

# 대화형 연산 후 수렴을 이용한 저장된 비디오의 효율적인 전송 스케줄 작성 방안

이 재 홍<sup>†</sup> · 김 승 환<sup>††</sup>

## 요 약

주문형 비디오 서비스에서는 이동 연산이나 고속 재생 연산과 같은 클라이언트에서의 대화형 연산이 종료된 후 선택된 새로운 프레임 위치에서 서버는 신속하게 정상 재생 모드로 복귀하여야 한다. 본 논문에서는 대화형 연산이 종료된 후 서버가 새로운 프레임 위치에서 재생을 재개하기 위한 전송 스케줄을 작성하기 위한 전송 스케줄을 효율적으로 작성하는 방안을 제안한다. 제안된 방안은 동일 비디오 스트림 내에서 서로 다른 재생 시작 프레임 위치를 가지는 전송 스케줄들이 특정 프레임 위치로부터 동일한 스케줄을 가진다는 수렴 특성을 기반으로 한다. 제안된 방안에서는 대화형 연산 후 선택된 프레임 위치에 대응하는 수렴 위치를 찾고 재생 시작 프레임 위치로부터 수렴 위치까지는 대역폭 평활화를 이용하여 새로운 전송 스케줄을 작성하고, 수렴 위치로부터 마지막 프레임 위치까지는 기존에 작성된 전송 스케줄을 이용한다. MPEG-1으로 부호화된 비디오 데이터를 대상으로 비디오 스트림 내 임의의 프레임 위치에 대응하는 수렴 위치가 존재함을 보였고, 선택된 프레임 위치로부터 잔여 프레임들을 모두 고려하는 경우와 제안된 방안 사이의 계산량을 비교하였다. 제안된 방안은 수렴을 이용함으로써 전송 스케줄을 작성하기 위하여 적용되는 대역폭 평활화로 인한 계산량을 대폭 줄일 수 있었으며, 대화형 연산이 가능한 비디오 스트림 내 모든 GoP의 I 프레임 위치마다 전송 스케줄을 오프라인으로 미리 작성하여 유지함으로써 대화형 연산이 종료된 후 정상 재생으로 복귀하는데 소요되는 재생 시작 지연을 대폭 줄일 수 있을 것으로 기대된다.

# An Efficient Scheme to write a Transmission Schedule using Convergence after Interactive Operations in a Stored Video

Jae-Hong Lee<sup>†</sup> · Seung-Hwan Kim<sup>††</sup>

## ABSTRACT

In a Video-on-Demand (VOD) service, a server has to return to the normal playback quickly at a certain new frame position after interactive operations such as jump or fast playback. In this paper, we propose an efficient scheme to write a transmission schedule for a playback restart of a video stream at a new frame position after interactive operations. The proposed scheme is based on convergence characteristics, that is, transmission schedules with different playback startup frame position in a video stream meet each other at some frame position. The scheme applies a bandwidth smoothing from a new frame position to a convergence position without considering all remaining frames of a video stream. And then the scheme transmits video data according to the new schedule from the new frame position to the convergence position, and then transmits the remaining video data according to the reference schedule from the convergence position to the last frame position. In this paper, we showed that there existed the convergence position corresponding to any frame position in a video stream through many experiments based on MPEG-1 bit trace data. With the convergence we reduced the computational overhead of a bandwidth smoothing, which was applied to find a new transmission schedule after interactive operations. Also, storage overhead is greatly reduced by storing pre-calculated schedule information up to the convergence position for each I frame position of a video stream with video data off-line. By saving information on a transmission schedule off-line along with the video data and searching the schedule corresponding to the specified restarting frame position, we expect the possibility of normal playback of a video stream with small tolerable playback startup delay.

※ 본 연구는 과학기술부 한국과학재단 지경 칭주대학교 정보통신 연구센터의 지원에 의한 것이다.  
† 경 회 원 : 진남도립 담양대학 전신 · 경보통신공학부 교수

†† 경 회 원 : 충청대학 컴퓨터학부 교수  
논문접수 : 1999년 10월 25일, 심사완료 : 2000년 6월 12일

## 1. 서 론

비디오 데이터는 음성 데이터와 비교하여 그 데이터 양이 방대하므로 망에 직접 데이터를 전송하기에는 적합하지 못하다 따라서, 이러한 방대한 양의 비디오 데이터는 효율적인 저장과 전송을 위해 일반적으로 MPEG 표준안으로 압축 부호화 된다[1, 2]. 한편, 비디오 데이터는 재생 중에 일정한 품질을 유지하기 위하여 가변 비트율(Variable Bit Rate) 모드의 MPEG 기법으로 부호화 된다. 그러나, VBR MPEG 기법으로 부호화 된 비디오 데이터는 공간 부호화만을 하는 [프레임과 예측 부호를 사용하는 P와 B 프레임의 비트율의 차이로 인해 매우 폭주적인 트래픽 특성을 보인다. 따라서, 이를 망에 그대로 전송하는 경우에는 매우 높은 망 대역폭이 필요하므로 전송 비용을 높이고 트래픽 제어를 어렵게 한다[2, 3].

실시간 비디오와는 달리 저장된 비디오를 대상으로 하는 주문형 비디오 (VOD : Video-On-Demand) 서비스에서는 서버가 클라이언트로 비디오 데이터를 전송하기 전에 전체 비디오 스트림의 통계적 특성을 알 수 있다. 그러므로 클라이언트에서 비디오 데이터를 재생하기 전에 서버는 충분한 데이터를 선인출(prefetch)하여 평활화함으로써 망에 대한 대역폭 요구를 줄일 수 있다. VOD 서비스에서 서버가 비디오 데이터를 클라이언트로 전송하기 위한 전송 스케줄을 작성하는 경우에, 망에 대한 대역폭 요구를 최소화하기 일반적으로 대역폭 평활화(bandwidth smoothing)를 이용한다[3-5]. 진정한 VOD 서비스를 위해서는 정상 재생뿐만 아니라 현재 재생 위치를 비디오 스트림 내 임의의 프레임 위치로 이동할 수 있는 전진 이동(jump forward), 후진 이동(jump backward)과 같은 이등 연산(jump operation)이나 영상을 보여 주면서 고속 전진(fast forward) 및 후진 재생(fast backward)과 같은 고속 연산 등의 대화형 연산을 지원해야 한다[6, 7].

