RSVP에서 묶음 자원 예약의 크기 결정을 위한 적응적 방법

정회원 박 태 근*, 김 치 하*

An Adaptive Method for Determining the Size of Aggregate Reservation in RSVP

Taekeun Park*, Cheeha Kim* *Regular Members*

요 약

RSVP는 강력한 QoS 보장 시그널링 프로토콜이기는 하지만 확장성에 문제점을 가지고 있기 때문에, IETF는 RSVP에 묶음 자원 예약 개념을 도입하였다. 그러나, 묶음 자원 예약에 관한 RSVP 규격은 경로에 대하여 예약되고 해제될 묶음 자원의 크기 계산 방법을 명시하지 않고, 묶음 자원의 크기가 주어진 경우에 자원을 예약하고 해제하는 절차에 대하여 서술하고 있다. 본 논문은 실시간 응용의 QoS 보장을 위하여 RSVP를 통해 묶음 단위로 자원을 예약하고 해제하고자 할 때, 경로의 상태에 적절하게 예약되고 해제될 묶음 자원의 크기 결정 방법을 제안한다. 시뮬레이션을 통하여 제안하는 방법을 적용하였을 때 묶음 단위 RSVP의 효율성이 향상됨을 보인다.

Key Words : Aggregate Reservation, QoS, RSVP, Scalability, Efficiency

ABSTRACT

Although RSVP is a strong signaling protocol in providing QoS support, it is not scalable. Thus, the IETF introduced the concept of aggregation into RSVP. However, the specification for the aggregation of RSVP does not specify how to compute the size of aggregate reservation for a path, but only describes how the resources are reserved and released with a given size. This paper proposes the method of properly determining the size of aggregate reservation on the path condition, when aggregate reservation is performed through RSVP for guaranteeing QoS of multimedia traffic. Simulation results show that the proposed method improves efficiency of the aggregation of RSVP.

Ⅰ. 서 론

인터넷 망에서 화상 전화, 주문형 비디오 및 VoIP와 같은 멀티미디어 서비스를 제공하고자 하는 요구가 늘어남에 따라, IETF는 실시간 응용에 필요한 자원을 예약하는 종합 서비스 모델 (IntServ: Integrated Services)[1]과 시그널링 프로토콜인 RSVP (Resource ReSerVation Protocol)[2, 3]를 정의하였다.

종합 서비스 모델에서, 송신자는 새로운 플로우를 시작하기에 앞서 RSVP Path 메시지를 이용하여 플로우의 QoS 요구 사항을 수신자에게 통지하고, 이에 대한 응답으로, 수신자는 RSVP Resv 메시지를 송신자에게 전송하여 새로운 플로우를 위한 자원 예약을 시도한다. 또한, 송신자와 수신자를 연결하는 경로상의 모든 라우터들은 Resv 메시지를 수신하였을 때, 수신자가 요구한 자원 예약이 가능한지를 판단하고 자원을 예약한다. 경로상의 모든 라우터들이 성공적으로 자원 예약을 마친 경우에 새로운 플로우의 요구 QoS는 보장된다. 이와 같

* 포항공과대학교 컴퓨터공학과 네트워크 및 분산시스템 연구실 ({tkpark99, chkim}@postech.ac.kr)
논문번호 : 030588-1231, 접수일자 : 2004년 1월 2일

은 QoS 보장 기법은 매우 강력한 것이기는 하지만, 경로상의 모든 라우터들이 개별 플로우에 대한 소프트 상태 정보 (soft-state)를유지해야 할 뿐만 아니라 RSVP 시그널링 메시지의 수가 망에 존재하는 플로우의 수에 비례하여 증가하기 때문에, 확장성 (Scalability) 측면에서 문제점을 내포하고 있다[4].

이러한 RSVP의 단점을 보완하기 위하여, IETF는 RSVP에 묶음 자원 예약 기법 (Aggregation of RSVP) [5]을 도입하였다. 이 기법은 개별 플로우에 대하여 자원을 예약하는 대신에 동일한 경로를 사용하는 플로우들의 집합에 대하여 묶음 자원을 예약하기 때문에, 경로상의 라우터들이 개별 플로우에 대한 소프트 상태 정보를 저장할 필요가 없어질 뿐만 아니라 새로운 플로우에 대한 수락 여부를 경계 노드 (Edge Node)에서 이미 예약된 묶음 자원 중 가용 자원의 크기를 검사하는 것만으로 간단히 판단할 수 있다는 장점을 가진다. "예약된 묶음 자원"이란 여러 번의 묶음 자원 예약 및 해제 절차 후에 누적 예약되어 있는 묶음 자원을 의미하고, "예약되고 해제될 묶음 자원"이란 한번의 묶음 자원 예약 또는 해제 절차에서 추가 예약되거나 해제될 자원을 의미한다.

그러나, 묶음 단위의 자원 예약을 수행하는 기존의 기법[5, 6, 7, 8, 9]들은 다음의 두 가지 문제점을 내포하고 있다. 첫째, 주어진 망 구조와 시간에 따라 변하는 트래픽 요구에 적합하게 예약되고 해제될 묶음 자원의 크기를 결정하는 것이 어려우며, 둘째, 결정된 묶음 자원의 크기가 그 시점에 적합한 값보다 큰 경우 사용률 손실 (under-utilization)이 발생 가능하고, 묶음 자원의 크기가 적합한 값보다 작은 경우 불필요한 자원 예약 및 해제 시그널링이 요구될 수 있다. 따라서, 기존 기법들은 시간에 따라 병목 링크가 바뀌는 경로에 대하여 예약되고 해제될 묶음 자원의 적절한 크기 계산 방법을 명시하지 않고 시그널링 절차만 제안하거나 [5, 7], 고정된 묶음 크기로 [6, 8] 또는 고정된 시간 간격마다 [9] 예약된 묶음 자원의 양을 증감시키면서 고정된 묶음 크기 또는 고정된 시간 값에 따라 변화하는 시그널링 비용과 사용률 사이의 역관계 (tradeoff)를 분석하는 수준의 연구 결과를 제시하고 있다.

