JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 18, No. 1 : 97~102 Jan. 2014

다중프로세서 시스템상의 개선된 합성 이용율을 이용한 실시간 비주기 태스크 스케줄링

문석환*

Real-Time Aperiodic Tasks Scheduling Using Improved Synthetic Utilization on Multiprocessor Systems

Seok-Hwan Moon*

Department of Embedded Software, Youngdong University, Youngdong 310, Korea

요 약

다중프로세서 시스템에서 임의의 시점에 비주기 태스크들의 스케줄링 가능성을 판단하기위한 알고리즘으로서 합성 이용율이 Abdelzaher등에 의해 제시되었는데, 이들은 임의의 시점에 합성이용율의 상한 값인 0.59를 넘지 않으면 비주기 태스크들이 스케줄링 가능 하다는 것을 증명 하였다. 하지만 이 방법은 비주기 태스크들의 프로세서 이용율 계산 시 태스크가 실제 모든 실행시간을 종료하여 더 이상의 실행시간을 갖지 않더라도 현재요청집합에 속해 있다면 실행시간과 종료시한을 합성 이용율에 포함하기 때문에 스케줄링 가능한 태스크들이 실행 불가능한 경우로 판단되는 문제점을 가지고 있다. 본 논문에서는 이러한 문제점을 해결하여 다중프로세서 시스템에서 더 많은 비주기 태스크들이 스케줄링 가능 하도록 개선된 합성 이용율 방법을 제시하였다.

ABSTRACT

Abdelzaher et al. proposed an algorithm to determine the schedulability of aperiodic tasks on multiprocessor systems, and proved that the aperiodic tasks are schedulable if the upperbound of synthetic utilization is less than or equal to 0.59. But this algorithm has a drawback in that if some tasks, even though they are completed and have no more execution times, are included in the current invocation set, their execution times and deadlines are added to the synthetic utilization. This may lead to a problem in which actually schedulable tasks are decided not to be schedulable. In this paper, we recognize the above mentioned problem and propose an improved synthetic utilization method that can be used to schedule aperiodic tasks more efficiently on multiprocessor systems.

키워드 : 실시간 스케줄링, 비주기 태스크, 합성 이용율, 다중프로세서 시스템

Key word : Real-Time Scheduling, Aperiodic Tasks, Synthetic Utilization, multiprocessor systems

접수일자 : 2013. 09. 24 심사완료일자 : 2013. 10. 27 게재확정일자 : 2013. 11. 13

*** Corresponding Author** Seok-Hwan Moon(E-mail : shmoon@yd.ac.kr, Tel:+82-43-740-1524)
Department of Embedded Software, Youngdong University, Youngdong 310, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2014.18.1.97 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

실시간 시스템(real-time system)은 태스크의 수행 결과의 정확성뿐만 아니라 처리시한의 제한인 종료시한(deadline)을 엄격하게 지키는 것이 요구되는 시스템을 말한다. 실시간 시스템에서 실행되는 태스크들은 일정 시간 마다 도착하여 반복적으로 실행되는 주기(periodic) 태스크와 도착 시간이 정해져 있지 않은 비주기(aperiodic) 태스크로 구분 할 수 있다. 본 논문에서는 합성 이용율을 기반으로 하여 다중프로세서 상에서 비주기 태스크들을 스케줄링 하는 기법을 다룬다.

이제까지 제안된 스케줄링 알고리즘들 중에서 Rate Monotonic (RM)[1] 스케줄링 알고리즘은 고정 우선순위 기반 알고리즘들 중 최적이고, Earliest Deadline First(EDF)[1-2] 스케줄링 알고리즘은 동적 우선순위 알고리즘들 중 최적인데 이들은 모두 주기적 태스크 모델을 기본으로 하고 있고, 프로세서 이용율을 이용하여 태스크들을 스케줄링 하는 방법이며, [3]에서 제시한 방법은 최악의 경우 응답시간(worst case responsetime)[4]를 이용하여 주기 태스크들의 스케줄링 가능성 여부를 판단하는 방법이다.

