화상회의를 위한 효율적인 온-라인 프레임 스케줄링 알고리즘

(An Efficient On-line Frame Scheduling Algorithm for Video Conferences)

안 성 용 † 이 정 아 †† 심 재 홍 †††
(Seong Yong Ahn) (Jeong A Lee) (Jaehong Shim)

요 약 본 논문에서는 화상회의 시스템의 서비스 품질(QoS)을 최대화하기 위해 프레임 복원 태스크들에게 프로세서 시간을 효율적으로 분산하는 방안을 제시하고자 한다. 압축된 프레임은 복원에 소요된 프로세서 시간이 증가함에 따라 복원된 영상의 QoS 또한 증가하는 특성을 가진다. 따라서 각 프레임별 복원 영상의 품질은 복원을 위해 주어진 서비스 시간의 QoS 함수로 표현될 수 있다. 또한 각 화상회의 스트림은 연속된 프레임간 밀접한 시간 의존성을 가진다. 이러한 특성과 QoS 함수를 기반으로 본 논문에서는 총 QoS를 최대화하면서 모든 프레임이 아닌 일부 프레임들만 스케줄링하는 온-라인 프레임 스케줄링 알고리즘을 제안한다. 실험 결과 시스템 부하가 증가함에 따라 제안 알고리즘은 기존 EDF에 비해 보다 부드럽게 화질이 감소하고, 연속된 프레임들이 끊김 없이 출력되어 보다 자연스런 화자의 동작을 보일 수 있었다.

키워드 : 화상회의, 서비스 품질, 프레임 스케줄링

Abstract In this paper, we propose an algorithm that distributes processor time to the tasks decoding encoded frames with a goal maximizing total QoS(quality of services) of video conference system. An encoded frame has such a characteristic that the QoS of recovered frame image also increases as the processor time given for decoding the frame gets to increase. Thus, the quality of decoded image for each frame can be represented as a QoS function of the amount of service time given to decode. In addition, every stream of video conference has close time-dependency between continuous frames belonging to the same stream. Based on the time-dependency and QoS functions, we propose an on-line frame scheduling algorithm which does not schedule all frames in the system but just a few frames while maximizing total QoS of video streams in the conference. The simulation results show that, as the system load gets to increase, the proposed algorithm compared to the existing EDF algorithm can reduce the quality of decoded frame images more smoothly and show the movements of conference attendees more naturally without short cutting.

Key words : Video conference, quality of service, frame scheduling

1. 서 론

인터넷의 급속한 확산과 통신 속도의 향상에 따라 영상이나 음성 등의 연속 미디어(continuous media) 서비스가 급증하고 있다. 이러한 연속 미디어 서비스로서 최근 관심이 집중되고 있는 대표적인 것이 인터넷을 통한 VOD 서비스나 영상전화, 화상회의 시스템 등이다. 연속 미디어 서비스 중에서도 VOD 서비스와 같은 단방향 서비스와는 달리 영상전화나 화상회의 시스템과 같은 양방향 서비스는 송수신측 모두 미디어를 생성하고 전송하며, 이와 동시에 수신된 미디어를 복원하고 출력한다. 화상회의 시스템은 서로 다른 원격지에 위치한 다수의 화자가 인터넷을 통하여 상호간 영상과 음성을 실시간으로 주고받는 서비스이며, 다양한 시간적 제약을 가진다. 영상 프레임의 생성 또는 출력(display) 주기인

† 비 회 원 : 조선대학교 전자계산학과
dis@chosun.ac.kr
†† 종신회원 : 조선대학교 전자계산학과 교수
jalee@chosun.ac.kr
††† 비 회 원 : 조선대학교 인터넷소프트웨어공학부 교수
jhshim@chosun.ac.kr
논문접수 : 2003년 11월 5일
심사완료 : 2004년 4월 13일

프레임 율(frame rate)이 초당 15, 24, 또는 30 프레임 등으로 기준이 정해져 있고, 송신측에서 압축되어 전송된 프레임이 수신측에서 복원되어 출력되기까지의 시간인 출력지연(display delay)에 대한 제한시간도 있다. 이러한 시간제약을 위반할 경우 출력되는 영상의 품질에 상당한 영향을 미치게 된다. 연속 미디어 서비스에서는 이러한 프레임 율을 준수하고 출력지연을 단축시키기 위해 다양한 기술을 적용하고 있다[1-3].

인터넷을 통한 화상회의 시스템의 기본적인 요구사항 중의 하나는 화상회의에 참여하는 화자(스트림)의 수가 늘어나거나 시스템의 성능저하 등으로 인해 모든 스트림별 정해진 수의 프레임을 주어진 시간 내에 복원하여 출력할 수 없을 경우, 이를 효과적으로 제어하고 적절한 QoS(quality of service)를 제공할 수 있는 기법을 요구한다. 이 경우 대처방안의 하나로 현재의 시스템 성능에서 처리 가능한 수의 프레임을 제외한 나머지 프레임은 버리는 것이다. 그러나 이 방법은 버퍼 범람(overflow)이 발생한 경우와 같이 출력되는 영상들의 끊김 현상이 자주 발생하게 된다.

시스템 성능을 향상시키고 보다 질 좋은 QoS를 제공할 수 있는 또 다른 방법은 각 프레임에 대한 처리시간을 절약하는 방법이다. 각 프레임에 대해 충분한 처리시간을 제공할 때 보장되는 최상의 화질은 얻을 수 없을지라도 각 프레임에 대한 처리시간을 조금씩 줄여 수용할 만한 화질을 제공받을 수 있다면, 시스템이 처리 가능한 프레임 수를 상대적으로 늘릴 수 있다. 이렇게 하려면 프레임 복원작업은 필요에 따라 복원을 완결하기 전에라도 강제로 중단될 수 있어야 한다. 이때 줄여진 프레임 처리시간은 그렇지 않으면 버려졌을 또 다른 프레임을 처리하는데 사용될 수 있다. 다행히 몇몇 영상 압축/복원 알고리즘에서 부분적으로 영상을 처리함으로써 수용할 만한 QoS를 제공하는 것이 가능해졌다[4,5].

부분적으로 작업을 수행하는 이러한 아이디어는 멀티미디어 응용[5], 정보 수집 및 추출, 데이타베이스 질의 처리, 자동 항법 계획, 실시간 휴리스틱(heuristic) 탐색, 의료 결정 시스템, 전통적인 반복적인 수치 알고리즘 계산 등과 같은 여러 응용에서 활용되어져 왔다[6]. 이러한 응용들에게 소속된 태스크들은 그들의 실행 결과로 생성되는 QoS를 가지게 되는데, 이는 만기 전까지 각 태스크에게 주어진 서비스 시간(실제 실행시간) 양의 함수 값으로 표현된다. 이 함수를 해당 태스크의 QoS 함수(QoS function)라 하며, 점진적 증가 볼록(strictly increasing concave) 함수 특성을 가진다. 즉, 각 태스크에게 주어지는 서비스 시간이 증가할수록 해당 태스크가 얻는 QoS 또한 증가한다. 이러한 태스크를 IRIS(Increasing Reward with Increasing Service) 태스크라 한다[6,7].