본 논문은 특정 경로를 지나는 실시간 응용의 QoS 보장을 위하여 RSVP를통해 묶음 단위로 자원 (대역폭)을 예약하고 해제하고자 할 때, 그 시점에 적절하게 예약되고 해제될 묶음 자원의 크기인 "경로 묶음 크기"를 결정하는 방법을 제안한다. 제안하는 방법은 특정 경로에 속한 모든 링크에 대하여 자원 예약 및 해제 시점에 적절한 "링크 묶음 크기"를 계산한 후, 계산된 "링크 묶음 크기"들 중에서 최소값을 "경로 묶음 크기"로 사용한다. 이와 같은 방법으로, 앞에서 언급한 묶음 단위의 자원 예약을 수행하는 기법들의 두 가지 문제점을 회피할 수 있다.

본 논문은 성능 비교를 통하여, 제안하는 방법이 고정된 묶음 크기로 자원을 예약하고 해제하는 방법에 비하여 효율성 (Efficiency) 측면에서 월등함을 보인다. 본 논문에서 효율성이란 평균 처리율 (Throughput)을 시그널링 비용으로 나눈 것으로 정의되며, 동일한 처리율을 보장하기 위하여 요구되는 시그널링 비용이 낮을 때 효율성이 높다고 표현한다.

본 논문의 구성은 다음과 같다. II장에서는 IETF가 정의한 묶음 단위의 RSVP를 시그널링 절차 중심으로 간략히 소개하고, III장에서는 본 논문이 제안하는 "경로 묶음 크기" 결정 방법을 서술하며, IV장에서는 수정된 묶음 자원 예약 및 해제 시그널링 절차를 서술한다. 그리고, V장에서는 제안하는 방법과 고정된 묶음 크기로 자원을 관리하는 방법의 성능을 비교하며, VI장에서 결론을 맺는다.

II. 묶음 단위의 RSVP

묶음 단위의 RSVP[5]는, ATM (Asynchronous Transfer Mode) 망의 가상 경로와 유사하게, 특정 라우팅 영역 내에서 복수개의 RSVP 자원 예약을 하나의 RSVP 묶음 자원 예약으로 처리하는 기법이다. 묶음 단위의 RSVP에 의하여 서비스되는 라우팅 영역을 "묶음 영역 (Aggregation Region)"이라 하고, "묶음 영역"의 입구(Ingress) 라우터 및 출구 (Egress) 라우터를 각각 "Aggregator" 및 "Deaggregator"라 한다.

묶음 단위의 RSVP에서, 주어진 Aggregator / Deaggregator 쌍에 대하여 최초의 E2E (End-to-End) Path 메시지가 "묶음 영역"에 도착하면, Aggregator와 Deaggregator는 그림 1과 같은 시그널링 절차를 통하여 묶음 자원을 예약한다. "묶음 영역"에서는 개별 플로우에 대한 RSVP 메시지를 처리하지 않으므로, Aggregator는 E2E Path 메시지를 수신하면 메시지에 포함된 IP 프로토콜 번호를 *RSVP(46)*에서 *RSVP-E2E-IGNORE(134)*로 대체한 후, Deaggregator로 송신한다. Deaggregator는 수정된 E2E Path 메시지를 수신하면, E2E PathErr 메시지의 오류 코드 값으로 *NEW-AGG-NEEDED*를 지정하고 *DCLASS* 오브젝트에 예약된 묶음 자원을 위한 DSCP (Differentiated Services Code Point)[10] 값을 지정하여 Aggregator에게 송신한다. 다음 단계로, Aggregator와 Deaggregator

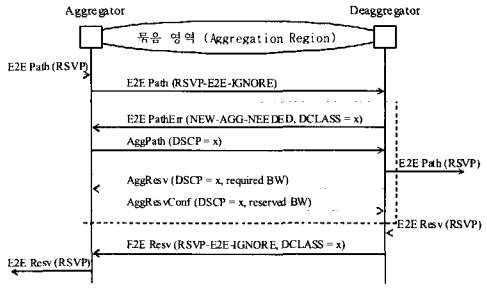


그림 1. IETF 묶음단위의 RSVP에서 초기 묶음 자원 예약 시그널링 절차

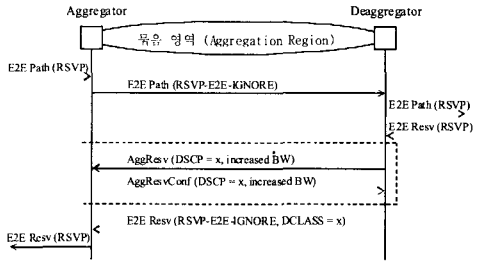


그림 2. IETF 묶음단위의 RSVP에서 묶음 자원 증가 시그널링 절차

는 AggPath, AggResv 및 AggResvConf 메시지를 이용하여 Aggregator / Deaggregator 쌍에 대한 묶음 자원을 예약한다.

Aggregator / Deaggregator 쌍에 대한 묶음 자원이 예약된 후에는, 새로 도착하는 플로우들에 의하여 예약된 자원이 모두 소진되거나 종료되는 플로우들에 의하여 예약되었지만 사용되지 않는 자원의 크기가 일정 수준을 넘어설 때까지 예약된 묶음 자원의 크기를 변화시키지 않는다. 그러나, 예약된 묶음 자원의 크기를 변화시킬 필요성이 있는 경우에, Deaggregator는 예약된 묶음 자원의 크기 변경을 위한 시그널링 절차를 시작한다. 그림 2는 예약된 묶음 자원의 크기를 증가시키려 하는 경우의 시그널링 절차를 보여준다. 묶음 자원 증가 시그널링 절차에서는, Aggregator / Deaggregator 쌍에 대한 RSVP 소프트 상태 정보가 이미 생성되어 있기 때문에, AggPath 메시지를 수신하지 않더라도 Deaggregator는 필요에 따라 AggResv 메시지를 송신함으로써 묶음 자원의 크기 증대를 요구할 수 있고, 묶음 자원 크기 증대를 확인한 Aggregator는 확인 메시지로 AggResvConf 메시지를 Deaggregator에게 전송한다.