비주기 태스크에 대한 스케줄링 분석이 주기 태스크에 비하여 상대적으로 어려운 이유는 태스크가 도착하기 이전에는 태스크의 도착 시점, 수행 시간, 종료시한을 주기 태스크처럼 미리 예측 할 수 없기 때문이다. 경성 실시간 비주기 태스크 스케줄링 방법으로는 연성 실시간 비주기 태스크에도 적용 가능한 슬랙(slack) 시간 분석을 통한 슬랙 스틸링 알고리즘[5-6]과 최근에 Abdelzaher등에 의해 제시 된 비주기 태스크들의 이용율 분석을 통한 합성 이용율 알고리즘[7-10]등이 있다.

본 논문의 구성은 2장에서 다중 프로세서 상에서의 합성 이용율과 이를 이용한 스케줄링 방법을 기술한 후 이의 문제점과 개선된 방법을 제시한다. 3장에서는 모의 실험결과를 보이고, 마지막으로 4장에서는 결론 및 향후 연구 과제에 대하여 기술한다.

Ⅱ. 비주기 태스크 이용율

2.1. 다중프로세서에서의 합성 이용율

Abdelzaher등에 의해 제시된 합성 이용율 방법[7]은 경성 비주기 태스크 집합에 대하여 스케줄링 가능한 합성 이용율의 상한 값을 제시하였다. 또한 이를 이용하여 비주기 태스크가 도착했을 때 합성 이용율을 상한 값과 비교하여 도착한 비주기 태스크의 수락여부를 인정할 것인지, 거절할 것인지를 결정하게 된다.

비주기 태스크 집합 T_a를 $\{T_1, T_2, T_3 \cdots T_i \cdots\}$라 할 때, T_1은 T_2보다 먼저 도착(arrival)된 태스크라 가정 한다. 즉 T_i는 T_{i+1}보다 먼저 도착한 비주기 태스크 이다. 비주기 태스크 T_i의 수행시간(execution time)을 $C_i(>0)$, 도착시간(arrival time)을 A_i, 상대적 종료시간(relative deadline)을 $D_i(>0)$ (여기서 절대적 종료시간(absolute deadline) d_i는 $A_i + D_i$로서 정의된다.), m을 프로세서의 개수라 할 때 다중프로세서 상에서 임의의 시점 t에서의 합성 이용율[9]은 다음과 같이 정의된다.

$$U(t) = \frac{1}{m} \sum_{T_i \in S(t)} \frac{C_i}{D_i} \tag{1}$$

식(1)은 Liu와 Layland가 정의한 주기태스크들의 스케줄링 알고리즘인 RM 스케줄링 알고리즘[1]을 다중프로세서 상에서 비주기 태스크에 확장하여 정의하였다. 식(1)에서 현재요청집합(current invocation set) $S(t)$는 임의의 시점 t를 기준으로 t이전에 도착한 태스크 중 아직 종료시한이 지나지 않은 태스크 집합을 나타내며 $S(t) = \{T_i | A_i \le t < A_i + D_i\}$ 로서 표현된다.

[9]에서는 다중프로세서 상에서 임의의 시점 t에서의 합성 이용율 식(1)이 상한값인 0.59를 넘지 않으면 비주기 태스크가 스케줄링 가능하다는 것을 증명하였다. 또한 합성 이용율을 이용한 스케줄링 분석은 $O(1)$에 수행할 수 있어 수락제어를 수행할 때 적합하다. 비주기 태스크들의 합성 이용율을 구하기 위해서는 임의의 시점 t에 대한 현재 요청 집합 $S_{(t)}$를 먼저 구해야한다.

다음은 t=3과 t=4에서 비주기 태스크 집합의 합성 이용율을 구하는 예제이다. 여기서 프로세서의 개수 m = 1이라고 가정 한다.

