論文 98-23-5-9

프록시 서버를 이용한 DAVIC VOD 시스템의 설계

正會員 안 경 아*, 최 훈*

Server Network Architectures for VOD Services

Kyung-Ah Ahn*, Hoon Choi* *Regular Members*

요 약

VOD 서비스 시스템에서의 시간 지연을 줄이기 위하여 캐싱 기능을 갖는 프록시 서버를 이용한 DAVIC VOD 서비스 제공 방식을 제안한다. 프록시 서버는 서비스 제공자와 사용자 사이에 위치하며 서비스 제공자가 보유한 일부 프로그램을 캐싱하여 사용자에게 제공한다. 프록시 서버를 사용하면 서비스 제공자와 네트워크의 부하를 줄일 수 있다.

프록시 서버의 동작은 요구된 프로그램이 자신의 디스크에 있는 경우와 없는 경우로 구분된다. 첫번째 경우 프록시 서버가 서비스를 제공하고, 두번째 경우는 서비스 제공자로 부터 프로그램을 수신하여 사용자에게 제공하고 동시에 자신의 디스크에 저장하여 캐싱한다. 프록시 서버의 저장 공간이 부족하면 캐시 대치 알고리즘을 수행한다. 캐시 대치 알고리즘으로 잘 알려진 LRU, LFU, FIFO중에서 제안한 프록시 서버에 가장 효과적인 방식이 LFU 방법을 이용하는 것이 3개중 가장 효과적임을 시뮬레이션을 이용하여 보였다.

ABSTRACT

In this paper, we provide a design of DAVIC VOD service system with proxy servers which perform caching of video streams. Proxy servers are placed between a service provider system and service consumer systems. They provide video services to consumers on behalf of the service provider, therefore they reduce the loads of service providers and network.

The operation of a proxy server depends on whether the requested program is in its storage. If this is the case, the proxy server takes all the controls, but if the proxy does not have the program, it forwards the service request to a service provider. While the service provider system provides the program to the consumer, the proxy copies and caches the program. The proxy server executes cache replacement, if necessary. We show by simulation that the LFU is the most efficient caching replacement algorithm among the typical algorithms such as LRU, LFU, FIFO.

*충남대학교 컴퓨터공학과
論文番號:97309-0904
接受日字:1997年 9月 4日

Ⅰ. 서 론

사용자의 요구에 의해 통신망을 통하여 영화, 비디오 게임, 홈 쇼핑 등의 비디오 관련 서비스를 제공받는 것을 주문형 비디오 서비스(VOD : Video On Demand)라고 한다. 이 서비스는 VOD 서비스 제공자가 사용자 요구를 수용하여 가상 VCR(Video Casette Record) 기능을 이용하여, 사용자가 원하는 내용을 선택하게 하는 특징을 갖는다. 피동적인 텔레비젼 방송 서비스와는 달리, VOD 서비스는 사용자가 언제든지 원하는 비디오 프로그램을 선택하여 서비스를 받을 수 있기 때문에 앞으로 대표적인 멀티미디어 정보 서비스중 하나가 될 것으로 전망되고 있다[1, 2, 3].

VOD 서비스를 위한 표준은 DAVIC(Digital Audio VIsual Council)에서 국제 표준 규격을 정하고, 이 표준에 따라 VOD 시스템이 개발되고 있다. DAVIC 표준에 의하면 VOD 시스템 참조 모델은 기능에 따라 내용 제공자 시스템(Content Provider System), 서비스 제공자 시스템(Service Provider System), 전달 시스템(Delivery System), 사용자 시스템(Service Consumer System)으로 구성되며[4], 지금까지는 수용된 많은 사용자에게 안정적인 스트림을 제공하기 위한 서비스 제공자 시스템 위주의 연구가 진행되었다[5, 6, 7]. 서비스 제공자 시스템의 효율성과 더불어 서비스 품질에 영향을 미치는 것은 시스템들간의 네트워크 구성 방식으로서, 서비스 제공자와 사용자간의 구성은 중앙 집중형 서비스 구조와 분산형 서비스 구조로 생각할 수 있다. 중앙 집중형 서비스 구조는 서비스 제공자가 사용자들에게 모든 서비스를 제공하는 것으로 서비스 제공자가 보유한 비디오 스트림, 즉 프로그램들이 다수의 사용자에 의해서 공유될 수 있는 것이 장점이다. 그러나 사용자의 요구가 급증하면 병목 현상이 발생하며, 병목 현상을 해결하기 위해 네트워크 전송 대역폭과 서버의 처리 능력을 확장해야 하는 문제점이 있다. 반면에 분산형 서비스 구조는 서비스 제공자가 지역적으로 분산되어 병목 현상을 해소하고 고장 감내한 서비스를 제공하는 장점이 있지만 같은 비디오 프로그램이 서로 다른 위치의 서비스 제공자 시스템에 중복 저장되어야 하는 문제점이 있다.

이러한 문제점들을 해결하기 계층형 분산 구조[8]가 제안되었다. 이 구조에서 서비스는 여러 계층의 서버에 의해 처리되는데, 하위 계층 서버는 상위 계층 서버가 보유한 프로그램의 일부를 디스크나, 테잎의 매체를 이용하여 저장한다. 이 구조의 경우, 상위 계층 서버에 프로그램이 존재하더라도 하위 계층의 서버에 저장되어 있지 않으면 하위 계층 서버는 그 프로그램을 사용자들에게 제공할 수 없다. 따라서 하위 계층 서버에 저장된 프로그램들은 실시간적으로 교체되지 못하여 사용자의 요구가 즉시 반영되지 못하는 또 다른 문제점을 가진다. 이와 유사한 연구[9]로 저장 장치를 서버 노드, 인터페이스 노드와 객체 노드로 계층적으로 구성하는 방법도 제안되었다. 이 방법은 객체가 어느 일정 기준의 액세스 횟수를 넘을 경우 중간 계층인 인터페이스 노드에 프로그램을 캐싱하는 기법이다. 이 기법의 적용은 기준의 변화에 따라 성능이 크게 의존되어 서비스 품질의 변화가 심한 약점이 있다. 계층형 분산 구조와 유사한 U.C. Berkely에서 연구중인 Distributed VOD 시스템[10]은 영구적인 객체는 Archive Server에, 임시적인 객체는 Virtual File Server에 저장되며 프로그램이 요구될 경우 Archive Server에 있는 객체를 Virtual File Server로 복사하는 방법을 이용한다. 이 구조는 Archive Server가 프로그램 저장소 역할만 할 뿐 직접 서버로서 사용자를 상대하는 것이 아니므로 Virtual File Server의 부하를 분담하지 못하는 제약이 있다.