Ⅲ. 경로 묶음 크기 결정 방법

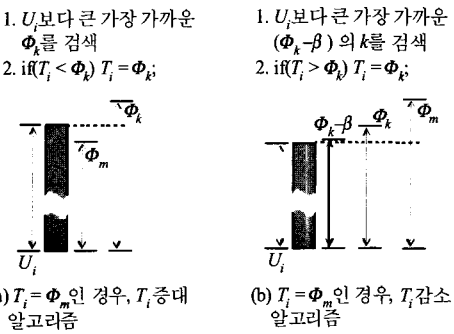
IETF에서 서술한 묶음 단위의 RSVP 문서에는 예약되고 해제될 묶음 자원 (대역폭)의 크기 결정 방법이 명시되어 있지 않다. 따라서, 본 장에서는 주어진 망 구조와 시간에 따라 변하는 트래픽 요구에 적합한, 예약하고 해제될 묶음 자원의 크기인 "경로 묶음 크기"를 계산하는 방법을 제안한다. 제안하는 방법에서는, 특정 경로에 속한 모든 링크에 대하여 자원 예약 및 해제 시점에 적절한 "링크 묶음 크기"를 계산한 후, 계산된 "링크 묶음 크기"들 중에서 최소값을 "경로 묶음 크기"로 결정한다.

1. 링크 묶음 크기 계산 함수

묶음 단위의 RSVP로 서비스하고자 하는 플로우들의 특성과 "묶음 영역" 내의 동작 환경이 다음과 같다고 가정한다.

- 링크 i의 대역폭 중에서 실시간 응용을 위하여 지정된 대역폭의 크기는 B_i이다.
- 링크 i를 사용하는 Aggregator / Deaggregator 쌍의 수는 N_i이다.
- 링크 i의 자원은 주어진 "링크 묶음 크기" L_i 크기로 예약 및 해제된다.
- Deaggregator는 묶음 자원의 해제 결정을 위하여 Hysteresis 값으로 α ($\alpha \geq 0$)를 사용한다[5, 6]. 즉, 사용되지 않으면서 소유되고 있는 가용 자원의 크기가 $(1+\alpha) \times L_i$보다 큰 경우에 Deaggregator는 묶음 자원을 해제한다. 자원 해제에 사용되는 Hysteresis는 짧은 기간 동안 반복적으로 발생 가능한 자원 예약 및 해제 요구의 수를 감소시켜준다.
- 실시간 응용 플로우의 요구 대역폭 (토큰 버킷 속도: Token Bucket Rate)은 음성 / 영상 코덱의 목표 전송률 (Target Rate)과 RTP, UDP, IPv4 헤더와 같은 패킷 헤더를 위한 전송률의 합으로 계산되고[11, 12], Deaggregator는 토큰 버킷 속도의 합과 토큰 버킷 크기(Token Bucket Depth)의 합으로 수락 제어를 수행한다[5, 13]. VBR (Variable Bit Rate) 코덱의 목표 전송률은 평균 전송률과 동일하다[14].

이와 같은 가정하에서, 링크 i의 자원이 예약 및 해제되다가 Deaggregator의 자원 예약 요구에 대하여 추가적으로 예약할 자원이 없는 시점에 발생 가능한 가장 낮은 링크 사용률 (Utilization) U_{ui}은 수식 (1)과 같이 표현된다.



(a) $T_i = \Phi_m$인 경우, T_i증대 알고리즘

(b) $T_i = \Phi_m$인 경우, T_i감소 알고리즘

그림 3. 목표 사용률 T_i값 변경 알고리즘

$1 \le N_i \le B_i/L_i$ 인 경우,

$$U_w = \frac{B_i - A_i}{B_i} = \frac{B_i - (N_i - 1) \times (1 + \alpha) \times L_i}{B_i} \quad (1)$$

수식 (1)에서 A_i는 자원 예약을 요구한 Aggregator / Deaggregator 쌍을 제외한 나머지 Aggregator / Deaggregator 쌍 ($N_i - 1$ 개)에 의해 사용되지 않으면서 소유되고 있는 가용 자원의 크기를 의미한다. "링크 묶음 크기" 계산 함수는, 수식 (1)의 U_w를 링크 i에 대한 목표 사용률 (Target Utilization) T_i로 대체함으로써, 수식 (2)와 같이 유도된다.

$$L_i = \begin{cases} \dfrac{(1 - T_i) \times B_i}{(N_i - 1) \times (1 + \alpha)} & , N_i > 1 \\ B_i & , N_i = 1 \end{cases} \quad (2)$$

수식 (2)로 계산되는 "링크 묶음 크기"를 사용하는 경우, 링크 사용률 U_i과 목표 사용률 T_i의 상관관계는 다음과 같다. 링크의 자원이 "링크 묶음 크기" 보다 같거나 작은 크기로 예약되고 해제되어 왔다면, Deaggregator가 요구한 자원 예약에 대하여 링크에 더 이상 예약할 자원이 없는 경우의 링크 사용률 U_i는 목표 사용률 T_i보다 항상 같거나 크다.

2. 목표 사용률 변경 방법

수식 (2)를 이용하여 "링크 묶음 크기"를 계산하기 위해서는 목표 사용률 T_i 값의 지정이 필요하다. 따라서, 목표 사용률 값 지정을 위하여, $a < b$에 대하여 $\Phi_a < \Phi_b$인, 임계치 집합 $S_T = \{\Phi_1, \ldots, \Phi_g\}$를 정의하고 임계치 hysteresis로 β를 사용한다.