표 1. 합성 이용율을 위한 비주기 태스크 집합
Table. 1 Aperiodic Tasks Set for Synthetic Utilization

	A_i	C_i	D_i
T_1	0	2	16
T_2	1	2	14
T_3	2	2	12
T_4	3	2	14

합성 이용율을 구하기 위해 먼저 현재 요청 집합을 구하면 $S(3)=\{T_1, T_2, T_3, T_4\}$이고, 이것을 식(1)에 적용하게 되면 $U(3)=\frac{2}{16}+\frac{2}{14}+\frac{2}{12}+\frac{2}{14}=0.57$ 이다.

$U(3)=0.57<0.59$ 이므로 시점 t=3에 스케줄링 가능하다. 즉, 태스크 T_4가 시점 t=3에 도착했을 때 이미 도착된 모든 태스크들의 스케줄링 가능성을 보장하므로 T_4는 수락된다. 또한 t=4의 시점에서 합성 이용율을 구하게 되면 집합 $S(4)$도 변화가 없고 $U(4)$의 값도 변화가 없으므로 모든 태스크들은 스케줄링 가능하다.

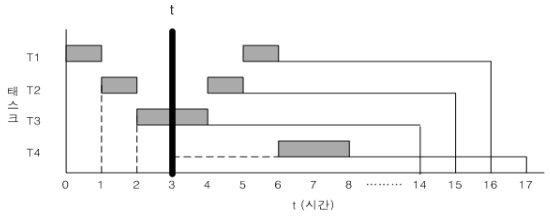

그림 1. t=3에서의 현재요청집합
Fig. 1 Current Invocation set at t=3

하지만 모든 태스크들의 종료시한이 1만큼씩 줄어들게 되면 $U(4)=\frac{2}{15}+\frac{2}{13}+\frac{2}{11}+\frac{2}{13}=0.62$이 되므로 상한값 0.59를 넘게 되어 스케줄링 불가능한 상태가 된다. 이 때, $S(4)$의 태스크 중 태스크 T_3는 실행시간이 종료되었지만, 종료시한을 넘지 않았기 때문에 실제 시점 t=4이후에는 실행되지 않더라도 집합 $S(4)$에 속하게 되고 이용율 계산 시 C_4값이 불필요하게 합성 이용율에 함께 계산 된다. 또한 t=4 이후에는 실행시간이 $C_1=1$, $C_2=1$만큼만 실행하면 되지만, 합성 이용율에서는 $S(t)$에 속하게 되면 모든 실행 시간을 포함하여 이용율을 계산하게 된다.

이러한 문제점으로 인해 특정시간 t에서의 합성 이용율은 계산할 때 t이후에 실제 스케줄링 가능하지만, 스케줄링 불가능하게 판단되는 경우가 발생하게 된다. 본 논문에서는 이러한 문제를 개선하여 스케줄링 가능성을 높이고자 한다.

2.2. 다중프로세서에서의 개선된 합성 이용율

임의의 시점 t에서 합성 이용율을 계산할 때 먼저 고려되어야 할 것은 시점 t이전에 도착하였으나 아직 시점 t이전에 종료시한을 넘기지 않은 비주기 태스크들의 집합, 즉, $S(t)=\{T_i | A_i \le t < A_i + D_i\}$로서 표현 되는 현재 요청 집합 $S(t)$의 원소에 해당하는 비주기 태스크들을 찾는 것이다. 이러한 $S(t)$에 속하는 비주기 태스크들 중 시점 t를 기준으로 먼저 도착 하였고, 종료시한은 넘지 않았지만, 시점 t이전에 실행시간을 모두 마친 태스크들이 포함될 수 있다. 이러한 태스크들은 임의의 시점 t이후부터 태스크의 종료시한까지 실제 프로세서를 이용하지 않지만 이용율 계산 시 실행 시간이 포함되는 문제점을 가지고 있다. 본 논문에서는 이러한 조건을 제거하여 합성 이용율을 계산한다. 개선된 현재 요청집합을$S'(t)=\{T_i | A_i \le t < A_i + D_i, t < C_{ie}\}$로 표현하고 여기서 C_{ie}는 비주기 태스크 T_i의 실행시간 C_i의 종료시점이다. C_i의 종료시점은 고정 우선순위를 이용하기 때문에 계산해낼 수 있다.