앞에서 언급한 여러 가지 문제점들을 해결하기 위하여 본 논문에서는 캐시 메카니즘을 갖는 프록시 서버를 이용하여 DAVIC VOD 서비스의 분산 처리 능력을 향상시키고 사용자의 만족을 충족시킬 수 있는 구조를 제안한다. 분산형 서비스 구조의 한 형태인 프록시 서버 구조는 네트워크상에 프록시 서버를 두어 같은 비디오 프로그램을 요구할 때 이전에 디스크에 저장된 비디오 프로그램을 전송하는 방법이다. 자주 이용되는 프로그램은 프록시 서버에 캐싱되어 서버 및 네트워크 부하를 경감시키고, 중앙 서버에 장애가 발생하면 프록시 서버가 서비스 사용자에게 원활하게 서비스를 제공할 수 있는 장점을 가진다. 프록시 서버의 구현을 위해 고려되어야 하는 것은 캐시 대치 알고리즘과 데이터 일치성 유지이다[11]. 프록시 서버에 서비스 제공자가 새로이 공급하는 프로그램을 저장하기 위한 디스크의 저장 공간을 확보하기 위

하여 필요한 것이 캐시 대치 알고리즘이다. 비디오 스트림의 특성에 맞는 캐시 대치 알고리즘을 수행하면 프록시 서버의 효과를 높일 수 있다. 프록시 서버와 서비스 제공자간의 데이터 불일치 문제는 서비스 제공자에 프로그램이 새로 추가되는 경우, 사용자의 요청에 의해 프록시 서버에 새로운 프로그램이 저장되므로 해결될 수 있다. 반면에, 프록시 서버에 이미 복사된 프로그램이 변경되는 경우는 서비스 제공자가 프록시 서버들에게 복사본이 더 이상 유효하지 않음을 통보하여 데이터 일치성을 유지한다. 이 문제는 VOD 서비스의 특성이 데이터에 대한 읽기 동작이 많으므로 효율성, 서비스 품질에 영향을 미치지 않을 것으로 예상된다. 프록시 서버는 사용자들이 선호하는 프로그램들을 동적으로 저장할 수 있어 요구된 비디오 프로그램을 보유하고 있는 경우 프록시 서버에서 바로 서비스를 제공할 수 있으며, 보유하고 있지 않은 경우 서비스 제공자로 부터 수신하여 사용자에게 실시간으로 전달하면서 동시에 자신의 디스크에 저장한다. 제안된 프록시 서버 구조는 사용자의 요구를 동적으로 반영할 수 있어 히트율을 높이고 사용자의 서비스 응답 시간을 줄일 수 있다. 물론 프록시형 분산 구조는 추가적인 저장 장치 설치에 따른 경제적 부담이 있다.

2장에서 DAVIC Spec. 1.0[4]을 기반으로 하는 VOD 시스템에 대하여 간략히 설명하고, 3장에서 프록시 서버를 도입한 새로운 구조에 대하여 설명한다. 4장에서 제안한 구조에 대한 성능을 분석하고, 5장에서 결론을 맺는다.

Ⅱ. DAVIC VOD 시스템 구조

DAVIC VOD 시스템은 영화 제작자와 같이 VOD 서비스 내용을 제공하는 내용 제공자 시스템, 서버와 같이 내용을 보유하여 서비스를 제공하는 서비스 제공자 시스템, 정보를 전달하는 네트워크인 전달 시스템, 그리고 서비스를 최종적으로 사용하는 소비자인 서비스 사용자 시스템으로 구성된다. 내용 제공자 시스템과 서비스 제공자 시스템간의 정보 교환은 빈번하게 발생하지 않고 온라인으로 연결될 필요가 없으므로 VOD 서비스의 동작 방식에 영향을 미치지 않는다. 따라서 본 논문의 연구 범위는 DAVIC에서 정의한 서비스 제공자 시스템과 서비스 사용자 시스템으로 한다. 이 절에서는 DAVIC VOD 시스템의 전반적인 기능과 구성 컴포넌트에 대하여 간략히 논한다 (그림 1 참조).

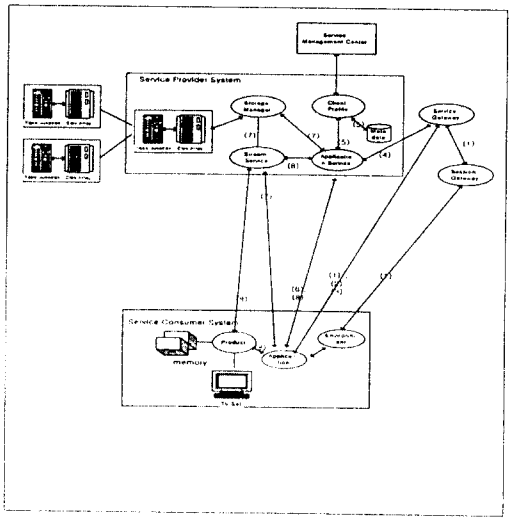


그림 1. DAVIC VOD 시스템 구조
Fig. 1 DAVIC VOD System Architecture

1. 서비스 제공자 시스템

서비스 제공자 시스템은 서비스 사용자에게 비디오 스트림을 전송하기 위한 고속의 저장장치로 구성되며, 기능 컴포넌트들은 아래와 같이 분류된다.

- Storage Manager: 비디오 스트림이 저장된 디스크 어레이와 테잎에 대한 관리를 한다.
- Application Service: 멀티미디어 정보에 대한 서비스 조작 정보를 처리한다.
- Meta data: 사용자에 대한 모든 정보(요금 등급, 서비스 등급등)를 의미한다.
- Stream Service: 스트림 계층에서 내용 흐름에 대한 처리를 수행한다.
- Service Gateway: 시스템에 접속된 모든 서비스를 등록받아 관리하여 사용자에게 메뉴 리스트를 보여주며 서비스 제공자 시스템들의 주소 정보를 사용자에게 제공한다.
- Session Gateway: 망내 시스템들간 연결을 위한 자원의 추가 및 제거를 담당한다.

• Client Profile: Meta data를 참조하여 서비스를 요구한 사용자에 대한 인증 검사를 수행한다. 사용자의 서비스 요구 빈도, 프로그램명등 여러 가지 정보를 기록하여 서비스 관리 센터로 전달하는 기능을 수행한다.