제안하는 방법은 시그널링 비용을 줄이기 위하여 링크의 초기 목표 사용률 T_i를 Φ_1로 설정한다. 그리고, Deaggregator의 자원 예약 요구에 대하여 링크가 예약할 자원이 없을 때까지, 그 링크의 목표 사용률 T_i를 변화시키지 않는다. 왜냐하면, 그 시점의 링크 부하는 낮은 상태이거나, 링크 부하가 높더라도 자원의 분배가 적절한 상태이기 때문이다. 그러나, 자원 예약 요구에 대하여 예약할 자원이 링크에 없다면, 그 링크의 자원을 사용하고 있는 모든 Aggregator / Deaggregator 쌍으로부터 가용 자원 중 여유분을 수집하고 재분배하기 위하여, 목표 사용률 T_i를 증대시킴으로써 "링크 묶음 크기" 인 L_i를 감소시켜야 한다. 반면에, 링크 사용률이 점차 낮아진다면, 시그널링 비용을 줄이기 위하여, 목표 사용률 T_i를 감소시킴으로써 "링크 묶음 크기"인 L_i를 증대시켜야 한다. 목표 사용률 변경 알고리즘은 그림 3과 같다. 목표 사용률 증대 알고리즘은 중간 라우터가 자원 예약 요구에 대하여 예약할 자원이 없는 경우에 수행되고, 목표 사용률 감소 알고리즘은 중간 라우터가 Deaggregator로부터 수신한 자원 해제 요구를 처리한 이후에 수행된다.

AGG_SIZE Object Length	Class	Type
Aggregation Size in a Path (ASP)		
Available Bandwidth (ABW)		

그림 4. *AGG_SIZE* 오브젝트 포맷

3. *AGG_SIZE* 오브젝트

경로를 구성하는 링크들은 각각 다른 수의 Aggregator / Deaggregator 쌍에 대하여 자원을 제공하고 있으며, 실시간 응용을 위하여 지정된 대역폭의 크기도 다를 뿐만 아니라 시간에 따라 변하는 트래픽 요구에 의해 다른 링크 사용률을 가질 수있다. 따라서, 특정 시점에 적합한 "경로 묶음 크기"는 그 경로를 구성하는 모든 링크들의 "링크 묶음 크기" 중 최소값이 되어야 한다. "경로 묶음 크기"는 RSVP AggPath 메시지에 그림 4와 같은 *AGG_SIZE* 오브젝트를 추가함으로써, RSVP Adspec의 "Path Bandwidth" 값을 구하는 것과 동일한 방법으로 결정될 수 있다.

"경로 묶음 크기" 값을 구하는 방법은 다음과 같다. Aggregator는 RSVP AggPath 메시지를 전송하기에 앞서, *AGG_SIZE* 오브젝트의 *ASP* (Aggregation Size in a Path) 필드에 AggPath 메시지가 전송될 경로의 첫 번째 링크의 "링크 묶음 크기" 값을 삽입한 후, 다음 라우터에게 AggPath 메시지를 전송한다. AggPath 메시지를 수신한 중간 라우터들은 자신이 관리하는 링크의 "링크 묶음 크기" 값이 *AGG_SIZE* 오브젝트의 *ASP* 값보다 작은 경우에 *ASP* 필드에 자신의 "링크 묶음 크기"

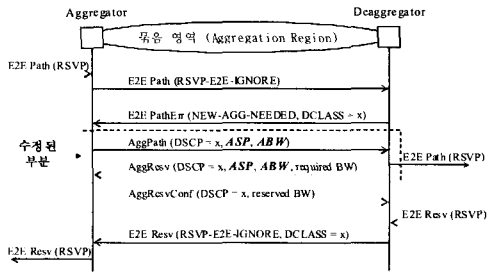


그림 5. 수정된 초기 묶음 자원 예약 시그널링 절차

값을 삽입한다. 이와 같은 방법으로 경로상의 모든 라우터들이 *ASP* 값을 수정하면, AggPath 메시지를 수신한 Deaggregator는 그 시점에 적합한 "경로 묶음 크기" 값을 *ASP*로부터 얻게 된다.

AGG_SIZE 오브젝트의 *ABW*(Available Bandwidth)는 경로상의 모든 링크들의 가용 자원 크기 중 가장작은 값을 포함하며 *ASP*와 동일한 방법으로 얻어진다. 링크의 가용 자원이란 각 링크에서 실시간 응용을 위하여 지정된 대역폭 중 예약되지 않은 대역폭을 의미한다.

AGG_SIZE 오브젝트는 RSVP AggResv 메시지에도 사용되는데, 첫 번째 목적은 Aggregator에게 최신의 *ASP*를 통지하는 것이고, 두 번째 목적은 묶음 자원 증가를 원하는 Deaggregator가 수신한 AggPath 메시지의 *ABW* 값이 0인 경우에 AggResv의 *ABW* 값을 -1로 지정하여 송신함으로써 가용 자원이 없는 중간 라우터로 하여금 목표 사용률 증대 알고리즘을 수행하도록 하는 것이다.

Ⅳ. 수정된 묶음 자원 예약 및 해제 절차

IETF에서 정의한 묶음 단위의 RSVP에서는, 그림 1 및 그림 2와 같이, 초기 묶음 자원 예약 시그널링 절차와 묶음 자원 증감을 위한 시그널링 절차에 약간의 차이가 있다. 초기 묶음 자원 예약 시그널링 절차는 AggPath 메시지의 사용을 필요로 하지만, 묶음 자원 증감 절차에서는 AggPath 메시지를 필요로 하지 않는다.

