

## 다중프로세서 시스템에서 개선된 합성 이용율을 이용한 혼합 태스크 스케줄링

문석환\*

### Mixed 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 실시간 비주기 태스크 스케줄링 방법 중 하나인 합성 이용율 방법은 주기 태스크들을 고려하지 않고 단지 비주기 태스크들을 위한 스케줄링 방식이다. 하지만 실제로 비주기 태스크는 대부분의 경우에 주기 태스크와의 혼합된 형태로 스케줄링이 이루어지며, 주기 태스크의 스케줄링을 보장하면서 비주기 태스크의 스케줄링 가능성을 판단해야 한다. 본 논문에서는 다중프로세서 시스템에서 주기태스크와 비주기 태스크가 혼합된 태스크 집합을 개선된 합성 이용율을 이용하여 스케줄링하기 위한 방법을 제시하였으며, 기존의 비주기 서버를 이용하여 혼합 태스크 집합을 스케줄링 하는 방법보다 스케줄링 성능이 향상됨을 보였다.

#### ABSTRACT

Synthetic utilization on multiprocessor system is not considered periodic tasks, except scheduling methods for aperiodic tasks where one of the real-time aperiodic tasks is a scheduling method. But really aperiodic tasks scheduling method is composed of mixed task types. Aperiodic task scheduling method guarantee an analysis of the schedulability of aperiodic task. The set of mixed tasks periodic and aperiodic tasks scheduling method uses improved synthetic utilization that is presented in this paper. The new method shows that schedulability increases aperiodic server method.

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

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

접수일자 : 2014. 10. 23 심사완료일자 : 2014. 11. 28 게재확정일자 : 2014. 12. 11

\* **Corresponding Author** Seok-Hwan Moon(E-mail:shmoon@yd.ac.kr ,Tel+843-740-1524  
Department of Embedded Software, Youngdong University, Youngdong 310, Korea

**Open Access** <http://dx.doi.org/10.6109/jkiice.2015.19.2.351>

print ISSN: 2234-4772 online ISSN: 2288-4165

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License(<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.  
Copyright © The Korea Institute of Information and Communication Engineering.

## I. 서 론

태스크의 수행결과와 정확성뿐만 아니라 처리시간의 제한인 종료시한(deadline)을 지키는 것을 요구하는 시스템을 실시간 시스템(real-time system)이라 한다. 이러한 실시간 시스템은 종료시한이 엄격하게 지켜져야 하는 경성 실시간(hard real-time) 시스템과 그렇지 않은 연성 실시간(soft real-time) 시스템으로 구분 할 수 있다. 또한 일정 시간마다 도착하여 반복적으로 실행되는 주기(periodic) 태스크와 도착 시간이 정해져 있지 않은 비주기(aperiodic) 태스크로 구분할 수 있다. 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 있다.

최근에 제시된 프로세서 이용율을 이용한 합성 이용율(synthetic utilization) 알고리즘은[3-5] 비주기 태스크로만 구성된 태스크 집합들을 스케줄링 하는 방법으로 제시되었으며, [6]에서는 개선된 합성 이용율을 통해 비주기 태스크들의 스케줄링 가능성을 증가시키는 방법이 연구되었다.

주기와 비주기 태스크가 혼합되어 스케줄링되는 경우에는 주기 태스크의 스케줄링 가능성을 우선 보장하면서 비주기 서버를 이용하여 비주기 태스크들을 스케줄링한다. 비주기 태스크의 수행은 비주기 서버의 대역폭에 의해 제한되기 때문에 주기 태스크의 스케줄링에 영향을 주지 않으면서 비주기 태스크들을 스케줄링할 수 있다. 이러한 방법을 사용하는 알고리즘으로 다중프로세서 시스템에서 대역폭 서버(total bandwidth server, TBS)[7]를 이용하는 알고리즘이 있다. 또한 이기종의 다중프로세서 환경에서의 실시간 스케줄링 방법도 연구되었다[9]. 본 논문에서는 경성 실시간 시스템에서 EDF 스케줄링에 의해 비주기 태스크들만을 스케줄링 하는 합성 이용율 방법을 확장하여 주기와 비주기 태스크가 혼합된 경우의 태스크 스케줄링 방법을 제시하고, 다중프로세서 시스템에서 TBS 알고리즘과 비교 분석하였다.