2. 서비스 사용자 시스템

서비스 사용자 시스템은 출력장치와 압축된 비디오 데이터를 재생하는 셋탑 박스로 구성되며, 기능 컴포넌트들은 아래와 같이 분류된다.

• Environment: 서비스 제공자 시스템과 서비스 품질(Quality of Service) 협상 및 세션을 설정하는 기능을 수행한다.
• Application: Service Gateway로부터 받은 메뉴 선택과 프로그램에 대한 네비게이션을 한다.
• Product: 서버로부터 받은 스트림을 디코딩하여 출력장치로 전송한다.

3. VOD 서비스 시나리오

이 절에서는 VOD 서비스를 위한 서비스 제공자 시스템과 서비스 사용자간 처리 순서에 대해 그림 1을 참조하여 설명한다. 서비스가 수행되는 순서는 그림 1에 표시된 번호 순이며, 표현을 간단히 하기 위해 이후에는 서비스 제공자 시스템을 서버라고 부르고 서비스 사용자 시스템을 사용자로 간략화한다.

(1) 사용자가 TV를 켜면 사용자의 Environment 요소에서 서버의 Session Gateway에게 세션 설정 요구를 한다. Session Gateway는 Service Gateway에게 세션을 넘겨주고, Service Gateway에서 사용자에게 세션 설정에 대한 응답을 한다.
(2) 응답을 받은 사용자는 Service Gateway에게 메뉴 리스트를 요구하고, Service Gateway는 사용자에게 메뉴 리스트를 준다.
(3) 사용자는 메뉴 리스트에서 VOD 서비스를 선택한다.
(4) Service Gateway는 VOD 서버와 사용자 사이에 세션을 연결한다.
(5) 서버의 Application Service는 요구받은 서비스에 대하여 사용자가 정당한 사용자인지 Client Profile에게 문의하여, 정당한 사용자이면 VOD 서비스의 프로그램 리스트를 사용자에게 전송한다. 정당하지 않은 사용자의 서비스 요청은 거절된다.
(6) 프로그램 리스트를 전송받은 사용자는 프로그램을 선택하여 서버에게 보낸다.
(7) 서버의 Application Service는 Storage Manager에게 프로그램 검색을 요구하고, Stream Service는 사용자의 Application에게 스트림 서비스 준비 통보를 한다.
(8) 사용자로 부터 재생 명령을 요구 받으면 실제적인 스트림을 전송한다.

Ⅲ. 프록시 서버를 이용한 VOD 시스템

이 절에서는 제안하는 프록시형 분산 구조의 기능, 컴포넌트, 시나리오, 그리고 캐시 대치에 대하여 논한다. VOD 서비스에서 비디오 액세스 시간을 줄이기 위하여 사용자 시스템에서, 서버 시스템에서, 프록시 서버 시스템에서 캐싱 기능을 가질 수 있다. 사용자 시스템에서 캐싱할 경우 사용자는 디스크 설치에 대한 비용 부담을 가지고 캐싱 히트율이 높지 않으므로 비효율적이다. 중앙 서버에 캐싱하는 경우 중앙 서버의 메모리내 캐싱만이 이루어져 네트워크 측면에서 큰 의미가 없으므로 사용자가 느끼는 비디오 액세스 시간에는 큰 변화가 없다. 그래서 프록시 서버를 두어 캐싱할 경우를 고려한다. 프록시 서버는 하나의 복사본만이 필요하므로 사용자들에 캐싱하는 경우와 비교해 디스크 공간을 절약할 수 있고, 프로그램이 히트되는 경우 중앙 서버의 부하를 줄일 수 있어 동시에 여러명의 사용자에게 서비스할 수 있다. 따라서 서버와 별도의 프록시 서버를 두어 캐싱하는 것이 효율적이므로 프록시 서버를 이용하여 DAVIC VOD 서비스를 제공하기 위한 새로운 망 구조를 구성한다.

프록시 서버 구조는 동적으로 사용자의 요구를 반영하여 저장할 수 있는 구조로서, 서버와 비교되는 점은 서버가 갖는 대부분의 기능들을 가지나 서버가 제공하는 모든 비디오 프로그램에 대한 일부분만을 보유하고 있어 용량면에서 적다. 또한, 사용자와 지역적으로 가까운 위치에 설치되어 서버 시스템보다 신속히 서비스에 대응할 수 있고 서버의 부하를 분담한다(그림 2 참조).

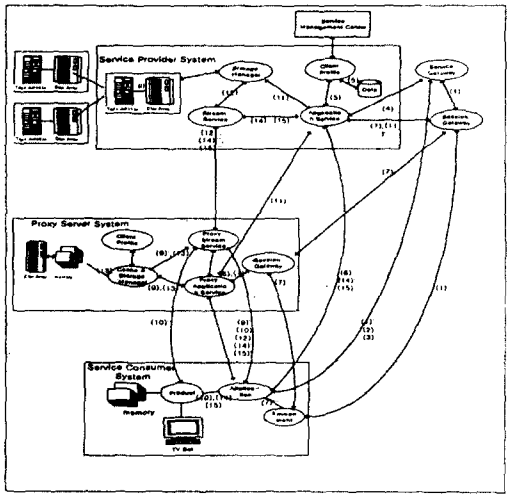


그림 2. 프록시 서버를 이용한 VOD 시스템 구조
Fig. 2 VOD System Architecture with Proxy Server

1. 프록시 서버의 기능

프록시 서버의 특징적 기능은 아래와 같다.

- 프록시 서버는 사용자가 요청한 프로그램이 자신의 디스크에 없을 경우 서버로 부터 프로그램을 받아 사용자에게 전송한다. 동시에 자신의 디스크에 저장하여 저장된 프로그램이 다시 요청될 경우 바로 서비스를 제공한다.
- 서버로 부터 프로그램을 받는 도중에 사용자로 부터 서비스 중단 요청을 받으면, 프록시 서버는 사용자에게 보내는 프로그램의 전송은 중단하고 서버로 부터 계속 프로그램을 받아 자신의 저장 장치에 한편의 프로그램으로 저장한다. 이때 사용자의 과금 정보는 서비스가 중단된 시점까지 기록한다.
- 프록시 서버에 프로그램을 저장할 때, 저장 공간을 확보하기 위하여 기존의 비디오 프로그램을 대치하는 캐시 대치 알고리즘을 수행한다.
- 서버는 사용자가 요구한 프로그램을 바로 재생하지 않고, 프록시 서버에게 제어권을 넘겨준다.
- 서버가 보유한 프로그램이 변경될 경우 복사본을 가진 프로그램이 더 이상 유효하지 않음을 프록시 서버에게 통보하여, 서버와 프록시 서버간의 데이터 일치성을 유지한다.
- 서버는 프록시 서버로 부터 사용자의 과금 정보를 받아 저장한다.

