논문 13-38B-12-04

한국통신학회논문지(J-KICS) '13-12 Vol.38B No.12
http://dx.doi.org/10.7840/kics.2013.38B.12.962

Q-CBQ기반 MPLS망에서 Q-MOTP의 멀티미디어 객체 데이터 전송 QoS 지원을 위한 자원 관리자

최 원 근◆

Resource Manager of QoS Supporting of Q-MOTP for Multimedia Object Data Transfer in MPLS Network Using Q-CBQ

Won-Keun Choi◆

요 약

차세대 인터넷은 음성이나 비디오 트래픽과 같은 실시간 트래픽을 원활히 처리할 수 있는 QoS 보장이 요구된다. 본 연구는 이미 검증된 Q-CBQ기법을 사용하는 MPLS 라우터에서 멀티미디어 객체 전송 프로토콜인 Q-MOTP 프로토콜에 QoS지원을 위한 자원관리자에 대한 연구로서 자원관리자는 주기적으로 시스템 자원 정보를 보고하거나 자원의 과부하가 발생했거나 또는 QoS 관리자의 요구에 의해서 시스템 자원 정보를 보고한다. 보고된 자원 정보를 이용해서 QoS 관리자는 합의된 QoS 요구 수준을 만족하도록 관리한다.

Key Words : Multimedia, QoS, MPLS router, CBQ, Q-CBQ, Q-MOTP

ABSTRACT

The internet will should require the QoS(quality of service) guarantees that the real-time traffic like as audio and video data will be operated well in the future. We had designed Q-MOTP which transfer multimedia object data and proved excellent performance. This paper describes an adaptive QoS management architecture and mechanism based on the information of system resources. Resource manager reports the system resource information periodically or when resources are in the overload state, or on demand by the QoS manager. By using this information, the QoS manager can predict QoS degradation and perform CAC.

Ⅰ. 서 론

QoS 보장을 위해 인터넷은 플로우들을 하나의 클래스로 묶어서 서비스해 주는 DiffServ를 지원하는 MPLS[11] 라우터를 백본 네트워크에 배치하여 기존의 IP 라우터가 제공하지 못했던 고속 전송과 QoS(quality of service)를 보장하고자 하는 노력이 진행되고 있다[10].

QoS보장을 위한 또 다른 연구로서는 멀티미디어 스트림 자체 특성에 대한 연구이다. 멀티미디어 통신 관점에서 볼 때, 미디어 데이터들의 결합과 동기화는 통신의 특정한 성능들을 요구한다. 일반적으로 통신성능 요구들은 QoS 매개변수들로 기술되며, 주로 고려되는 QoS 매개변수들은 처리율, 전송지연, 지터(jitter). 신뢰성, 및 스큐(skew) 등이다[1-4].

연구[1,2]에서는 Q-CBQ기법을 사용하는 MPLS망에서 멀티미디어 객체 데이터를 효율적으로 전송하기 위한 프로토콜에 대한 연구와 기존 프로토콜들과의 성능을 비교하는 연구를 수행하였다.

본 연구에서는 연구[2] 제안한 Q-MOTP에 대한 QoS

◆ First Author : 인하공업전문대학 정보통신과, wkchoi@inhatc.ac.kr, 종신회원
논문번호 : KICS2013-09-413, 접수일자 : 2013년 9월 16일, 심사일자 : 2013년 12월 6일, 최종논문접수일자 : 2013년 12월 11일

를 지원하기 위한 연구로서 자원관리자를 통해서 QoS 관리자에게 시스템 자원정보를 갖게 함으로써, QoS 관리자가 QoS 및 자원의 2개의 정보를 갖는다. 2개의 정보를 이용해서 QoS에 대해서 종합적이고 빠른 판단이 가능하도록 하며, 예방적 대응이 가능하고, 실시간으로 대응할 수 있도록 함으로써 QoS의 관리를 수행하도록 하였다. 2장에서는 기존 수행된 연구인 Q-MOTP에 대해서 요약 설명하고, 설계된 QoS 관리 원리에 대해서 설명하며 자원 관리자 및 QoS 관리자에 대해서 기술하고 3장에서는 자원관리자에 대한 기능과 역할을 상세한 내용을 설명하고 마지막으로 4장에서는 본 연구에 대한 결론 및 향후과제를 논할 것이다.

Ⅱ. QoS 관리 구조

그림1은 본 연구에서 설계한 QoS 관리 시스템의 구조를 보여준다. 트랜스포트 계층의 프로토콜과 QoS 관리자와 자원관리자의 모델을 보여 준다.

