韓國經營科學會誌
第29卷 第4號
2004年 12月

MF-TDMA 광대역 위성시스템을 위한 QoS 기반 최적 타임슬롯 할당 체계*

장근녕** · 이기동*** · 박유진****

QoS-based Optimal Timeslot Allocation for MF-TDMA Broadband Satellite Systems*

Kun-Nyeong Chang** · Ki-Dong Lee*** · You-Jin Park****

Abstract

In this paper, we consider broadband satellite systems using MF-TDMA(Multi-Frequency Time Division Multiple Access) scheme. First, we analyze return link, superframe structure, and QoS(Quality of Service) parameters in broadband satellite systems, and mathematically formulate the QoS-based optimal timeslot allocation problem as a nonlinear integer programming problem for broadband satellite systems with clear-sky and rain-fade satellite terminals, and multiple data classes. Next, we modify the proposed problem to solve it within in a fast time, and suggest the QoS-based optimal timeslot allocation scheme. Extensive simulation results show that the proposed scheme finds an optimal solution or a near optimal solution within 5ms at Pentium IV PC.

Keyword : Optimal Timeslot Allocation, Broadband Satellite System, MF-TDMA, QoS

1. 서 론

위성통신서비스에 대한 수요가 급격히 증가함에 따라 미국, 유럽, 일본 등 위성기술 관련 선진국에서는 Ka 대역을 활용하여 멀티미디어 서비스를 제공하는 광대역 위성시스템 개발을 본격적으로 추진

논문접수일 : 2004년 4월 24일 논문게재확정일 : 2004년 11월 6일
* 이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음(KRF-2002-041-B00146).
** 연세대학교 정경대학 경영학과
*** 한국전자통신연구원 무선방송연구소
**** 연세대학교 경영학과

하고 있다[4]. 이러한 광대역 위성시스템의 효율적인 구현을 위해서는 해결해야 할 여러 가지 과제가 남아 있는데, 그 중에 가장 중요한 부분 중의 하나가 위성 액세스 자원의 효율적인 활용을 위한 다중접속 체계의 개발이다[2, 7, 10]. 현재 개발 중인 다중접속 프로토콜에는 MF-CDMA(Multi-Frequency Code Division Multiple Access)방식, MF-TDMA(Multi-Frequency Time Division Multiple Access)방식 등이 있다. 광대역 위성시스템 개발 분야에서 널리 활용되고 있는 ETSI의 DVB-RCS (Digital Video Broadcasting-Return Channel via Satellite) 표준에서는 MF-TDMA 방식을 사용하고 있다[1]. 따라서 이 방식 하에서 위성 액세스 자원을 가장 효율적으로 활용할 수 있는 방안을 찾는 것이 중요한 이슈라고 할 수 있고, 이에 대한 연구가 다양하게 이루어지고 있다[3, 5, 6, 8, 9].

지금까지의 연구 결과를 종합해 보면, 다양한 특성을 갖는 멀티미디어 서비스를 제공해야 하는 환경하에서는 CFDAMA(Combined Free/Demand Assignment Multiple Access) 방식이 가장 효율적인 것으로 나타나고 있다[8]. CFDAMA는 단말이 요청하는 타임슬롯 수요를 결정하는 방법, 요청된 타임슬롯 수요를 기반으로 각 단말에 타임슬롯을 할당하는 방법, 요청된 타임슬롯 수요를 충족시키고 남은 여유타임슬롯(free timeslots)을 할당하는 방법 등의 차이에 여러 가지로 나누어진다. 현재까지 CFDAMA에 기반을 두어 개발된 대표적인 방식을 살펴보면, P_CFDAMA(Pure CFDAMA), R_CFDAMA(Round-robin CFDAMA), W_CFDAMA (Weighted CFDAMA)[3, 8], RTA(Real-Time Algorithm for Timeslot Assignment)[5, 6] 등이 있다.

P_CFDAMA, R_CFDAMA, W_CFDAMA 등의 방식에서 각 단말의 타임슬롯 수요는 현재 버퍼에서 대기하고 있는 데이터의 양으로 결정된다. 이용 가능한 타임슬롯의 수가 전체 단말의 타임슬롯 수요보다 적으면, 각 단말에 할당되는 타임슬롯의 수는 요청 수요의 비율에 따라 결정된다. 이용 가능한 타임슬롯의 수가 전체 타임슬롯 수요보다 많으면 전체 수요를 충족시키고 타임슬롯이 남는데, 이 남은 여유타임슬롯은 P_CFDAMA에서는 할당되지 않고 버려지고, R_CFDAMA에서는 각 단말에 균등하게 할당되고, W_CFDAMA에서는 단말의 타임슬롯 요청 수요의 비율에 따라 할당된다. 그런데 이들 방법들은 단말 유형, 서비스 유형, 지연 정도와 같은 특성을 고려한 최적의 타임슬롯 할당체계를 제시하지 못한다.

이들 방법들의 단점을 보완하기 위해서 RTA에서는 단말 유형, 서비스 유형, 지연 정도와 같은 특성을 고려한 타임슬롯 할당체계를 제시하고 있다. 그런데 RTA에서는 보다 실제적인 시스템 제약을 반영하지 못하고 단순화하고 있다. 즉, CS 단말(clear-sky 단말 ; clear-sky 지역에 있는 단말)과 RF 단말(rain-fade 단말 ; rain-fade 지역에 있는 단말)에 할당될 수 있는 최대 타임슬롯 수를 공히 한 개의 RF용 타임-주파수 블록에 의해 생성되어지는 RF TRF(traffic) 타임슬롯의 수 이내로 한정하여 시스템을 단순화하고 있다. 실제로는 CS 단말에 할당될 수 있는 최대 타임슬롯 수를 한 개의 CS용 타임-주파수 블록에 의해 생성되어지는 CS TRF 타임슬롯의 수 이내로 한정하는 것이 보다 실제적이다. 둘째, RTA에서는 QoS(Quality of Service) 요구사항을 제대로 반영하지 못하고 있다. 전송되지 못하고 손실되는 데이터 수를 일정 수준 이하로 유지함으로써 시스템의 안정성을 높일 수 있는데, 이러한 점을 반영하지 못하고 있다. 셋째, RTA에서는 요청한 수요보다 이용 가능한 타임슬롯이 많을 경우, 이 남은 여유타임슬롯을 할당하기 위한 효율적인 방안은 제시하지 못하고 있다. 넷째, RTA에서는 CS TRF 타임슬롯의 집합과 RF TRF 타임슬롯의 집합을 사전적으로 결정하고 있으나, 이 보다는 최적 타임슬롯 할당 모형의 일부분으로 정형화하는 것이 보다 효율적인 타임슬롯 할당 체계를 수립할 수 있다.

본 논문에서는 먼저 CS SaT(Satellite Terminal ; 단말)과 RF SaT이 존재하고 복수의 데이터 클래스가 존재하는 MF-TDMA 광대역 위성 시스템을 대

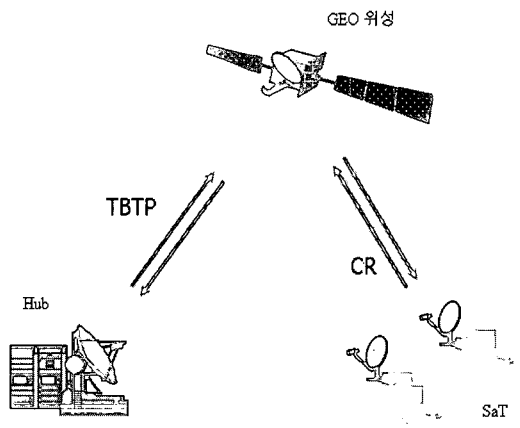
상으로, 슈퍼프레임 구조와 QoS 패러미터를 분석하고, 이를 토대로 앞에서 언급한 방식들에서 고려하지 못하고 있는 사항들을 고려하여 CS TRF 타임슬롯의 집합과 RF TRF 타임슬롯의 집합 그리고 타임슬롯 할당 스케줄을 함께 결정하는 QoS 기반 최적 타임슬롯 할당 모형을 정형화한다. 다음으로 이 모형을 보다 빠른 시간 내에 풀 수 있도록 여러 단계에 걸쳐 모형을 완화하고, QoS 기반 최적 타임슬롯 할당 체계를 제시한다. 제시된 타임슬롯 할당 체계는 CS 타임슬롯의 집합과 RF 타임슬롯의 집합 그리고 CS SaT의 집합 및 RF SaT의 집합에 할당되는 타임슬롯 수를 결정하고, 각 SaT의 각 데이터 클래스에 할당되는 타임슬롯의 수를 결정하고, 각 SaT의 각 데이터 클래스에 할당된 타임슬롯들의 스케줄을 결정한다. 마지막으로 제시한 최적 타임슬롯 할당 체계의 성능을 분석한다.

본 논문은 다음과 같이 구성되어 있다. 제2절에서는 리턴 링크 모형, 슈퍼프레임 구조, QoS 패러미터 등 광대역 위성 시스템 모형을 설명한다. 제3절에서는 CS SaT와 RF SaT가 존재하고 복수의 데이터 클래스가 존재하는 MF-TDMA 광대역 위성 시스템을 대상으로 QoS 기반 최적 타임슬롯 할당 모형을 정형화한다. 제4절에서는 QoS 기반 최적 타임슬롯 할당 체계를 제시하고, 제5절에서는 테스트를 통해 제시된 할당 체계의 성능을 분석한다. 제6절에서는 결론을 제시한다.

2. 시스템 모형

2.1 리턴 링크 모형

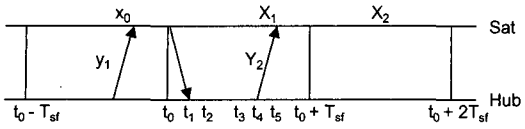
본 연구에서는 [그림 1]과 같이 하나의 지구국(earth station ; Hub), 하나의 GEO 위성, 복수의 SaT로 구성된 대화형 광대역 위성 액세스 네트워크를 대상으로 연구를 수행한다. SaT는 CS SaT와 RF SaT로 구분되어진다.