사용자의 요청에 의해 비디오 스트림 내 임의의 프레임 위치로 재생 위치를 이동한 후에 정상 재생을 시작하는 경우에, 그 위치로부터 서버는 가능한 빨리 정상 재생을 위한 전송 스케줄로 복귀하여야 한다[3, 8]. 이 때, 서버가 새로운 프레임 위치로부터 정상 재생을 시작하는 전송 스케줄을 작성하지 않고 비디오 스트림의 첫 번째 프레임 위치로부터 재생을 시작하도록 한 전송 스케줄을 그대로 사용하는 경우에 재생 시작 지연이 매우 커지게 된다[8, 9]. 이것은 대역폭 평활화의

특성상 망에 대한 대역폭 요구를 최소화하기 위하여 재생 시작 전에 클라이언트의 버퍼가 허용하는 한 서버가 보다 많은 데이터를 선인출하도록 하기 때문에 기존 스케줄로 복귀하기 위하여 선택된 프레임 위치에서 일정 수준의 버퍼 점유율을 유지하여야 하는 데에서 발생하는 지연 때문이다. 따라서, 이 지연을 줄이기 위해서 서버가 기존 스케줄을 사용하지 않고 새로운 프레임 위치로부터 재생을 시작하도록 하는 새로운 전송 스케줄을 작성하는 것이 효율적이다. 그러나 전송 스케줄을 작성하기 위해서는 대역폭 평활화를 적용하여야 하는데, 대역폭 평활화의 계산상의 오버헤드로서 서버가 동적으로 전송 스케줄을 작성하는 것은 어렵다 [4]. 그러므로 서버는 대화형 연산 후 정상 재생이 가능한 모든 프레임 위치에 대하여 오프라인으로 전송 스케줄을 미리 작성하여 놓고 필요한 경우에 이를 이용하는 방안이 효율적일 것이다.

본 논문에서는 비디오 스트림 내 임의의 프레임 위치에서 중단된 비디오 데이터의 재생을 시작할 때, 새로운 프레임 위치에서 비디오 스트림 내 잔여 프레임 모두를 고려하지 않고도 효율적으로 전송 스케줄을 작성하는 방안에 대하여 기술한다. 제안된 방안에서는 전송 스케줄의 작성에 이용하는 대역폭 평활화의 계산상의 오버헤드와 전송 스케줄 정보량을 줄이기 위해 새로운 프레임 위치에 대응하는 수렴 위치까지는 새로운 전송 스케줄을 구하고, 수렴 위치로부터 비디오 스트림 내 잔여 프레임들에 대한 전송 스케줄은 기존 전송 스케줄 정보를 이용한다.

본 논문의 구성은 다음과 같다 2장에서는 대역폭 평활화 기반의 전송 스케줄을 정의하고, 3장에서는 수렴을 이용한 전송 스케줄 작성 방안에 대하여 기술한다. 4장에서는 MPEG-1으로 부호화된 비디오 스트림에 대하여 서로 다른 재생 시작 위치를 가지는 전송 스케줄들이 서로 수렴하는 것을 실험적으로 보였으며, 전송 스케줄 작성 시 시작 프레임 위치로부터 잔여 프레임들을 모두 고려하는 경우와 본 논문에서 제안된 수렴을 이용하는 경우에 대하여 실제적인 계산량을 비교하고, 5장에서 결론을 맺는다.

## 2. 전송 스케줄과 대역폭 평활화

저장된 비디오는 실시간 비디오와는 달리 전송 전에 각 프레임의 크기 정보를 알 수 있다. 따라서, 서버는 최소한의 대역폭으로 비디오 데이터를 전송하기 위해

클라이언트에서 비디오 데이터를 재생하기 전에 서버에서 충분한 데이터를 선인출한 후 평활화를 한다[3-5].

MPEG으로 부호화된 비디오 스트림은 GoP(Group of Pictures) 패턴에 따라 IPBBP.. 순서로 클라이언트로 전송된다[1, 2]. 전송 초기에 I와 P 프레임 전송하기 위해서는 초기 대역폭 요구가 매우 크다. 이러한 초기 대역폭 요구를 줄이기 위하여 서버는 재생 시작 시각 전에 비디오 데이터를 선인출하여 클라이언트의 버퍼로 전송한다.

대역폭 평활화는 주어진 클라이언트의 버퍼 크기와 초기 재생 시작 지연 내에서 버퍼 오버 플로우와 버퍼 언더 플로우 없이 최대 비트율, 대역폭 변경 횟수, 대역폭 변화 변위, 대역폭 증가 횟수 등의 평활화 조건들을 충족하면서 비디오 스트림을 전송하기 위한 최적의 대역폭 할당 스케줄을 구하는 데 사용된다[3-5]. 대역폭 평활화 방안에서는 초기 대역폭 요구를 줄이기 위하여 초기 재생 시작 지연(initial playback startup delay)  $d_s$ 를 도입하기도 한다[10]. 그러나, 이 방안은 망 대역폭 요구를 줄일 수는 있으나 클라이언트 측의 초기 재생 시작 지연을 크게 하여 재생 지연에 민감한 대화형 서비스에는 적용하기 어렵다

주문형 비디오 서비스에서 서버는 비디오 스트림을 클라이언트로 전송하기 위하여 전송 스케줄을 작성하여야 한다. 전송 스케줄은 특정 프레임들을 클라이언트로 전송하기 위하여 망에 요구할 대역폭 정보들을 포함하고 있다[5]. 서버는 전송 스케줄을 이용하여 망에 최적의 대역폭을 요구하면서 재생 시작 프레임 위치로부터 비디오 데이터를 클라이언트로 전송한다. 서버는 대역폭 평활화를 이용하여 동일한 대역폭이 할당되는 세그먼트의 시작 프레임 번호와 마지막 프레임 번호, 그리고 그 세그먼트에 할당될 대역폭을 구하여 전송 스케줄을 작성한다. 서버는 각 비디오 스트림에 대하여 대용량 저장 장치에 비디오 데이터와 함께 전송 스케줄에 관한 정보를 유지한다.