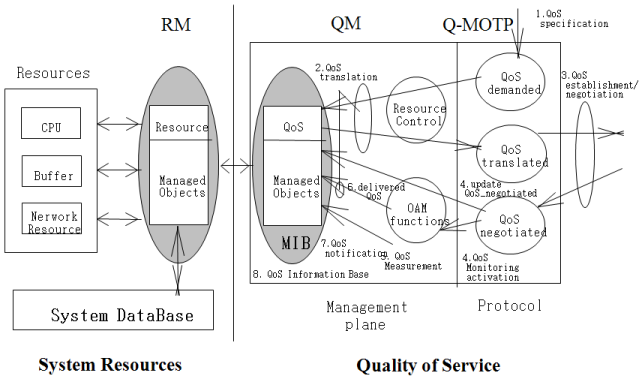

그림 1. QoS 관리구조
Fig. 1. QoS management structure

제안된 QoS 관리 구조는 자원관리자(RM: resource manager) QoS 관리자(QM:QoS manager), 그리고 Q-MOTP로 구성된다. QM은 QoS의 규격, 번역, 협상, 관리 등에 관여한다. 특히 제안된 기법에서는 시스템 자원(들)의 과부하 상태가 보고되면, Q-MOTP에게 QoS와 관련된 통계적 데이터를 수집하도록 요구한다. 사용자와 합의된 QoS값과 프로토콜에 의해서 측정된 QoS값을 비교한다. QoS 감쇠가 인지되면 RM에게 자원 동조를 요구하는 방법으로 QoS의 수준을 관리한다.

2.1. Q-CBQ 개요

제안된 Q-CBQ 방식의 계층적 링크 공유 구조를 보면 다음과 같다.

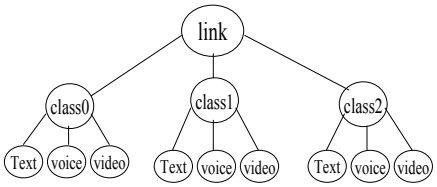

그림 2. Q-CBQ의 구조
Fig. 2. The structure of Q-CBQ

그림2의 링크 공유 구조에서 보는 바와 같이 기존 연구들과 달리 Q-CBQ에서는 미디어 트래픽 특성에 따라서 3가지의 형태의 데이터로 구분하여 처리하고 관리한다[4].

2.2. 제안한 QoS 관리의 원리

QoS 관리관점에서 사건들의 발생은 합의된 QoS 수준의 위반으로 정의할 수 있다. 대부분의 사건은 연결에 할당된 시스템 자원들의 고갈이나 병목으로 인해 발생한다. 자원의 고갈이 QoS 감쇠보다는 시간적으로 먼저 발생하게 된다.

기존의 QoS 관리 기법들의 문제점들을 살펴보면 다음과 같다.

첫째 기존의 QoS 감쇠에 대한 대응은 엄격한 의미에서 예방적 대응이라 보기 어렵다. 사용자가 QoS 감쇠를 인지하고 QoS 감쇠에 대한 조정을 요구한다. QoS 관리자가 프로토콜에게 QoS에 관련된 통계적 데이터를 수집하도록 요구하면, 프로토콜이 통계적 데이터를 수집한다.

둘째 기존의 QoS 관리기법에서는 QoS와 자원과의 관계를 고려하지 않았다. QoS 감쇠가 일어나기 전에 이미 자원의 고갈이나 병목이 먼저 일어나게 된다. 예를 들어서 처리율이 합의된 QoS 수준이하로 떨어진 경우, 원인은 CPU나 대역폭의 과부하를 생각할 수 있으나 또 다른 QoS 감쇠현상들을 분석해서 정확한 원인을 찾아야 한다. 그러나 본 논문에서 제안한 QoS 관리기법에서는 CPU 혹은 대역폭의 과부하가 이미 보고되었으며 이를 통해서 이미 낮은 처리율은 예측되어 있었다

Ⅲ. 자원 관리자

3.1. 사건과 동작 모델

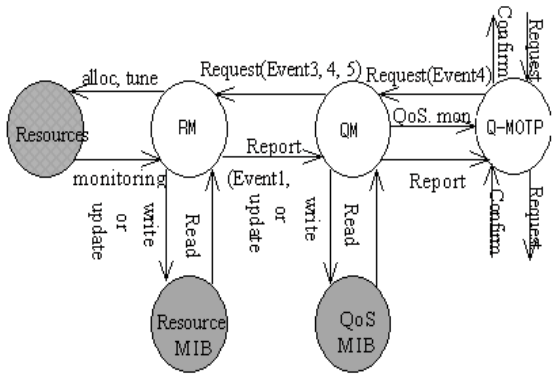

그림 3. QoS 관리 사건/동작 다이어그램
Fig. 3. QoS management event/action daigram

그림3은 제안된 QoS 관리구조에서의 사건/동작을 보여 준다.