IRIS 태스크 모델은 기존의 부정확 계산 모델(imprecise computation model)[8,9]과는 다음과 같은 측면에서 차이를 가진다. 부정확 계산 모델은 하나의 태스크가 다시 필수 서브태스크(mandatory subtask)와 선택 서브태스크(optional subtask)로 분리되며, 필수 서브태스크는 태스크의 만기(deadline) 이전에 반드시 실행되어야하며 선택 서브태스크는 다른 필수 서브태스크의 만기에 영향을 미치지 않는 범위 내에서 선택적으로 실행될 수 있다. 각 서브태스크의 실행시간은 사전에 알려져 있어야 하며, 태스크의 QoS 함수는 일반적으로 선형(linear) 함수를 가정하고 있다. 선택 서브태스크의 실행되지 않은 부분의 실행시간을 그 태스크의 에러(error)라 불리우며, 총 에러의 최소화[10-12], 최대 에러의 최소화[13-15], 평균 에러의 최소화[16], 평균 응답시간의 최소화[17]를 위한 다양한 알고리즘들이 제안되었다.

이에 비해 IRIS 태스크 모델은 필수/선택 서브태스크들의 분리가 필요하지 않으며, 태스크의 실행시간 또한 사전에 결정되는 것이 아니라, 스케줄러에 의해 동적으로 결정된다. 또한 태스크의 QoS 함수는 점진적으로 증가하는 볼록형 함수이다. 따라서 스케줄러는 총 QoS의 최대화를 위해 모든 태스크들의 만기를 넘기지 않는 범위 내에서 각 태스크에게 할당할 서비스 시간을 동적으로 결정해야 한다[6,7,18].

화상회의 시스템의 경우 각 화자에게 제공되는 QoS는 주로 해당 스트림의 각 프레임에게 할당된 프로세서 시간(실행시간)의 양에 의존한다. 즉, 영상 복원작업에 주어지는 실행시간이 증가함에 따라 그 실행결과로 얻어지는 영상의 품질 또한 함께 증가된다. 그러나 기존의 어떠한 연구도 각 프레임의 처리 단계를 어떻게 나누고, 각 프레임에게 얼마만큼의 실행시간을 할당하고, 화상회의 시스템의 QoS를 어떻게 최대화할 수 있는지에 대한 방안을 제시하지 못했다.

본 논문에서는 화상회의 시스템의 QoS를 최대화하기 위해 프레임 복원 태스크들에게 프로세서 시간을 효율적으로 분산하는 방안을 제시하고자 한다. 프레임을 복원하는 각 태스크는 하나의 IRIS 태스크로 모델링 된다. 하나의 스트림은 연속된 프레임간 밀접한 시간 의존성을 가지며, 이러한 특성을 기반으로 하여 본 논문에서는 모든 프레임이 아닌 일부 프레임들만 스케줄링하여 그들의 실행시간을 결정하는 온-라인 프레임 스케줄링 알고리즘을 제안한다. 매 스케줄링 시점에서 스케줄링되는 프레임들의 수를 줄임으로써 알고리즘의 평균 계산 복잡도를 줄일 수 있으며, 또한 주어진 시스템 성능에 대하여 화상회의에 참여한 모든 화자 영상의 총체적 화질을 향상시킬 수 있음을 모의실험을 통해 확인하였다.

본 논문은 다음과 같은 공헌도를 가진다. 먼저 복원되는 프레임 영상의 품질을 지수함수 형태의 QoS 함수로 맵핑할 수 있음을 보이고, 영상 복원 태스크를 IRIS 태스크로 모델링 하였다. 또한 인터넷 화상회의 시스템에 적용 가능한 온-라인 프레임 스케줄링 알고리즘을 제안하고, 복원영상의 품질측정을 위한 몇 가지 측정요소를 제안한다.

본 논문의 구성은 다음과 같다. 2절에서는 화상회의 시스템에서 화자 영상에 대한 스트림 특성과 각 스트림의 QoS 함수 도출 방안에 대해 기술하고, 3절에서 화상회의 시스템의 QoS를 최대화하는 효율적인 온-라인 프레임 스케줄링 알고리즘을 제안한다. 4절에서는 제안 알고리즘을 적용한 실험결과를 보이고, 마지막 5절에서 결론을 맺는다.

2. 화상회의 프레임의 특성과 QoS 함수

인터넷을 통해 화상회의 서비스를 지원할 경우 동영상은 제한된 통신망 속도에 비해 과대한 대역폭을 요구하므로 어떤 방법으로든 영상을 압축할 필요가 있다. 영상 압축 알고리즘은 크게 정지영상과 동영상을 기반으로 하는 두 종류가 있다. 동영상을 기반으로 하는 대부분의 압축 알고리즘은 연속된 두 영상의 차이를 활용하는 기술이며, 가장 대표적인 것으로 MPEG[19]을 들 수 있다.

일반적으로 인간은 영상의 저주파 성분보다는 고주파 성분에 더 둔감하다. 정지 영상을 기반으로 하는 압축 알고리즘은 이러한 인간의 시각적 특성을 이용한다[20]. 즉, 저주파에서 고주파에 이르기까지 영상을 주파수 성분별로 분해하고 시각에 둔감한 고주파 성분을 제거하여 정보량을 줄임으로써 데이타 압축 효과를 거둘 수 있다. 이러한 방법을 이용하는 압축 방안에는 JPEG[21]과 Wavelet[4] 등이 있다. 두 방안의 차이점으로 Wavelet은 JPEG에 비해 영상의 주파수 성분을 서브밴드로 나누어 별도로 처리함으로써 서브밴드별로 화질을 조절할 수 있다는 장점을 가진다. 이때 각 서브밴드를 말릿블록(Mallat Block)이라 한다. 일반적으로 Wavelet은 하나의 주어진 프레임에 대하여 다섯 처리 단계를 거쳐 14개의 말릿블록을 생성한다.

그림 1은 말릿블록 수를 하나씩 증가하면서 복원된 영상들을 보여준다. 그림에서는 최고주파 서브밴드를 위한 두개의 말릿블록을 제외한 12개의 말릿블록에 대한 영상만을 표시하였다. 압축된 하나의 프레임에 대하여 각 말릿블록별로 데이타를 처리하여 순서적으로 영상을 복원할 수가 있는데, 그림은 12개의 말릿블록에 대해 저주파부터 고주파 성분으로 단계적으로 말릿블록을 하나씩 증가하면서 복원한 영상을 보이고 있다. 그림에서 복원에 사용된 말릿블록의 수가 증가함에 따라 영상의 화질도 함께 증가함을 확인할 수 있다. 주목할 만한 것은 여섯 개의 말릿블록만을 이용해 복원된 영상일지라도 납득할 만한 화질을 제공한다는 것이다.