비디오 스트림  $V$ 에 대한 전송 스케줄 정보  $M$ 은 식 (1)과 같이 클라이언트의 버퍼 크기  $B_c$ , 초기 재생 시작 지연  $d_s$ , 각 세그먼트에 대한 대역폭 할당 스케줄  $TS$ 로 이루어진다. 식 (1)에서  $K$ 는 비디오 스트림  $V$ 의 세그먼트의 수를 나타낸다.

$$M = \{ B_c, d_s, TS \} \quad (1)$$

$$TS = \{ S_1, \dots, S_k, \dots, S_K \}$$

세그먼트  $S_k$ 에 대한 대역폭 할당 정보는 식 (2)로 표현된다.

$$S_k = \{ F_k, L_k, R_k \} \quad (2)$$

식 (2)에서  $F_k, L_k, R_k$ 는 각각 세그먼트  $S_k$ 의 시작 프레임 위치와 마지막 프레임 위치, 그리고  $S_k$ 에 할당된 망 대역폭을 나타낸다.

서버는 식 (1)과 식 (2)에 의해 표현된 전송 스케줄 정보를 이용하여, 세그먼트  $S_k$ 의 시작 프레임  $F_k$ 에서 마지막 프레임  $L_k$ 사이의 프레임들을 클라이언트로 전송하기 위하여 망에 대하여 대역폭  $R_k$ 를 요구한다. 서버는 세그먼트  $S_k$ 의 프레임들을 모두 전송한 후에 다음 세그먼트  $S_{k+1}$ 의 프레임들을 전송하기 전에 필요한 대역폭  $R_{k+1}$ 을 확보하기 위해 망과 자원 협상을 시도한다. 이를 위해 세그먼트  $S_k$ 에 대해 확보한 대역폭  $R_k$ 을 망에 반환하고 망에 새로운 대역폭  $R_{k+1}$ 을 요구한다. 서버가 요구한 대역폭을 망이 제공할 수 있는지에 대해 서버는 망과의 협상을 시도하며, 이를 위해 RCBR 서비스[11]를 사용할 수 있다.

### 3. 수렴을 이용한 전송 스케줄 작성

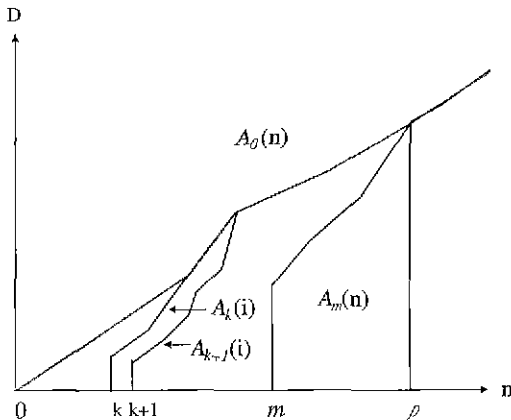
본 장에서는 수렴을 정의하고 이를 이용하여 전송 스케줄을 작성하는 방안을 기술한다. 본 장에서 사용하는 기호들을 <표 1>에 요약하였다.

<표 1> 기호 요약

기 호	의 미
$L$	GoP내 프레임 수
$N$	비디오 스트림의 총 프레임 수
$f$	전신 또는 후진 이동 후 선택된 프레임 위치
$f_s$	$f$ 가 속한 GoP의 I 프레임 위치
$p$	$f$ 에 대응하는 수렴 위치
$x_i$	$i$ 번째 프레임의 데이터 양
$D$	비디오 스트림의 시작 프레임으로부터 $n$ 번째 프레임까지의 누적 데이터 양
$C_{total}$	비디오 스트림 내 프레임들의 총 누적된 데이터 양
$TS_0$	재생 시작 프레임 위치가 0인 전송 스케줄
$TS_f$	재생 시작 프레임 위치가 $f$ 인 전송 스케줄
$TS_{f-p}$	프레임 위치 $f$ 로부터 대응하는 수렴 위치 $p$ 까지의 전송 스케줄
$K$	$TS_0$ 의 세그먼트 수
$P_f(n)$	프레임 위치 $f$ 로부터 $n$ 까지 클라이언트가 재생된 누적 데이터 양
$U_f(n)$	프레임 위치 $f$ 로부터 $n$ 까지 클라이언트가 수신할 수 있는 최대 누적 데이터 양
$A_f(n)$	서버가 $n$ 번째 프레임 위치까지 전송 스케줄 $TS_f$ 에 따라 클라이언트로 전송한 누적 데이터 양

(그림 1)은 동일 비디오 스트림에 대하여 재생 시작 프레임 위치가 다른 전송 스케줄들이 특정 프레임 위치  $p$ 로부터 서로 동일한 스케줄을 가지는 것을 보여 준다. (그림 1)에서  $A_0(n)$ ,  $A_k(n)$ ,  $A_{k+1}(n)$ ,  $A_m(n)$ 은 각각 재생 시작 프레임 위치  $0, k, k+1, m$ 부터 프레임 위치  $n$ 까지 전송 스케줄에 따라 클라이언트로 전송된 누적 데이터 양을 나타내며,  $A_k(n)$ ,  $A_{k+1}(n)$ ,  $A_m(n)$ 이  $A_0(n)$ 의 세그먼트의 시작 위치들에서  $A_0(n)$ 에 모두 수렴하고 있음을 보여 준다. (그림 1)에서 재생 시작 프레임 위치가 다른 전송 스케줄들이 특정 프레임 위치에서 서로 만나는 것은  $A(n)$ 의 기울기가 변경되는 프레임 위치, 즉 새로운 세그먼트의 시작 위치에서 대역폭 평활화의 평활화 조건을 만족하기 때문이다. 이런 특성을 이용하여 대화형 연산 후 새로운 프레임 위치  $f$ 로부터 새로운 전송 스케줄을 작성할 때, 서버는  $f$ 로부터 잔여 프레임들을 모두 고려한 전송 스케줄을 작성하지 않고도 전송 스케줄들이 동일하게 되는 프레임 위치를 찾고, 이 위치 이후의 전송 스케줄은 기존 전송 스케줄을 이용할 수 있게 된다.

본 장에서는 비디오 스트림 내 임의의 선택된 프레임 위치  $f$ 에 대한 전송 스케줄  $TS_f$ 를 작성할 때 비디오 스트림의 첫 번째 프레임 위치로부터 재생을 시작하는 전송 스케줄  $TS_0$ 를 참조함으로써  $f$  이후의 잔여 프레임들을 모두 고려하지 않고도 쉽게 전송 스케줄  $TS_f$ 를 구할 수 있음을 보인다



(그림 1) 재생 시작 프레임 위치가 다른 전송 스케줄들의 수렴

본 장에서는 새로운 전송 스케줄을 작성하기 위하여

기준 스케줄과 수렴(convergence)을 정의한다.