[그림 1] 광대역 위성 액세스 네트워크

리턴 링크(return link ; 위성을 거쳐 SaT에서 지구국으로의 링크)에서의 다중접속체계(multiple access scheme)는 MF-TDMA에 근거를 두고 있다. 리턴 링크에 할당되는 라디오 자원은 다수의 SaT에 의해 공유된다. SaT는 Hub에 CR(capacity request) 메시지를 전송하고, 이 메시지를 받은 Hub는 TBTP(terminal burst time plan)를 생성하여 SaT에 전송한다. TBTP를 받은 SaT는 이를 통해 자신에게 어떤 타임슬롯들이 할당되었는지를 파악하게 된다. 이러한 프로시저는 매 슈퍼프레임(superframe)마다 실시된다.

SaT는 필요할 때마다 Hub에 CR 메시지를 전송한다. Hub는 SaT가 슈퍼프레임 단위 시간 $(t_0 - T_{sf}, t_0)$동안 요청한 수요를 축적한다([그림 2] 참조). Hub는 SaT가 요청한 수요를 이용하여 t_2시점까지 최적 할당량을 결정하고, t_3시점까지 TBTP 테이블을 생성한다. SaT는 생성된 TBTP를 t_5시점에 받게 되고, 이 테이블에 따라 자신에게 할당된 타임슬롯을 기다리게 된다. 시간 간격 $(t_5, T_{sf} + t_0)$가 너무 짧으면, SaT는 스케줄을 제대로 읽을 수 없게 되고, 따라서 이 스케줄을 다음 슈퍼프레임에서 사용할 수 없게 된다. SaT에게 수신한 TBTP 테이블을 읽을 수 있는 충분한 시간을 제공하기 위해서는 타임슬롯 할당 시간을 최대한 줄일 필요가 있다.



[그림 2] SaT와 Hub간의 메시지 전송 체계

2.2 슈퍼프레임의 구조

슈퍼프레임의 구조는 시스템에 따라 다양하게 정의될 수 있지만, 본 논문에서는 아래와 같이 정의하고 논문을 전개한다. 물론 본 논문에서 제시되는 내용은 슈퍼프레임의 구조에 관계없이 쉽게 적용될 수 있다.

하나의 슈퍼프레임은 시간-주파수 도메인(time-frequency domain)에서 특정한 시간-주파수 블록(time-frequency block) $T_{sf} \times W_{sf}$ (예를 들어 $T_{sf} = 800ms$, $W_{sf} = 64MHz$)으로 정의되고, 이 시간-주파수 블록(time-frequency block) $T_{sf} \times W_{sf}$는 $\overline{b}$개의 시간-주파수 블록 $T_{sf} \times W_f$ ($W_f = W_{sf}/\overline{b}$)로 나누어진다. 블록 $T_{sf} \times W_f$는 CS 프레임(CS 타임슬롯) 또는 RF 프레임(RF 타임슬롯)을 생성하기 위해 사용된다.

블록 $T_{sf} \times W_f$가 CS 프레임을 생성하기 위해 사용되는 경우, 특정한 시간-주파수 블록 $T_f^c \times W_f$ ($T_f^c = T_{sf}/n_c$)로 정의되는 n_c개의 CS 프레임으로 나누어진다. 이 CS 프레임은 하나의 CSC(common signaling channel) 타임슬롯, 하나의 ACQ (acquisition) 타임슬롯, 두 개의 SYNC(synchronization) 타임슬롯, n_{ct}개의 TRF(traffic) 타임슬롯으로 구성되어진다.

블록 $T_{sf} \times W_f$가 RF 프레임을 생성하기 위해 사용되는 경우, 특정한 시간-주파수 블록 $T_f^r \times W_f$ ($T_f^f = T_f/n_r$)로 정의되는 n_r개의 RF 프레임으로 나누어진다. 이 RF 프레임은 n_b개의 시간-주파수 블록 $T_f^r \times W_f^r$ ($W_f^r = W_f/n_b$)로 나누어지고, 각각의 블록은 하나의 CSC 타임슬롯, 하나의 ACQ 타임슬롯, 두 개의 SYNC 타임슬롯, n_{rt}개의 TRF (traffic) 타임슬롯으로 구성되어진다.

2.3 QoS 패러미터

SaT는 서로 다른 전송 모드를 사용하는 CS SaT와 RF SaT로 구분되어 진다. 또한, 각 SaT에서 전송하고자 하는 데이터는 여러 가지 형태의 클래스로 나누어진다. 실시간(real-time) 또는 비실시간 서비스, 서비스 유형 클래스(service type class), 지연 클래스(delay class) 등이 여기에 포함될 수 있다. 서비스 유형 클래스는 e-mail, 채팅, FTP 등과 같이 서비스 형태별로 나누어진다. 지연 클래스는 요청된 타임슬롯의 지연 정도에 따라 나누어진다.

시스템의 QoS를 보장하기 위해 클래스별 가중치 w_{jk}(SaT j의 클래스 k 데이터에 대한 가중치, $k = (k_1, k_2, \cdots)$), 클래스별 최소 타임슬롯 할당량 m_{jk} (SaT j의 클래스 k 데이터에 대한 최소 타임슬롯 할당량 ; SaT j의 클래스 k 데이터의 수요량에 대한 타임슬롯 할당량 비율의 최소값 α_{jk}에 의해 결정), SaT 별 최소 타임슬롯 할당수 m_j(SaT j에 대한 최소 타임슬롯 할당량) 등의 퍼러미터를 도입한다.

w_{jk}는 평균 대기 시간(average waiting time) 등의 여러 가지 요인에 의해 결정된다. 일반적으로 RF SaT의 GOS(Grade of Service)가 CS SaT의 GOS보다 더 중요하기 때문에, RF SaT의 가중치가 CS SaT의 가중치보다 크다. 본 연구에서는 RF SaT에 절대적인 우선권을 주어, RF SaT의 가중치가 데이터 클래스에 관계없이 CS SaT의 가중치보다 월등히 큰 것으로 가정한다. 즉, RF SaT에 우선적으로 타임슬롯이 할당된다.

α_{jk}는 수요량 대 할당량을 어느 정도 이상의 수준으로 유지하기 위하여 도입한 패러미터로, 이는 시스템의 효율적인 운용 차원에서 필요하다. α_{jk}가 0.5라면, SaT j의 클래스 k 데이터의 수요인 d_{jk}의 50%에 해당되는 타임슬롯은 최소한 할당되어야한다는 것을 의미한다. 이 값은 서비스 제공업자의 정

책에 따라 정해진다.

SaT j에 대한 최소 타임슬롯 할당량 m_j는 두 개의 패러미터에 의해 결정된다. 즉 $m_j = \max\{m_j^a, m_j^b\}$이다. m_j^a는 SaT와 지구국간 통신을 위해 필요한 최소 타임슬롯 할당량을 나타낸다. SaT가 추가적인 자원요청을 할 때, CR 메시지 등을 원활하게 송신하도록 하기 위해서는 클래스에 관계없이 SaT별로 최소한의 타임슬롯이 무조건 필요하다. 일반적으로 m_j^a는 단말에 관계없이 계속해서 일정한 값을 가진다.

m_j^a는 버퍼에서 대기하고 있는 데이터 양(패킷 수)이 주어진 임계값 B_j^T를 넘지 않을 확률(임계값 B_j^T를 버퍼 크기(buffer size) B_j^S와 동일하게 두면, 데이터 손실이 발생하지 않을 확률을 나타냄)을 일정 수준 이하로 유지하는데 필요한 최소 타임슬롯 할당량을 나타낸다. [그림 2]에서 t_0시점에 버퍼에서 대기하고 있는 데이터 양을 O_0 라 하자. x_0를 $(t_0 - T_{sf}, t_0)$에 발생한 데이터 양이라 하고, y_1을 $(t_0, t_0 + T_{sf})$에 사용하도록 할당된 타임슬롯 수라 하자. 또한, X_1과 X_2를 $(t_0, t_0 + T_{sf})$와 $(t_0 + T_{sf}, t_0 + 2T_{sf})$에 발생할 것으로 예상되는 데이터 양이라 하고, Y_2를 $(t_0 + T_{sf}, t_0 + 2T_{sf})$에 사용하도록 할당되어야 하는 타임슬롯 수라 하자. 이 때 $t_0 + T_{sf}$ 시점에 버퍼에서 대기하고 있는 데이터 양은 $O_1 = \min\{\max\{O_0 + X_1 - y_1, 0\}, B_j^S\}$ 이다. 버퍼에서 대기하고 있는 데이터 양이 주어진 임계값 B_j^T를 넘지 않을 확률을 임계값 β_j보다 크게 되도록 하기 위해서는 $P\{O_1 + X_2 \le Y_2 + B_j^T\} \ge \beta_j$를 만족하도록 Y_2를 설정하면 된다. 그런데

$$P\{O_1 + X_2 \le Y_2 + B_j^T\} =$$

$$\begin{cases} P\{X_2 \le Y_2 + B_j^T\}, \\ \quad \text{if } O_0 + X_1 - y_1 \le 0 \\ P\{X_1 + X_2 \le Y_2 + B_j^T + y_1 - O_0\}, \\ \quad \text{if } 0 < O_0 + X_1 - y_1 \le B_j^S \\ P\{X_2 \le Y_2 + B_j^T - B_j^S\}, \\ \quad \text{if } O_0 + X_1 - y_1 > B_j^S \end{cases}$$

이다. 따라서

$$\begin{aligned} P\{O_1 + X_2 \le Y_2 + B_j^T\} \\ &= P\{X_1 \le y_1 - O_0\} * P\{X_1 \le Y_2 + B_j^T\} \\ &\quad + P\{y_1 - O_0 < X_1 \le B_j^S + y_1 - O_0\} \\ &\quad * P\{X_1 + X_2 \le Y_2 + B_j^T + y_1 - O_0\} \\ &\quad + P\{X_1 > B_j^S + y_1 - O_0\} \\ &\quad * P\{X_2 \le Y_2 + B_j^T - B_j^S\} \end{aligned}$$

이다. 이제 X_1과 X_2를 예측하여, $P\{O_1 + X_2 \le Y_2 + B_j^T\} \ge \beta_j$ 를 만족하는 Y_2를 구할 수 있고, 이 값이 SaT j에 대한 m_j^b가 된다.