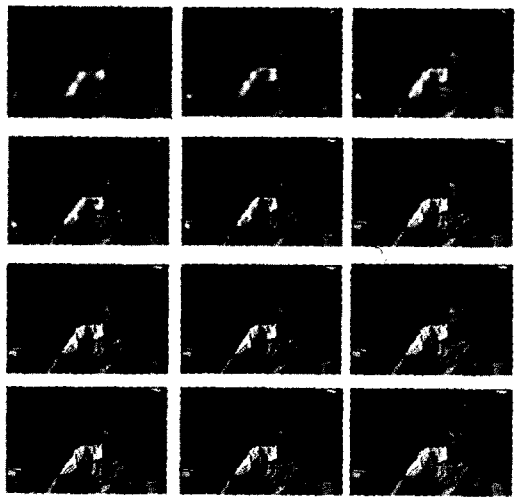

그림 1 말릿블록 수를 하나씩 증가하면서 복원한 영상

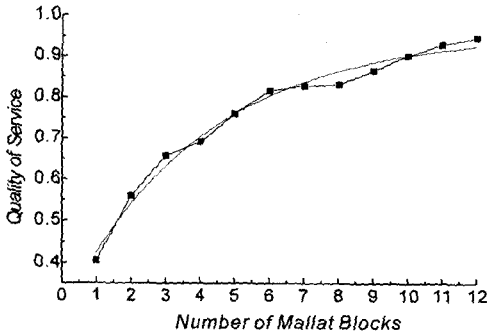

그림 2 말릿블록 수의 증가에 따른 영상의 화질 변화

그림 2는 그림 1에 보인 각 영상의 화질을 QoS 함수 값으로 표현할 경우 말릿블럭 수의 증가에 따른 영상의 화질변화를 보여 준다. 이때 사용된 QoS 함수는 $(S-N)/S = 1 - N/S$ 이다. 여기서 N/S는 원영상과 복원영상의 신호 대 잡음비의 역수로서 S는 신호성분을, N은 잡음성분을 표현하고 있다. S는 원영상의 신호성분으로서 영상이 정해지면 함께 결정되는 상수이며, N은 말릿블록의 수에 따라 변하는 변수로서 말릿블록의 수가 증가함에 따라 감소하는 특성을 가진다. 즉, 화질이 좋아짐에 따라서 N 값은 감소하게 된다. 그림에서 보는 바와 같이 영상의 화질변화는 말릿블록 수의 증가에 대하여 점진적으로 증가하는 볼록한 형태의 그래프로 나타나고 있다.

Wavelet 방식으로 압축된 프레임을 복원하는 알고리즘을 저주파 성분으로부터 고주파 성분으로 단계적으로 14개의 말릿블록들을 복원하도록 수정할 경우, 프로세서 시간이 부족하여 만기 전까지 완벽하게 프레임 복원을 완료하지 못할지라도 그때까지 복원된 프레임의 중간영상을 출력할 수 있다. 이렇게 수정된 Wavelet 복원 알고리즘을 사용할 경우, 하나의 프레임을 복원하는데 보다 많은 실행시간이 주어질수록 보다 우수한 화질의 프레임 영상을 재생할 수 있다.

프레임 복원에 사용되는 말릿블록 수의 증가는 처리시간의 증가로 표현할 수 있으므로, 프레임의 화질은 만기 전까지 프레임 복원을 위해 주어진 프로세서 시간의 양(x)의 함수 값 $f(x)$로 표현할 수 있다. 이때의 함수 $f(x)$를 그 프레임의 QoS 함수라 한다. 복원된 영상의 품질은 주어진 처리시간의 증가에 따라 점진적으로 증가하는 볼록한 형태(strictly increasing concave)를 가지므로, QoS 함수 그래프는 2차 함수나 지수함수로 표현할 수 있다. 본 논문에서는 그림 2의 그래프를 $f(x)=y-ae^{-(x-x0)/t}$의 지수함수로 모델링 하였으며, 그 결과 $f(x)=0.97-0.5e^{-0.25(x-1)}$을 얻었다. 이는 시간에 대한 위상 천이를 무시하면 $f(x)=1-ae^{-wx}$의 형태로 표현되며, a 및 w의 값에 따라 최소값으로 0.2 ~ 0.5의 수치를 가지고 최대값 1에 접근하는 래프로 묘사할 수 있다.

화상회의 시스템에서 다루는 영상의 특성 중 주목할 만한 것은 하나의 스트림 내의 연속된 프레임간의 변화가 크지 않다는 것이다. 즉, 대부분의 경우 영상의 배경은 변화하지 않으며 배경 안에서의 화자만이 매우 느린 속도로 움직이는 경우가 대부분이다. 그림 3은 Wavelet 알고리즘에 의하여 복원된 동일한 스트림 내의 여러 프레임들을 보여준다. 그림 4는 그림 3의 각 프레임별로 말릿블록 수의 증가에 따르는 영상의 화질 변화를 보여

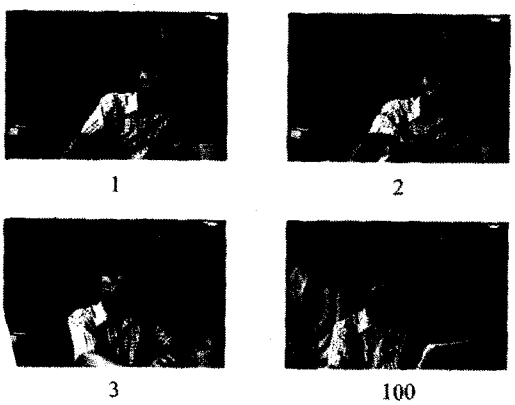

그림 3 동일 스트림에 속한 서로 다른 프레임 영상들

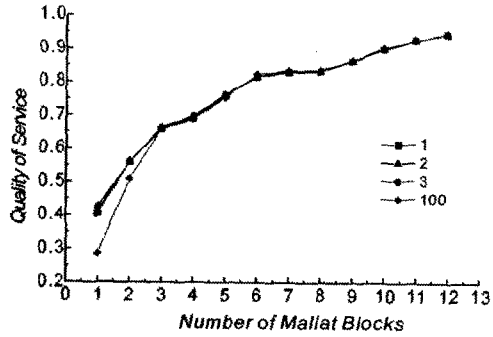

그림 4 동일 스트림의 서로 다른 프레임들에 대한 영상의 화질 변화

준다. 그림에서 프레임들의 화질은 거의 유사하다는 것을 확인할 수 있다. 이는 동일 스트림 내의 서로 다른 프레임들의 복원을 위하여 동일한 복원시간을 할당하였을 경우 거의 비슷한 수준의 화질을 제공할 수 있음을 의미하며, 또한 동일한 스트림 내의 서로 다른 프레임에 대하여 동일한 QoS 함수를 이용할 수 있음을 의미한다.

3. 화상회의 스트림을 위한 프레임 스케줄링

화상회의에 참여하는 사람(화자)의 수가 증가하거나 (화면에 보여야 하는 스트림들의 수가 증가) 또는 시스템에 일시적인 과부하가 발생할 경우 시스템에 도착한 모든 프레임들을 완벽하게 복구하는 것이 어려울 수 있다. 이 경우 화질은 약간 떨어지더라도 가능한 많은 프레임들을 출력시켜서 화자들의 움직임이 자연스럽게 보이게 하는 방법과 움직임은 약간씩 끊어져 자연스럽지 못해도 선명한 영상을 출력시키는 방법이 있다. 불행이도 이 두 방법은 동시에 만족될 수 없는 상호 배타적이다. 즉, 보다 선명한 영상을 위해 보다 많은 실행시간을 할당하는 대신 일부 프레임들은 버려야 하며, 반대로 보다 자연스러운 동작을 위해선 실행시간을 줄이는 대신 저화질의 보다 많은 프레임을 출력해야 한다. 시스템에 과부하가 발생할 경우 얼마 정도의 프레임을 누락시키고 어떤 정도의 화질로 영상을 출력할 것인지를 결정하는 것은 매우 어려운 일이다. 이런 문제를 해결할 수 있는 한 가지 대안은 프레임들에게 상대적으로 부족한 프로세서 시간을 적절히 분산함으로써 프레임들이 획득한 QoS들의 총합이 최대화되도록 하는 것이다.