[정의 1] 전송 스케줄  $TS_i$

재생 시작 프레임 위치가  $i$ 인 전송 스케줄을  $TS_i$ 로 정의한다.

[정의 2] 기준 스케줄  $TS_0$

재생 시작 프레임 위치가 0인 전송 스케줄  $TS_0$ 를 기준 스케줄로 정의한다.

[정의 3] 수렴(Convergence)

프레임 위치  $p$ 에서 기준 스케줄  $TS_0$ 와 재생 시작 프레임 위치가  $f$ 인 전송 스케줄  $TS_f$ 가 식 (3)을 만족하는 경우, 두 스케줄은 프레임 위치  $p$ 에서 수렴한다고 한다. 식 (3)에서  $A_0(n)$ 은 비디오 서버가 비디오 스트림의 첫 번째 프레임부터  $n$ 번째 프레임까지 기준 스케줄  $TS_0$ 에 따라 클라이언트로 전송한 누적된 데이터 양이며,  $A_f(n)$ 은 서버가  $f$ 번째 프레임부터  $n$ 번째 프레임까지 전송 스케줄  $TS_f$ 에 따라 클라이언트로 전송한 누적된 데이터 양이다.

$$A_0(n) = A_f(n), \quad \forall n \in [p, N] \quad (3)$$

두 전송 스케줄이 수렴하는 위치  $p$ 는 식 (4)에 의해 구할 수 있다.

$$p = \min_{f \leq b_1 \leq N} \{b_1 \mid A_0(b_1) = A_f(b_1)\} \quad (4)$$

MPEG으로 부호화된 비디오 데이터는 부호화 방법에 따라 I, P, B 프레임들로 부호화된다[1]. I 프레임은 공간 부호화만으로 생성되며 GoP 패턴 내의 P나 B 프레임들은 I 프레임을 이용하여 예측 부호화된다. 그러므로 MPEG의 부호화 특성을 고려하여 비디오 스트림 내 임의의 프레임 위치로 이동한 후 재생을 다시 시작할 수 있는 프레임 위치  $f$ 는 B나 P 프레임이 아니라 I 프레임이 되어야 한다. 따라서, 프레임 위치  $f$ 는 식 (5)에 의해 인접 I 프레임 위치  $f_s$ 에 대한 전송 스케줄 정보로 대체되어 새로운 위치  $f_s$ 로부터 비디오 스트림의 재생이 시작된다. 식 (5)에서  $L$ 은 전송하고자 하는 비디오 스트림의 GoP 패턴의 길이를 나타낸다

$$f_s = \lfloor \frac{f}{L} \rfloor \times L \quad (5)$$

식 (5)에 의해 구해진 프레임 위치  $f_s$ 에서 비디오 서버

의 새로운 전송 스케줄을 구하기 위해  $P_f(n)$ 와  $U_f(n)$ 를 구한다.

클라이언트에서  $f$ 번째 프레임으로부터  $n$ 번째 프레임 까지 재생된 누적 데이터 양  $P_f(n)$ 는 식 (6)에 의해 구한다. 클라이언트가 비디오 스트림을 연속적으로 재생을 할 수 있도록 보장하기 위해, 서버는 항상 재생 전에 클라이언트에서 버퍼 인더 플로우되지 않도록 비디오 데이터를 전송해야 한다. 즉, 서버는 프레임 위치  $n$ 에서 식 (6)의  $P_f(n)$ 보다 많은 비디오 데이터를 클라이언트로 전송해야 한다.

$$P_f(0) = P(f_s), P_f(n) = \sum_{i=1}^n x_i \quad (6)$$

클라이언트의 버퍼 크기가  $B_c$ 일 때, 클라이언트는 서버로부터  $B_c$ 이상의 데이터를 수신할 수 없다. 따라서 클라이언트의 버퍼가 오버 플로우하지 않고  $f$ 번째 프레임으로부터  $n$ 번째 프레임까지 비디오 데이터를 수신할 수 있는 누적 데이터 양  $U_f(n)$ 은 식 (7)로 표현된다.

$$U_f(0) = P_f(0) + B_c$$

$$U_f(n) = \min\{ B_c + P_f(n-1), C_{total} \}, \quad 1 \leq n \leq N-f \quad (7)$$

서버의 대역폭 할당 스케줄은 식 (8)과 같이  $P_f(n)$ 과  $U_f(n)$  사이에 위치하면서 대역폭 평활화 조건들[3]을 만족하는 복수의 세그먼트들로 이루어진  $A_f(n)$ 을 구하는 것이다.

$$P_f(n) \leq A_f(n) \leq U_f(n) \quad (8)$$

(그림 2)는 비디오 스트림 내 임의의 프레임 위치  $f$ 로 이동한 후 식 (5)에 의해 인접 I 프레임 위치  $f_s$ 가 구해지고 기준 스케줄  $TS_0$ 를 이용하여 새로운 전송 스케줄  $TS_{f-p}$ 를 구하는 알고리즘을 나타낸다.

```

Procedure Schedule_for_Jump( $f, TS_0$ )
begin
    Find  $f_s$ , Constructs  $P_f(n)$  and  $U_f(n)$ 
     $TS_{f-p} \leftarrow \{ \}$ ,  $n \leftarrow f_s$ 
    while ( $n < N$ ) begin
        Find a segment  $S_j$  by applying Smoothing( $P_f, U_f, n$ )
         $n \leftarrow F_j + L_j$ 
         $m \leftarrow$  Find_Start_Segment( $f_s, TS_0$ )
        Add  $S_j$  to  $TS_{f-p}$ 
         $p \leftarrow$  Find_Convergence_Position( $f_s, m, TS_0$ )
        if  $p$  exists return ( $p, TS_{f-p}$ )
    
```

```

end while
end procedure
Procedure Find_Start_Segment( $f_s, TS_0$ )
begin
    // Find the index of the segment at the frame position after  $f_s$ 
     $i \leftarrow 1$ 
    while ( $i < K$ ) begin
        if ( $f_s$  equals  $F_i$ ) return  $F_i$ 
        else if ( $f_s > F_i$ ) and ( $f_s < (F_{i+1})$ ) return  $F_{i+1}$ 
        else  $i \leftarrow i+1$ 
    end while
end procedure
Procedure Find_Convergence_Position( $R_j, m, TS_0$ )
begin
    // Find the convergence position  $p$  corresponding to  $f_s$ 
     $i \leftarrow m$ 
    while ( $i < K$ ) begin
        if ( $R_j$  equals  $R'_i$ ) return  $p \leftarrow F_i$ 
        else  $i \leftarrow i+1$ 
    end while
end procedure
    
```