또한 합성 이용율 $U(t)=\sum_{T_i \in S(t)} \frac{C_i}{D_i}$에서 C_i와 D_i값은 태스크 T_i가 $S(t)$의 조건을 만족하여 $S(t)$의 원소가 되면 항상 동일한 값을 가진다. 즉 시점 t가 증가하더라도 $S(t)$에 속하면 항상 동일한 합성 이용율값을 가지게 되는데 이것은 시점 t이전에 이미 실행 완료된 태스크의 실행시간 값까지 합성 이용율에 포함되기 때문에 실제 남은 실행시간 값이 매우 작더라도 시점 t에서 스케줄링이 불가능하게 판단되는 경우가 발생할 수 있다. 이러한 문제를 개선하기 위해 C_i값과 D_i값을 시점 t를 기준으로 하여 변경한 C_i와 D_i값을 이용하여 합성 이용율을 구한다.

시점 t를 기준으로 하여 태스크 T_i의 실행을 마치기 위해 필요한 최대 수행시간[11]를 잉여 수행시간 $e_{i,t}$라 하고, 시점 t를 기준으로 하여 태스크 T_i의 절대 종료시한 d_i와 현재 시점 t와의 차이, 즉 $d_i - t$를 리드시간(lead time)[12] $D_{i,t}$라고 정의하면 다중프로세서 상에서 임의의 시점 t에서의 합성 이용율은 다음과 같이 새롭게 표

현할 수 있다.

$$U(t) = \frac{1}{m} \sum_{T_i \in S'(t)} \frac{e_{i,t}}{D_{i,t}} \quad (2)$$

다음은 새롭게 정의된 $S'(t)$, $e_{i,t}$, $D_{i,t}$를 이용해서 개선된 합성 이용율을 구하는 예제이다. 여기서 프로세서의 개수 m = 1이라고 가정 한다.

그림 2에서처럼 시점 t=4에서 개선된 현재 요청 집합은 $S'(4) = \{T_1, T_2, T_4\}$이 된다. T_3이 개선된 현재 요청 집합에서 제외된 이유는 t=4에서 종료시한은 지나지 않았지만, 실행시간이 모두 종료되었기 때문이다.

표 2. 개선된 합성 이용율을 위한 비주기 태스크 집합
Table. 2 Aperiodic Tasks Set for Improved Synthetic Utilization

	A_i	C_i	D_i	d_i
T_1	0	2	16	16
T_2	1	2	14	15
T_3	2	2	12	14
T_4	3	2	14	17

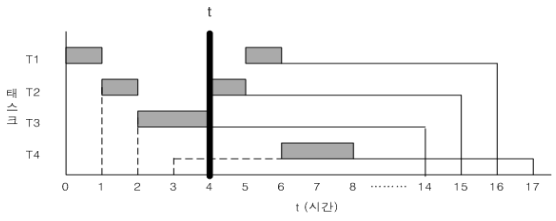

그림 2. t=4에서의 개선된 합성 이용율
Fig. 2 Improved Synthetic Utilization at t=4

식(2)를 이용하면 t=4에서의 개선된 합성 이용율은 다음과 같다.

$$U(4) = \frac{e_{1,4}}{D_{1,4}} + \frac{e_{2,4}}{D_{2,4}} + \frac{e_{4,4}}{D_{4,4}} = 0.328$$

$U(1), U(2)$의 경우에는 $S'(t)$값의 변화는 없으나 시간의 흐름에 따라 이용율이 변하게 되고, $U(5), U(6), U(7)$의 경우는 $S'(t)$값도 변하고, 이용율도 변하게 되어 $U(1) = 0.488$, $U(2) = 0.448$, $U(5) = 0.257$ $U(6) = 0.181$, $U(7) = 0.1$의 값을 가진다.