3. 최적 타임슬롯 할당 모형

여기에서는 QoS 기반 최적 타임슬롯 할당문제(Timeslot Assignment Problem : TAP)를 수리적으로 정형화한다. 타임슬롯 할당문제 정형화에 사용되는 패러미터를 정리하면 다음과 같다.

R : 활성(active) SaT의 집합 ($R = R_c \cup R_r$)

R_c : 활성 CS SaT의 집합

R_r : 활성 RF SaT의 집합

R_c' : $\max\{m_j, \sum_{k \in C} m_{jk}\} > N_r$인 활성 CS SaT들의 집합

C_i : i번째 데이터 클래스의 집합, $C = (C_1, C_2, \cdots)$

$\overline{b}$: 전체 타임-주파수 블록 $T_{sf} \times W_f$의 수 ($\overline{b} = b_c + b_r$)

b_c : CS 프레임을 생성하는데 사용되는 타임-주파수 블록 $T_{sf} \times W_f$의 수

b_r : RF 프레임을 생성하는데 사용되는 타임-주파수 블록 $T_{sf} \times W_f$의 수

N_c : 한 개의 CS 타임-주파수 블록 $T_{sf} \times W_f$에 의해 생성되어지는 CS TRF 타임슬롯의 수 ($= n_c \times n_{ct}$)

N_r : 한 개의 RF 타임-주파수 블록 $T_{sf} \times W_f^r$에 의해 생성되어지는 RF TRF 타임슬롯의 수

$(=n_r \times n_{rt})$

S : TRF 타임슬롯의 집합($S=S_c \cup S_r$)

S_c : CS TRF 타임슬롯의 집합, $S_c=\{1,\cdots,N_c \times b_c\}$

S_r : RF TRF 타임슬롯의 집합, $S_r=\{N_c \times b_c+1,\cdots,N_c \times b_c+n_b \times N_r \times b_r\}$

F_c^t : 각 CS 타임-주파수 블록 $T_{sf} \times W_f$에 의해 생성되어지는 t번째 CS TRF 타임슬롯들의 집합 ($t=1,\cdots,N_c$)

F_r^t : 각 RF 타임-주파수 블록 $T_{sf} \times W_f^r$에 의해 생성되어지는 t번째 RF TRF 타임슬롯들의 집합 ($t=1,\cdots,N_r$)

d_{jk} : SaT j의 클래스 k 데이터 전송을 위해 요청된 TRF 타임슬롯의 수($k=(k_1,k_2,\cdots)$)

Q_j^c : CS SaT j에 할당될 수 있는 최대 타임슬롯 수($Q_j^c \le N_c,\ j \in R_c$)

Q_j^r : RF SaT j에 할당될 수 있는 최대 타임슬롯 수($Q_j^r \le N_r,\ j \in R_r$)

w_{jk} : SaT j의 클래스 k 데이터에 대한 가중치

α_{jk} : SaT j의 클래스 k 데이터의 수요량에 대한 타임슬롯 할당량 비율의 최소값

m_{jk} : SaT j의 클래스 k 데이터에 필요한 최소 타임슬롯 할당량($m_{jk}=\lceil \alpha_{jk} d_{jk} \rceil$)

m_j : SaT j에 대한 최소 타임슬롯 할당량($m_j=\max\{m_j^a, m_j^b\}$)

m_j^a : SaT와 지구국간 통신을 위해 필요한 최소 타임슬롯 할당량

B_j^S : SaT j의 버퍼 크기 (buffer size)

β_j : 버퍼에서 대기하고 있는 데이터 양(패킷 수)이 주어진 임계값 B_j^T를 넘지 않을 확률 ($B_j^T=B_j^S$이면, 데이터 손실이 발생하지 않을 확률을 나타냄)에 대한 임계값

m_j^b : 주어진 임계값 β_j를 달성하는 데 필요한 최소 타임슬롯 할당량

CS TRF 타임슬롯들의 집합인 S_c와 RF TRF 타임슬롯들의 집합인 S_r은 CS 프레임을 생성하는데 사용되는 타임-주파수 블록 $T_{sf} \times W_f$의 수인 b_c와 RF 프레임을 생성하는데 사용되는 타임-주파수 블록 $T_{sf} \times W_f$의 수인 b_r에 의해 정해진다. 한편, 일관성의 결여 없이 $S_c=\{1,\cdots,N_c \times b_c\}$와 $S_r=\{N_c \times b_c+1,\cdots,N_c \times b_c+n_b \times N_r \times b_r\}$로 표현할 수 있다.

TAP의 목적은 다음과 같이 가중 페널티(weighted sum)의 합을 최소화하는 것으로 정의한다.

$$\sum_{j \in R} \sum_{k \in C} w_{jk} \cdot \max\{d_{jk} - \sum_{i \in S} x_{ijk}, 0\}.$$

여기서 x_{ijk}은 타임슬롯 할당 여부를 나타내는 이진 정수(binary integer) 변수로, x_{ijk}가 1이면 TRF 타임슬롯 i가 SaT j의 클래스 k 데이터에 할당되었다는 것을 의미하고 x_{ijk}가 0이면 그렇지 않다는 것을 의미한다. $\sum_{i \in S} x_{ijk}$는 SaT j의 클래스 k 데이터에 할당된 타임슬롯의 총 수를 나타내고, 따라서 $\max\{d_{jk} - \sum_{i \in S} x_{ijk}, 0\}$는 SaT j의 클래스 k 데이터에서 타임슬롯 할당으로 충족되지 않은 타임슬롯의 수를 나타낸다.

각 SaT이 요청한 수요를 모두 충족시킬 수 있는 것이 가장 바람직하고, 그렇지 못할 경우 가능한 한 충족되지 못하는 수요를 줄이는 것이 바람직하다. 다만 이 때, SaT 유형이나 데이터 클래스 유형에 따라 수요 충족의 중요도가 다를 수 있기 때문에, 단순히 충족되지 못하는 수요의 합을 최소화하는 것보다는 각 SaT의 우선순위와 각 데이터 클래스의 우선순위를 고려하는 가중합을 최소화하는 것이 시스템 측면에서 더 바람직하다. 물론 모든 가중치가 동일하다고 가정하면 단순히 충족되지 못하는 수요의 합을 최소화하는 것과 동일하게 된다.

이제, TAP는 다음과 같은 수리모형으로 정형화되어진다.

(TAP)

$$Min \sum_{j \in R} \sum_{k \in C} w_{jk} \cdot \max\{d_{jk} - \sum_{i \in S} x_{ijk}, 0\} \quad (1)$$

$$s.t. \sum_{j \in R} \sum_{k \in C} x_{ijk} \le 1, \quad \forall i \in S \quad (2)$$

$$\sum_{i\in S}\sum_{k\in C} x_{ijk} \le Q_j^c (\le N_c), \quad \forall j \in R_c \tag{3}$$

$$\sum_{i\in S_r}\sum_{k\in C} x_{ijk} \le Q_j^r (\le N_r), \quad \forall j \in R_r \tag{4}$$

$$z_j \sum_{i\in S_c}\sum_{k\in C} x_{ijk} + (1-z_j)\sum_{i\in S_r}\sum_{k\in C} x_{ijk} = 0, \ \forall j \in R \tag{5}$$

$$\sum_{i\in F_c^t}\sum_{k\in C} x_{ijk} \le 1, \quad \forall j \in R, t = 1, \cdots, N_c \tag{6}$$

$$\sum_{i\in F_r^t}\sum_{k\in C} x_{ijk} \le 1, \quad \forall j \in R, t = 1, \cdots, N_r \tag{7}$$

$$\sum_{i\in S} x_{ijk} \ge m_{jk}, \quad \forall j \in R, k \in C \tag{8}$$

$$\sum_{i\in S}\sum_{k\in C} x_{ijk} \ge m_j^a, \quad \forall j \in R \tag{9}$$

$$P_j\left(\sum_{i\in S}\sum_{k\in C} x_{ijk}\right) \ge \beta_j, \quad \forall j \in R \tag{10}$$

$$b_c + b_r = \overline{b} \tag{11}$$

$$z_j = 1, \quad \forall j \in R_r \tag{12}$$

$$z_j = 0, \quad \forall j \in R_c' \tag{13}$$

$$z_j = 0 \text{ or } 1, \quad \forall j \in R_c \tag{14}$$

$$x_{ijk} = 0 \text{ or } 1, \quad \forall i \in S, j \in R, k \in C \tag{15}$$

$$b_c, b_r: \text{ nonnegative integers.} \tag{16}$$

여기서 z_j는 이진 정수 변수로, z_j가 1이면 SaT j에 RF 타임슬롯이 할당된다는 것을 나타내고, z_j가 0이면 SaT j에 CS 타임슬롯이 할당된다는 것을 나타낸다.

제약식 (2)는 하나의 타임슬롯이 두 번 이상 할당되어서는 안 된다는 것이다. 제약식 (3)은 CS SaT j의 데이터에 할당되는 타임슬롯의 수는 최대용량 $Q_j^c(\le N_c)$ 보다 많아서는 안 된다는 것이고, 제약식 (4)는 RF SaT j의 데이터에 할당되는 타임슬롯의 수는 최대용량 $Q_j^r(\le N_r)$ 보다 많아서는 안 된다는 것이다. 제약식 (5)는 하나의 SaT가 CS 타임슬롯과 RF 타임슬롯을 같이 사용해서는 안 된다는 것이다. 제약식 (6)과 (7)은 하나의 SaT에 동일한 타임 도메인(time domain)을 사용하는 타임슬롯이 두 개 이상 할당되어서는 안 된다는 것이다. 제약식 (8)은 SaT j의 클래스 k 데이터에 할당되는 타임슬롯의 수는 최소 요구량 m_{jk}이상이어야 한다는 것이다. 제약식 (9)는 SaT j에 할당되는 타임슬롯의 수 $\sum_{i\in S}\sum_{k\in C} x_{ijk}$는 최소 요구량 m_j^a이상이어야 한다는 것이다. 제약식 (10)은, SaT j에 $\sum_{i\in S}\sum_{k\in C} x_{ijk}$개의 타임슬롯이 할당될 때 SaT j의 버퍼에서 대기하고 있는 데이터 양이 주어진 임계값 B_j^T를 넘지 않을 확률($B_j^T = B_j^S$이면, 데이터 손실이 발생하지 않을 확률을 나타냄)을 의미하는 $P_j(\sum_{i\in S}\sum_{k\in C} x_{ijk})$가 임계값 β_j이상이어야 한다는 것이다. 이 제약식은 $\sum_{i\in S}\sum_{k\in C} x_{ijk} \ge m_j^b$ 형태로 변형되어질 수 있다. 제약식 (11)은 CS 프레임과 RF 프레임을 생성하는데 사용되는 타임-주파수 블록의 수의 합은 전체 타임-주파수 블록의 수 $\overline{b}$라는 제약식이다. 제약식 (12)는 RF SaT에는 RF 타임슬롯만을 할당해야 한다는 것이고, 제약식 (13)은 SaT $j \in R_c'$에는 CS 타임슬롯만이 할당되어야 한다는 것이다. 제약식 (13)은 제약식 (8)과 (9)가 만족되면 제약식 (6)과 (7)에 의해 자동적으로 만족되기 때문에 제외해도 상관없지만, 모형의 이해를 높이기 위해서 첨가한 것이다.