(그림 2) 전송 스케줄  $TS_{f-p}$ 을 구하는 알고리즘

(그림 2)에서  $P_f(n)$ 과  $U_f(n)$ 은 각각 프레임 위치  $f_s$ 에서 식 (6)과 식 (7)에 의해 구해지며,  $B_c$ 는 클라이언트의 버퍼 크기이며,  $K$ 는 기준 스케줄  $TS_0$ 의 세그먼트 수이다

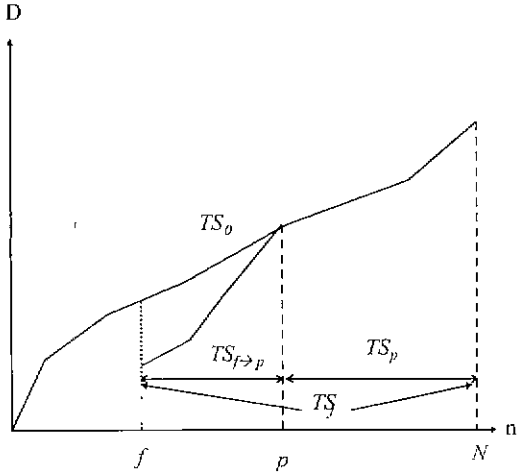
서버가 수렴을 이용하여 비디오 스트림 내 임의의 프레임 위치  $f$ 에서의 새로운 전송 스케줄  $TS_f$ 를 구하기 위한 (그림 2)의 절차를 요약하면 다음과 같다.

- 1) 식 (5)에 의해 인접 I 프레임 위치  $f_s$ 를 구한다.
- 2) 식 (6)과 식 (7)을 이용하여  $P_f(n)$ 과  $U_f(n)$ 을 작성한다.
- 3) 대역폭 평활화를 적용하면서 세그먼트  $S_j$ 를 구하고 이를  $TS_{f-p}$ 에 추가한다.
- 4) 세그먼트  $S_j$ 에 할당될 대역폭  $R_j$ 를 이용하여 식 (4)를 만족하는 프레임 위치가 있는지를 결정한다. (그림 2)에서  $R'_i$ 는  $TS_0$ 의  $i$ 번째 세그먼트에 할당될 대역폭을 나타낸다
- 5) 식 (4)를 만족하는 프레임 위치가 있으면 이 위치가  $f_s$ 에 대응하는 수렴 위치가 되며 다음 단계 6)으로 진행하고, 식 (4)를 만족하는 프레임 위치를 찾지 못하면 다음 세그먼트를 구하기 위하여 단계 3)으로 진행한다.
- 6) 수렴 위치  $p$ 이후의 전송 스케줄  $TS_p$ 는 기준 스케줄  $TS_0$ 의 정보를 이용한다.

(그림 3)에서 볼 수 있듯이 전송 스케줄  $TS_f$ 는 식

(9)와 같이 수렴 위치  $p$ 까지의 전송 스케줄  $TS_{f \rightarrow p}$ 와 수렴 위치로부터 비디오 스트림의 마지막 프레임 위치  $N$ 까지의 전송 스케줄  $TS_p$ 로 이루어진다

$$TS_f = TS_{f \rightarrow p} \cup TS_p \quad (9)$$



(그림 3)  $TS_f$  구하기

서버는 하나의 비디오 스트림에 대하여 기준 스케줄에 관한 정보  $M$ 과 함께 비디오 스트림 내 모든 GoP의 I 프레임 위치  $i$ 마다  $TS_i$ 를 구하여  $M_i$ 을 작성하여 비디오 스트림과 함께 대용량 저장 장치에 유지한다.  $M_i$ 은 식 (10)과 같이 비디오 스트림 내 GoP들의 I 프레임 위치에서의 전송 스케줄  $TS_i$ 들로 이루어진다. 이때  $TS_i$ 는 수렴 특성을 이용하여 식 (9)에 의하여 구한다. 식 (10)에서  $G$ 는 전송 스케줄을 작성할 비디오 스트림의 GoP 수이다.

$$M_i = \sum_{i=1}^G TS_i \quad (10)$$

클라이언트에서의 대화형 연산이 종료된 후, 서버가 수렴을 이용한 전송 스케줄 정보  $M_i$ 을 검색하여 비디오 데이터를 클라이언트로 전송하기 위한 절차는 다음과 같다.

- 1) 대화형 연산 후 프레임 위치  $f$ 가 새로운 재생 시작 위치로 주어지면, 식 (5)에 의해 인접 I 프레임 위치  $f_s$ 를 찾는다.
- 2) 오프라인으로 이미 작성되어 서버에 유지되고 있는  $M_i$  정보를 검색하여  $f_s$ 에 대응하는 전송 스케

줄  $TS_{f_s}$ 를 찾는다.

- 3) 서버는  $TS_{f_s}$ 에 따라 새로운 재생 시작 프레임 위치  $f_s$ 로부터 비디오 스트림의 마지막 프레임 위치까지의 비디오 데이터를 전송한다
- 4) 전송 중에 클라이언트가 대화형 연산을 요청하는 경우, 서버는 전송을 즉시 중단한다. 대화형 연산이 종료된 후 서버는 1)에서 3)의 과정을 반복한다

#### 4. 실험 및 평가

본 논문에서 제안된 방안의 성능 평가를 위해, MPEG-1으로 부호화된 "star wars"[12]와 "lamb", "term", "soccer"[13]의 프레임 크기 정보를 대상으로 실험을 수행하였다.

<표 2>에 실험에 사용된 데이터들의 부호화 특성들을 보였다.