본 절에서는 화상회의 시스템의 QoS를 최대화하면서 제한된 프로세서 시간을 적절히 분산할 수 있는 방안을 제시하고자 한다. 프레임 스케줄링 알고리즘의 목적은 프레임들의 총 QoS를 최대화시키는 것이며, 이를 위해 각 프레임의 복원에 할당할 실행시간을 결정하고, 또한 그들의 실행 순서를 결정해야 한다. 그러나 프레임들이

언제 도착하고 그들의 만기가 얼마인지 사전에 알려져 있지 않으므로, 최적(optimal)의 총 QoS를 생성하도록 스케줄링한다는 것은 매우 어렵다(NP-hard). 따라서 본 논문에서는 더 이상의 프레임들이 도착하지 않는다는 가정 하에서 현재 시스템에 도착해 있는 프레임들을 상대로 총 QoS가 최대화되도록 스케줄링하는 온-라인 프레임 스케줄링 알고리즘을 제안한다.

3.1 프레임 스케줄링 문제

N명의 화자가 참석한 화상회의 시스템을 고려해 보자. 각 화자의 시스템에는 단일화면에 N개의 화자 영상이 출력된다. 각 화상회의 스트림(각각의 화자)은 동일하거나 또는 서로 다른 시스템에서 생성되어 인터넷을 통하여 전송되어진다. 수신된 각 스트림은 Wavelet으로 압축된 연속된 프레임들로 구성된다. 프레임들은 송신시스템에서 주기적으로 생성되어 압축된 상태로 목적시스템에 보내지며, 수신시스템에서는 이를 복원하여 주기적으로 화면에 출력한다. 인터넷의 전송지연으로 인하여 프레임의 도착시간과 출력 만기는 불규칙적이고 비주기적이다.

앞 절에서 언급했듯이 화상회의용 스트림들은 동일한 스트림 내의 모든 프레임들이 거의 동일한 QoS 함수를 가지므로, 스트림당 하나의 QoS 함수를 가진다. 스트림의 QoS 함수는 선형함수(linear function) 형태가 아닌 점진적으로 증가하는 볼록한 형태(monotonically increasing concave)를 가진다. 따라서 실행시간을 많이 받으면 받을수록 그 프레임이 획득하는 QoS는 점점 증가하게 된다. 그러나 실행시간이 증가할수록 QoS 증가율은 점점 감소하게 된다. 스트림 S_i(또는 그 스트림에 속한 각 프레임)의 QoS 함수를 $f_i(x)$로 표현하자. QoS 함수 $f_i(x)$의 도함수를 $g_i(x)$로 표현하고, $g_i(x) = df_i(x)/dx$라 정의할 때, 도함수 $g_i(x)$는 $f_i(x)$가 점진적 증가 볼록형이므로 점진적 감소 오목형(monotonically decreasing convex) 함수이다. 도함수 $g_i(x)$의 역함수는 $g_i^{-1}(y)$로 표현하며, $g_i(0)$를 임의의 스케줄링 시점(τ_0)에서의 순간 가치 증가율이라 한다.

위와 같은 시간적 제약과 QoS 함수를 가진 프레임을 처리(복원)하는 태스크는 IRIS 실시간 태스크[6]로 모델링될 수 있다. IRIS 태스크는 임의의 도착시간과 만기를 가지며, 실행시간이 사전에 고정된 것이 아니라 스케줄러에 의해 동적으로 결정된다. 주어진 실행시간이 증가함에 따라 해당 태스크 출력의 QoS 또한 증가하는 특성을 가진다. 따라서 인터넷 화상회의 시스템에서 수신된 프레임 스케줄링을 위한 알고리즘으로 총 가치(total reward or QoS)를 최대화시키는 IRIS 태스크용 온-라인 스케줄링 알고리즘들을 사용할 수 있다[6,22,7]. 이들 알고리즘은 상하계층으로 분리된 두 개의 알고리즘으로 구성된다. 상위계층 알고리즘은 태스크들의 만기를 고려하여 최대화된 총 가치를 생성할 수 있도록 태스크들에게 할당할 실행시간을 결정한다. 하위계층 알고리즘은 태스크들의 실행순서(프로세서의 할당순서)를 결정한 후, 각 태스크에게 상위계층에서 결정된 실행시간만큼만 프로세서를 할당한다. 하위 계층 알고리즘으로 주로 EDF(earliest deadline first)[23] 알고리즘을 사용한다.

상위계층 알고리즘은 더 이상의 태스크들이 도착하지 않는다는 전제 하에 태스크들의 만기를 고려하여 실행시간을 결정하기 때문에 새로운 태스크가 도착할 때마다 총 QoS를 최대화하기 위해 태스크들에게 할당할 실행시간을 다시 결정한다. 이처럼 총 가치를 최대화하는 상위계층 스케줄링 문제는 다음과 같은 온-라인 최적화 문제와 동일하다: "만기(τ_i)와 가치함수($f_i(x_i)$)가 주어진 n개의 태스크들이 시스템에 존재한다고 가정할 때, 새로운 태스크가 도착할 때마다 태스크들의 총가치가 최대화($max(\sum_{i=1}^{n} f_i(x_i))$)되도록 각 태스크에게 할당할 실행시간($x_i$)을 결정하라." 이 문제를 해결하기 위해 기존의 연구에서 서로 다른 방법들이 제시되었다[6,22,7]. 이들 상위계층 알고리즘들의 계산 복잡도는 모두 $O(n^2)$이다. 따라서 이들을 많은 화자가 참여하는 인터넷 화상회의 시스템에 적용하기에는 계산 복잡도가 너무 높다.

화상회의용 프레임을 처리하는 태스크와 일반적인 IRIS 태스크간의 명백한 차이는 화상회의의 경우 태스크들간의 실행순서에 상당한 의존성이 존재하는 것이다. 기존 연구에서 각각의 IRIS 태스크는 도착시간과 만기 등에서 다른 태스크들과 독립적이라고 가정한다. 이는 곧 태스크들이 임의의 시간에 도착하고 소멸함을 의미한다. 이에 비해 화상회의용 태스크들은 다음과 같은 특징을 가진다. 첫째, 동일한 스트림(화자)에 포함된 프레임들은 송신측에서 보내진 순서와 동일하게 수신측에서 출력되어야 한다. 이는 인터넷의 과부하 및 패킷 충돌로 인해 프레임들이 임의 순서로 도착해도 지켜져야 한다. 둘째, 동일한 스트림의 연속된 프레임 영상들은 주기적으로 화면에 출력되어야 하므로 이들을 처리하는 태스크들의 만기는 시간 축에서 일정한 간격으로 분포되어 있다.

위와 같이 연속하여 빠른 주기로 도착하는 프레임들의 특성과 다수의 화자가 참여하는 인터넷 화상회의 시스템의 특성을 면밀히 분석해 보면, 거의 대부분 이전 스케줄링 시점에서 스케줄링된 프레임들이 모두 서비스되기(복원 완료) 전에 새로운 프레임들이 도착한다. 이 경우 기존 알고리즘들은 재 스케줄링을 실시하여 모든 태스크들에게 할당할 실행시간을 다시 결정한다. 따라서

스케줄링 복잡도를 줄이기 위해서는 매 스케줄링 시 모든 프레임들의 실행시간을 결정하지 않고, 만기를 고려한 QoS 증가율이 큰 가장 몇 개의 프레임들에 대해서만 실행시간을 결정하는 것이 보다 효율적이며, 그 만큼 스케줄링 오버헤드도 줄일 수 있다. 다음 절에서 제안하는 알고리즘은 이러한 아이디어를 기반으로 하고 있다.