본 논문은 2장에서 다중프로세서 시스템에서 비주기 태스크만을 대상으로 하는 합성 이용율을 기술한 후 이

것을 확장하여 주기태스크와 비주기 태스크가 혼합된 태스크 집합들을 스케줄링 하는 방법을 제시한다. 3장에서는 모의실험 결과를 보이고, 마지막으로 4장에서는 결론 및 향후 연구 과제에 대하여 기술한다.

## II. 다중프로세서 시스템에서의 혼합 태스크 스케줄링

다중프로세서 시스템에서 기존의 합성 이용율 방법은 비주기 태스크들만을 고려하였기 때문에 주기태스크와 혼합된 태스크 집합에 적용하기 어려웠다. 합성 이용율을 이용하여 주기 태스크를 스케줄링하기 위해서는 주기 태스크를 고려한 구간에서 합성 이용율을 적용할 수 있어야 하고 주기 태스크를 비주기 태스크로 변환해야 한다.

기존의 합성 이용율은 임의시간  $t$ 의 시점에서 프로세서의 이용율을 구하는 방법으로  $t+1$ 과 같은  $t$ 이후에 도착되는 태스크들에 대한 고려를 하지 않는다. 이러한 문제를 해결하기 위해 특정구간  $[t, t']$ 에서의 합성 이용율을 제시하고 이것을 확장하여 주기 태스크에 적용하였다.

주기 태스크의 특성상, 앞으로 발생하는 인스턴스들의 도착을 미리 예측하여 이것을 이용율 계산에 적용할 수 있다면 주기 태스크들에 대해서도 합성 이용율을 이용한 스케줄링 기법을 제시할 수 있다.

비주기 태스크 집합  $T_a$ 를  $\{T_1, T_2, T_3, \dots, T_i, \dots\}$ 라 할 때,  $T_1$ 은  $T_2$ 보다 먼저 도착(arrival)된 태스크라 가정 한다. 즉  $T_i$ 는  $T_{i+1}$ 보다 먼저 도착한 비주기 태스크 이다. 비주기 태스크  $T_i$ 의 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)  $d_i$ 는  $A_i + D_i$ 로서 정의된다.),  $m$ 을 프로세서의 개수라 할 때 다중프로세서 시스템에서 임의의 시점  $t$ 에서의 비주기 태스크를 위한 합성 이용율[8] 은 다음과 같이 정의한다.

$$U(t) = \frac{1}{m} \sum_{T_i \in S(t)} \frac{C_i}{D_i} \leq 0.59 \quad (1)$$

식(1)을 주기 태스크에 적용하기 위해서는 임의의

구간에서 합성 이용율을 계산할 수 있어야 한다. 그 이유는 주기 태스크는 매주기마다 인스턴스들이 도착되어 실행되기 때문에 각 주기의 구간마다 스케줄링 가능성을 보장해 주기 위해서이다. 먼저 특정 구간에서 비주기 태스크의 합성 이용율을 구하기 위해  $m=1$ 이라 가정하고, 현재 요청집합  $S(t)$ 를  $S(t, t')$ 로 확장한다.  $S(t, t') = \{T_i | A_i \leq t', D_i > t\}$ 로 정의할 수 있으며 여기서  $t \leq t'$ 이다.

그림 1은 특정구간에서의 현재 요청 집합에 대한 예제이다.

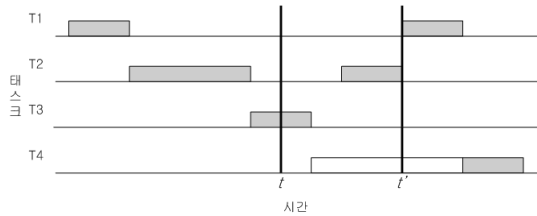


그림 1.  $[t, t']$ 에서의 합성 이용율  
Fig. 1 Synthetic Utilization at  $[t, t']$

여기서  $S(t) = \{T_1, T_2, T_3\}$ 이지만  $S(t, t') = \{T_1, T_2, T_3, T_4\}$ 이다. 이러한 방법으로 특정구간의 현재 요청 집합을 구하여 합성 이용율을 계산할 수 있다. 또한  $S(t, \infty)$ 를 생각한다면, 시점  $t$ 에서 그 이후의 모든 태스크들에 대해서 합성 이용율을 정의할 수 있다. 주기 태스크를 비주기 태스크로 변환하면 합성 이용율을 적용할 수 있는데 주기 태스크 집합  $T_p = \{T_1, T_2, \dots, T_n\}$ 에서 주기 태스크의 모든 인스턴스들을 비주기 태스크로 생각할 수 있다.