2. 프록시 서버 컴포넌트

프록시 서버의 기능 컴포넌트들을 다음과 같다.

- Proxy Stream Service: 서버의 Stream Service와 동일한 기능이다.
- Proxy Session Gateway: 서버의 Session Gateway와 동일한 기능이다.
- Client Profile: 서버의 Client Profile와 동일한 기능이다.
- Proxy Application Service: 사용자가 선택한 프로그램이 프록시 서버에 존재 여부를 검색하여, 존재하는 경우 Cache & Storage Manager에게 프로그램의 스트림을 요구한다. 존재하지 않는 경우 자신과 연결된 서버의 Application Service에게 해당 프로그램을 요청한다.
- Cache & Storage Manager: 서버로 부터 받은 새로운 프로그램명을 저장하고 저장된 프로그램명을 Proxy Application Service에게 통보한다. 캐시 대치 알고리즘을 수행하여 저장 공간을 확보한다.

3. 프록시 서버를 이용한 VOD 서비스 시나리오

다음은 프록시 서버에 의한 VOD 서비스의 수행 절차에 대한 설명으로 연결 단계, 스트림 전송 및 제어 단계, 연결 해제 단계별로 서술되며 서비스 순서는 그림 2에 표시된 번호 순이다.

3.1 사용자와 서버간의 연결단계

(1)~(6) 프록시 서버가 없는 경우의 시나리오 (Ⅱ장 3절의 (1)~(6))와 동일하다.

3.2 서버와 프록시 서버간의 연결 단계

(7) 서버는 사용자 사이와의 세션 연결을 서버와 프록시 서버, 프록시 서버와 사용자의 연결로 전환하고 프록시 서버에게 사용자 식별 번호와 요청된 프로그램명을 넘겨준다.

(8) 프록시 서버의 Proxy Application Service는 캐싱 테이블을 이용하여 요청된 프로그램의 유무를 검사한다.

(9) 그 결과 프로그램이 있으면 프록시 서버의 Cache & Storage Manager에게 프로그램을 요구하고,

Cache & Storage Manager는 Proxy Stream Service로 스트림 서비스 요청을 한다. Proxy Stream Service는 사용자의 Application에게 스트림 서비스 준비 통보를 한다.

(10) 사용자로 부터 재생 명령를 받으면 실제적인 스트림을 전송한다.

(11) 프로그램이 없으면 서버의 Application Service에게 프로그램을 요구하고, Application Service는 Storage Manager에게 프로그램을 요구한다.

3.3 프로그램 스트림 전송 및 제어 단계

(12) 프록시 서버가 비디오 프로그램을 보유한 경우, Proxy Stream Service가 사용자에게 스트림을 전송하고, 아닌 경우 프록시 서버는 서버의 Stream Service를 통하여 비디오 프로그램을 받아서 사용자에게 전송한다.

(13) 프록시 서버의 Cache & Storage Manager는 서버로 부터 받은 프로그램을 저장하고, Proxy Application은 캐싱 테이블을 수정한다. 이때, Cache & Storage Manager에서 디스크에 저장할 공간이 없을 경우 캐시 대치 알고리즘을 이용하여 대치할 프로그램 목록을 선택한다.

(14) 스트림에 대한 제어는 프록시 서버에 스트림이 있는 경우 프록시 서버에서 수행하고, 없는 경우 서버에게 제어권을 넘겨준다.

- 프록시 서버가 서버로 부터 스트림을 받는 중 사용자가 빨리감기 명령을 요청하면 빨리감기 지점의 스트림이 프록시 서버에 없을 수 있다. 이 경우 프록시 서버는 빨리감기 명령을 서버에게 넘겨주어 서버에서 제공하는 스트림을 받아, 자신의 디스크에 저장하지 않고 바로 사용자에게 넘겨준다. 프록시 서버는 프로그램의 저장이 중지된 지점의 NPT(Normal Play Time) 값을 저장한다. NPT는 스트림이 처음부터 정상적인 속도로 재생될 때 이에 상응하는 상대적인 시간값을 의미한다.
- 서비스가 끝난 후 서버와 별도의 연결을 시도하여 저장된 프로그램의 NPT 값 이후의 비디오 스트림을 받아 자신의 디스크에 저장하고, NPT와 대응되는 MPEG-TS 번호를 구하여 내부적으로 사용되는 캐싱 테이블을 수정한다.
- 사용자가 A(그림 3 참조) 부분에서 C 부분으로 빨리감기를 요청한 경우 프록시 서버는 서버로 부터 C 부분을 받아, 저장하지 않고 바로 사용자에게 넘겨준다. 이후 서버와 별도의 연결을 통해 나머지 B, C부분을 저장한다.

(15) 프록시 서버가 서버로 부터 스트림을 받는 상태에서 사용자가 되감기 명령을 요청한 경우 다시 재생되는 비디오 스트림 부분을 서버로 부터 받아, 저장하고 사용자에게 전송한다. 사용자가 B(그림 4 참조) 부분까지 되감기한 경우 서버로 부터 새로 재생되는 B, C 부분을 받아 자신의 디스크에 저장하고 사용자에게 전송한다.

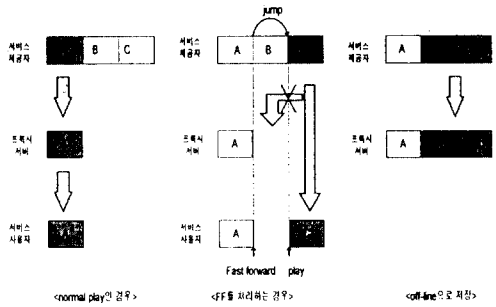


그림 3. 빨리감기에 대한 제어
Fig. 3 Control for fast-forward function

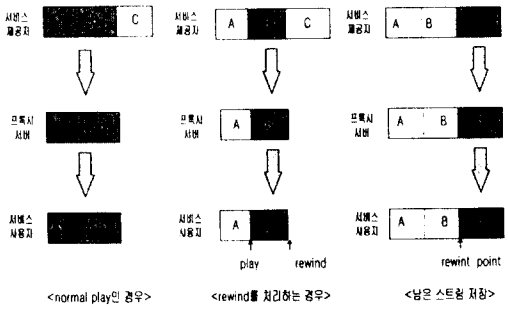


그림 4. 되감기에 대한 제어
Fig. 4 Control for rewind function

3.4 사용자와 프록시 서버간의 해제 단계

(16) 프록시 서버가 직접 프로그램을 제공하는 경우 사용자가 정지 명령을 요청하면 프록시 서버는 사용자와의 모든 연결을 해제하여 서비스를 종료한다.