RM은 자원의 모든 정보를 갖고 있으며, 사용자 요구 서비스를 만족시키기 위해서 필요한 경우 자원을 할당(allocate)하고, 동조(tune)하고, 양도(release)한다. 또한 RM은 자원의 이용도를 감시(monitoring)하고 자원부하상태(RLS: resource load state)를 변경시키는 사건을 발견 시, 자원부하상태에 대한 정보를 QM에게 전달한다. 제안된 QoS 관리기법에서는 기존의 사건 정의를 확장해서 5가지 사건으로 정의하였다. 사건1,2는 RM이 QM에게 자원 정보를 보고하는 사건이고 사건3은 QM이 RM에게 자원 정보를 요구하는 사건이다.

사건 1 : 주기적인 자원상태보고(RM-> QM)
사건 2 : 한 개 이상의 자원이 과부하 상태임을 보고하는 비주기적인 보고 (RM -> QM)
사건 3 : 자원상태정보 요구(QM ->RM)
사건 4 : 연결수락제어(CAC)를 위한 자원할당요청 (QM -> RM)
사건 5 : QoS 조정위한 자원동조요청(QM -> RM)

사건/동작 모델을 정의하는데 필요한 시간들은 다음과 같다.
Tbeg 사용자가 처음으로 통신을 요청할 때의 시간
Trep 주기적인 자원상태 보고 시간
Titv RM의 자원 상태 정보의 주기 보고 시간 간격
Tcr 자원중 하나이상이 과부하 상태를 보고한 시간
Treq QM이 RM에게 자원 상태 정보를 요구한 시간
Tres Treq에 대한 응답 시간

연결이 설정될 때, QM은 Titv값으로 RM을 호출하며, RM은 Tbeg및 Titv를 저장한다. RM의 다음 보고 시간은 Trep=Trep+Titv가 된다. 다음의 Trep이전에 사건3, 사건4 또는 사건5가 발생하면, RM은 사건 요청에 대해서 응답(Tres)하며, 다음 보고 시간은 Trep=Tres+Titv로 바뀐다. 자원 중 하나 이상이 과부하 상태일 때, RM은 자원 상태(Tcr)을 보고하며, 다음 보고 시간은 Trep=Tcr+Titv로 바뀐다.

3.2. 자원정보 보고

멀티미디어 응용들을 지원하기 위한 시스템 자원들은 QoS 특성과 관련되어 3가지로 분류된다. 자원 자체의 특성에 의한 분류는 아니며 QoS 분류에 따라서 자원들을 할당한 것에 의한 자원들의 분류이다. 본 논문에서는 멀티미디어 통신에 있어서 비교적 더 중요하다고 생각되는 CPU, 버퍼, 대역폭만을 고려하며 구조체는 그림4와 같다.

```
typedef  struct {
      float  det_a;
      float  det_b;
      float  best_c;
      } resource;
```

```
typedef  struct {
      int      rls_a;
      int      rls_b;
       int      rls_c;
       } rls;
```

그림 4. 자원 정보와 자원 정보 상태 구조체
Fig. 4. Structure of RI and RI state

3.3.1. 주기적인 자원정보 보고 (사건1: RM -〉 QM)

RM은 자원의 이용율과 부하를 관측한다. 감시 정보는 주기적 또는 요청에 의해서 보고된다. 사건1은 시간 간격 Titv로 자원의 상태들을 보고한다.

```
typedef sruct{
int report_type;
int report_time;
int connect_id;
int connect_type:
int cpu;
int     s_buffer;
int     s_bandwidth;
 } periodic_idrep;
```

```
typedef      struct{
      int report_type;
       int report_time;
      int total_cn;
      resource cpu;
       resource   buffer;
      resource   bandwidth;
       rls          cpu;
       rls          buffer;
       rls          bandwidth;
        }   periodic_report;
```

그림 5. 주기적인 보고 형태
Fig.5. Periodic report form

그림5는 각각의 연결 및 연결 전체의 자원에 대한 주기적인 자원정보의 보고 형태를 나타낸다. 그림5a는 각각의 연결에 대한 자원 정보를 표시한다. connect_id는 할당된 연결 번호를 표시하고, connect_type은 연결의 종류가 보증 서비스 인가,

best-effort 서비스 인가를 구분한다. s_cpu, s_buffer, 그리고 s_bandwidth는 CPU, 버퍼 및 대역폭의 할당된 자원에 대한 사용율을 나타낸다.