즉 주기 태스크의 임의의 인스턴스  $T_{i,j}$ 는 도착시각이  $A_k = A_i + (j-1)T_i$ 이고, 종료시한이  $d_k = T_i$ 인 비주기 태스크  $T_k$ 로 생각할 수 있다. 이 두 가지 방법을 이용하고, 비주기 태스크 집합  $T_a$ 가 EDF에 의해 스케줄링 될 때 합성 이용율  $U(t) = \sum_{T_i \in S(t)} \frac{C_i}{D_i} \leq 1$ 을 이용하여 다중프로세서 시스템에서 주기 태스크와 비주기 태스크가 혼합된 태스크 집합에 대해 합성 이용율을 적용할 수 있다. 프로세서의 개수  $m = 1$ 이고, 비주기 태스크 집합을  $T_a$ , 주기 태스크 집합을  $T_p$ (단,  $T = D$ ), 그

리고 변환된 주기 태스크 집합을  $T_p'$ 이라할 때  $T_a$ 와  $T_p'$ 의 합집합을  $T_s$ 라 하자. 비주기 태스크가 도착할 때 마다 모든  $T_i \in T_s$ 에 대해 EDF에 의한 방식을 사용하므로 합성 이용율  $U_{T_s}(t, t') \leq 1$ 가 만족되면 비주기 태스크  $T_i$ 는 스케줄링 가능하다. 하지만 이 방법은 모든 주기 태스크의 인스턴스에 대해 구간별로 모두 검사해야 하므로 실제로는 불가능하다. 또한 구간  $[t, t']$ 에서 현재 시간을  $t$ 라 할 때 현재 시간  $t$ 부터  $t'$ 까지 태스크 정보를 변환된 주기 태스크는 알 수 있지만 비주기 태스크는 전혀 알 수 없기 때문에 구간을 이용한 방법은 실제 불가능하다.

하지만 주기 태스크와 비주기 태스크와의 관계를 이용하면 이러한 문제를 해결할 수 있다. 주기 태스크 집합  $T_p$ 가 EDF에 의해 스케줄링 될 때 프로세서 이용율은

$$U_{T_p} = \sum_{i=1}^N \frac{C_i}{T_i}$$

이다. 또한 변환된 주기 태스크  $T_p'$ 의 합성 이용율은  $U_{T_p'}(t, t') = \sum_{T_i \in S(t, t')} \frac{C_i}{D_i}$ 로 표현된다. 이

때  $U_{T_p'}(t, t') \leq U_{T_p}$ 의 관계를 가진다. 즉, EDF 알고리즘에 의해 스케줄되는 원래의 주기 태스크의 프로세서 이용율은 비주기 태스크로서 변환되어 합성 이용율을 적용한 프로세서 이용율 보다 항상 크거나 같다.

다음 표 1은  $U_{T_p'}(t, t') \leq U_{T_p}$ 의 예이다.  $T_1, T_4$ 는 비주기 태스크로  $(A_i, C_i, D_i)$ 가 각각 (1,3,11), (4,2,10)이고  $T_2, T_3$ 은 주기 태스크로  $(A_i, C_i, P_i)$ 가 각각 (3,2,7), (5,1,9)이다. 여기서  $P_i$ 는 태스크  $T_i$ 의 주기다. 먼저  $U_{T_p(2,4)}$ 를 구하면 변환된 주기 태스크  $T_2, T_3$ 에 대해 현재 요청집합은  $S_{(2,4)} = \{T_2\}$ 이므로

$$U_{T_p(2,4)} = \frac{C_2}{T_2} = \frac{2}{7} = 0.28 \text{이다.}$$

표 1. 혼합 태스크 집합  
Table. 1 Mixed Tasks Set

	$A_i$	$C_i$	$D_i(P_i)$	비고
$T_1$	1	3	11	비주기
$T_2$	3	2	7	주기
$T_3$	5	1	9	주기
$T_4$	4	2	10	비주기