타임슬롯 할당은 매 슈퍼프레임마다 실시간으로 이루어져야 하고, 따라서 아주 빠른 시간 이내에 해를 구할 수 있어야 한다(빠를수록 좋고, 시스템에 따라 다를 수 있지만 최소한 수십 ms 이내에 해를 구해야 함). 그런데 TAP는 비선형 정수계획문제(nonlinear integer programming problem)로, 이진 정수변수 m_{ijk}는 $2^{|S|*|R|*|C|}$ 개의 경우의 수를 가지고 이진 정수변수 z_j는 $2^{|R|}$개의 경우의 수를 가진다. 또한 TAP의 경우 문제의 특성상 동일한 목적식 값을 갖는 해가 매우 많이 존재한다. 예를 들어, SaT #1에 1번째에서 10번째까지의 타임슬롯이 할당되든 101번째에서 110번째까지의 타임슬롯이 할당되든 목적식 값은 동일하다. 이러한 특성 때문에 TAP를 직접 다루는 것 보다 각 SaT j의 클래스 k에 할당되는 타임슬롯의 수 y_{jk}를 결정하는 문제 RTAP로 바꾸어 접근하는 것이 보다 효율적이다. 한편 Lemma 1에 의해 RTAP는 TAP와 동등한 문제(equivalent problem)임을 알 수 있고, 4.4절에 제시된 프로시저 PTAP를 이용하여 RTAP의 최적해로부터 TAP의 최적해를 쉽게 구할 수 있다.

(RTAP)

$$Min \quad \sum_{j\in R}\sum_{k\in C} w_{jk}\cdot \max\{d_{jk}-y_{jk},0\} \tag{17}$$

$$s.t. \quad \sum_{k\in C} y_{jk}\le Q_j^c, \quad \forall j\in R_c \tag{18}$$

$$\sum_{k\in C} y_{jk}\le Q_j^r, \quad \forall j\in R_r \tag{19}$$

$$\sum_{j\in R}(1-z_j)\sum_{k\in C} y_{jk}\le b_c\times N_c \tag{20}$$

$$\sum_{j\in R} z_j\sum_{k\in C} y_{jk}\le n_b\times b_r\times N_r \tag{21}$$

$$y_{jk}\ge m_{jk}, \quad \forall j\in R, k\in C \tag{22}$$

$$\sum_{k\in C} y_{jk}\ge m_j^a, \quad \forall j\in R \tag{23}$$

$$P_j(\sum_{k\in C} y_{jk})\ge\beta_j, \quad \forall j\in R \tag{24}$$

$$b_c+b_r=\overline{b} \tag{25}$$

$$z_j=1, \quad \forall j\in R_r \tag{26}$$

$$z_j=0, \quad \forall j\in R_c \tag{27}$$

$$z_j=0 \text{ or } 1, \quad \forall j\in R_c \tag{28}$$

$$y_{jk}, b_c, b_r, : \text{nonnegative integers}$$
$$\forall j\in R, k\in C. \tag{29}$$

여기서 제약식 (24)는 $\sum_{k\in C} y_{jk}\ge m_j^b$로 변형되어질 수 있다.

Lemma 1. RTAP는 TAP와 동등한 문제(equivalent problem)이다.

증명. RTAP의 최적해를 $y_{jk}^*, z_j^*, b_c^*, b_r^*$라 하자. $z_j^*=1$인 SaT($z_j^*=0$인 SaT)에 대해서는 S_r에 속한 타임슬롯(S_c에 속한 타임슬롯)을 첫 타임슬롯부터 순차적으로 y_{jk}^* 만큼씩 할당하면(4.4절의 프로시저 PTAP 참조), 제약식 (2), (5)~(7)을 만족하는 x_{ijk}^*를 구할 수 있다. 이때 $z_j^*=1$인 SaT에 대해서는 $\sum_{i\in S_r} x_{ijk}^*=y_{jk}^*$가 되고 $z_j^*=0$ 인 SaT에 대해서는 $\sum_{i\in S_c} x_{ijk}^*=y_{jk}^*$ 가 된다. 한편, y_{jk}^*가 RTAP의 제약식 (18), (19), (22)~(24)를 만족하므로, 대입을 통해 x_{ijk}^*도 TAP의 제약식 (3), (4), (8)~(10)을 만족함을 간단히 알 수 있다. 따라서 $x_{ijk}^*, z_j^*, b_c^*, b_r^*$는 TAP의 실행가능해이다.

이제 $x_{ijk}^*, z_j^*, b_c^*, b_r^*$가 TAP의 최적해가 아니고, $\overline{x_{ijk}}, \overline{z_j}, \overline{b_c}, \overline{b_r}$가 최적해라 하자. 이 경우 $\overline{y_{jk}}$ $(=\sum_{i\in S}\overline{x_{ijk}})$, $\overline{z_j}, \overline{b_c}, \overline{b_r}$ 가 RTAP의 실행가능해이고 $\overline{y_{jk}}, \overline{z_j}, \overline{b_c}, \overline{b_r}$ 에서의 목적식 값이 $y_{jk}^*, z_j^*, b_c^*, b_r^*$에서의 목적식 값보다 크므로, $y_{jk}^*, z_j^*, b_c^*, b_r^*$가 RTAP의 최적해라는 것과 모순이다. 따라서 $x_{ijk}^*, z_j^*, b_c^*, b_r^*$는 TAP의 최적해이다.

한편, TAP의 최적해가 $x_{ijk}^*, z_j^*, b_c^*, b_r^*$라 하면, $y_{jk}^*(=\sum_{i\in S} x_{ijk}^*), z_j^*, b_c^*, b_r^*$는 RTAP의 최적해가 된다. 따라서 RTAP는 TAP와 동등한 문제이다. ■

$y_{jk}^*, z_j^*, b_c^*, b_r^*$를 RTAP의 최적해라 할 때, $\sum_{j\in R_c^*}\sum_{k\in C} y_{jk}^* < b_c^*\times N_c$또는 $\sum_{j\in R-R_c^*}\sum_{k\in C} y_{jk}^* < n_b\times b_r^*\times N_r$ 이면 할당되지 않고 남은 타임슬롯이 존재하는 것이다. 여기서 R_c^c는 CS 타임슬롯이 할당되는 CS SaT들의 집합($z_j^*=1$인 SaT들의 집합)을 나타내고, $R-R_c^c$는 RF 타임슬롯이 할당되는 SaT들의 집합($z_j^*=0$ 인 SaT들의 집합)을 나타낸다. 남은 타임슬롯을 할당 할 때 여러 가지 기준을 사용할 수 있지만, 버퍼에서 대기하고 있는 데이터 양이 주어진 임계값 B_j^T를 넘지 않을 확률($B_j^T=B_j^S$이면, 데이터 손실이 발생하지 않을 확률을 나타냄)을 줄이는 방향으로 할당하는 것이 바람직하다. 아래의 FTAP는 이러한 관점에서 남은 타임슬롯의 할당 방법을 결정하는 문제이다.

(FTAP)

$$Max \quad \min\{\frac{1}{w_j}P_j(\sum_{k\in C}(y_{jk}^*+y_{jk}^F))\} \tag{30}$$

$$s.t. \quad \sum_{k\in C}(y_{jk}^*+y_{jk}^F)\le Q_j^c, \quad \forall j\in R_c \tag{31}$$

$$\sum_{k\in C}(y_{jk}^*+y_{jk}^F)\le Q_j^r, \quad \forall j\in R_r \tag{32}$$

$$\sum_{j\in R_c^c}\sum_{k\in C}(y_{jk}^*+y_{jk}^F)\le b_c^*\times N_c \tag{33}$$

$$\sum_{j\in R-R_c^c}\sum_{k\in C}(y_{jk}^*+y_{jk}^F)\le n_b\times b_r^*\times N_r \tag{34}$$

$$y_{jk}^F : \text{nonnegative integers},$$
$$\forall j\in R, k\in C. \tag{35}$$

여기서 y_{jk}^{F}는 남은 타임슬롯 중에 SaT j의 클래스 k에 할당되는 타임슬롯의 수를 나타내고, w_j는 남은 타임슬롯을 할당할 때 사용되는 SaT j에 대한 가중치를 나타낸다.

Lemma 2. $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$가 RTAP의 최적해이고 y_{jk}^{F*}가 FTAP의 최적해일 때, $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$는 RTAP의 최적해이다.

증명. 먼저 $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$는 RTAP의 실행가능해이다. y_{jk}^{*}가 제약식 (31)~(34)를 만족하므로 $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$는 제약식 (18)~(21)을 만족하고, 또한 $y_{jk}^{*}+y_{jk}^{F*}\geq y_{jk}^{*}$ 이므로 나머지 제약식 (22)~(29)도 만족한다. 따라서 $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$는 RTAP의 실행가능해이다.

