	변화 없음
	칼라값을 가졌으나 그렇지 않은 경우,	기존 링크로 수신되면, 칼라 변경없이 extending하고, 아니면 칼라 변경후 extending한다.	변화 없음
Rewound		이전 홉들에게 스레드 rewinding하고, 입·출력 링크의 칼라를 투명으로 설정한다. Hmax+1<Hout면 투명 스레드를 확장하고, C-flag=0인 출력링크의 스레드를 철수한다.	transparent
Next hop acquisition	링크 존재이면	어떤 것도 하지 않음(보존모드일때).	변화 없음
	그 외	칼라 스레드를 생성하고 확장한다.	변화 없음
Next hop loss	출력 링크가 투명이고, 링크 유지 가능이며, 다음 홉 생존일 때	어떤 것도 하지 않는다.	변화 없음
	그렇지 않고, Ni=0일 때	다음 홉에 대한 스레드 철수를 시작한다.	null
	egress 노드	Rewound 이벤트를 스케줄한다.	transparent
그 외		이벤트를 무시한다.	변화 없음

(표 3) Merging 상태의 동작

이벤트	조건	동 작	상태 변경
스레드 수신	투명이고, Hmax+1<Hout<unknown	칼라 스레드 생성, extending한다.	변화 없음
	칼라값을 가졌고, Hmax<Hout	스레드 Merging한다.	merging
	칼라값을 가졌고, Ni=0, 루프 형성, 리프 노드 적격	스레드 철수를 초기화한다.	stalling
	칼라값을 가졌고, Ni>0, Hrec<unknown	Reset to unknown 이벤트 스케줄한다.	변화 없음
	칼라값을 가졌고 그렇지 않은 경우,	기존 링크로 수신 시 칼라 변경없이 스레드 extending 하고, 아니면 변경 후 extending한다.	변화 없음
Rewound		이전 홉들에게 스레드 rewinding을 전파하고, 입·출력 링크의 칼라를 투명으로 변경하고, Hmax+1<Hout면 투명 스레드를 확장. C-flag=0 인 출력링크의 스레드를 철수한다.	transparent
Next hop acquisition	링크 존재하면	어떤 것도 하지 않음(보존모드일때).	변화 없음
	그 외	칼라 스레드를 생성, extending한다.	변화 없음
Next hop loss	출력 링크가 투명이고, 링크 유지 가능, 다음 홉 생존이면	어떤 것도 하지 않는다.	변화 없음
	egress 노드이면	Rewound 이벤트를 생성한다.	null
	Ni=0	출력 링크 제거한다.	transparent
	그렇지 않으면	다음 홉에 대한 스레드 철수를 시작한다.	
그 외		이벤트를 무시한다.	변화 없음

(표 4) Stalling 상태의 동작

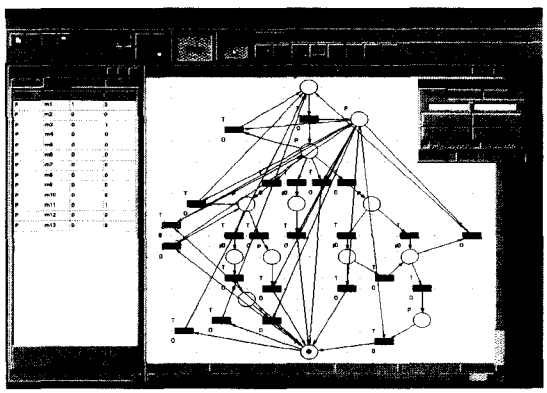
이벤트	조 건	동 작	상태 변화
Colored Thread	루프형성, 새 링크로 수신	홉 수가 Unknown인 새로운 칼라 스레드 생성, extending한다.	변화 없음
	Ni=0, 리프 부적	Stall, 스레드 철수를 시작한다.	null
	Ni>0, 스레드 홉 수가 Unknown이 아니면	Unknown 홉 카운트의 칼라 스레드 생성, extending한다.	변화 없음
rewound		스레드 rewinding한다.	transparent
reset to unknown		Unknown 홉 카운트의 칼라 스레드를 생성, extending한다.	변화 없음
Next hop acquisition	링크 존재이면	어떤 것도 하지 않는다(보존 모드).	변화 없음
	그 외	칼라 스레드를 생성하고 확장한다.	변화 없음
Next hop loss	출력 링크가 투명이고, 링크 유지 가능, 다음 홉 생존가능	어떤 것도 하지 않는다.	변화 없음
	egress 노드이면	Rewound 이벤트를 생성한다.	변화 없음
	Ni=0	출력 링크 제거한다.	null
Next hop change		칼라 스레드를 생성, extending한다.	extending
그외		이벤트를 무시한다.	