$U(1)$부터 $U(7)$까지의 합성 이용율을 비교해 보면 $U(1)$보다 $U(7)$이 더 낮은 이용율을 보이는데 이것은 이용율과 비례관계에 있는 실행 시간이 줄어들기 때문이다. 다른 비주기 태스크가 도착하여 수락 여부를 판단하는 경우에 시점 t=3 보다 시점 t=4에서 비주기 태스크가 수락될 가능성이 높아진다고 볼 수 있다. 예제처럼 재 정의된 $S'(t)$, $e_{i,t}$, $D_{i,t}$를 이용하여 합성 이용율을 계산한 후 합성 이용율의 상한 값인 0.59와 비교하여 임의의 시점 t에 도착한 비주기 태스크의 수락 여부나, 이미 도착되어 실행 중인 비주기 태스크들의 합성 이용율을 이용하여 스케줄링 가능성 여부를 판단한다.

Ⅲ. 모의 실험

3.1. 모의실험 방법

본 장에서는 모의실험을 통해 프로세서 개수의 변화에 따른 기존의 합성 이용율과 본 논문에서 제시한 방법을 비교한다. 모의실험에서는 도착시간(A_i), 수행시간($C_i > 0$), 상대적 종료시한($D_i > 0$)을 가지는 비주기 태스크 1000개를 랜덤하게 생성하고 임의의 시점 t에서의 기존의 합성 이용율과 본 논문에서 제시한 방법으로 구한 합성 이용율 값을 비교 하였다. 또한 아래와 같이 프로세서 개수별로 워크로드를 변화시키고, 기존의 합성 이용율과 개선된 방법에서의 합성 이용율 변화를 그래프로서 비교 하였다. C_i / $\min(d_i, T_i)$로 표현되는 태스크 밀도는 일반적인 0.1로 하였고, 입력 워크로드는 모든 도착된 태스크들의 수행시간의 합과 태스크들이 도착한 시간구간의 비율로 표현할 수 있는데 실험 범위는 40% ~ 150%까지 10%씩 증가되는 시점에서 이용율을 측정 하였다.

3.2. 실험 결과

다음의 그림 3와 그림 4는 프로세서의 개수별 기존의 합성 이용율과 개선된 방법을 적용한 합성 이용율에 대한 워크로드의 변화에 따른 비주기 태스크들의 프로세서 이용율을 보여주고 있다.

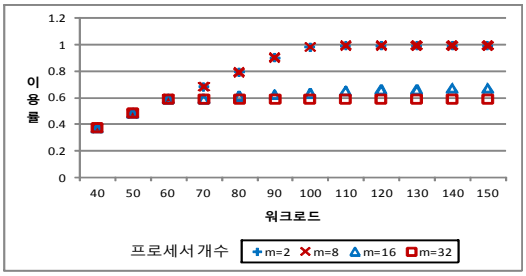

그림 3. 기존의 합성 이용율
Fig. 3 Synthetic Utilization

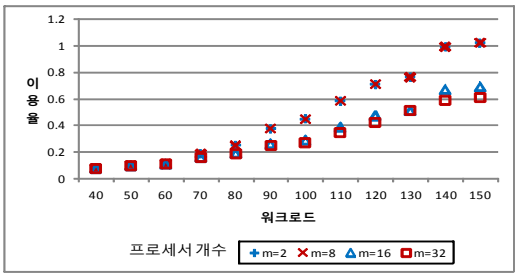

그림 4. 개선된 합성 이용율
Fig. 4 Improved Synthetic Utilization

그림 3에서 입력 워크로드가 클수록 프로세서의 개수에 따라 수락율의 많은 차이를 보인다. 그 이유는 입력워크로드가 크다는 의미는 시간구간에 비해 도착되는 비주기 태스크들이 많이 도착된다는 의미이므로 프로세서 이용율이 올라가기 때문이며, 또한 프로세서의 개수가 적을수록 이용율이 높아지기 때문이다. 실험결과에서도 볼 수 있듯이 기존 합성 이용율과 개선된 합성 이용율을 비교해 보면 개선된 방법의 합성 이용율이 좀더 낮은 이용율을 갖는다는 것을 알 수 있다.