그러나, 제안하는 방법에 따라 계산된, 현재의 상황에 적합한 "경로 묶음 크기"를 Deaggregator에 통지하기 위해서는 AggPath 메시지의 사용이 필수적이다. 물론, RSVP의 Refresh 기능을 이용하여 주기적으로 "경로 묶음 크기"를 Deaggregator에게 통지할 수도 있지만, Refresh 주기 값의 크기에 관계없이 동일한 수준의 성능을 보장하기 위하여, 본 논문에서는 IETF의 초기

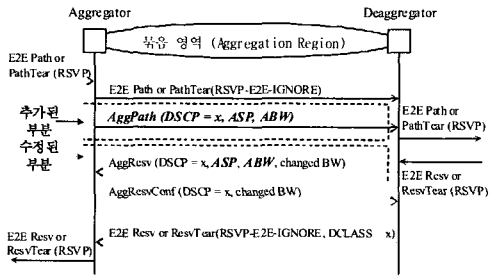


그림 6. 수정된 묶음 자원 증감 시그널링 절차

묶음 자원 예약 절차와 묶음 자원 증감 절차를 수정 제안한다. 또한, Deaggregator의 자원 예약 요구에 대하여 중간 라우터가 예약할 자원이 없어서 "링크 묶음 크기"인 L_i를 감소시킨 경우에 Aggregator / Deaggregator 쌍들로부터 여유 자원을 회수할 수 있도록 여유 자원 회수 시그널링 절차를 제안한다.

수정된 초기 묶음 자원 예약 시그널링 절차는 그림 5와 같다. AggPath 및 AggResv 메시지에 *ASP*와 *ABW* 필드가 추가되었다. 수정된 절차에서, Deaggregator는 AggPath 메시지의 *ASP* 값을 통하여 현재 자원 예약에 적합한 "경로 묶음 크기"를 인지하고, *ABW* 값을 통하여 예약 가능한 자원의 크기를 인지하며, 그 결과로 *min*(*ASP*, *ABW*) 크기의 묶음 자원 예약을 AggResv 메시지를 통하여 요구하게 된다. 즉, *"required BW"*의 값으로 *min*(*ASP*, *ABW*)가 사용된다. 대부분의 경우, AggResv 메시지의 *ASP*와 *ABW*에는 AggPath 메시지의 값이 복사되지만, AggPath 메시지의 *ABW*의 값이 0인 경우에는 예약 가능한 자원이 없는 중간 라우터가 목표 사용률 증대 알고리즘을 구동하도록 Deaggregator는 AggResv 메시지의 *ABW* 값을 -1로 지정하여 송신한다.

수정된 묶음 자원 증감 시그널링 절차는 그림 6과 같다. 수정된 절차에서 Aggregator는 E2E Path 또는 PathTear 메시지를 수신하면 Deaggregator가 묶음 자원을 증감시킬 것인지 여부를 판단한다. 이러한 판단이 가능한 것은 Aggregator와 Deaggregator가 동일한 정보를 가지고 있기 때문이다. Aggregator는 Aggregator / Deaggregator 쌍에 대하여 예약된 자원의 크기와 현재 서비스 중인 플로우들이 요구하는 자원의 합을 알고 있기 때문에, Deaggregator의 묶음 자원 증가 요구 시점을 정확히 예측할 수 있다. 또한, Aggregator는 Deaggregator가 묶음 자원의 감소 요구를 판단하는 기준이 되는 "경로 묶음 크기" 값을 AggResv 메시지의 *ASP* 필드를 통하여 인지하고 있기 때문에,

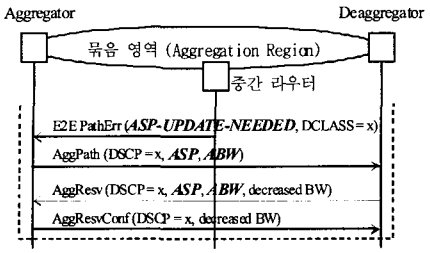


그림 7. 여유 자원 회수 시그널링 절차

Deaggregator의 묶음 자원 감소 요구 시점 (사용되지 않으면서 소유되고 있는 가용 자원의 크기가 $(1+\alpha)\times ASP$ 보다 큰 시점)도 예측 가능하다.

Deaggregator가 E2E Path 또는 PathTear 메시지를 수신한 후 묶음 자원 증감 시그널링 절차를 시작할 것이라고 판단한 Aggregator는 그림 6과 같이 AggPath 메시지를 Deaggregator에게 전송한다. AggPath 메시지를 수신한 Deaggregator는 다음과 같은 방법으로 *"changed BW"*를 계산하고 AggPath 메시지의 *ASP*와 *ABW* 값을 AggResv 메시지에 복사한 후, Aggregator에게 AggResv 메시지를 전송한다.

- 묶음 자원 증가 경우:
  $changedBW = currentBW + min(ASP, ABW)$;
- 묶음 자원 감소 경우 (플로우들에 의해 사용되는 대역폭의 합을 R이라 할 때):
  $changedBW = (\ (R/ASP + 1)\times ASP$;

단, *ASP* 값이 동적으로 변화 가능하기 때문에 Aggregator가 자원 해제를 예상하였다 하더라도 AggPath 메시지에서 추출한 새로운 *ASP*로는 자원 해제 조건이 만족되지 않을수 있는데, 이 경우에 Deaggregator는 AggResv 메시지를 송신하지 않는다. 또한, 묶음 자원 증가를 원하는 Deaggregator가 *ABW*의 값이 0인 AggPath 메시지를 수신하면, 예약 가능한 자원이 없는 중간 라우터가 목표 사용률 증대 알고리즘을 구동할 수있도록, AggResv 메시지의 *ABW* 값을 -1로 지정하여 Aggregator에게 송신한다.