(표 5) Transparent 상태

스레드 수신	투명이고, Hmax+1<Hout	투명 스레드 extending한다.	변화 없음
	칼라값을 가졌고, egress 노드며, Hmax+1<Hout	수신 링크의 칼라를 투명으로 변경하고, 스레드 rewinding한다.	변화 없음
	칼라값을 가졌고, 그렇지 않은 경우,	새 링크로 수신되면, 칼라 변경후 스레드 extending하고, 아니면 칼라 변경없이 스레드 extending한다.	extending
withdrawn	Ni=0, 리프 노드면	다음 홉으로 스레드 withdrawing한다.	null
	Ni>0, Hmax 변화	투명 스레드 생성, extending 하여 홉 수를 갱신한다.	변화 없음
	그 외	상응하는 입력 링크를 제거한다.	null
Next hop acquisition		칼라 스레드를 생성하고, extending한다.	extending
Next hop loss	링크 유지 가능, 다음 홉 생존	아무것도 하지 않는다.	변화 없음
	Ni=0	스레드 withdrawing 초기화한다.	null
그 외		이벤트를 무시한다.	변화 없음

4. 알고리즘의 평가

제안한 칼라 스레드 메커니즘의 평가를 위하여 표 1에서부터 표 5에 의해 정의된 알고리즘을 마크트 페트리 네트로 그림 4와 같이 모델링하였다. 페트리 네트 시뮬레이터는 Visual Object Net++ Version 2.0(Evaluation)을 사용하였다. 페트리 네트에서 플레이스(Place)는 알고리즘의 상태 또는 조건을 나타내며, 트랜지션(Transition)은 시스템의 상태를 변화시키는 동작을 추상화하였다. 토큰(Token)은 플레이스의 조건의 진위 혹은 시스템의 가용 자원을 나타낸다. FSM에 따라 마크드 패트리 네트 모델을 생성하고, 페트리 네트 도구인 Visual Object Net++을 이용하여 안정된 알고리즘이 갖추어야 할 6가지 성질을 검증한 결과 표 6과 같은 결과를 확인하였다



(그림 4) MPN 도구를 이용한 제안 알고리즘의 페트리 네트 모델

(표 6) 제안 알고리즘의 페트리 네트 시뮬레이션

성 질	설 명	평 가
안전성	플레이스의 토큰이 1개를 초과하지 않으면, 플레이스는 안전하고 페트리 네트도 안전하다.	모든 플레이스에서 1개의 토큰이 존재하여 안전성을 만족함.
유한성	플레이스의 토큰의 수가 무한정 늘어나는 현상으로 시스템의 버퍼 및 레지스터의 오버플로우 발생	1개의 토큰만이 존재하여 유한성을 만족함.
보존성	페트리 네트에서 토큰 수가 일정하게 흐르는 성질	1~2개 토큰이 유통되어 만족함.
생동성	초기 상태에서 점화 순서에 따라 계속적으로 트랜지션이 일어나는지 점검하여 시스템의 교착상태의 발생 여부를 검증	계속적으로 점화가 일어나면서 모델이 수행되어 만족함.
도달 가능성	초기 상태에서 모든 플레이스로의 도달이 이루어지는지 점검	모든 플레이스로의 도달을 확인함.
가역성	초기 상태로의 복귀 가능성을 점검	초기 마킹 상태로 도달을 확인함.

5. 결론 및 향후과제

MPLS 망의 LSP 설정 과정에서 발생할 수 있는 루프의 형성을 미연에 방지함으로써, 네트워크 자원의 낭비로 인한 네트워크 기능의 저하를 감소시킬 수 있는 방안으로 Ohba에 의해 칼라 스레드 알고리즘이 제안되었다. 이 알고리즘은 기존의 경로 벡터 방식의 알고리즘에 비하여 네트워크의 크기에 무관하여 보다 유연하며, 보다 견고한 방법이다. 그러나, Ohba는 스레드의 상태를 Null, Colored, Transparent의 세 가지로 정의함에 따라 링크상의 스레드 상태를 표현하기에 모호하거나, 부적절한 경우가 발생하였다. 즉, 스레드의 루프 상태나, 병합 상태를 모두 Colored라는 하나의 상태로 정의함으로써, 상태의 모호성을 유발하고, 이로 인하여 복잡한 조건식을 만들게 되어 잘못된 동작 수행이나 오버로드를 유발할 수 있다. 그래서 스레드의 상태를 Extending, Merging, Stalling, Null, Transparent의 5가지로 확장하여 정의하고, 이에 따라 관련 FSM과 TCB를 재정의하여 스레드 상태의 모호성으로 인한 잘못된 동작과 오버로드의 발생을 개선코자 하였다. 그리고, Stalling 상태에서 불필요한 스레드를 네트워크에 생성시키지 않음으로써, 과도한 스레드 발생을 억제하였다. 또한 LSR의 구현을 위하여 아키텍처에서 TCB 에이전트와 TIB를 독립시켜 보다 효율적인 스레드 운용과 관리를 할 수 있도록 제안하였다.