그림5b는 연결 전체에 대한 자원의 상태를 보고하는 형태이고 이는 자원 전체에 대한 연결 전체에 할당된 자원정보를 표시한다. cpu.det_a는 구분A에서 사용하는 CPU 자원의 용량을 표시하고, cpu.det_b는 구분B에서 사용하는 CPU자원의 용량을 나타낸다. 그리고 cpu.best_c는 구분C에서 사용하는 CPU 자원의 용량을 표시한다. 버퍼와 대역폭도 CPU와 같은 의미를 갖는다.

메시지에 포함된 데이터들은 QM으로 전달되며, 궁극적으로는 QM MIB에 저장된다. 저장된 데이터들은 새로운 연결 수락을 결정하거나, 자원 동조에서의 판단 정보로 사용된다.

3.3.2. 과부하 자원정보 보고 (사건 2 : RM -〉 QM)

사건2는 할당된 자원들 중에서 하나 또는 그 이상의 자원이 과부하일 때 발생한다. 그러므로 사건2는 하나의 연결 또는 연결 전체에 대해서 발생할 수 있다. 하나의 연결에서 발생하는 과부하는 연결에 할당된 자원을 초과할 때 발생한다. 그리고 연결 전체에 대해서 발생하는 과부하는 연결 전체에 대해서 할당된 자원을 초과할 때 발생한다. 이와 같이 자원부하상태를 변경시키는 사건을 발견 하면 RM은 자원 과부하 상태에 대한 정보를 QM에게 전달한다. 그림6a는 하나의 연결에서 할당된 자원에 대한 과부하 상태의 정보 보고 형태를 나타내고, 그림6b는 연결 전체에 할당된 자원에 대한 과부하 상태가 발생했을 때의 정보 보고 형태를 보여준다. 자원의 병목이나 고갈은 통상적으로 QoS 감쇠(degradation)의 원인이 된다. 보고 메시지 포맷은 사건 1과 거의 유사하며, 차이점은 하나 또는 그 이상의 자원이 과부하 상태이고, 사용 가능한 용량이 거의 없다는 점이다. 모든 데이터는 QM에게 보고되며, 지역 QoS MIB에 저장된다.

```
typedef   struct{
int     report_type; ;
int     report_time;
int     connect_id;
int connect_type:
ints_cpu;
int     s_buffer;
int     s_bandwidth;
}critical_idrep;
```

```
typedef     struct{
   int     report_type;
  int     report_time;
   resource    cpu;
   resource    buffer;
   resource    bandwidth;
   rls          cpu;
  rls          buffer;
   rls          bandwidth;
    }  critical_report;
```

그림 6. 과부하 보고 형태
Fig. 6. overstate report fom

3.3.3. 자원 상태 요구(사건 2 : QM -〉RM)

사건3은 QM이 필요하다고 판단될 때마다, 비주기적으로 RM에게 자원 정보를 요구하는 사건이다. 주기적으로 보고된 자원 정보의 신뢰성에 문제가 있다고 판단되는 경우에 QM이 RM에게 현재의 자원에 관한 정보를 요구한다. 예를 들어서 사건4의 경우(연결 수락제어)에 현재시간에서 가장 최근에 보고된 시간을 뺀 시간이 보고 간격의 절반보다 크다면, QM은 RM에게 새로운 자원 정보 보고를 요구한다. 자원 정보 요구형태는 그림7과 같고, RM이 응답하는 형태는 주기적인 자원 정보 보고 형태와 같다.

connect_id가 0일 경우는 연결 전체에 대한 자원 정보를 요구하고, 0이외의 값은 연결번호로써 지정할 수 있다. 즉 각각의 연결에 대한 자원 정보를 필요로 할 때는 connect_id 값을 지정함으로써 각각의 연결에 대한 자원 정보를 요구할 수 있다.

```
typedef   struct{
int   request_type;
int   request_time; /* 자원정보 요청 시간 */
int   connect_id;   }   resour_request;
```