3.2 온-라인 프레임 스케줄링 알고리즘

각 프레임이 수신시스템에 도착할 때마다 이를 처리하는 하나의 태스크(t_i)가 준비되고(released), 모든 태스크들이 만기 순서로 정렬(즉, $\tau_i < \tau_{i+1}$)되어 있다고 가정하자. 그림 5는 더 이상의 태스크들이 도착하지 않는다는 가정 하에서 현재 시스템에 도착해 있는 태스크들을 상대로 총 QoS가 최대화되도록 스케줄링하는 온-라인 프레임 스케줄링(on-line frame scheduling: OFS) 알고리즘을 보여준다. 제안 알고리즘 역시 기존 알고리즘과 같이 상하계층으로 구성되며, 하위계층 알고리즘으로 EDF를 사용한다. 제안 알고리즘은 매 스케줄링 시점에서 실행된다. 여기서 스케줄링 시점이란 새로운 태스크가 시스템에 도착한 시점 또는 새로운 태스크가 도착하기 전에 이전 스케줄링 시점에서 스케줄링된 태스크들의 실행을 모두 완료한 시점을 의미한다. 알고리즘에서 첨자 i, k 등은 해당 태스크 리스트 내에서 태스크들의 만기 순서를 의미한다.

총 QoS를 최대화하기 위해 우리는 임의의 스케줄링 시점(τ_0)에서 각 태스크에게 만기를 넘기지 않는 범위 내에서 임의의 실행시간(y_i)을 할당할 수 있고, 이를 이용해 실행시간이 할당된 모든 태스크들 중 가장 큰 가치 도함수 값, 즉 $max_{i=1,\cdots,n}\{g_i(y_i) | \sum_{i=1}^{k} y_i \le \tau_k - \tau_0, \forall k=1, 2,\cdots,n\}$을 구할 수 있다. 여기서 n은 현재 시스템에 도착해 있는 프레임(태스크)들의 개수이다. 모든 태스크들에게 매번 다른 임의의 서비스 시간을 할당할 수 있고, 그때마다 가장 큰 가치 도함수 값을 구할 수 있다. 이러한 최대 도함수 값들 중 최소값을 $\phi(\tau_0) = min\{max_{i=1, \cdots, n}\{g_i(y_i)\}\}$라 표현한다. $\phi(\tau_0)$는 모든 태스크들의 만기를 준수하면서 현 스케줄링 시점에서 획득 가능한 가장 큰 QoS 증가율이다. 이 값을 각 프레임의 도함수의 역함수의 매개변수로 사용하여 해당 프레임에게 할당될 실행시간을 구할 수 있다.

Let T be a list of tasks in the system and sorted in increasing order of their deadlines
Let τ_0 be the current time;
(1) Get $\phi(\tau_0)$, where $\phi(\tau_0)$ is the min. max. reward derivatives for the tasks in T
(2) Get a task list $S = \{t_i \mid g_i(0) \ge \phi(\tau_0), t_i \in T\}$;
(3) Get service time $y_i = g_i^{-1}(\phi(\tau_0))$ for all $t_i \in S$
(4) Select k's tasks such that $\sum_{i=1}^{k} y_i = \tau_k - \tau_0$, for $t_i \in S$
Set the next scheduling point to τ_k
(5) Schedule k's tasks using EDF;
Execute each scheduled task for y_i time units;

그림 5 제안 OFS 알고리즘

단계 (1)에서 $\phi(\tau_0)$를 찾는다. $\phi(\tau_0)$를 찾기 위해 계산 복잡도가 낮은 이분법(bisection method)[10]을 적용하였다. $\phi(\tau_0)$를 결정하면 단계 (2)에서 현 스케줄링 시점(τ_0)에서의 순간 가치 증가율 $g_i(0)$가 $\phi(\tau_0)$보다 같거나 큰 태스크들의 리스트 S를 구할 수 있다. 단계 (3)에서는 리스트 S의 모든 태스크들에 대해 할당 가능한 실행시간(y_i)을 계산한다. 이때 y_i는 $\phi(\tau_0)$를 매개변수로 사용하는 태스크 t_i의 도함수의 역함수($g_i^{-1}()$)를 이용하여 구한다. 단계 (4)에서는 리스트 S 내의 모든 태스크들의 실행시간 y_i에 대해 조건 $\sum_{i=1}^{k} y_i = \tau_k - \tau_0$을 만족하는 $k(1 \le k \le n)$개의 태스크들을 선택한다. 또한 선택된 마지막 태스크의 만기 τ_k를 다음 스케줄링 시점으로 설정한다. 만약 새로운 태스크가 다음 스케줄링 시점 τ_k 이전에 도착하면 도착한 그 시점이 다음 스케줄링 시점이 된다. 제안 알고리즘은 단계 (4)에서 선택된 단지 k개의 태스크에 대해서만 실행시간(y_i)을 최종적 확정 짓는다. 실행시간이 확정된 k개의 스케줄링된 태스크들은 하위 계층인 단계 (5)의 EDF 알고리즘에 의해 그들의 실행순서가 결정되고 실제 프로세서가 할당된다. 이때, 태스크들은 만기 전까지 단계 (4)에서 확정된 실행시간 이상을 수행할 수 없다.

만약 다음 스케줄링 시점 τ_k까지 스케줄링된 k개의 태스크들이 모두 실행을 완료할 동안 새로운 태스크가 도착하지 않을 경우, 제안 알고리즘은 재 스케줄링을 실시하여 또 다른 k(이전 스케줄링 시점에서의 k와는 다름)개의 태스크들에 대해 실행시간을 결정한다. 시스템 내의 모든 태스크들의 실행을 완료할 때까지 새로운 태스크가 전혀 도착하지 않는 극단적인 경우에, 제안 알고리즘은 반복하여 재 스케줄링을 실시하게 되며, 궁극적으로 시스템 내의 모든 태스크들에 대해 그들의 실행시간(y_i)을 결정하게 될 것이다.

제안 알고리즘은 참고문헌 [22]과 동일하게 $\phi(\tau_0)$를 찾기 위해 계산 복잡도가 낮은 이분법을 사용한다. 그러나 참고문헌 [22]에서 제시된 알고리즘은 [6,7]의 알고리즘과 마찬가지로 매 스케줄링 시점에서 모든 태스크들에 대해 그들의 실행시간을 결정한다. 따라서 이들의 계산 복잡도는 $O(n^2)$이다. 그러나 제안 알고리즘은 시스템 내의 모든 태스크가 아니라, 단지 k개의 태스크에 대해서만 실행시간을 결정한다. 이는 앞서 기술한 화상회의 프레임들의 특성을 기반으로 하고 있다.

태스크 리스트 T와 S에 존재하는 태스크들이 기존

연구에서처럼 그들의 만기 순서로 적절히 관리된다고 가정할 경우, 매 스케줄링 시점에서 제안 알고리즘의 계산 복잡도는 $O(n)$이다. 즉, 단계 (1)의 이분법에 의해 요구되는 계산 복잡도가 $O(n)$[22]이고, 단계 (2)와 (3)은 n번의 반복실행(iteration)을 요구하고, 단계 (4)와 (5)는 기껏해야 k번의 반복실행을 요구하기 때문이다. 그러나 만약 매 스케줄링 시점에서 단지 하나의 태스크만 스케줄링되고 시스템 내의 모든 태스크가 수행될 때까지 새로운 태스크가 도착하지 않는 극단적인 경우에, 제안 알고리즘의 수행 총 회수는 시스템 내의 태스크들의 총 개수와 같다. 따라서 제안 알고리즘의 최악의 경우 계산 복잡도는 $O(n^2)$이다. 그러나 스케줄링된 k개의 태스크들의 완료 직전에 항상 새로운 태스크가 도착한다면, 이는 가장 이상적인 경우이며 계산 복잡도는 $O(n)$이 된다. 따라서 제안 알고리즘의 평균 계산 복잡도는 $O(n^2)$보다는 작고 $O(n)$보다는 크다.