*ABW*의 값이 -1인 AggResv 메시지를 수신하여 "링크 묶음 크기"를 감소시킨 중간 라우터의 여유 자원 회수 시그널링 절차는 그림 7과 같다. "링크 묶음 크기"를 감소시킨 중간 라우터는 E2E PathErr 메시지의 오류 코드 값으로, 본 논문에서 정의한 값인, *ASP-UPDATE-NEEDED*를 지정한 후Aggregator에게 송신한다. 이를 수신한 Aggregator가 AggPath 메시지

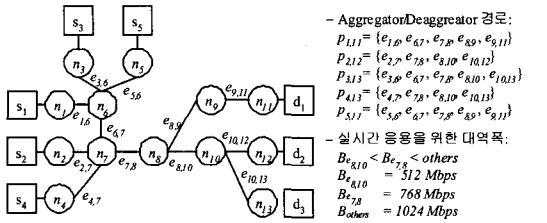


그림 8. "묶음 영역" 구성도

를 Deaggregator에게 송신하면, Deaggregator는 AggPath 메시지의 ASP 값으로 "경로 묶음 크기"를 갱신한 다음 묶음 자원 해제 여부를 판단한다. 묶음 자원 해제 조건이 만족된 경우에 송신되는 AggResv 메시지의 *"decreased BW"*의 값은 수정된 묶음 자원 증감 절차에서 묶음 자원 감소를 위하여 *"changed BW"*를 계산하는 방법과 동일하게 구해진다.

V. 성능 분석

성능 분석을 위하여 그림 8과 같이, 5개의 Aggregator(n_1, n_2, n_3, n_4, n_5)와 3개의 Deaggregator (n_{11}, n_{12}, n_{13}) 및 5개의 중간 라우터(n_6, n_7, n_8, n_9, n_{10})로 구성된 "묶음 영역"에서 시뮬레이션을 수행한다. 각 Aggregator와 Deaggregator는 시작점 노드 s_i와 목적지 노드 d_i로 표현되는 액세스 망을 가지고 있다. 이 "묶음 영역"에는 두 개의 잠재적인 병목 링크들인, $e_{7,8}$과 $e_{8,10}$이 존재한다. 만일, s_1와 s_5가 트래픽 부하를 증대시킨다면, 실시간 응용을 위해 지정된 $e_{7,8}$의 대역폭이 $e_{8,10}$의 것보다 큼에도 불구하고, $e_{8,10}$ 대신에 $e_{7,8}$이 모든 경로에 대한 병목 링크가 된다.

실제 망에서 수집된 묶음 트래픽의 요구 형태[9]는 주기적인 특성을 가지며 주기 내에서 최소 값과 최대 값 사이에서 증감을 반복하는 형태이다. 따라서, 본 논문은 Aggregator / Deaggregator 경로에 대한 부하 생성을 위해 *cos()* 함수를 사용하며, i 번째 경로의 트래픽 요구량인 r_i를 수식 (3)과 같이 모델링한다.

$$r_i(t) = s\cdot\ (m_i + a_i\cos((\frac{2\pi}{T_i})t + \theta_i)) \quad (3)$$

수식 (3)에서, s와 m_i는 각각 규모 인자 (Scaling Factor)와 플로우의 평균 수를 의미하고, a_i는 주기 T_i와 위상 θ_i을 가지는 *cos()* 함수의 진폭을 나타낸다. 각 플로우의 요구 대역폭은 128Kbps이고, 플로우 지속 시간 (Duration)은 $1/\mu$ (120초)를 평균으로 하는 지수 분

표 1. 부하 생성을 위한 변수 값

	m_i	a_i	T_i	θ_i
1st path: $p_{1,11}$	800	800	2 × 3600	1.000 × π
2nd path: $p_{2,12}$	950	950	3 × 3600	1.250 × π
3rd path: $p_{3,13}$	950	950	4 × 3600	1.500 × π
4th path: $p_{4,13}$	900	900	6 × 3600	0.333 × π
5th path: $p_{5,11}$	900	900	12 × 3600	0.166 × π

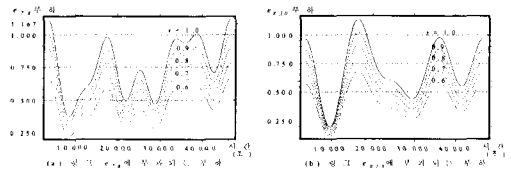


그림 9. 규모 인자 s의 변화에 따른 잠재적 병목 링크에 부과되는 부하

포이다. 또한, 플로우 생성 간격 (Inter-Arrival Time)은 t 시간에 경로 i에 대하여 $1/\lambda_i(t)$를 평균으로 하는 지수 분포를 따르는데, $1/\lambda_i(t)$는 $1/\lambda_i(t) = 1/(\mu \times r_i(t))$와 같이 계산된다.

표 1은 부하 생성을 위한 변수들을 위한 값을 보여주며, 그림 9는 표 1과 수식 (3)에 의해 생성되어 잠재적인 병목 링크 $e_{7,8}$과 $e_{8,10}$에 부과되는 부하를 보여준다. 그림 9에서 규모 인자의 값으로 0.6, 0.7, 0.8, 0.9, 1.0이 사용되었는데, 규모 인자의 사용 목적은 동일한 패턴의 부하에 대하여 평균 부하가 낮아짐에 따라 제안하는 방법의 성능 변화를 분석하기 위함이다.

시뮬레이션에서 임계치 집합 $S_T = \{\Phi_1, \Phi_2, \Phi_3\}$에 대하여 Φ_1와 Φ_2는 각각 0.50와 0.75로 고정하고, Φ_3의 값은 0.875, 0.900, 0.925, 0.950, 0.975로 변화하여 성능을 분석한다. 임계치 hysteresis β의 값으로는 0.02를 사용하며, Deaggregator의 묶음 자원해제를 위한 hysteresis α의 값으로는 0.1을 사용한다. 성능 비교 대상이 되는 고정된 묶음 크기를 사용하는 방법에 대해서는 128Kbps의 배수인 3.2, 6.4, 12.8, 16.0, 19.2Mbps의 묶음 크기를 사용한다. 61시간의 시뮬레이션 시간 중 처음 1시간은 안정화 시간 (Warm-up Time)이며, 그림 11과 그림 12의 결과는 플로우 생성 간격 (Inter-Arrival Time)과 플로우 지속 시간 (Duration)을 위한 랜덤 시드를 변화시키면서 수행한 다섯 번 시뮬레이션 결과의 평균치이다.