(17) 프록시 서버가 서버로 부터 프로그램을 받는 중 정지 명령을 수신하면, 프록시 서버는 사용자와의 연결을 해제하여 서비스를 종료하고 서버와의 연결은 유지하여 프로그램의 나머지 부분을 저장한다.

(18) 사용자의 사용 기록은 주기적으로 센터에 전송하여 과금 정책, 향후 서비스 개선등에 사용될 수 있도록 지원한다.

4. 프록시 서버의 캐시 대치

프록시 서버의 캐시 대치는 두가지로 구분된다. 첫째, 해당 프로그램이 프록시 서버의 디스크에 저장되어 있는지 여부에 대한 프로그램 캐싱이다. 둘째, 프로그램의 비디오 스트림이 프록시 서버의 메모리에 존재하는지 여부에 대한 메모리 캐싱이다. 캐시 대치는 프록시 서버의 Cache & Storage Manager가 PPHIT (Per-Program High Index Table)를 참조하여 수행한다. PPHIT 테이블은 프로그램 식별번호, 프로그램이 저장된 디스크의 start block number, highest index로 이루어진다. 일반적으로 프로그램들은 MPEG-TS 형태로 디스크에 저장되므로 한편의 프로그램은 일련의 MPEG-TS들의 집합이다. Highest Index는 현재까지 프록시 서버에 저장된 프로그램의 마지막 MPEG-TS의 번호이다.

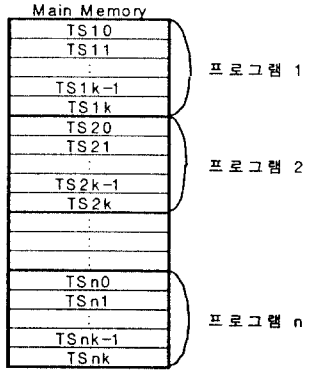
프로그램에 대한 캐싱은 테이블의 프로그램 식별번호 필드를 검색하여 프로그램의 유무를 조사한다. 프로그램이 없는 경우 서버로 부터 프로그램을 받아 테이블의 빈 엔트리에 새로운 프로그램을 등록한다. 테이블에 빈 엔트리가 없는 경우 프로그램 캐시 대치 기법에 따라 프로그램을 선택하여 대치한다. 프로그램 캐시 대치는 프로그램의 접근 시간을 비교하여 가장 오래된 접근 시간을 가지는 프로그램을 대치하는 LRU(Least Recently Used) 알고리즘, 프로그램의 접근 빈도를 비교하여 가장 작은 접근 빈도를 가지는 프로그램을 대치하는 LFU(Least Frequently Used) 알고리즘, 가장 먼저 재생된 프로그램을 대치하는 FIFO(First In First Out) 알고리즘등이 적용된다. 각 알고리즘별로 프록시 서버의 캐싱 성능은 다음 4장에서 설명한다.

사용자가 스트림 제어를 요구할 경우 프록시 서버는 디스크에서 MPEG-TS를 메모리로 읽어 사용자에게 전송한다. 그러나 요구되는 MPEG-TS 번호가 테이블에 저장된 highest index 보다 클 경우, 프록시 서버는 서버로 부터 MPEG-TS를 받아 디스크에 저장하고, 캐싱 정보 테이블의 highest index를 수정한다. 메모리 캐싱인 경우 일반적인 운영체제에서 메모리와 디스크 사이는 페이지 대치가 발생하므로 재생하려는 MPEG-TS가 메모리에 적재된 페이지에 있으면 그것을 사용하고 없으면 디스크에서 페이지 단위로 읽어 들인다(그림 5 참조).

Program ID.	Start Block No.	Highest Index
프로그램 1	B_{11}	8924100
프로그램 2	B_{21}	1784820
프로그램 3	B_{31}	78482
:	:	:
프로그램 n	B_{n1}	78482

(a) Per-Program High Index Table

(a) Per-Program High Index Table



(b) MPEG-TS의 메모리내 배열

(b) The memory array of MPEG-TS

그림 5. 캐싱 정보 테이블

Fig. 5 Caching Information Table

Ⅳ. 성능 분석

1. 모 델

제안된 구조에 대하여 시뮬레이션을 통하여 성능 분석에 대해 논한다. 성능 평가 항목으로서 응답률, 첫 화면에 대한 응답 시간, 스트림 제어에 대한 응답 시간을 고려한다. 시뮬레이션 툴로 SLAM II[13]를 이용하고 사용된 컴퓨터 시스템은 Sun SPARCserver 670MP이다.

다음 그림 6는 프록시 서버를 사용한 DAVIC VOD 시스템에서 사용자가 첫 화면을 수신하는 응답시간에 대한 성능 분석 모델을 나타내며(서비스 수행절차로는 Ⅲ장 3절의 (1)~(12)에 해당됨), atrib(2)에 저장된 값은 서비스의 흐름을 제어하기 위하여 내부적으로 사용되는 값이다. 그림 7은 사용자와 프록시 서버가 연결된 상태에서 스트림 제어의 응답시간에 대한 모델을 나타낸다(서비스 수행절차로는 Ⅲ장 3절의 (10)~(15)에 해당됨). 모델에서 알파벳 Q자와 같이 생긴 기호는 큐(Queue)를 의미하며 프록시 서버의 전체 구성도인 그림 2에 나타나는 시스템들의 컴포넌트들을 큐로 나타내었다. 화살표에 표시된 숫자는 큐의 서비스 시간과 시스템간의 전송 지연을 나타낸다.

시뮬레이션의 가정은 서비스를 요청한 사용자에 대한 인증 검사에서 99%는 정당한 사용자이고, 1%는 정당하지 못한 사용자라고 가정하고 파라메터들은 아래와 같다[12].