<표 2> 실험에 사용된 데이터들의 부호화 특성

비디오 스트림	star wars	lamb	term	soccer
해상도 (픽셀)	508 x 408	384 x 288		
프레임 수	174,136 (약 119분 분량)	40,000 (약 27분 분량)		
GOP 크기 (L)	12	12		
재생 속도 ( $R_{rate}$ )	24	25		
평균 비트율 (Mbps)	0.374	0.183	1.989	0.628
최대 비트율 (Mbps)	4.446	3.356	0.273	4.757

<표 3>은 비디오 스트림 내 임의의 프레임 위치  $f$ 로부터 새로운 전송 스케줄  $TS_f$ 를 작성하는 경우에  $TS_f$ 의 처음 10개 세그먼트의 시작 프레임 위치  $f_s$ 를 보여 준다. 전송 스케줄 작성을 위한 대역폭 평활화 방안으로는 초기 재생 시작 지연  $d_s$ 가 0과 1초인 OSA (Optimal Smoothing Algorithm)[14]를 적용하였다. <표 3>에서 시작 프레임 위치  $f$ 가 0인 전송 스케줄이 기준 스케줄  $TS_0$ 이 되며,  $f_s$ 에 대한 전송 스케줄  $TS_{f_s}$ 가  $TS_0$ 에 수렴하는 위치를 회색으로 표시하였다. <표 3>에서 볼 수 있듯이 각 재생 시작 프레임 위치  $f_s$ 에 대응하는 수렴 위치가 존재하며, 이 위치는  $f$  이후의  $TS_0$ 의 인접 세그먼트의 시작 프레임 위치임을 알 수 있다. 그러나, <표 3> (b)에서  $f$ 가 1200과 2400인 경우에는 수렴 위치까지 길이가 짧은 여러 개의 세그먼트들이 존재하며, 수렴 위치는  $TS_0$ 의 최근접 세그먼트의 시작 프레임 위치인 3517이 아니라 5431임을 볼 수 있다.

대화형 연산 후 선택된 프레임 위치로부터 전송 스케줄을 작성하는 경우에, 비디오 스트림 내 잔여 프레임들을 모두 고려하여야 하는 기존 방안과 수렴 위치까지의 프레임들만 고려하는 제안된 방안 사이의 계산량을 비교 실험하였다. 실험에는 "starwars" 데이터를

대상으로 초기 재생 시작 지연  $d_s = 0$ 인 대역폭 평활화 방안 OSA[14]를 적용하였다. 실험에서는 클라이언트 버퍼 크기가 1~8 Mbyte일 때, 두 방안에 대하여 전송 스케줄을 구하기 위하여 대역폭 평활화가 적용되는 프레임 수를 구하였다

<표 3> 재생 시작 위치  $f_0$ 에 대한 수렴 위치

(a) "term",  $B_c = 2$  MBytes,  $d_s = 0$

$f$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$	$S_{10}$
0	1225	1237	1249	2476	4009	5908	6355	6361	34825	36949
250	253	373	1165	1225	1237	1249	2476	4009	5908	6355
500	502	1165	1225	1237	1249	2476	4009	5908	6355	6361
750	793	796	805	817	1165	1225	1237	1249	2476	4009
1000	1159	1165	1225	1237	1249	2476	4009	5908	6355	6361
1200	1201	1219	1225	1237	1249	2476	4009	5908	6355	6361
1225	1237	1249	2476	4009	5908	6355	6361	34825	36949	38233
1250	2476	4009	5908	6355	6361	34825	36949	38233	38776	
1300	1333	1336	2476	4009	5908	6355	6361	34825	36949	38233
1550	2233	2476	4009	5908	6355	6361	34825	36949	38233	38776
1750	1753	1801	1816	2197	2233	2476	4009	5908	6355	6361
2000	2152	2197	2233	2476	4009	5908	6355	6361	34825	36949
2250	2251	2257	2476	4009	5908	6355	6361	34825	36949	38233
2500	4009	5908	6355	6361	34825	36949	38233	38776		
2750	4009	5908	6355	6361	34825	36949	38233	38776		
4000	4009	5908	6355	6361	34825	36949	38233	38776		
5900	5905	5908	6355	6361	34825	36949	38233	38776		
6250	6253	6325	6355	6361	34825	36949	38233	38776		
6350	6355	6361	34825	36949	38233	38776				
6500	6505	34825	36949	38233	38776					

(b) "soccer",  $B_c = 2$  MBytes,  $d_s = 0$

$f$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$	$S_{10}$
0	1	3517	3523	3526	5431	16597	28477	34405	39985	39997
250	637	652	3517	3523	3526	5431	16597	28477	34405	39985
600	601	637	652	3517	3523	3526	5431	16597	28477	34405
840	841	3517	3523	3526	5431	16597	28477	34405	39985	39997
1020	1021	3523	3526	5431	16597	28477	34405	39985	39997	
1080	1081	3526	5431	16597	28477	34405	39985	39997		
1200	1201	1255	1264	1297	1309	1333	3526	5431	16597	28477
2400	2401	2551	5431	16597	28477	34405	39985	39997		
3600	3601	3679	3928	3937	3940	4297	4645	4714	4717	5431
4800	4801	5401	5404	5407	5431	16597	28477	34405	39985	39997
6000	6001	16597	28477	34405	39985	39997				
8400	8401	9877	9889	9901	9913	16597	28477	34405	39985	39997
9600	9601	9721	9727	9733	9865	9877	9901	9913	16597	28477
11760	11761	28477	34405	39985	39997					
12000	12001	12085	12148	12157	28477	34405	39985	39997		
18000	18001	18313	19852	21277	21304	21313	21409	34405	39985	39997
24000	24001	24016	24025	32836	34405	39985	39997			
36000	36001	36625	36637	37381	38197	39985	39997			