4. 성능 분석

모의실험에서는 동적으로 도착하는 1~4개의 화상회의 스트림(Wavelet 압축방식 사용)을 복원하여 출력하는 단일 프로세서 시스템을 시뮬레이션 한다. 일반적으로 스트림의 화질은 두 가지로 평가할 수 있다. 첫 번째 평가요소는 시스템 내 서비스 중인 스트림들의 모든 프레임들 중 프로세서 시간 부족으로 누락된 프레임들의 비율(ratio of dropped frames: RDF)이며, 이는 $RDF = NDF / \sum_{i=1}^{N} N_i$로 정의된다. 여기서 N은 시스템 내의 스트림의 수, N_i는 스트림 S_i의 모든 프레임들의 수, NDF는 누락된 프레임들의 수이다. 누락된 프레임이란 프레임의 만기까지 프로세서로부터 서비스를 전혀 받지 못한 프레임들을 의미한다. RDF가 낮으면 낮을수록 단위 시간동안 누락되지 않고 출력된 프레임의 수가 많다는 것을 의미하며, 누락된 프레임의 수가 적기 때문에 연속적으로 출력되는 화자들의 움직임이 끊김없이 자연스럽게 출력된다. 따라서 RDF는 출력된 연속된 프레임들간의 자연스러운 연결성(smoothness)을 평가하는 측도로 사용될 수 있다.

두 번째 평가요소는 모든 스트림들의 프레임들 중 누락되지 않고 출력된 프레임들의 화질(quality of displayed frames: QDF)이며, 이는 $QDF = \sum_{i=1}^{N} \sum_{k=1}^{N_i} f_i(x_{ik}) / (\sum_{i=1}^{N} N_i - NDF)$로 정의된다. 여기서 x_{ik}는 스트림 S_i의 k번째 프레임에게 할당된 실행시간이며, $f_i()$는 스트림 S_i의 QoS 함수이다. 프레임들에게 할당되는 실행시간이 많으면 많을수록 프레임들의 품질은 증가하며, 그 만큼 출력된 프레임들이 깨끗하고 선명하게 보인다. 따라서 QDF는 출력된 프레임들의 영상 선명도(clearness)를 평가하는 측도로 사용될 수 있다.

스트림 S_i의 화질은 RDF와 QDF가 동시에 고려되어야 하며, 가장 이상적인 것은 RDF는 낮고 QDF는 높게 되는 것이다. RDF와 QDF 모두를 종합적으로 평가한 스트림 S_i의 서비스 품질은 $QoS_i = \sum_{k=1}^{N_i} f_i(x_{ik}) / N_i$로 정의된다. 스트림 S_i의 QoS_i가 높다는 것은 전체적으로 누락된 프레임들의 수가 적고 출력된 프레임들의 화질도 좋다는 것을 의미한다. 반면 QoS_i가 낮다는 것은 누락된 프레임의 수가 많거나 또는 전체적으로 출력된 프레임들의 화질이 좋지 않다는 것을 의미한다. 한 시스템에 동시에 여러 스트림들이 출력된다면, 시스템의 평균 서비스 품질은 $QoS = \sum_{i=1}^{N} \sum_{k=1}^{N_i} f_i(x_{ik}) / \sum_{i=1}^{N} N_i$로 정의된다.

각 시뮬레이션에서는 본 논문에서 제시한 온-라인 프레임 스케줄링(OFS) 알고리즘과 기존의 EDF 스케줄링 알고리즘을 사용했을 때의 각각의 스케줄링 성능을 측정하였다. 스케줄링 성능 측정 요소로는 각 스트림의 QoS_i와 시스템 전체의 RDF, QDF, QoS를 측정하였다.

시뮬레이션에서 각 스트림당 10분간 출력될 수 있는 14400개의 프레임을 생성하였다. 각 스트림의 프레임들은 초당 24 프레임(p_i = 41.67 ms)이 생성되고, 1초의 출력 지연시간을 가진다. 프레임들의 도착은 평균 도착율 0.024(= $1/p_i$)를 가진 포아송(Poisson) 분포를 따른다. OFS 스케줄링을 위해 4개의 화상회의 스트림들을 사전에 분석하여 이들의 QoS 함수를 구하였으며, 이들은 각각 $f_1(x) = 1 - e^{-0.41x}$, $f_2(x) = 1 - e^{-0.39x}$, $f_3(x) = 1 - e^{-0.36x}$, $f_4(x) = 1 - e^{-0.31x}$이다. 본 연구에서는 기존의 EDF를 수정하여 만기 전까지 중간 복원된 프레임의 영상 역시 출력 가능하도록 하였다.

본 연구에서는 시뮬레이션에 사용되는 4개의 스트림들을 구성하는 모든 프레임들의 시뮬레이션 변수들(생성시간, 도착시간, 만기 등)을 미리 생성하고, 동일한 스트림들에 대해 다양한 시스템 부하에서 스케줄링 성능을 측정하였다. 강제로 시스템 부하를 생성하기 위해 가장 우선순위가 높은 하나의 주기적 태스크 t_l을 시스템에 추가하였다. 부하 태스크 t_l이 시스템 내에서 주기적으로 실행될 경우, 스트림들을 서비스하기 위해 사용 가능한 시스템 활용도(usable system utilization: U_u)는 $1 - u_l$이다. 여기서 u_l은 c_l과 p_l을 각각 태스크 t_l의 실행시간과 주기라고 했을 때 c_l/p_l로 정의되는 태스크 t_l의 활용도(utilization)이다. p_l은 항상 5ms으로 고정시킨 채 c_l을 조절함으로써 사용 가능한 시스템 활용도(U_u)를 다

양하게 생성하였다. U_u가 높을수록 시스템 부하가 낮다는 것을 의미하며, U_u가 낮을수록 시스템 부하는 반대로 높다는 것을 의미한다. 스케줄러는 부하 태스크 t_i이 도착하자마자 현재 실행중인 작업을 중단하고 무조건 c_i 시간 동안 t_i을 서비스한다.

그림 6은 시스템 내의 스케줄링해야 할 화상회의 스트림 수가 2개(NS2), 4개(NS4)일 때의 OFS와 EDF 스케줄링 알고리즘에 의해 서비스된 모든 프레임들의 평균 QoS인 시스템 QoS를 보여준다. 그림에서 U_u가 감소함에 따라 시스템 QoS 역시 감소하고, 동일한 U_u에서는 화상회의 스트림 수가 많을수록 시스템 QoS는 더 감소한다. 그림에서 또한 동일한 스트림 개수에 대해 U_u가 매우 크거나(NS2인 경우 $U_u \geq 0.4$, NS4인 경우 $U_u = 1.0$) 또는 매우 작을 경우 ($U_u = 0.1$), 스케줄링 알고리즘에 상관없이 QoS는 비슷한 값을 가진다. 스트림의 개수가 동일할 경우 시스템 부하에 상관없이 항상 OFS의 QoS가 EDF보다 높다. 또한 U_u가 감소함에 따라 OFS의 QoS는 점진적으로 감소하는 반면 EDF는 OFS에 비해 상대적으로 큰 폭으로 감소한다. 이 실험 결과를 통해 EDF의 경우 프레임들의 화질이 시스템 부하에 따라 민감하게 영향을 받는 대신, OSF의 경우 프레임들의 화질이 소폭으로 변화한다는 것을 알 수 있다. 또한 시스템 내의 스케줄링해야 할 스트림의 수가 증가할수록 OSF와 EDF에 의해 스케줄링된 프레임들의 화질은 더 큰 차이를 보인다.