- 프로그램 한편의 크기 : 2Mbps로 코딩된 100분 영화를 가정하면 2Mbps/8bit = 0.25Mbyte이므로 0.25 Mbyte * 60sec. * 100min. = 1.5Gbyte이다.
- 시간 파라메터 : 각 기능 컴포넌트들의 처리 시간에

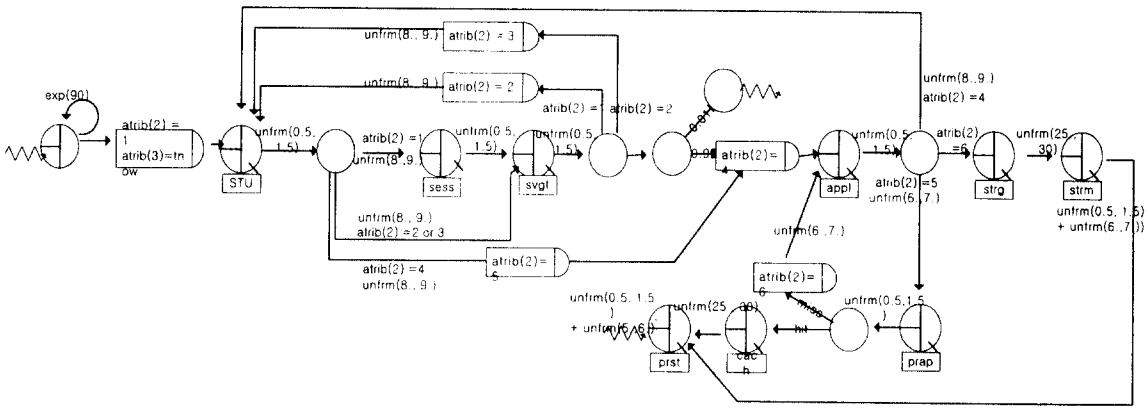


그림 6. 첫 화면의 응답시간에 대한 성능 분석 모델
Fig. 6 The model for response time of the first frame

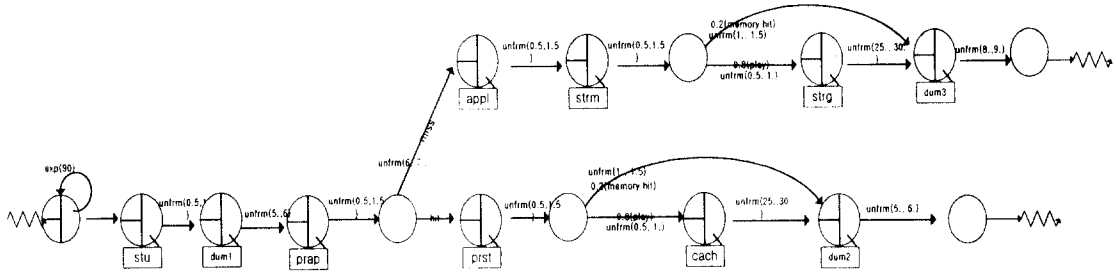


그림 7. 스트림 제어의 응답시간에 대한 성능 분석 모델
Fig. 7 The model for response time of video stream controls

대한 임의 변수는 0.5~1.5msec., 사용자와 서버간은 8~9msec., 사용자와 프록시 서버간은 5~6msec., 그리고 프록시 서버와 서버간은 6~7msec. 범위내에서 균등 분포(Uniform Distribution)로 가정한다. 또한, 각 시스템의 디스크 액세스 시간은 25~30msec. 범위내에서 균등 분포로 가정한다.

- 가입자의 수:1000명
- 서비스 요청 빈도 (request rate):가입자당 평균 하루에 한 번 요구하는 포아송 분포(Poisson Distribution)로 가정하여
  24hour*3600sec/1000 request (1회*1000명)=86.4sec.
  이므로 약 90초 마다 하나의 서비스 요구가 발생된다고 가정한다.
- 프록시 서버의 최대 디스크 용량:80편*1.5Gbyte =120Gbyte
- 서버에서 보유하는 최대 영화 편수:총 1000편
- 프로그램별 사용자 선호도:일반적인 비디오 관람자는 최신 영화에 대한 수요가 많을 것으로 예상되므로 프로그램 Ⅰ 그룹은 최근 히트하고 있는 영화로 전체 1000편 가운데 80편의 영화가 차지한다고 가정한다. 프로그램 Ⅱ 그룹은 히트 시기가 지난 일반적인 영화로서 720편, 프로그램 Ⅲ 그룹은 오래된 영화로서 200편 정도로 가정하고 각 그룹의 서비스 요구 확률은 아래와 같이 가정한다.

표 1. 프로그램 그룹별 서비스 요구 확률
Table 1. Service Request Rate of Program Gruops

프로그램 그룹	등 급	서비스 요구 확률
Ⅰ	최신 영화	85%
Ⅱ	일반 영화	10%
Ⅲ	오래된 영화	5%

- 메모리내의 MPEG-TS hit 확률:0.2

2. 분 석

본 절에서는 프록시를 사용하지 않고 서버와 사용자만 연결된 경우와 프록시 서버를 사용한 경우에 대하여 비교한다. 앞에서 제시한 파라메터들을 적용하여 LRU, LFU, FIFO 캐시 대치 알고리즘에 따른 프록시 서버의 디스크 크기와 히트율의 관계를 시뮬레이션 하였다(그림 8 참조). 시뮬레이션 결과는 LFU 방법이 다른 두가지 방법에 비해 히트율이 높은 것으로 나타났다. VOD 서비스는 최신작이나 인기 프로그램에 대한 요구가 증가하는 특징을 가지므로 캐시 대치 알고리즘에서 프로그램이 재생된 시각을 비교하는 것보다 프로그램이 재생된 빈도를 비교하는 것이 더 타당한 것임을 이 결과를 통해 알 수 있다. 또한, 프록시 서버의 하드 디스크 크기가 커질 수록 히트율이 증가하는 것을 알 수 있고 프록시 서버에 설치할 디스크 용량도 추정할 수 있다. 사용자는 프로그램 그룹 I인 최신 영화를 가장 많이 선호할 것이다. 프로그램 그룹 I의 서비스 요구 확률이 85%라고 가정하였다면 히트율이 85%에 근접하는 디스크 사이즈를 프록시 서버의 용량으로 결정하면 효과적이다.