제안 알고리즘의 평가를 위하여 모델링 도구인 마크드 페트리 네트을 이용하여, 안정성, 유한성, 보존성, 생동성, 도달가능성, 가역성의 6가지 기본 성질에 만족하는 것을 확인하였다. 그러나, 일반적으로 루프방지 알고리즘은 루프 경로가 많이 형성되는 망이나, 루프 경로가 오래 지속되는 망에 적절하지만 알고리즘의 오버헤드는 망의 성능을 떨어지게 한다. 따라서, 최근 IETF 등의 워킹 그룹에서는 루프 제어의 다른 방법으로 루프 탐사(Loop Probe) 알고리즘과 루프 회피(Loop Avoidance) 알고리즘에 관한 연구가 더불어 진행되고 있다. 향후, 이와 같은 알고리즘들과 연계하여 오버헤드를 감소시키는 방안을 칼라 스레드 알고리즘에 적용하여 보다 다양한 망에 유연하게 적용할 수 있는 방안이 제시되어야 할 것이다.

참 고 문 헌

[1] Y. Ohba, Y. Katsube, "MPLS Loop Prevention Mechanism", IETF RFC 3063, Feb. 2001.

[2] Y. Ohba, "Issues on Loop prevention in MPLS Networks", IEEE Communication Magazine, pp 64-68, Dec. 1999.

[3] B.Davie, Y. Rekhter, "MPLS Technolgy and Applications", Morgan Kaufmann Publishers.

[4] 이강수, "패트리 넷(Petri Net)에 관한 기술해설", 정보과학회지, 제1권 제2호, pp. 39-47, 6월 1983년.

[5] 이강수, "페트리넷트의 분류법", 한국정보과학회논문지, 제21권 제8호, p1379~1389, 8월 1994년.

[6] 이성창, "고속 인터넷 서비스를 위한 MPLS", 텔레콤 제15권 제1호, pp. 37-54, 대한전자공학회, 6월 1999년.

[7] L. Andersson, P. Doolan, N. Feldman, A.Fredette, B. Thomas, "LDP Specification", IETF RFC 3036, Jan. 2001.

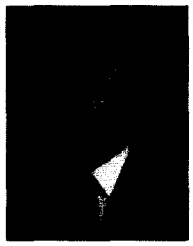
[8] R. Callon, N. Feldman, A. Fredette, G. Swallow, A.Viswanathan, "A Framework for Multiprotocol Label Switching", IETF Draft draft-ietf-mpls-framework-03.txt, Jun. 1999.

[9] G. Hagard, M. Wolf, "Multiprotocol Label Switching in ATM networks", Ericsson Review No. 1 1998.

[10] Eric C. Rosen, Arun Viswanathan, Ross Callon, "Multiprotocol Label Switching Architecture", IETF RFC 3031, Jan. 2001.

◐ 저 자 소 개 ◑

전 환 식
1998년 국립 창원대학교 전자계산학과 졸업(학사)
2001년 국립 창원대학교 대학원 전자계산학과 졸업(석사)
2000년~현재 : 영우통신㈜ 연구소 연구원
관심분야 : MPLS, ATM,인터넷 보안
E-mail : wschun@ywtc.com



김 한 경
1973년 서울대학교 원자력공학과(공학사)
1987년 충북대학교 대학원 전산통계학과(이학석사)
1992년 충북대학교 대학원 전자계산학과(이학박사)
1973년 4월~1977년 9월 한국유니시스(주), 시스템엔지니어
1977년 9월~1983년 2월 삼성전자(주), 기술과장
1983년 3월~1997년 8월 한국전자통신연구원, 책임연구원
1997년 9월~현재 : 국립 창원대학교 공과대학 컴퓨터공학과 부교수
관심분야 : ATM, MPLS, SDR, 프로토콜 공학, 소프트웨어공학
E-mail : hkim@sarim.changwon.ac.kr