(c) "starwars",  $B_c = 8$  MBytes,  $d_s = 1$  sec

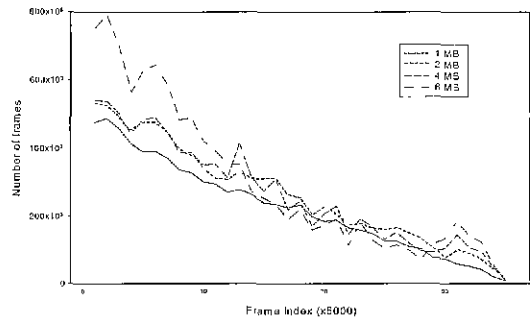
$f$	$S_1$	$S_2$	$S_3$	$S_4$	$S_5$	$S_6$	$S_7$	$S_8$	$S_9$	$S_{10}$
0	2267	2279	2291	2303	2315	11003	11015	12251	33611	73043
600	2219	2243	2255	2267	2279	2291	2303	2315	11003	11015
2160	2315	11003	11015	12251	33611	73043	92579	98687	98831	98843
2256	11003	11015	12251	33611	73043	92579	98687	98831	98843	107432
2268	11003	11015	12251	33611	73043	92579	98687	98831	98843	107432
2280	11003	11015	12251	33611	73043	92579	98687	98831	98843	107432
2292	11003	11015	12251	33611	73043	92579	98687	98831	98843	107432
2400	11003	11015	12251	33611	73043	92579	98687	98831	98843	107432
10992	12251	33611	73043	92579	98687	98831	98843	107432	144080	144923
12000	12215	12251	33611	73043	92579	98687	98831	98843	107432	144080
18000	24071	33587	33602	33611	73043	92579	98687	98831	98843	107432
24000	33587	33602	33611	73043	92579	98687	98831	98843	107432	144080
50000	98687	98831	98843	107432	144080	144923	145088	147959	166142	168647
70000	107432	144080	144923	145088	147959	166142	168647	168663	168671	173855
90000	144923	145088	147959	166142	168647	168663	168671	173855	173867	173879
98000	147959	166142	168647	168663	168671	173855	173867	173879	173882	173882
99000	147959	166142	168647	168663	168671	173855	173867	173879	173882	173882

(그림 4(a))는 선택된 프레임 위치로부터 비디오 스트림 내 잔여 프레임들을 모두 고려하는 경우에 버퍼 크기에 따른 계산량의 변화를 보여 준다. 동일 시작 프레임 위치에서 버퍼 크기가 커짐에 따라 보나 많은 데이터를 버퍼링할 수 있기 때문에 계산량이 많아지는 것을 볼 수 있으며, 시작 프레임 위치가 100,000 이상인 경우에는 잔여 프레임 수가 적어지므로 계산량도 대폭 감소하는 것을 볼 수 있다.

(그림 4(b))는 선택된 프레임 위치로부터 수렴 위치 사이의 프레임들만 고려하는 경우에 버퍼 크기에 따른 계산량의 변화를 보여 준다. (그림 4(b))에서는 버퍼 크기가 커짐에 따라 시작 프레임 위치에 대응하는 수렴 위치까지 대역폭 평활화로 인한 계산량이 커지는 것을 볼 수 있다. 이런 결과는 버퍼 크기가 커짐에 따라 대역폭 평활화의 특성상 기준 스케줄  $TS_0$ 의 세그먼트 수는 적어지고 세그먼트의 길이는 길어지므로 선택된 프레임 위치에 대응하는  $TS_0$ 의 수렴 위치의 길이가 더욱 커져서 대역폭 평활화가 적용되는 프레임 수가 커지기 때문이다[9].

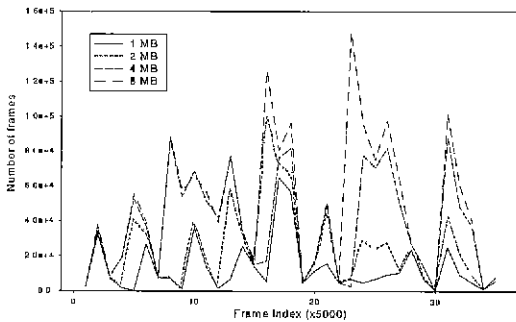
(그림 4(c))와 (그림 4(d))는 클라이언트의 버퍼 크기가 1 MByte와 8 MByte일 때, 선택된 프레임 위치에 대하여 잔여 프레임들을 모두 고려하는 방안과 수렴을 이용하는 방안의 계산량 비교를 보여 준다. 8 MByte 버퍼 크기의 경우에 수렴을 이용하는 방안이 기존 방안보다 선택된 프레임 위치가 0에서 10,000 사이에 제

산량이 최대 800,000에서 300,000개까지 감소하고 있음을 볼 수 있다. 그러나 프레임 위치가 110,000 이후로는 대역폭 평활화가 적용되는 프레임 수의 감소로 계산량의 차이가 작아짐을 보여 준다. 1 MByte 버퍼 크기의 경우에는 선택된 프레임 위치가 0에서 110,000로 증가해도 두 방안의 계산량이 최대 500,000에서 150,000의 차이가 있다. 버퍼 크기가 증가하면 기준 스케줄의 세그먼트 수가 감소하기 때문에 대역폭 평활화가 적용되는 프레임 수가 증가한다. 따라서, 수렴을 이용한 방안이 상대적으로 계산량이 많아져서 선택된 프레임 위치의 증가에 따라 8 MByte의 경우가 1 MByte의 경우보다 두 방안의 계산량 차이가 급격히 작아지는 것을 (그림 4(d))에서 볼 수 있다.

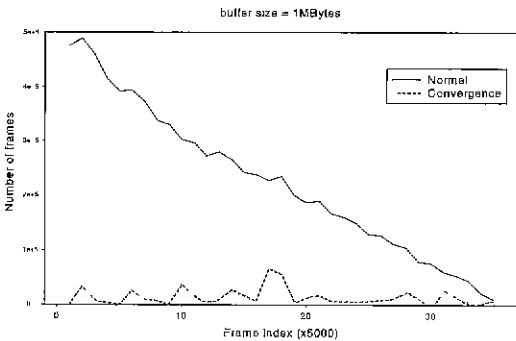


(a) 시작 프레임 위치로부터 잔여 프레임들을 모두 고려한 경우

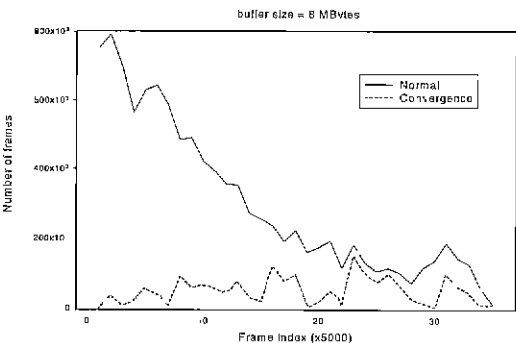




(b) 시작 프레임 위치로부터 수렴 위치까지의 프레임들만 고려한 경우



(c) 수렴을 이용한 계산량 감소(버퍼 크기 = 1Mbyte)



(d) 수렴을 이용한 계산량 감소(버퍼 크기 = 8Mbyte)

(그림 4) 계산량 비교

### 5. 결 론

본 논문에서는 이동이나 고속 연산 등의 대화형 연산 후 새로운 프레임 위치로부터 정상적인 비디오 스트림의 재생을 위한 전송 스케줄을 효율적으로 작성하는 방안에 대해 기술하였다. 본 논문에서는 대화형 연