디스크 용량의 변화에 따라 첫 화면을 수신하는 응답 시간도 LFU 방법이 다른 두가지 방법에 비해 작은 것으로 나타났다(그림 9 참조). 세가지 캐시 대치 방법의 결과는 프로그램이 hit한 경우와 miss한 경우를 모두 고려한 평균 응답 시간을 나타내며 히트율이 100%인 경우는 프록시 서버에 요청한 프로그램이 모두 존재하는 경우의 응답 시간이다. 첫 화면 수신 응답 시간은 프록시 서버를 사용하는 경우가 사용하지 않는 경우 보다 응답 시간이 큰 것으로 나타났다. 이러한 현상은 프록시 서버를 사용하는 경우에도 사용자와 처음 접속되는 것은 서버이고, 서버에서 프록시 서버에게 사용자와의 연결을 넘겨주므로 이에 대한 오버헤드가 발생하여 나타나는 것이다.

그러나, 사용자와 프록시 서버 사이에 이미 연결된 상태에서 스트림 제어(빨리감기, 되감기, 일시정지, 등)에 대한 응답시간을 비교하면 차이를 볼 수 있다(그림 10 참조). 세가지 캐시 대치 방법의 결과는 프로그램이 hit한 경우와 miss한 경우를 모두 고려한 평균 응답 시간을 나타내며 히트율이 100%인 경우는 프록시 서버에 요청한 스트림이 모두 존재하는 경우의 응답 시간이다. 프록시 서버를 사용하는 경우가 비록 히트율이 낮아도 프록시 서버를 사용하지 않는 경우보다 스트림 제어에 대한 응답시간이 작은 것으로 나타난다. 요청한 비디오 스트림이 프록시 서버에 있으면 프록시 서버가 스트림을 사용자에게 넘겨주고, 없으면 서버로 부터 받아서 넘겨주므로 히트율에 따른 응답 시간의 차이가 발생한다. 일반적으로 사용자들

은 첫 화면에 대한 응답 시간보다 영화 시청중 요청하는 스트림 제어에 대한 응답 시간에 대하여 더 민감하다고 볼 수 있으므로, 프록시 서버를 이용하여 사용자에게 스트림 제어에 대한 빠른 응답 시간을 제공하는 것이 효과적이다.

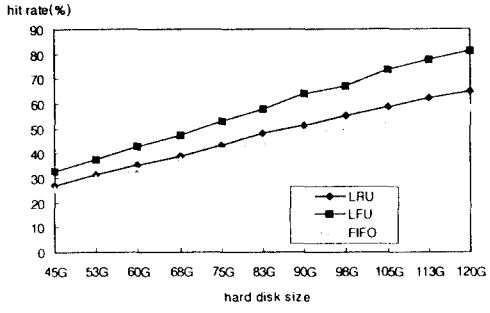


그림 8. 하드 디스크 크기와 히트율의 관계
Fig. 8 Hard disk size vs Hit rate

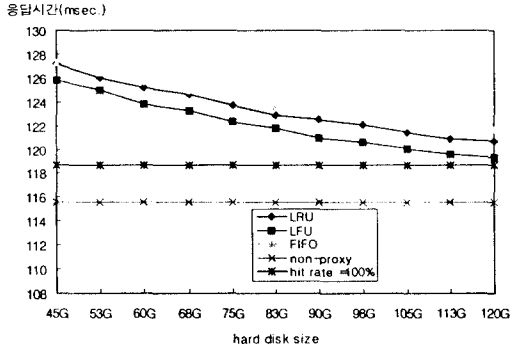


그림 9. 하드 디스크 크기와 첫 화면 응답시간의 관계
Fig. 9 Hard disk size vs The fist response time

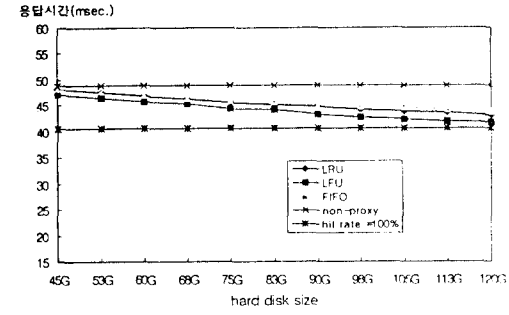


그림 10. 스트림 제어에 대한 응답시간
Fig. 10 Hard disk size vs The response time of stream controls

서버의 부하의 정도에 따라 초기 화면 응답 시간의 변화에 대하여 살펴본다. 앞 그림 9에서 나타난 것 같이 초기 화면 응답 시간은 프록시 서버를 사용하지 않는 경우가 빠르다. 그러나 사용자의 요구가 증가하게 되면 서버의 부하가 높아지고 처리 속도도 늦어지게 되어 응답 시간이 커지는 현상이 발생한다. 그림 11에서 서비스 요청 빈도, 즉 가입자당 하루 평균 요구 빈도가 낮을 경우 서버에서의 응답 시간이 빠른 것으로 나타났다. 그러나, 서비스 요청 빈도가 커지면 서버의 부하가 증가하여 프록시 서버를 사용하지 않는 것이 프록시 서버를 사용하는 경우보다 응답 시간이 커지는 것으로 나타났으며 과부하가 발생될 가능성을 보여준다. 프록시 서버를 사용하면 서버의 부하를 분담하여 많은 사용자를 수용할 수 있고, 히트율이 높을수록 응답 시간이 작은 것으로 나타났다.

스트림 제어에 대한 응답 시간의 변화에서도 프록시 서버를 사용하는 경우가 프록시 서버를 사용하지

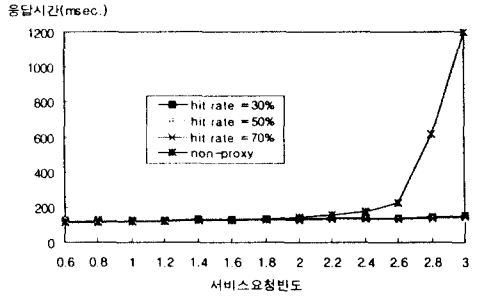


그림 11. 서비스 요청 빈도에 따른 첫 화면 응답시간
Fig. 11 First response time of first vs Service request rate

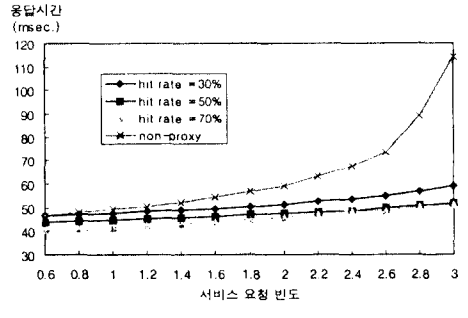


그림 12. 서비스 요청 빈도에 따른 스트림 제어 응답시간
Fig. 12 Response time of stream controls with service request rate