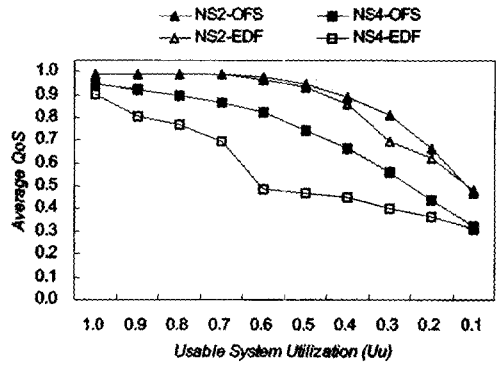

그림 6 시스템 QoS

그림 7은 화상회의 스트림 수가 4개 일 때(그림 6의 NS4에 대한 실험 결과임) 4개의 스트림별 평균 QoS 중 가장 좋은 QoS_i(Best-NS4)와 가장 나쁜 QoS_i(Worst-NS4)를 보이고 있다. 그림에서 OFS는 시스템 부하가 적을 경우($U_u \geq 0.6$) 가장 좋은 QoS_i와 가장 나쁜 QoS_i 차이가 크지 않다가 시스템 부하가 증가할수록 점점 그 격차가 벌어지다가 궁극적으로는($U_u = 0.1$) EDF와 동일한 격차가 된다. 반면, EDF의 경우 두 스트림간의 QoS 차이가 시스템 부하에 상관없이 매우 크다. 이 실험 결과를 통해 EDF의 경우 시스템 부하에 상관없이 특정 스트림의 화질은 매우 좋은데 비해 어떤 스트림은 매우 나쁜 화질을 가진다. 그러나 OFS는 시스템 부하가 매우 심할 때를 제외하고는 각 스트림별 QoS_i의 차이가 크지 않다는 부수적인 효과를 함께 가진다.

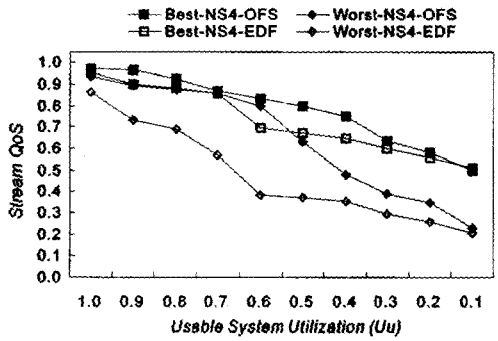

그림 7 가장 좋은 QoS_i 대 가장 나쁜 QoS_i

그림 8에서는 스케줄링해야 할 스트림 수가 4개 일 때(그림 6의 NS4에 대한 실험 결과임), 누락되지 않고 출력된 프레임들의 품질(QDF)과 출력되지 않고 누락된 프레임들의 비율(RDF)을 보여준다. 그림에서 각 스케줄링 알고리즘별 QDF는 실제 출력된 프레임들만의 평균 QoS이므로, 그림 6의 모든 프레임들의 평균 QoS인 시스템 QoS(NS4)에 비해 상대적으로 높은 값을 가진다. 그림에서 EDF의 경우 시스템 부하가 적을 경우($U_u \geq 0.7$) OFS와 비슷한 QDF를 가지지만, 시스템 부하가 증가할수록 OFS보다는 높은 QDF 값을 가진다. 그러나 누락되는 프레임의 비율(RDF)이 OFS보다 훨씬 높게 나타난다. 이는 시스템 부하가 증가할수록 EDF에 의해 출력되는 프레임의 화질은 OFS보다 좀더 선명하고 깨끗하지만 반대로 누락된 프레임의 수가 급격히 증가하여 화자의 움직임이 자연스럽지 못하거나 영상의

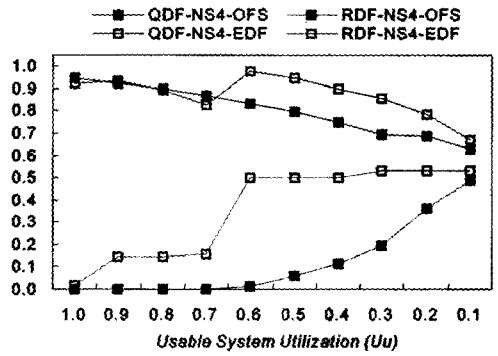

그림 8 QDF와 RDF

끊김이 길어지는 현상이 발생하게 됨을 의미한다.

이상의 시뮬레이션을 통해 화상회의를 위한 여러 스트림의 압축을 복원하는 스케줄링 알고리즘으로써 본 논문에서 제안한 온-라인 프레임 스케줄링 알고리즘을 사용할 수 있음을 확인했다. 이 스케줄링 기법을 사용할 경우 시스템 부하가 증가함에 따라 화질은 부드럽게 떨어지고 연속된 프레이임들을 지속적으로 출력함으로써 자연스런 화자의 동작을 볼 수 있음을 확인했다. 또한 한 화면에 다수의 화자 화상을 볼 경우(다자간 화상회의를 할 경우), 각 스트림별로 화질의 차이가 크게 발생하지 않는 부수적인 이점도 있다.

5. 결 론

본 논문에서는 인터넷상에서 화상회의 시스템을 위한 효율적인 온-라인 프레임 스케줄링 알고리즘을 제안하였다. 이를 위해 압축된 영상의 복원 과정에서 프로세서의 처리시간에 따르는 결과 영상의 품질에 대한 QoS 함수를 도출하고, 압축된 영상을 복원하는 태스크를 IRIS 태스크로 모델링하였다. 또한 복원영상의 품질측정을 위한 몇 가지 측정요소를 정의하고, 제안 알고리즘의 효율성을 검증하기 위하여 품질평가를 위한 시뮬레이션을 실시하였다. 실험결과 제안된 스케줄링 기법을 사용할 경우 시스템 부하가 증가함에 따라 화질은 부드럽게 감소하고, 연속된 프레임들을 지속적으로 출력함으로써 자연스런 화자의 동작을 볼 수 있음을 확인하였다. 또한 한 화면에 다자간 화상회의를 할 경우, 각 스트림별로 화질의 차이가 크게 발생하지 않는 부수적인 이점도 가진다.

제안된 알고리즘이 보다 효율적인 스케줄링 결과를 생성하기 위해서는 말릿블록 단위로 프레임을 처리함으로써 발생하는 프로세서 낭비시간을 최소화할 수 있는 방안이 지속적으로 연구되어야 한다. 또한 보다 실용적인 알고리즘이 되기 위해서는 자동으로 QoS 함수를 도출할 수 있는 방안도 함께 연구되어야 한다.