다음으로 $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값과 $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값은 동일하다. 우선 $y_{jk}^{*}+y_{jk}^{F*}\geq y_{jk}^{*}$이므로 $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값은 $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값보다 작거나 같다. 그런데, $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값이 $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값보다 작다면, $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$가 RTAP의 최적해라는 것과 모순이다. 따라서 $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값과 $y_{jk}^{*}, z_j^{*}, b_c^{*}, b_r^{*}$일 때의 목적식 값이 동일하고, $y_{jk}^{*}+y_{jk}^{F*}, z_j^{*}, b_c^{*}, b_r^{*}$는 RTAP의 최적해이다. ▣

한편, 비선형 정수계획문제인 RTAP의 경우 z_j변수로 인해 비선형 문제로 정형화되어 있다. 따라서 이 변수를 제외할 수 있다면 보다 쉽게 문제를 풀 수 있을 것으로 보인다. 이에 따라 RTAP의 제약식 (20)과 (21)을 완화하여 RTAP를 아래의 RRTAP로 완화한다. RRTAP에서는 RTAP에서와는 달리 CS SaT의 경우 CS 타임슬롯과 RF 타임슬롯을 동시에 사용할 수 있도록 하고 있다. 따라서 RRTAP의 최적 목적식 값은 RTAP의 최적 목적식 값보다 작거나 같다.

(RRTAP)

$$Min \sum_{j\in R}\sum_{k\in C} w_{jk}\cdot \max\{d_{jk}-y_{jk},0\} \tag{36}$$

s.t. & (18) – (19), (22) – (24), (29)

$$\sum_{j\in R}\sum_{k\in C} y_{jk}\leq b_c\times N_c+n_b\times b_r\times N_r \tag{37}$$

$$\sum_{j\in R_r}\sum_{k\in C} y_{jk}\leq n_b\times b_r\times N_r \tag{38}$$

$$\sum_{j\in R_c'}\sum_{k\in C} y_{jk}\leq b_c\times N_c. \tag{39}$$

Remark 1. RRTAP의 최적 목적식 값은 RTAP의 하한(lower bound)이다.

증명. RTAP의 모든 실행가능해(feasible solution)는 RRTAP의 실행가능해이다. 반면에 RRTAP의 실행가능해 중에는 RTAP의 실행가능해가 아닌 경우가 있다. 즉, RRTAP에서 CS 타임슬롯과 RF 타임슬롯을 동시에 사용하는 CS SaT가 있는 경우 RTAP의 실행가능해가 될 수 없다. 따라서 RRTAP의 최적 목적식 값은 RTAP의 하한이 된다. ▣

4. 최적 타임슬롯 할당 체계

여기에서는 빠른 시간 내에 TAP를 해결하기 위한 QoS 기반 최적 타임슬롯 할당체계를 제시한다. 먼저 RRTAP의 해를 구하고(CS 타임슬롯의 집합 및 RF 타임슬롯의 집합 결정, CS SaT의 집합 및 RF SaT의 집합에 할당되는 타임슬롯 수 결정), 다음으로 RTAP와 FTAP의 해를 구하고(각 SaT의 각 데이터 클래스에 할당되는 타임슬롯의 수 결정), 마지막으로 TAP의 해를 구한다(각 SaT의 각 데이터 클래스에 할당된 타임슬롯들의 스케줄 결정).

4.1 RRTAP 해결 체계

RF SaT들의 가중치가 CS SaT들의 가중치보다 월등히 크다고 가정하였기 때문에, RRTAP에서 제약식이 만족되는 범위 내에서 RF SaT에 최대한 많은 타임슬롯을 할당하는 것이 최적이다. 즉, y_{jk}, $j\in R_r$의 값을 크게 할수록 좋다. 따라서 타임슬롯

집합을 결정하는 패러미터인 b_c와 b_r는 다음의 TSDP를 통해서 구할 수 있다.

(TSDP)

$$
\begin{aligned}
&Max\ \ M\times y_r + y_c \\
&s.t.\ \ y_c \le \min\{\sum_{j\in R_c}\max\{\sum_{k\in C} d_{jk}, m_j\},\ \sum_{j\in R_c} Q_j^c\} \\
&\qquad y_r \le \min\{\sum_{j\in R_r}\max\{\sum_{k\in C} d_{jk}, m_j\},\ \sum_{j\in R_r} Q_j^r\} \\
&\qquad y_c + y_r \le b_c\times N_c + n_b\times b_r\times N_r \\
&\qquad y_r \le n_b\times b_r\times N_r \\
&\qquad \sum_{j\in R_c'} m_j' \le b_c\times N_c \\
&\qquad y_c \ge \sum_{j\in R_c} m_j' \\
&\qquad y_r \ge \sum_{j\in R_r} m_j' \\
&\qquad b_c + b_r = 4 \\
&\qquad y_c, y_r, b_c, b_r:\ \text{nonnegative integers}.
\end{aligned}
$$

여기서 M은 다른 계수에 비해 상대적으로 아주 큰 수를 나타내고, y_r은 RF SaT의 집합에 할당되는 타임슬롯의 수를 나타내고, y_c는 CS SaT의 집합에 할당되는 타임슬롯의 수를 나타낸다. 그런데 TSDP에서 b_c와 b_r이 $\overline{b_c}$와 $\overline{b_r}$로 주어졌을 때 (단, $\overline{b_c}+\overline{b_r}=4$, $\sum_{j\in R_c'} m_j' \le \overline{b_c}\times N_c$),

$$
\overline{y_r} = \min\{\sum_{j\in R_r}\max\{\sum_{k\in C} d_{jk}, m_j\},\ \sum_{j\in R_r} Q_j^r,
$$

$$
n_b\times\overline{b_r}\times N_r - \max\{\sum_{j\in R_c} m_j' - \overline{b_c}\times N_c, 0\}\} \ge \sum_{j\in R_r} m_j',
$$

$$
\overline{y_c} = \min\{\sum_{j\in R_c}\max\{\sum_{k\in C} d_{jk}, m_j\},\ \sum_{j\in R_c} Q_j^c,
$$

$$
\overline{b_c}\times N_c + n_b\times\overline{b_r}\times N_r - \overline{y_r}\} \ \ge\ \sum_{j\in R_c} m_j'
$$

를 만족하면 $\overline{y_c}$와 $\overline{y_r}$는 $\overline{b_c}$와 $\overline{b_r}$하에서 TSDP의 최적해가 된다. 따라서 TSDP는 쉽게 해결될 수 있는 문제이다. TSDP의 최적해 y_c^*, y_r^*, b_c^*, b_r^* 하에서 RRTAP는 다음의 RRTAP^{R}로 바뀐다. RRTAP^{R}은 프로시저 PRRTAP에 의해 쉽게 해결된다.

(RRTAP^R)

$$
Min\ \sum_{j\in R}\sum_{k\in C} w_{jk}\cdot\max\{d_{jk}-y_{jk}, 0\} \qquad (40)
$$

s.t. (18)~(19), (22)~(24)

$$
\sum_{j\in R_c}\sum_{k\in C} y_{jk} \le y_c^* \qquad (41)
$$

$$
\sum_{j\in R_r}\sum_{k\in C} y_{jk} \le y_r^* \qquad (42)
$$

$$
\sum_{j\in R_c'}\sum_{k\in C} y_{jk} \le b_c^*\times N_c \qquad (43)
$$

y_{jk} : nonnegative integers , $\forall j\in R, k\in C$. (44)

프로시저 PRRTAP의 Step 1에서는 RF SaT에 타임슬롯을 할당한다. Step 1.1에서는 최소 요구량을 할당한다. 먼저 SaT j의 클래스 k에 대한 최소 요구량 m_{jk}를 할당한다(제약식 (22)). 이를 통해 SaT j에 할당된 양($\sum_k m_{jk}$)이 SaT j의 최소 요구량 m_j보다 작으면 w_{jk}가 큰 SaT j의 클래스 k 부터 순서대로 d_{jk}범위 내에서 추가적으로 할당하고, 추가 할당 후에도 m_j를 만족시키지 못하면 w_{jk}가 가장 큰 클래스 k에 부족한 양을 할당하여 최소 요구량 m_j를 충족시킨다(제약식 (23)~(24)). Step 1.2에서는 최소요구량을 충족시키고 남은 타임슬롯을 w_{jk}가 큰 큰 SaT j의 클래스 k부터 순서대로 d_{jk}와 Q_j^r범위 내에서 할당한다(제약식 (19)). Step 2에서는 CS SaT에 타임슬롯을 할당한다. Step 2.1에서는 최소 요구량을 할당하고(제약식 (22)~(24)), Step 2.2에서는 최소요구량을 충족시키고 남은 타임슬롯을 w_{jk}가 큰 SaT j의 클래스 k 부터 순서대로 d_{jk}, Q_j^c, $b_c^*\times N_c$ 범위 내에서 할당한다(제약식 (18)과 (43)).

Lemma 3. 프로시저 PRRTAP는 RRTAP^{R}의 최적해를 구한다(결국, RRTAP의 최적해를 구하게 된다).

증명. RRTAP^{R}은 두 개의 문제로 완전히 분리된다. 하나는 타임슬롯 y_r^*개를 타임슬롯 할당량에 대한 최소값 및 최대값 제약(제약식 (19), 그리고 제약식 (22)~(24) 중 RF SaT 관련 제약식) 하에서 RF SaT에 할당하는 문제이고, 다른 하나는 타임슬롯 y_c^*개를 타임슬롯 할당량에 대한 최소값 및 최대값 제약(제약식 (18)과 (43), 그리고 제약식 (22)~(24) 중 CS SaT 관련 제약식) 하에서 CS SaT에 할당하는 문제이다.

먼저, 타임슬롯 y_c^*개를 RF SaT에 할당하는 문제의 경우, 타임슬롯 할당량에 대한 최소값 및 최대값 관련 제약식 (19), (22)~(24)를 고려하지 않는다면, y_r^*개 범위 내에서 w_{jk}가 큰 SaT j의 클래스 k부터 순서대로 d_{jk}개 까지 할당하면 최적해를 구할 수 있다. 그런데 제약식 (19)로 인해 SaT j에는 Q_j^r개 까지만 할당할 수 있다. 한편 제약식 (22)로 인해 SaT j의 클래스 k에는 최소한 m_{jk}개가 할당되어야 하고, 제약식 (23)과 (24)로 인해 SaT j에는 최소한 m_j개가 할당되어야 한다. 결국 타임슬롯 y_r^*개를 RF SaT에 할당하는 문제는, 최소값 m_{jk}와 m_j를 만족시키고 최대값 Q_j^r범위 내에서, w_{jk}가 큰 SaT j의 클래스 k부터 순서대로 가능하면 많이 (d_{jk}개 까지) 할당하면 최적해를 구하게 된다. 프로시저 PRRTAP의 Step 1은 이러한 원칙에 따라 타임슬롯 y_r^*개를 RF SaT에 할당하므로 최적해를 구하게 된다(앞에서 제시한 프로시저 PRRTAP에 대한 설명 참조).