그림 10은 잠재적 병목 링크들의 사용률에 따라서

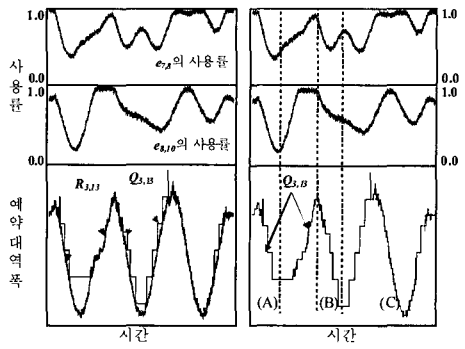


그림 10. 규모 인자 s가 1.0일 때, Aggregaor(n_3) / Deaggregator (n_{11}) 쌍에 예약된 묶음 자원 크기 $Q_{3,13}$변화

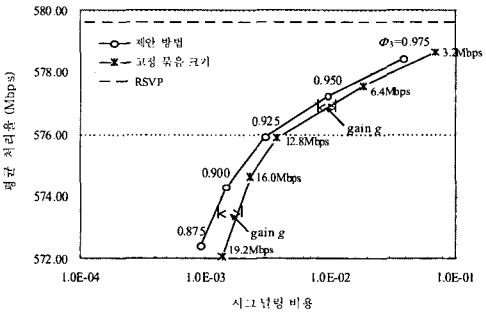
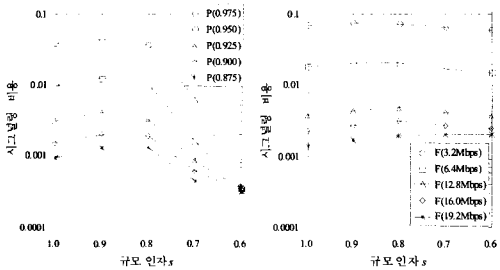


그림 11. 규모 인자 s가 1.0일 때, 시그널링 비용 대 평균 처리율

Aggregaor(n_3) / Deaggregator (n_{13}) 쌍에 예약된 묶음 자원 크기 $Q_{3,13}$과 Deaggregator에 의해 수락된 플로우들이 사용하는 대역폭의합 $R_{3,13}$의 변화를 보여준다. 그림 10의 (C)와 같이 잠재적 병목 링크의 사용률이 높은 경우에는 예약된 묶음 자원 크기 $Q_{3,13}$이 작은 단위로 자주 변화하는 것을 확인할 수 있는데, 이는 작은 가용 자원을 여러 Aggregator / Deaggregator가 나누어 쓸 수 있도록 "경로 묶음 크기"가 감소되었기 때문이다. 반면에, 그림 10의 (A) 또는(B)와 같이 잠재적 병목 링크의 사용률이 낮아지는 경우에는 예약된 묶음 자원 크기 $Q_{3,13}$이 자주 변화하지 않을 뿐만 아니라 한번 변화할 때 변화의 폭이 큰 것을 확인할 수 있다. 이는 시그널링 비용을 절감하기 위하여 "경로 묶음 크기"가 증대되었기 때문이다. 이와 같은 결과를 모든 Aggregator / Deaggregator 쌍의 예약된 묶음 자원 크기 변화에서 동일하게 확인할 수 있다.

그림 11은 규모 인자 s가 1.0일 때(높은 부하일 때), 제안하는 방법과 고정된 묶음 크기를 사용하는 방법을



(a) 제안하는 방법 (b) 고정된 묶음 크기 사용 방법

그림 12. 규모 인자 s의 변화에 따른 시그널링 비용 변화

적용했을 때, 시그널링 비용과 평균 처리율을 보여주며, 플로우 단위로 RSVP 자원 예약을 수행한 경우의 평균 처리율도 보여준다. 평균 처리율은 "묶음 영역"에서 서비스되는 모든 실시간 응용 플로우들의 처리율 합을 시뮬레이션 시간 동안 평균한 값으로 정의되고, 시그널링 비용은 수락된 호를 위하여 묶음 자원의 예약과 해제에 사용된 AggPath, AggResv, AggResvConf, E2E PathErr(*NEW-AGG-NEEDED* or *ASP-UPDATE-NEEDED*) 메시지 수를 수락된 플로우의 수로 나눈 값으로 정의된다[6].

그림 11로부터 제안하는 방법이 고정된 묶음 크기를 사용하는 방법보다 더 낮은 시그널링 비용으로 동일한 평균 처리율을 보장함을 확인할 수 있다. 제안하는 방법의 성능 향상 정도를 분석하기 위하여, 이득 $g = 1 - (cost_p(T)/cost_f(T))$ 값을 계산해 본다. 함수 $cost_p(T)$와 $cost_f(T)$는 각각 평균 처리율 T를 보장하기 위하여 필요한 제안하는 방법의 시그널링 비용과 고정된 묶음 크기를 사용하는 방법의 시그널링 비용 값을 나타낸다. 그림 11의 결과에 따르면, 제안하는 방법이 얻는 이득은 0.17 에서 0.31 수준이다. 즉, 제안하는 방법을 사용함으로써 고정된 묶음 크기를 사용하는 방법보다 동일한 처리율 보장에 요구되는 시그널링 비용을 17% 에서 31% 정도 절약할 수 있다.