그림 4에서 개선된 합성 이용율은 그림 3의 합성 이용율과 비교할 때 프로세서 개수의 변화와 워크로드의 변화에 따른 프로세서의 이용율의 차이가 크지 않게 나타난다. 그 이유는 기존의 합성 이용율에서는 임의의 시점 t에서 이용율을 측정할 때 태스크의 종료시한을 넘기지 않았다면 태스크의 실행이 완료 되었다 하더라도 실행시간을 이용율에 포함시키기 때문이다. 개선된 방법에서는 남은 실행시간만을 이용하여 프로세서 이용율을 계산하기 때문에 실제 임의의 시점 t에서 기존의 합성 이용율보다 개선된 합성 이용율의 값이 낮은 값을 가지는데, 이것은 이미 입력된 비주기 태스크들의 스케줄링을 보장하면서 시점 t에 입력되는 다른 비주기 태스크들의 수락율을 높일 수 있다는 의미를 가진다.

그림 5, 그림 6, 그림 7은 프로세서의 개수별로 기존의 합성 이용율과 개선된 합성 이용율의 워크로드별 변화를 보여준다. 그림에서 보듯이 워크로드별로 비교해 볼 때 개선된 스케줄링 방법이 기존의 방법보다 더 많은 태스크 집합들을 스케줄링 가능하다는 것을 알 수 있다. 또한 m=2인 경우에 비해 m=16, m=32처럼 m의 개수가 커질수록 기존의 방법에 비해 성능이 우수하다는 것을 알 수 있다.

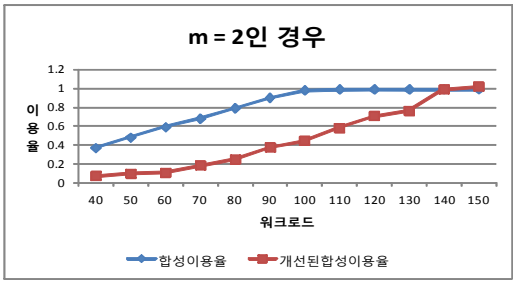

그림 5. m = 2인 경우 프로세서 이용율
Fig. 5 Processor Utilization in m = 2

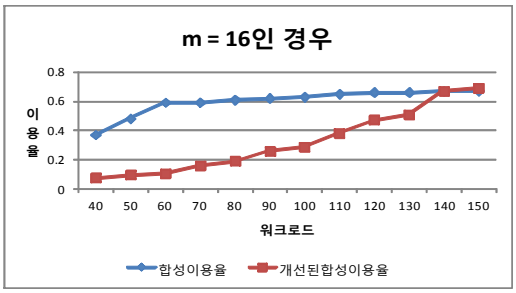

그림 6. m = 16인 경우 프로세서 이용율
Fig. 6 Processor Utilization in m = 16

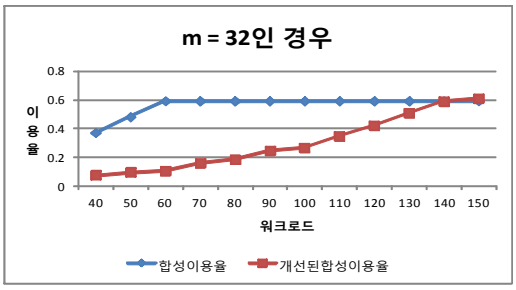

그림 7. m = 32인 경우 프로세서 이용율
Fig. 7 Processor Utilization in m = 32

Ⅳ. 결 론

본 논문에서는 다중프로세서 시스템에서 시점 t에서 현재요청집합의 조건을 변화시켜 합성 이용율에 의한 프로세서 이용율을 개선함으로써 기존의 합성 이용율을 이용하면 스케줄링이 불가능하던 비주기 태스크들을 스케줄링 가능하게 하는 방법을 제시하였다. 또한 실험을 통해 개선된 스케줄링 방법이 프로세서 개수와 워크로드의 변화에 따른 프로세서 이용율이 기존의 방법보다 낮은 값을 갖는다는 것을 확인 하였으며, 이것은 더 많은 비주기 태스크들을 수락할 수 있다는 의미를 가진다.