그림 7. 자원 정보 요구형태
Fig. 7. RI request form

3.3.4. 자원관리자 기본 알고리즘

```
while(1)   {
if ((Event occurs) >= 0 ) {
        switch(Event number) {
        Case periodic report & request RLS;
              gather the values (CPU, buffer, bandwidth);
              decide the RLS value;
              update the values and RLS value in RM
MIB;
              report to QoS manager; break;
        Case Event report;
              gather the values(CPU, buffer, bandwidth);
              decide the RLS value;
              write the RLS value;
              update the values and RLS values in
RMMIB;
              report to QoS manager; break;
        Case CAC;
              alloc_buffer;
              alloc_schedularbility;
              alloc_bandwidth;
              update RM MIB;
              response to QoS manager;
        Case resource tuning request;
              realloc_related resources;
              update RM MIB;
              response to QoS manager; break;
        Case connection end;
              delloc_resources;
```

```
        update RM MIB;
        report to QoS manager; break;
    }}}
```

그림 8. 자원관리자의 기본 알고리즘
Fig. 8. Basic Algorithm for RM

V. 결 론

본 연구에서는 자원 관리자는 주기적으로, 자원의 과부하가 발생했을 때, 혹은 QoS 관리자의 요구에 의해서 시스템 자원 정보를 보고한다. 보고된 자원 정보를 이용해서, QoS 관리자는 합의된 QoS 요구 수준을 만족하도록 관리한다. 이렇게 함으로써 프로토콜, QoS 관리자 및 자원 관리자를 같은 시간 영역으로 강력하게 결합시켰으며 프로토콜 시간 영역에서 발생할 수 있는 성능 변동(performance fluctuation)들을 실시간으로 감지하고 대응할 수 있도록 하였다.

앞으로의 과제는 제안된 구조에서 QoS관리자에 대한 연구가 수행되어 전체적인 기능의 구현을 수행할 것이다.

References

[1] W. K. Choi, "A study of transfer delay of Q-MOTP for multimedia object streams in MPLS network," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 38, no. 1, pp. 28-32, Feb. 2013.

[2] W. K. Choi, "Effective multimedia object data transport protocol in MPLS network using Q-CBQ method," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 37, no. 4, pp. 180-184, Aug. 2012.

[3] W. K. Choi, "Performance evaluation of Q-MOTP for multimedia object data transfer in MPLS network," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 37, no. 4, pp. 175-179, Aug 2012.

[4] W. K. Choi, "Performance evaluation of Q-CBQ method for multimedia streams in MPLS Router," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 37, no. 1, pp. 1-8, Feb. 2012.

[5] W. K. Choi, "Effective QoS supporting scheme for multimedia streams in MPLS router," *J. Korea Inform. Commun. Soc. (KICS)*, vol. 34, no. 8, pp. 260-266, Aug. 2009.

[6] W. K. Choi and S. S. Ahn "Performance analysis of multimedia-oriented error control mechanism over ATM network," *J. Korean Inst. Inform. Sci. Eng. (KIISE) (A)*, vol. 26, no. 7, pp. 827-838, July 1999.

[7] W. K. Choi and S. S. Ahn, "An adaptive QoS management based on resource information for multimedia streams over ATM," *J. Korean Inst. Inform. Sci. Eng. (KIISE) (A)*, vol. 25, no. 6, pp. 593-605, June 1998.

[8] G. Armitage, "MPLS: the magic behind the myths," *IEEE Commun. Mag.*, vol. 38, no. 1, pp. 124-131, Jan. 2000.

[9] G. M. Lee, and J. K. Choi, "Flow-based admission control for multiple service classes in the ATM-based MPLS network," in *Proc. IEEE Int. Conf. ATM (ICATM '01) High Speed Intell. Internet Symp.*, pp. 37-41, Seoul, Korea, Apr. 2001.

[10] C. Lin and E. C. M. Lam, "Dynamic queue length thresholds for scheduling real-time in ATM networks," in *Proc. IEEE Int. Conf. Commun. (ICC '99)*, vol. 2, pp. 869-874, Vancouver, Canada, June 1999.

최 원 근 (Won-keun Choi)

1982년 2월 아주대학교 전자공학과 졸업
1986년 2월 고려대학교 전자공학과 석사
1999년 8월 고려대학교 전자공학과 박사
1991년 9월~인하공업전문대학 정보통신과 근무
<관심분야> 멀티미디어, QoS, 트래픽관리