Procedure PRRTAP
Step 1. (RF SaT에 대한 할당)
Step 1.1 (최소 요구량 할당)
$r=y_r^*$.
FOR($\forall j\in R_r, \forall k\in C$){$y_{jk}=m_{jk}$. $r=r-y_{jk}$. $y_j=y_j+y_{jk}$.}
FOR($\forall j\in R_r, \forall k\in C$) /* 단, w_{jk}가 큰 것부터 */[1]
  IF($m_j>y_j$){$\overline{m}=\min(m_j-y_j, d_{jk}-y_{jk})$. $y_{jk}=y_{jk}+\overline{m}$. $y_j=y_j+\overline{m}$. $r=r-\overline{m}$.}
FOR($\forall j\in R_r$)
  IF($m_j>y_j$){$y_{jk^*}=y_{jk^*}+(m_j-y_j), y_j=y_j+(m_j-y_j), r=r-(m_j-y_j)$.(단, $k^*=\arg\max_k\{w_{jk}\}$)}
Step 1.2 (잔여 타임슬롯 할당)
FOR($\forall j\in R_r, \forall k\in C$) /* 단, w_{jk}가 큰 것부터 */[1]
IF($d_{jk}-y_{jk}>0, Q_j^r-y_j>0$){
  $\overline{m}=\min(r, d_{jk}-y_{jk}, Q_j^r-y_j)$. $y_{jk}=y_{jk}+\overline{m}$. $y_j=y_j+\overline{m}$. $r=r-\overline{m}$.
  IF($r\le 0$){*go to Step* 2.}
}
Step 2. (CS SaT에 대한 할당)
Step 2.1 (최소 요구량 할당)
$r=y_c^*, r'=0$.
FOR($\forall j\in R_c, \forall k\in C$){ $y_{jk}=m_{jk}$. $r=r-y_{jk}$. $y_j=y_j+y_{jk}$. *IF*($j\in R_c'$){$r'=r'+y_{jk}$.}}
FOR($\forall j\in R_c, \forall k\in C$) /* 단, w_{jk}가 큰 것부터 */[2]
  IF($m_j>y_j$){$\overline{m}=\min(m_j-y_j, d_{jk}-y_{jk})$. $y_{jk}=y_{jk}+\overline{m}$. $y_j=y_j+\overline{m}$. $r=r-\overline{m}$.
    IF($j\in R_c'$){$r'=r'+\overline{m}$.}}
FOR($\forall j\in R_c$)
  IF($m_j>y_j$){$y_{jk^*}=y_{jk^*}+(m_j-y_j), y_j=y_j+(m_j-y_j), r=r-(m_j-y_j)$.(단, $k^*=\text{argmax}_k\{w_{jk}\}$)
    IF($j\in R_c'$){$r'=r'+(m_j-y_j)$.}}
Step 2.2 (잔여 타임슬롯 할당)
FOR($\forall j\in R_c, \forall k\in C$) /* 단, w_{jk}가 큰 것부터 */[2]
IF($d_{jk}-y_{jk}>0, Q_j^c-y_j>0$){
  IF($j\in R_c', b_c^*\times N_c>r'$){
    $\overline{m}=\min(r, d_{jk}-y_{jk}, Q_j^c-y_j, b_c^*\times N_c-r')$.
    $y_{jk}=y_{jk}+\overline{m}$. $y_j=y_j+\overline{m}$. $r=r-\overline{m}$. $r'=r'+\overline{m}$.
  }
  Else IF($j\notin R_c'$){
    $\overline{m}=\min(r, d_{jk}-y_{jk}, Q_j^c-y_j)$. $y_{jk}=y_{jk}+\overline{m}$. $y_j=y_j+\overline{m}$. $r=r-\overline{m}$.
  }
  IF($r\le 0$){*terminate*.}
}

주1) 정렬 작업을 통해 w_{jk}값이 작은 것부터 (j,k), $k=(k_1,k_2,\cdots)$, R_c또는 R_r에의 포함 여부 등을 저장하는 구조체 배열을 생성하고, 생성된 구조체 배열의 역순으로 적용하면 된다. 5절에서와 같이 $w_{jk}=M+(k_1-1)|C_2|+k_2$, $j\in R_r$로 가정하면(단, 여기서 M은 다른 수에 비해 상대적으로 매우 큰 수를 나타내고, 데이터 클래스 집합은 2개인 것으로 가정함), 정렬 작업이 필요 없다. 이와 같이 가정하더라도 충분히 의미 있는 결과를 얻을 수 있고, 본 논문에서도 이와 같은 가정 하에 실험하였다.
주2) 주1)과 동일한 방법을 사용하면 된다. 단, 5절에서 $w_{jk}=(k_1-1)|C_2|+k_2$, $j\in R_c$로 가정하고 있다.
주3) 프로시저 PRRTAP의 계산 복잡도(computational complexity)는 정렬 작업이 필요한 경우에는 $\max\{O(|R_r|\times|C|\log|R_r|\times|C|), O(|R_c|\times|C|\log|R_c|\times|C|), O(|R|\times|C|)\}$ 이고, 정렬 작업이 필요 없는 경우에는 $O(|R|\times|C|)$이다.

마찬가지로, 타임슬롯 y_c^*개를 CS SaT에 할당하는 문제의 경우, 타임슬롯 할당량에 대한 최소값 및 최대값 관련 제약식 (18), (22)~(24), (43)을 고려하지 않는다면 y_c^*개 범위 내에서 w_{jk}가 큰 SaT j의 클래스 k부터 순서대로 d_{jk}개 까지 할당하면 최적해를 구할 수 있다. 그런데 제약식 (18)로 인해 SaT j에는 Q_j^c개 까지만 할당할 수 있고, 제약식 (43)으로 인해 R_c'에 속한 SaT에는 $b_c^* \times N_c$개 까지만 할당할 수 있다. 한편 제약식 (22)로 인해 SaT j의 클래스 k에는 최소한 m_{jk}개가 할당되어야 하고, 제약식 (23)과 (24)로 인해 SaT j에는 최소한 m_j개가 할당되어야 한다. 결국 타임슬롯 y_c^*개를 CS SaT에 할당하는 문제는, 최소값 m_{jk}와 m_j를 만족시키고 최대값 Q_j^r과 $b_c^* \times N_c$범위 내에서, w_{jk}가 큰 SaT j의 클래스 k부터 순서대로 가능하면 많이 (d_{jk}개 까지) 할당하면 최적해를 구하게 된다. 프로시저 PRRTAP의 Step 2는 이러한 원칙에 따라 타임슬롯 y_c^*개를 CS SaT에 할당하므로 최적해를 구하게 된다(앞에서 제시한 프로시저 PRRTAP에 대한 설명 참조). ■

4.2 RTAP 해결 체계

4.1절에서 프로시저 PRRTAP를 이용해 구한 해 y_{jk}(RRTAP의 최적해)가 아래의 Lemma 4를 만족하는 경우 RTAP의 최적해가 구해진 것이다. Lemma 4에 의해 최적해로 판명되지 않은 경우에는 RRTAP의 최적해로부터 RTAP의 실행가능해(또는 최적해)를 프로시저 PRTAP에 의해 구한다. 프로시저 PRTAP에서 R_c^c는 CS 타임슬롯이 할당되는 CS SaT들의 집합을 나타내고, $R_c - R_c^c$는 RF 타임슬롯이 할당되는 CS SaT들의 집합을 나타낸다.

프로시저 PRTAP의 Step 1에서는 R_c^c를 생성한다. 여기서 y_j는 프로시저 PRRTAP에서 구한 SaT j에 할당된 타임슬롯 수를 나타낸다($y_j = \sum_{k \in C} y_{jk}$). Step 1에서 $r = 0$이면 프로시저 RRTAP에 의해 구한 각 CS SaT에 대한 할당량 y_j를 이용하여 $\sum_{j \in R_c^c} y_j = \overline{N_c}$인 R_c^c를 구할 수 있다는 것을 나타내고, 따라서 프로시저 PRRTAP에 의해 구한 해(RRTAP의 최적해)가 RTAP의 최적해가 된다(Lemma 5 참조). Step 2에서는 y_{jk}값 조정을 통해 RTAP의 실행가능해를 구한다. 물론 우연히 최적해가 얻어질 수도 있다(Remark 2 참조).

Lemma 4. (최적조건 A & B)

$\sum_{j \in R_r} \sum_{k \in C} y_{jk} = n_b \times b_r^* \times N_r$(최적조건 A) 또는 $\sum_{j \in R_r} \sum_{k \in C} y_{jk} \le b_c^* \times N_c$(최적조건 B) 이면, RRTAP의 최적해는 RTAP의 최적해이다.

증명. $\sum_{j \in R_r} \sum_{k \in C} y_{jk} = n_b \times b_r^* \times N_r$ 또는 $\sum_{j \in R_c} \sum_{k \in a} y_{jk} \le b_c^* \times N_c$이면, $z_j^* = 1$, $j \in R_r$이고, $z_j^* = 0$, $j \in R_c$이다. 즉, 집합 R_r에 속한 SaT에는 RF 타임슬롯만이 할당되고, 집합 R_c에는 CS 타임슬롯만이 할당된다. 따라서 RRTAP의 최적해는 RTAP의 실행가능해이고, Remark 1에 의해 RTAP의 최적해가 된다. ■

Lemma 5. (최적조건 C) 프로시저 PRRTAP에 의해 구한 각 CS SaT에 대한 할당량 $y_j = \sum_{k \in C} y_{jk}$를 이용하여 $\sum_{j \in R_c^c} y_j = b_c^* \times N_c$인 R_c^c가 존재하면 프로시저 PRRTAP에 의해 구한 해(RRTAP의 최적해)가 RTAP의 최적해이다.