않는 경우보다 응답 시간이 작다. 프록시 서버의 히트율이 높을수록 응답 시간이 작고 사용자의 요청 빈도가 커지면 응답 시간이 급격히 커지는 것을 볼 수 있다.(그림 12 참조)

V. 결 론

본 논문에서는 DAVIC에서 권고하는 VOD 서버와 사용자 사이에 프록시 서버를 두어 사용자로부터 자주 액세스되는 스트림들에 대하여 캐싱 기능을 가지는 프록시 서버를 설계하고, 그 서비스 구조에 대해 논하고 그리고 성능을 분석하였다. 프록시 서버의 기능은 요구된 프로그램의 저장 여부를 나타내는 프로그램 캐싱와 메모리 안에 MPEG-TS의 존재 여부를 나타내는 메모리 캐싱로 나누어 볼 수 있다. 성능 분석 결과 프로그램 캐싱은 LFU 대치 알고리즘을 사용하는 것이 프로그램 히트율과 응답 시간면에서 효과적임을 보였다. 사용자들이 선호하는 프로그램이 접근 빈도가 높으므로 접근 시각보다 접근 빈도를 대치 기준으로 결정하는것이 타당하다. 메모리 캐싱의 경우 디스크와 메모리사이에 페이지 단위로 대치가 일어나므로, 재생하고자 하는 MPEG-TS가 메모리안에 없을 경우 MPEG-TS가 포함된 페이지를 메모리에 적재한다. 이 때 재생된 MPEG-TS를 대치하는 것이 타당하므로 FIFO가 적당한 방법이다.

본 고에서 제안하는 구조는 프록시 서버를 설치함으로 인하여 서버와 사용자에게 추가적인 기능을 최소화할 수 있다. 사용자들로 부터 지역적으로 근접한 곳에 프록시 서버를 두어 사용자가 첫 화면을 수신하는 응답 시간의 감소, 서버의 부하 분산 효과, 많은 사용자의 수용 가능성의 장점을 가짐을 보였다. 물론 프록시 서버 시스템을 설치하면 시스템 구축 비용이 증가한다. 그러나 서버 시스템의 부하가 줄어들므로 서버 시스템 설치 비용을 절감하는 이점도 있다. 시스템 구축 비용 측면 보다는 빠른 서비스 시간을 추구한다면 분산 구조를 도입해야 하는데 본 논문에서 밝힌바와 같이 프록시 서버형 분산 구조가 효과적이다.

향후 서버와 사용자 간의 네트워크 비용등을 고려하여 프록시 서버의 위치 결정, 수용할 사용자수 결정에 대한 비교 분석에 대한 연구가 필요하며 서버가 사용자와 가장 근접한 프록시 서버를 결정하는 방법과 프록시 서버간에 정보를 교환할 수 있는 메카니즘에 대한 연구가 진행되어야 할 것으로 생각된다.

참 고 문 헌

1. T.D.C. Little and D. Venkatesh, "Prospects for Interactive Video-on-Demand," IEEE Multimedia, Vol. 1, No. 3, pp. 14-24, 1994.
2. D. Deloddere, W. Verbiest, H. Verhile, "Interactive Video On Demand", IEEE Communication Magazine, Vol. 32, No. 5, pp. 82-88, May, 1994.
3. P. RangGan, H, Vin and S. Ramanathan, "Designing an On-Demand Multimedis Service," IEEE Communications, Vol. 30, No. 7, pp. 56-64, July, 1992.
4. Digital Audio-Visual Council, DAVIC 1.0 Specification, Geneva, 1995.
5. F.A. Tobagi et al., "Streaming RAID : A Disk Storage System for Video and Audio Files," Proc. ACM Multimedia 93, ACM Press, New York, pp. 393-400, 1993.
6. P.B. Berra, C-Y. R. Chen, A. Ghafoor, and T. D. C. Little, "Issues in Networking and Data Management of Distributed Multimedia Systems," Symposium on Hign-Perfromance Distributed Computing, Syracuse New York, September, 1992.
7. Asit Dan, Dinkar Sitaram and Perwez Shahabuddin, "Scheduling Policices for an On-Demand Video Server with Batching," ACM Multimedia 94, pp. 15-21, October, 1994.
8. Chatschik C. Bisdikian and Baiju V. Patel, "Cost-Based Program Allocation for Distributed Multimedia-on-Demand Systems," IEEE Multimedia Fall 1996, pp. 62-72, 1996.
9. M. M. Buddhikot, G. M. Parulkar, and J. K. Cox, "Design of a Large Scale Multimedia Storage Server," Computer Networks and ISDN Systems, Vol. 27, No. 3, pp. 503-517, 1995.
10. L. A. Rowe, et. al. "A Distirbuted Hierachical Video-on-Demand System," Proceedings of the International Conference on Image Processing,

Washington DC, October, 1995. *http://bmrc.berkerley.edu/papers/105/105.html.*

11. A. Luotonen and K. Altis, "World-Wide Web Proxies," Computer Network and ISDN systems, Vol. 27, No. 2, pp. 1-8, 1994.
12. 안 경아, 최 훈, "프록시 서버를 이용한 DAVIC VOD 시스템의 성능 분석," 내부문서, 충남대학교 컴퓨터공학과, September, 1997.
13. A. Alan B. Pritsker, *Introduction to Simulation and SLAM II*, Halsted Press, 1986.

안 경 아(Kyung-Ah Ahn) 정회원
1965년 9월 20일생
1988년 2월 : 경북대학교 통계학과 학사
1988년~1998년 : 한국전자통신연구원 SW공학연구부 근무
1998년~현재 : 충남대학교 컴퓨터공학과 재학
※주관심분야 : 분산 시스템, 컴퓨터 네트워크
e-mail : kaahn@comeng.chungnam.ac.kr

최 훈(Hoon Choi) 정회원
1960년 5월 19일생
1983년 2월 : 서울대학교 컴퓨터공학과 학사
1990년 12월 : Duke University 전산학과 석사
1993년 5월 : Duke University 전산학과 박사
1983년~1996년 : 한국전자통신연구원 광대역통신망연구부 근무
1996년~현재 : 충남대학교 컴퓨터공학과 조교수
※주관심분야 : 분산 시스템, Fault-tolerant 시스템, 컴퓨터 네트워크, Stochastic Petri Nets
e-mail : hchoi@comeng.chungnam.ac.kr