본 논문에서는 영상의 주파수 성분별로 화질을 점진적으로 조절할 수 있는 Wavelet 알고리즘의 특성을 기반으로 하여 화상회의 시스템에 IRIS 태스크 스케줄링 모델을 적용할 수 있음을 보였다. 따라서 이러한 특성을 지원하지 않는 인코딩-디코딩 알고리즘의 경우 제안 알고리즘을 적용하기에는 한계가 있다. 그러나 제안 알고리즘은 MPEC의 DCT(Discrete Consine Transform)용 압축 모듈이나[5], 정보 수집 및 추출, 데이타베이스 질의어 처리, 자동 항법 계획, 실시간 휴리스틱 탐색, 의료 결정 시스템, 전통적인 수치 알고리즘 계산 등과 같이 태스크에게 주어지는 서비스 시간이 증가할수록 해당 태스크가 얻는 QoS 또한 증가하는 IRIS 태스크용 스케줄링 모델이 적용될 수 있는 다양한 응용에 활용될 수 있다.

참 고 문 헌

[1] N. Laoutaris and I. Stavrakakis, "Intrastream Synchronization for Continuous Media Streams: A Survey of Playout Schedulers," *IEEE trans. on Networks*, Vol. 16, No. 3, pp. 30-40, May 2002.

[2] D. Stone and K. Jeffay, "An Empirical Study of Delay Jitter Management Policies," *Multimedia Systems*, Vol. 2, No. 6, pp. 267-279, Jan. 1995.

[3] G. Jung, K. Yim, J. Shim, K. Choi, et al., "A Jitter Management Policy for Monitoring System using JPEG," *IEEE Proceedings on Consumer Electronics*, pp. 214-215, Oct. 1996.

[4] C. S. Burrus, R. A. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelet Transforms*, Prentice-hall, 1998.

[5] K. Yim, J. Shim, G. Jung, and K. Choi, "An Imprecise DCT Computation Model for Real-time Applications," *Multimedia Technology and Applications*, pp. 153-161, Springer, 1997.

[6] J. K. Dey, J. F. Kurose, and D. Towsley, "On-line Scheduling Policies for a Class of IRIS(Increasing Reward with Increasing Service) Real-Time Tasks," *IEEE Trans. Computers*, Vol. 45, No. 7, pp. 802-813, July 1996.

[7] J. K. Dey, J. F. Kurose, D. Towsley, C.M. Krishna, and M. Girkar, "Efficient On-line Processor Scheduling for a Class of IRIS (Increasing Reward with Increasing Service) Real-Time Tasks," *Proc. ACM Sigmetrics Conf. Measurement and Modeling of Computer Systems*, pp. 217-228, Santa Clara, Calif., May 1993.

[8] J.Y. Chung, J.W.S. Liu, and K.J. Lin, "Scheduling Periodic Jobs That Allows Imprecise Results," *IEEE Trans. Computers*, Vol. 19, No. 9, pp. 1156-1173, Sept. 1990.

[9] J.W.S. Liu, K.J. Lin, W.K. Shih, A.C.S. Yu, J.Y. Chung, and W. Zhao, "Algorithms for Scheduling Imprecise Computations," *IEEE Computer*, Vol. 24, No. 5, pp. 58-68, May 1991.

[10] W.K. Shih, J.W.S. Liu, and J.Y. Chung, "Algorithms for Scheduling Imprecise Computations with Timing Constraints," *SIAM J. Computing*, Vol. 20, No. 3, pp. 537-552, June 1991.

[11] W.K. Shih and J.W.S. Liu, "On-Line Scheduling of Imprecise Computations to Minimize Error," *Proc. 13th IEEE Real-Time Systems Symp.*, pp. 280-289, Los Alamitos, Calif., Dec. 1992.

[12] K. Song, K. Choi, S. Park, D. Choi, and K. Yun, "A Heuristic Scheduling Algorithm for Reducing the Total Error of an Imprecise Multiprocessor System with 0/1 Constraint," *J. Electrical Engineering and Information Science*, Vol. 2, No. 6,

Dec. 1997.
[13] K. Choi, S. Yun, G. Jung, and N. Kim, "Scheduling Algorithm for Real-Time Imprecise Computations to Minimize Maximum Weighted Error Using the Linear Programming Method," *Electronics Letters*, Vol. 33, No. 15, pp. 1301-1302, July 1997.
[14] W.K. Shih and J.W.S. Liu, "Algorithms for Scheduling Imprecise Computations with Timing Constraints to Minimize Maximum Error," *IEEE Trans. Computers*, Vol. 44, No. 3, March 1995.
[15] K.I.J. Ho, J.Y.T. Leung, and W.D. Wei, "Minimizing Maximum Weighted Error for Imprecise Computation Tasks," *J. Algorithms*, Vol. 16, pp. 431-452, 1994.
[16] W.K. Shih, J.W.S. Liu, J.Y. Chung, and D.W. Gillies, "Scheduling Tasks with Ready Times and Deadlines to Minimize Average Error," *ACM Operating Systems Review*, July 1989.
[17] J. Du and J.Y.T. Leung, "Minimizing Mean Flow Time in Two-Machine Open Shops and Flow Shops," *J. Algorithms*, Vol. 14, pp. 24-44, 1993.
[18] H. Aydin, R. Melhem, D. Mossé, and P. Mejía-Alvarez, "Optimal Reward-Based Scheduling for Periodic Real-Time Tasks," *IEEE Trans. Computers*, Vol. 50, No. 2, pp. 111-130, Feb. 2001.
[19] ISO/IEC JTC1/SC29/WG11, "Coding of Moving Pictures and Associated Audio For Digital Storage Media at up to About 1.5 Mbit/s, Part 2: Video," ISO/IEC 11172-2 Information Technology, 1993.
[20] S. J. P. Westen, R. L. Lagendijk, and J. Biemond, "Perceptual Optimization of Image Coding Algorithms," *IEEE Proceedings of the International Conference on Image Processing*, pp. 69-72, 1995.
[21] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, pp. 395-404, 1992.
[22] K. Choi and G. Jung, "Comment on On-line Scheduling Policies for a Class of IRIS Real-Time Tasks," *IEEE Trans. Computers*, Vol. 50, No. 5, pp. 526-528, May 2001.
[23] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," *J. ACM*, Vol. 20, No. 1, pp. 46-61, Jan. 1973.

안 성 용
1996년 2월 조선대학교 전자계산학과 졸업(이학사). 1998년 2월 조선대학교 대학원 전자계산학과 석사과정 졸업(이학석사). 1999년 3월~현재 조선대학교 대학원 전자계산학과 박사과정 재학 중. 관심분야는 내장형 시스템, 시스템 수준 설계, 재구성 가능한 시스템, 실시간 시스템

이 정 아
1982년 2월 서울 대학교 컴퓨터공학과 졸업(공학사). 1985년 6월 Indiana University 컴퓨터과학과 석사과정 졸업(공학석사). 1990년 11월 UCLA 컴퓨터공학과 박사과정 졸업(공학박사). 1990년 8월~1995년 2월 미국 텍사스주 휴스턴주립대학 전기전산공학과 조교수. 2000년 8월~2002년 2월 Stanford University Visitng Professor. 2000년 10월~현재 조선대학교 정교수. 관심분야는 재구성 가능한 시스템, 고속 연산기 설계, CORDIC, 적응형 시스템, 실시간 시스템

심 재 홍
1987년 서울대학교 전산과학과 졸업(학사). 1989년 아주대학교 컴퓨터공학과 졸업(석사). 2001년 아주대학교 컴퓨터공학과 졸업(박사). 1989년~1994년 서울시스템(주) 공학연구소. 1999년~2000년 University of Arizona 객원연구원. 2001년~2001년 9월 아주대학교 정보통신전문대학원 BK21 전임연구원. 2001년 10월~현재 조선대학교 인터넷소프트웨어공학부 조교수. 관심분야는 운영 체제, 실시간 및 멀티미디어 시스템, 내장형시스템