산 후 선택된 프레임 위치로부터 서버가 전송 스케줄을 작성할 때, 동일 비디오 스트림에 대하여 재생 시작 프레임 위치가 다른 전송 스케줄들이 서로 수렴한다는 특성을 이용하여 선택된 프레임 위치에 대응하는 수렴 위치까지는 전송 스케줄을 구하고, 수렴 위치로부터 마지막 프레임까지는 기존의 스케줄을 그대로 이용하는 방안을 제안하였다. 제안된 방안은 선택된 프레임 위치로부터 수렴 위치까지만 대역폭 평활화를 적용함으로써 전송 스케줄 작성 시의 계산상의 오버헤드를 대폭 줄일 수 있었다. 또한, 비디오 스트림 내 모든 GoP의 I 프레임 위치에 대하여 그 위치에 대응하는 수렴 위치까지의 전송 스케줄 정보만을 오프라인으로 미리 계산하여 유지하는 것이 가능하므로 선택된 프레임 위치로부터 마지막 프레임 위치까지의 전송 스케줄 정보를 저장할 필요가 없게 되어 전송 스케줄을 저장하는 데 있어서의 오버헤드를 대폭 줄일 수 있다. 따라서, 클라이언트에서 현재의 재생 위치를 임의의 프레임 위치로 이동한 후 그 위치로부터 정상적인 재생을 할 수 있도록 서버가 전송 스케줄을 작성하는 경우에, 제안된 방안을 사용하여 미리 작성된 전송 스케줄을 쉽게 검색하여 사용하는 것만으로 전송 스케줄 작성으로 인한 지연을 제거할 수 있게 되어 클라이언트에서의 재생 시작 지연을 대폭 줄일 수 있을 것으로 기대된다.

### 참 고 문 헌

- [1] 정제창 역, '그림으로 보는 최신 MPEG', 교보문고, 1996.
- [2] Jean-Pierre Leduc, 'Digital Moving Pictures-Coding and Transmission on ATM Networks', Elsevier Science B. V., 1994.
- [3] Wu-chi Feng, 'Buffering Techniques for the Delivery of Compressed Video in Video-on-Demand Systems', Kulwer Academic Publishers, 1997.
- [4] Wu-chu Feng, Jennifer Rexford and Farnam Jahanian, "A Comparison of Bandwidth Smoothing Techniques for the Transmission of Pre-recorded Compressed Video," in Pro. of IEEE INFOCOM'97, pp 58-66, Kobe, Japan, April 1997.
- [5] Jac-Hong Lee, Sun-Sook Lee, Tae-Kyun Kim, "Bandwidth Smoothing Scheme for VBR Video

Streams in Video-On-Demand Service,” 한국통신학회 논문지, 제23권 제3호, pp.687-700, 1998.

- [6] Jae-Hong Lee, Sun-Sook Lee, “A GoP-based Dynamic Scheduling Scheme for Fast Scan Functions with a Certain Speedup Playback Rate,” in Proc. of International Conference on Computer Communication 1999 (ICCC’99), Vol.II, pp.333-340, Tokyo, Japan, Sept 1999.
- [7] 이재홍, “주문형 비디오에서 m배속 고속 재생을 지원하기 위한 GoP 기반 동적 전송 스케줄 작성”, 한국통신학회 논문지, 제24권 제9B호, pp 1643-1651, 1999.
- [8] Jayanta K. Dey, Subhabrata Sen, James F. Kurose, Don Towsley, “Playback Restart in Interactive Streaming Video Applications,” in Pro. of IEEE International Conference on Multimedia Computing and Systems, pp.458-465, Ottawa, Canada, June 1997.
- [9] Jae-Hong Lee, Sun-Sook Lec and Tae-Kyun Kim, “A Rewriting Scheme for a transmission schedule of playback restart of a stored video after VCR operations utilizing bandwidth smoothing,” in Proc. of International Conference on Advanced Communication Technology (ICACT’99), pp.422-427, Muju Resorts, Korea, Feb. 1999.
- [10] Wei Zhao, Tarun Seth, Michelle Kim and Marc Willebeek-LeMar, “Optimal Bandwidth/Delay Tradeoff for Feasible-Region-Based Scalable Multimedia Scheduling,” in Proc. of IEEE INFOCOM’98, pp 1131-1138, 1998.
- [11] M. Grossglauser, S. Keshav, and D. Tse. “RCBR : A simple and efficient service for multiple time-scale traffic,” in Proc. of ACM SIGCOMM Symposium on Communications Architectures and Protocols, pp.219-230, Aug 1995.
- [12] M. W. Garrett and A. Fernandez, ‘Variable Bit Rate Video Bandwidth Trace Using MPEG Code’, 4 Nov. 1994.
- [13] O. Rose, “Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems,” in Proc. of IEEE 20th Conference on Local Computer Networks, pp.397-406, Oct. 1995.
- [14] J. D. Salehi, Z.-L. Zhang, J. F. Kurose and D Towsley, “Supporting stored video : Reducing rate variability and end-to-end resource requirements through optimal smoothing,” in Proc. of ACM SIGMETRICS. pp.222-231, May 1996.

### 이재홍



e-mail : jhlee@danyang.damyang.ac.kr  
 1986년 충남대학교 전자공학과 졸업 (공학사)  
 1988년 충남대학교 대학원 전자공학과 졸업(공학석사)  
 1999년 충남대학교 대학원 컴퓨터공학과 졸업(공학박사)

1988년~1994년 국방과학연구소 연구원  
 1994년~1995년, 1999년 (주)한국인식기술 개발실장  
 2000년~현재 전남도립 담양대학 진산·정보통신공학부 전임강사  
 관심분야 : 멀티미디어 통신, 무선 ATM, 전자상거래

### 김승환



e-mail shkim@chch.ac.kr  
 1982년 충남대학교 전자공학과 졸업(공학사)  
 1988년 충남대학교 대학원 전자공학과 졸업(공학석사)  
 1994년 청주대학교 대학원 전자공학과 졸업(공학박사)

1989년~1990년 한국전자통신연구원 연구원  
 1997년~1998년 미국펜실바니아대학교 객원연구원  
 1990년~현재 충청대학 컴퓨터학부 부교수  
 관심분야 : 무선 ATM, 초고속정보통신, 스위칭 이론