증명. $\sum_{j \in R_c^c} y_j = b_c^* \times N_c$인 R_c^c가 존재하면, $z_j^* = 1$, $j \in R - R_c^c$이고, $z_j^* = 0$, $j \in R_c^c$이다. 즉, 집합 $R - R_c^c$에 속한 SaT에는 RF 타임슬롯만이 할당되고, 집합 R_c^c에는 CS 타임슬롯만이 할당된다. 따라서 프로시저 PRRTAP에 의해 구한 해(RRTAP의 최적해)는 RTAP의 실행가능해이고, Remark 1에 의해 RTAP의 최적해가 된다. ■

Procedure PRTAP

Step 1. (R_c^c 생성)
$R_c^c = R_c'$. $R^t = R_c'$. $r = y_c^* - \sum_{j \in R_c} y_j$.
WHILE($R_c - R^t \neq \Phi$){
  $j^* = \text{argmax}_j\{y_j, j \in R_c - R^t\}$.
  IF($\sum_{j \in R_c^c + \{j^*\}} y_j \leq y_c^*$){$R_c^c = R_c^c + \{j^*\}$. $r = r - y_{j^*}$.}
  $R^t = R^t - \{j^*\}$.
}
IF($r = 0$){*terminate with current optimal solution.*}

Step 2. (실행가능해 발견)
$j^* = \text{argmin}_j\{y_j - r, j \in R_c - R_c^c\}$.
IF($\sum_{j \in R_c^c} m_j' + m_{j^*}' \leq b_c^* \times N_c$){$R_c^c = R_c^c + \{j^*\}$, $r = y_{j^*} - r$, *go* to *Step* 2.1.}
ELSE IF ($\sum_{j \in R_c - R_c^c} m_j' \leq b_r^* \times N_r - y_r^*$){*go* to *Step* 2.2.}
ELSE{*go* to *Step* 2.3.}

Step 2.1.
$p = 0$.
FOR($\forall j \in R_c^c, k \in C$) /* 단, w_{jk}가 작은 것부터 */[1]
WHILE($y_{jk} > m_{jk}, y_j > m_j'$){
  $y_{jk} = y_{jk} - 1$. $y_j = y_j - 1$. $p = p + 1$.
  IF($p \geq r$){*exit.*}
}
$p = 0$.
FOR($\forall j \in R_c - R_c^c, k \in C$) /* 단, w_{jk}가 큰 것부터 */[2]
WHILE($y_j < Q_j^c, y_{jk} < d_{jk}$){
  $y_{jk} = y_{jk} + 1$. $y_j = y_j + 1$. $p = p + 1$.
  IF($p \geq r$){*exit.*}
}

Step 2.2.
Substitute $R_c - R_c^c$ and R_c^c for R_c^c and $R_c - R_c^c$, *respectively*, in *Step* 2.1.

Step 2.3.
IF($b_c^* \leq \overline{b} - 1$){
  $b_c^* = b_c^* + 1$, $b_r^* = b_r^* - 1$.
  IF(*a feasible solution is found* in *TSDP*){*go* to Pr*ocedure* PRRTAP.}
  ELSE{*go* to *Step* 2.3.}
}
ELSE{*terminate with no feasible solution.*}

주1) 프로시저 PRRTAP의 주2)에서 생성한 구조체 배열을 순서대로 적용한다. 단, 이 구조체 배열 속에 R_c^c에의 포함 여부를 표시하는 항이 추가로 필요하다(프로시저 PRRTAP를 수행할 때 구조체 배열 속에 미리 설정해둔다).
주2) 프로시저 PRRTAP의 주2)에서 생성한 구조체 배열을 역순으로 적용한다.
주3) 프로시저 PRTAP의 계산 복잡도(computational complexity)는 $\max\{O(|R_c - R_c'|\log|R_c - R_c'|), O(|R_c| \times |C|)\}$ 이다.

Remark 2. (최적조건 D) 프로시저 PRTAP에 의해 구한 실행가능해의 목적식 값이 프로시저 PRRTAP에 의해 구한 해의 목적식 값과 동일하면 프로시저 PRTAP에 의해 구한 해는 RTAP의 최적해가 된다.

4.3 FTAP 해결 체계

여기에서는 RTAP에서 할당하고 남은 타임슬롯을 할당하는 문제인 FTAP의 해결 체계를 제시한다. FTAP는 RTAP에서 RF 타임슬롯이 할당된 SaT($R - R_c^c$에 속한 SaT들)에 대해 남은 RF 타임슬롯을 할당하는 문제와 CS 타임슬롯이 할당된 SaT(R_c^c에 속한 SaT)에 대해 남은 CS 타임슬롯을 할당하는 문제로 나누어진다. 이들 각 문제는 프로시저 PFTAP에서와 같이 추가적인 타임슬롯 할당

이 가능한 SaT들 중에서 $\frac{1}{w_j} P_j(y_j^* + y_j^F)$값이 가장 작은 SaT j에 한 개의 타임슬롯을 추가로 할당하는 과정을 계속 반복하는 방법에 의해 해결되어진다. 여기서 $y_j^* = \sum_{k \in C} y_{jk}^*$이고, y_{jk}^*는 프로시저 PRTAP를 이용해 구한 값이다.

Procedure PFTAP

Step 1.(*RF* 타임슬롯 할당)
IF($n_b \times b_r^* \times N_r - \sum_{j \in R - R_c^c} y_j^* > 0$) {
  FOR($i = 1; i \le n_b \times b_r^* \times N_r - \sum_{j \in R - R_c^c} y_j^*$; i++) {
    $S_F = \{j | j \in R - R_c^c, y_j^* + y_j^F < Q_j^r$(or $Q_j^c)\}$.
    IF($S_F = \phi$){*exit*}.
    $j^* = \text{argmin}_{j \in S_F}\{\frac{1}{w_j} P_j(y_j^* + y_j^F)\}$.
    $y_{j^*}^F = y_{j^*}^F + 1$. $y_{j^*k^*}^F = y_{j^*k^*}^F + 1$ (단, $k^* = \text{argmax}_k\{w_{jk}\}$).
  }
}

Step 2.(*CS* 타임슬롯 할당)
IF($b_c^* \times N_c - \sum_{j \in R_c^c} y_j^* > 0$) {
  FOR($i = 1; i \le b_c^* \times N_c - \sum_{j \in R_c^c} y_j^*$; i++) {
    $S_F = \{j | j \in R_c^c, y_j^* + y_j^F < Q_j^c\}$.
    IF($S_F = \phi$){*exit*}.
    $j^* = \text{argmin}_{j \in S_F}\{\frac{1}{w_j} P(y_j^* + y_j^F)\}$.
    $y_{j^*}^F = y_{j^*}^F + 1$. $y_{j^*k^*}^F = y_{j^*k^*}^F + 1$ (단, $k^* = \text{argmax}_k\{w_{jk}\}$).
  }
}

주1) 프로시저 PFTAP의 계산 복잡도(computational complexity)는 $\max\{O(|R - R_c^c| \times (n_b \times b_r^* \times N_r - \sum_{j \in R - R_c^c} y_j^*)),$ $O(|R_c| \times (b_c^* \times N_c - \sum_{j \in R_c^c} y_j^*))\}$ 이다.

4.4 TAP 해결 체계

여기에서는 각 SaT의 각 클래스별 타임슬롯 할당량만큼 특정 타임슬롯을 지정한다. 이를 위해 RF 타임슬롯과 CS 타임슬롯을 시간과 주파수에 따라 번호를 매긴다. CS 타임슬롯과 RF 타임슬롯 각각에 대해서 가장 낮은 주파수와 가장 이른 시간대의 타임슬롯을 0번으로 가장 높은 주파수와 가장 늦은 시간대의 타임슬롯을 $b_c^* \times N_c - 1$과 $n_b \times b_r^* \times N_r - 1$로 번호를 매긴다. 타임슬롯 스케줄링은 프로시저 PTAP에 의해 순차적으로 이루어진다. 프로시저 PTAP에서 y_{jk}^*는 프로시저 PRTAP를 이용해 구한 값이고, y_{jk}^{F*}는 프로시저 PFTAP를 이용해 구한 값이다.

Procedure PTAP

Step 1. (RF 타임슬롯 스케줄링)

$r = 0$;
FOR($\forall j \in R_r + R_c - R_c^c, \forall k \in C$){
  FOR($i = r$; $i \le y_{jk}^* + y_{jk}^{F*}$; i++){$x_{ijk} = 1.$}
  $r = r + y_{jk*}$.
}

Step 2. (CS 타임슬롯 스케줄링)

$r = 0$;
FOR($\forall j \in R_c^c, \forall k \in C$){
  FOR($i = r$; $i \le y_{jk}^* + y_{jk}^{F*}$; i++){$x_{ijk} = 1.$}
  $r = r + y_{jk*}$.
}

주1) 프로시저 PTAP의 계산 복잡도(computational complexity)는 $O(|S|)$ 이다.

5. 실험 결과

여기에서는 실험 데이터를 이용하여 성능을 분석하고자 한다. 데이터 클래스는 지연 클래스(k_1)와 서비스 유형 클래스(k_2) 두 가지를 고려하였다. 성능 분석에 사용되는 주요 데이터는 다음과 같다. 먼저 지연 클래스 수와 서비스 유형 클래스 수는 각각 5라고 가정하였고, RF SaT의 수와 CS SaT의 수는 각각 150과 90으로 가정하였다. 슈퍼프레임 구조를 결정하는 패러미터들은 $\overline{b} = 4$, $n_c = 16$, $n_{ct} = 2000$, $n_r = 10$, $n_b = 8$, $n_{rt} = 250$등으로 가정하였다. 다음으로 각 SaT의 지연 및 서비스 유형 클래스에 대한 수요 $d_{j(k_1, k_2)}$는 다음과 같이 구하였다.