그림 12는 규모 인자가 작아짐에 따라 (부하가 낮아짐에 따라) 제안하는 방법과 고정된 묶음 크기를 사용하는 방법의 시그널링 비용 변화를 보여준다. 제안하는 방법은 "묶음 영역"에 부과되는 부하가 낮아지면 "경로 묶음 크기"를 증대시켜 시그널링 비용을 절감하려 하기 때문에, 규모 인자 값이 작아짐에 따라 시그널링 비용을 급격하게 감소시킨다. 그러나, 고정된 묶음 크기를 사용하는 방법에서는 "묶음 영역"에 부과되는 부하에 관계 없이 동일한 묶음 크기를 사용하기 때문에 시그널링 비용이 거의 변화하지 않는다.

그림 11과 그림 12의 성능 분석 결과로부터, 제안하는 방법을 묶음 단위의 RSVP에 적용하면, "묶음 영역"에 부과되는 부하에 관계없이, 고정된 묶음 크기를 사용하는 RSVP보다 효율적으로 실시간 응용을 위한 자원 예약이 가능하다는 것을 확인할 수 있다.

V. 결 론

본 논문은 실시간 응용의 QoS 보장을 위하여 RSVP를 통해 묶음 단위로 자원 (대역폭)을 예약하고 해제하고자 할 때, 그 시점에 적절하게 예약되고 해제될 묶음 자원의 크기인 "경로 묶음 크기"를 결정하는 방법을 제안하였다. 또한, 제안하는 방법을 묶음 단위의 RSVP에 적용하기 위하여 IETF가 서술한 묶음 자원 예약 및 해제 절차를 수정하였다. 묶음 단위의 자원 예약을 수행하는 기존의 기법들은 주어진 망 구성과 시간에 따라 변하는 트래픽 요구에 적합한 묶음 자원 크기 결정이 어렵다는 문제점과 결정된 묶음 자원 크기가 주어진 상황에 적절한 값이 아닌 경우에 낮은 사용률 또는 불필요한 시그널링 비용을 야기하는 문제점을 가지고 있는데, 제안하는 "경로 묶음 크기" 결정 방법과 수정된 묶음 자원 예약 및 해제 절차를 사용하면 이상의 문제점을 회피할 수 있다. "묶음영역"에 부과되는 부하가 높은 경우에는 "경로 묶음 크기" 값의 감소를 통하여 높은 처리율의 보장이 가능하고, "묶음 영역"에 부과되는 부하가 낮아짐에 따라 "경로 묶음 크기" 값의 증대를 통하여 시그널링 비용의 절감이 가능하다. 즉, 제안하는 방법을 대규모 망 ("묶음 영역")에 적용하면, 보다 효율적으로 실시간 응용을 위한 자원 예약이 가능하다.

참 고 문 헌

[1] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture," *RFC 1633*, 1994.

[2] Paul P. White, "RSVP and Integrated Services in the Internet: A Tutorial," *IEEE Communications Magazine*, vol.35, no.5, pp. 100-106, May 1997.

[3] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP)," *RFC 2205*, September 1997.

[4] S. Sargento, R. Valadas and E. Knightly, "Call Admission Control in IP networks with QoS support," *in the 3rd Conference on Telecommunications*, 2001.

[5] F. Baker, C. Iturralde, F. Le Faucheur, and B. Davie, "Aggregation of RSVP for IPv4 and IPv6 Reservations", *RFC 3175*, September 2001.

[6] Z. L. Zhang, Z. Duan and Y. T. Hou, "On Scalable Design of Bandwidth Brokers," *IEICE Transactions on Communications*, vol.E84-B, no. 8, pp. 2026-2032, August 2001.

[7] S. Salsano, E. Sangregorio, and M. Listanti, "COPS DRA: a protocol for dynamic DiffServ Resource Allocation," *Joint Planet-IP NEBULA workshop*, Courmayeur, Gennaio, 2002.

[8] M. Menth, "A Scalable Protocol Architecture for End-to-End Signaling and Resource Reservation in IP Networks," *Technical Report No. 278*, University of Würzburg, July 2001.

[9] H. Fu and E. W. Knightly, "Aggregation and Scalable QoS: A Performance Study," *in Proceedings of IWQoS*, 2001.

[10] S. Blake,D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *RFC 2475*, December 1998.

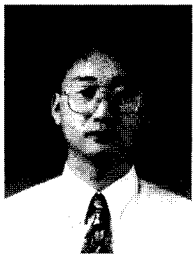
[11] 3GPP, "Quality of Service (QoS) concept and archiectre," *3GPP TS 23.107 v5.7.0*, December 2002.

[12] 3GPP, "Packet-switched conversational multimedia applications - transport protocols," *3GPP TS 26.236 v5.1.0*, December 2002.

[13] Rui Prior, Susana Sargento, Sérgio Crisóstomo, and Pedro Brandão, "End-to-End QoS with Scalable Reservations," *Technical Report Series DCC-2003-01*, University of Porto, April 2003.

[14] F. H. P. Fitzek and M. Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," *IEEE Network*, vol.15, no.6, pp. 40-54, November 2001.

박 태 근(Taekeun Park) 정회원



1991년 2월 : 포항공과대학교 컴퓨터공학과 졸업
1993년 2월 : 포항공과대학교 컴퓨터공학과 석사
2004년 2월 : 포항공과대학교 컴퓨터공학과 박사
2004년 2월 ~ 현재 : 포항공과대학교 컴퓨터공학과 박사후 과정 연구원
<관심분야> 이동 통신 프로토콜, 유/무선 망 QoS, IP 기반 통합망, 멀티미디어 통신망

김 치 하(Cheeha Kim) 정회원

1974년 2월 : 서울대학교 전자 공학과 졸업
1984년 8월 : University of Maryland Computer Science 석사
1986년 8월 : University of Maryland Computer Science 박사
1986년 9월 ~ 1989년 12월: State University of New York 교수
1989년 12월 ~ 현재 : 포항공과대학교 컴퓨터공학과 교수
<관심분야> 모바일 컴퓨팅 및 네트워킹, 컴퓨터 통신, 분산 시스템, 성능 평가