본 논문에서는 다중프로세서 시스템에서 비주기 태스크만을 고려하여 임의의 시간에서의 합성 이용율을 통하여 스케줄링의 가능성 여부를 판단하였지만, 주기 태스크들과의 혼합된 태스크 집합에서의 이용율 측정을 통한 스케줄링 가능성 판단 여부 및 수락율에 대한 연구가 필요하다.

REFERENCES

[1] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in hard real time environment", *Journal of the ACM*, vol. 20, pp. 46-61, Jan. 1973.

[2] H. Chetto and M. Chetto, "Some results of the earliest deadline first scheduling algorithm", *IEEE Transactions on Software Engineering*, vol. 15, no. 10, pp. 1261-1268, Oct. 1989.

[3] Kim, In-Guk, Kim, Dong-Yoon and Hong, Man-pyo, "Real-time scheduling of tasks that contain the external blocking intervals", *Proceedings Second International Workshop on RTCSA '95*, pp. 54-59, 1994.

[4] A. Wellings, M. Richardson, A. Burns, N. Audsley, K. Tindell, "Applying new scheduling theory to static priority preemptive scheduling." Report RTRG 1921120, Dept. of Computer Science, Univ. of York.

[5] J. P. Lehoczky, and S. Ramos-Thuel, "An optimal algorithm for scheduling soft-aperiodic tasks in fixed-priority preemtive systems", in *Proceedings of the real-Time Systems Symposium*, pp. 110-123, Dec. 1992.

[6] S. R. Thuel and J. P. Lehoczky, "Algorithms for scheduling hard aperiodic tasks in fixed-priority systems using slack stealing", in *Proceedings of IEEE Real-Time Systems Symposium*, pp. 22-33, Dec. 1995.

[7] T. F. Abdelzaher, V. Sharma, and C. Lu, "A Utilization bound for aperiodic tasks and priority driven scheduling", *IEEE Transactions on Computers*, vol. 53, no. 3, pp. 334-350, Mar 2004.

[8] T. F. Abdelzaher and C. Lu, "Schedulability analysis and utilization bounds for highly scalable real-time services", in *Proceedings of IEEE Real-Time Technology and Applications Symposium*, May 2001.

[9] T. F. Abdelzaher and B. Anderson, J. Jonsson, V. Sharma, and M. Nguyen. "The aperiodic multiprocessor utilization bound for liquid tasks." in *Real-time and Embedded Technology and Applications Symposium*, San Jose, California, September 2002.

[10] T. F. Abdelzaher and V. Sharma. "A synthetic utilization bound for aperiodic tasks with resource requirements." in 1*5th Euromicro Conference on Real-Time Systems*, porto, Portugal, July 2003.

[11] J. Park, M. Ryu, and S. Hong, "Deterministic and statistical admission control for QoS-aware embedded systems", *Journal of Embedded Computing*, vol. 1, no. 1, 2004.

[12] J. P. Lehoczky, "Real-time queueing theory", in *Proceedings of IEEE Real-Time Systems Symposium*, pp. 186-195, Dec. 1996.

문석환(Seok-Hwan Moon)

2001년 단국대학교 전자계산학과 이학석사
2009년 단국대학교 전자계산학과 이학박사
2006년 ~ 현재 영동대학교 임베디드소프트웨어학과 교수
※관심분야 : 실시간 스케줄링, 실시간 운영체제, 임베디드 시스템