- $k_1 = 1$이면,

$\frac{1}{\text{주기}}$의 확률로 일양분포 U(하한, 상한)으로부터 $d_{j(k_1, k_2)}$ 생성, $\left(1 - \frac{1}{\text{주기}}\right)$의 확률로 $d_{j(k_1, k_2)} =$ 하한,

- $k_1 \ge 2$이면,

$$d_{j(k_1, k_2)} = \left[\frac{1}{k_1^2} \times d_{j(k_1 - 1, k_2)}\right].$$

〈표 1〉 서비스 유형 클래스별 하한, 상한, 주기

서비스 유형 클래스	하한	상한	주기(period)
1	20	100	200
2	20	60	50
3	20	300	15
4	20	450	12
5	20	800	6

또한, $\alpha_{j(k_1,k_2)}$은 다음과 같이 가정하였다. 즉, $k_2 > \frac{|C_2|}{2}+1$이면 $\alpha_{j(k_1,k_2)}=0.7$로 가정하고, $k_2 \le \frac{|C_2|}{2}+1$이면 $\alpha_{j(k_1,k_2)}=0.5$로 가정하였다. Q_j^r, $j \in R_r$와 $Q_j^c, j \in R_c$는 각각 N_r과 N_c로 가정하였고, $m_j^a = m_j^b = 100, j \in R$로 가정하였다. $w_{jk}=M+(k_1-1)|C_2|+k_2$, $j \in R_r$, $w_{jk}=(k_1-1)|C_2|+k_2$, $j \in R_c$으로 가정하였다. 여기서 M은 다른 수에 비해 상대적으로 매우 큰 수를 나타낸다. w_j, β_j, B_j^S, B_j^T는 모든 SaT j에 대해서 동일하다고 가정하였고, 현재 버퍼에서 대기하고 있는 데이터의 양과 현재 이용가능한 타임슬롯의 수도 모든 SaT j에 대해서 동일하다고 가정하였다.

제시한 최적 타임슬롯 할당 체계를 이용해 구한 타임슬롯 할당 결과를 정리하면 <표 2>와 같다. 먼저 TSDP를 통해서 RF 타임슬롯의 수와 CS 타임슬롯의 수가 각각 60000개와 32000개로 결정되었다. 다음으로 프로시저 PRRTAP를 수행한 결과 최적조건 B를 만족하는 것으로 나타났고, 따라서 Gap(=(목적식 값-하한)/하한)은 RF SaT와 CS SaT 모두 없다. 최적조건 B를 만족했다는 것은 CS SaT에 할당된 타임슬롯 모두를 CS 타임슬롯만으로 충족시킬 수 있다는 것을 의미한다. 마지막으로 요청한 수요를 얼마나 충족시키는가를 나타내는 충족률은 전체 SaT의 평균 충족률, RF SaT의 평균 충족률, CS SaT의 평균 충족률 모두 1.0인 것으로 나타났고, 요청한 수요에 대한 실제 할당된 타임슬롯 수의 비율을 나타내는 할당률은 전체 SaT의 평균 할당률, RF SaT의 평균 할당률, CS SaT의 평균 할당률이 각각 1.22, 1.24, 1.77인 것으로 나타났다. 할당률이 1보다 큰 것은 각 단말의 타임슬롯 수요의 합이 이용 가능한 타임슬롯 수보다 작아서, 수요를 충족시키고 남은 타임슬롯을 각 단말에 추가로 할당했기 때문이다.

〈표 2〉 타임슬롯 할당 결과

구 분	총 타임슬롯 수	Gap	충 족 률	할 당 률	최적조건
RF	60000($b_r^*=3$)	0.0 !	1.0 +	1.21816 *	B !!
CS	32000($b_c^*=1$)	0.0 !	1.0 ++	1.24311 **	
전 체			1.0 +++	1.17658 ***	

! (목적식 값 -하한)/하한

!! 만족하는 최적조건

+ $\sum_{j\in R}\sum_{k\in C}\min\{\frac{y_{jk}}{d_{jk}},1\}/(|R|\times|C|)$

* $\sum_{j\in R}\sum_{k\in C}\frac{y_{jk}}{d_{jk}}/(|R|\times|C|)$

++ $\sum_{j\in R_r}\sum_{k\in C}\min\{\frac{y_{jk}}{d_{jk}},1\}/(|R_r|\times|C)$

** $\sum_{j\in R_r}\sum_{k\in C}\frac{y_{jk}}{d_{jk}}/(|R_r|\times|C|)$

+++ $\sum_{j\in R_c}\sum_{k\in C}\min\{\frac{y_{jk}}{d_{jk}},1\}/(|R_c|\times|C|)$

*** $\sum_{j\in R_c}\sum_{k\in C}\frac{y_{jk}}{d_{jk}}/(|R_c|\times|C|)$

<표 3>은 서비스 유형 클래스별 주기가 감소함에 따라(발생확률이 증가함에 따라, 수요가 증가함에 따라) 타임슬롯 할당 결과에 어떤 변화가 있는지를 나타낸다. 표에서 주기의 기준값은 <표 1>에 제시된 값을 의미한다. 표에서 알 수 있듯이 서비스 유형 클래스별 주기가 감소함에 따라 예상대로 충족률은 감소한다. 한편, 서비스 유형 클래스별 주기에 관계없이(수요의 많고 적음에 관계없이) 제시한 최적 타임슬롯 할당 체계는 최적해를 구하거나 Gap이 아주 작은 실행가능해를 구한다는 것을 알 수 있다.

<표 4>는 최소 타임슬롯 할당수 m_j의 변화에 따른 타임슬롯 할당 결과에 어떤 변화가 있는 지를 나타낸다(단, 주기 = 기준값 * 0.5). 최소 타임슬롯 할당수 변화에 관계없이 제시한 최적 타임슬롯 할당 체계는 최적해를 구하거나 Gap이 아주 작은 실행가능해를 구한다는 것을 알 수 있다.

마지막으로 제시한 최적 타임슬롯 할당 체계의 계산 시간을 살펴본 결과, Pentium IV PC 2.0GHz를 이용하여 평균적으로 5ms 이내에 해를 구하는 것을 확인할 수 있었다.

〈표 3〉 서비스 유형 클래스별 주기가 변하는 경우의 타임슬롯 할당 결과

주 기	b_r^*	b_c^*	최적 조건	Gap		충 족 률		
				RF	CS	전 체	RF	CS
기준값+*0.7	3	1	B++	0.0	0.0	1.0	1.0	1.0
기준값*0.6	3	1	-	0.0	0.000181	0.99918	1.0	0.99780
기준값*0.5	3	1	A	0.0	0.0	0.95061	0.99632	0.87443
기준값*0.4	3	1	A	0.0	0.0	0.79702	0.79763	0.79600

+ <표 1>에 주어진 주기
++ 만족하는 최적조건

〈표 4〉 최소 타임슬롯 할당수가 변하는 경우의 타임슬롯 할당 결과

m_j	b_r^*	b_c^*	최적 조건	Gap		충 족 률		
				RF	CS	전 체	RF	CS
100	3	1	A	0.0	0.0	0.95061	0.99632	0.87443
150	3	1	A	0.0	0.0	0.95304	0.98126	0.90600
200	3	1	A	0.0	0.0	0.92004	0.93458	0.89580

6. 결 론

본 논문에서는 먼저 리턴 링크 모형, 슈퍼프레임 구조, QoS 패러미터 등을 분석하고, CS SaT와 RF SaT가 존재하고 복수의 데이터 클래스가 존재하는 광대역 위성 시스템을 대상으로 MF-TDMA 체계하에서 QoS 기반 최적 타임슬롯 할당 문제를 수리적으로 정형화하였다.

다음으로 최적 타임슬롯 할당 체계를 제시하였다. 먼저 CS 타임슬롯의 집합과 RF 타임슬롯의 집합 그리고 CS SaT의 집합 및 RF SaT의 집합에 할당되는 타임슬롯 수를 결정한다. 다음으로 각 SaT의 각 클래스에 할당되는 타임슬롯의 수를 결정하고, 마지막으로 각 SaT의 각 클래스에 할당된 타임슬롯들의 스케줄을 결정한다.

끝으로 불균등 수요 데이터를 활용하여 제시한 최적 타임슬롯 할당 체계의 성능을 분석하였다. 성능 분석 결과 제시한 최적 타임슬롯 할당 체계는 빠른 시간 내에 최적해 또는 최적해와 아주 가까운 해를 안정적으로 구한다는 사실을 확인할 수 있었다.

참 고 문 헌

[1] ETSI, Digital video broadcasting (DVB) : Interaction channel for satellite distribution systems, ETSI EN 301 790 (v.1.2.2), 2000.

[2] Farserotu, J. and R.A. Prasad, "A survey of future broadband multimedia satellite systems, issues and trends," IEEE Commun. Mag., (2000), pp.128-133.

[3] Jiang, Z. and V.C.M. Leung, A predictive demand assignment multiple access protocol for Internet access over broadband satellite networks, Int. J. Satell. Commun. Network, Vol. 21(2003), pp.451-467.

[4] Lee, H.-J., S.K. Shin, D.-G. Oh and J.-M. Kim, "Development of satellite broadband interactive multimedia access technology," Proceedings of APSCC, 2002.

[5] Lee, K.-D. and K.-N. Chang, "A real-time algorithm for timeslot assignment in multirate return channels of interactive satellite multimedia," IEEE J. Select. Areas Commun., Vol. 22, No.3(2004), pp.518-528.

[6] Lee, K.-D., K.-N. Chang and H.-J. Lee, "Timeslot assignment for an interactive satellite multimedia network with multiclass RCSTs and services," Proceedings of VTC, 2003.

[7] Le-Ngoc, T., V. Leung, P. Takats and P. Garland, "Interactive multimedia satellite access communications," IEEE Commun. Mag., Vol.41, No.7(2003), pp.78-85.

[8] Le-Ngoc, T. and S.V. Krishnamurthy, "Performance of combined free/demand assignment multiple access scheme in satellite communications," International J. Satellite Commun., Vol.14(1996), pp.11-21.

[9] Mobasseri, v. and V.C.M. Leung, "Bandwidth Assignment for VBR Traffic in Broadband Satellite Networks," Proceedings of IEEE CCECE, 2000.

[10] Neale, J., R. Green and A. Landovskis, "Interactive channel for multimedia satellite networks," IEEE Commun. Mag., (2001), pp. 192-198.