또한  $U_{T_p} = \sum_{i=1}^N \frac{C_i}{T_i}$  이므로  $U_{T_p} = \frac{2}{7} + \frac{1}{9} = 0.39$ 이다. 그러므로  $U_{T_p}' < U_{T_p}$  이다. 만일 주기 태스크  $T_3$ 이 존재하지 않는다면  $U_{T_p(2,4)} = U_{T_p} = \frac{2}{7}$  이다. 그러므로  $U_{T_p}' = U_{T_p}$  이다. 또는 현재 요청 집합 구간이  $S_{(2,6)}$ 으로 변경되더라도  $U_{T_p}' = U_{T_p}$  이다. 즉,  $U_{T_p}'$ 는 최대  $U_{T_p}$  값을 가질 수 있다.  $U_{T_S}$ 가 EDF로 스케줄링 될 때  $U_{T_S} = U_{T_a} + U_{T_p}'$ 로 표현할 수 있고  $U_{T_p}'$ 는 최대  $U_{T_p}$ 를 가질 수 있으므로  $U_{T_S} = U_{T_a} + U_{T_p} \leq 1$ 이고  $U_{T_a} \leq 1 - U_{T_p}$ 의 값을 가진다. 즉, 합성 이용율을 이용한 비주기 태스크가  $1 - U_{T_p}$ 의 범위에서 프로세서 이용율을 가진다면 스케줄링 가능하다. 정리하면 주기 태스크와 비주기 태스크의 혼합 집합  $T_s = T_a + T_p$ 는 모든 비주기 태스크  $T_i \in Q$ 에 대해  $U_{T_S} \leq 1$ 을 만족하면 스케줄링 가능하다.

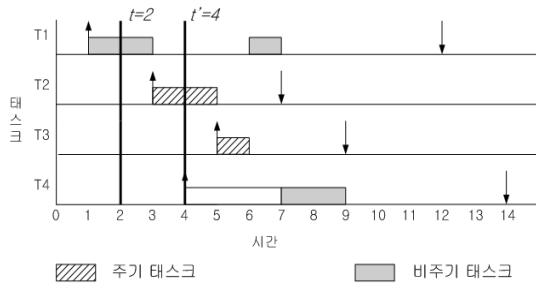


그림 2. 구간 [2,4]에서 주기 태스크의 합성 이용율  
Fig. 2 Periodic Tasks Synthetic Utilization at [2,4]

그림 2에서처럼 EDF에 의한 주기 태스크의 이용율이 0.39이고, 임의의 시점에서 비주기 태스크의 이용율이  $1 - 0.39$ 를 넘지 않으면 주기 태스크 스케줄링을 보장하면서 비주기 태스크들도 스케줄링 가능하다. 이 방법은 앞에서 제시한 구간을 이용하는 방법처럼 모든 구간을 점검하지 않고 주기 태스크의 이용율을 이용해서 비주기 태스크와 주기 태스크가 혼합된 태스크 집합에 적용할 수 있다. 또한 합성 이용율을 [6]에서 제시한 개선된 합성 이용율 방법을 사용하면 비주기 태스크들의 스케줄링 가능성이 증가하게 된다.

### III. 모의 실험

#### 3.1. 모의실험 방법

본 장에서는 주기 태스크와 비주기 태스크가 혼합된 태스크 집합을 다중프로세서 시스템의 TBS 알고리즘과 본 논문에서 제시한 방법을 이용하여 스케줄링 가능성 여부를 비교한다. 모의실험은 인텔 CPU CORE i5, 윈도우즈7 환경에서 실험 하였으며, 주기 태스크는 EDF 방법을 이용하여 주기 태스크의 이용율이 낮은 0.3과, 이용율이 높은 0.7인 경우로 나누어 실험하였다. 또한 비주기 태스크는 1000개를 랜덤하게 생성하여 주기 태스크와 함께 스케줄링 하였다.

입력 워크로드는 비주기 태스크로서 모든 도착된 태스크들의 수행시간의 합과 태스크들이 도착한 시간구간의 비율로 표현할 수 있는데 실험 범위는 50%~150%까지 10%씩 증가되는 시점에서 태스크들의 스케줄링 가능성을 측정하였다.

#### 3.2. 실험 결과

그림 3, 그림 4는  $m=2$ 일 때 주기 태스크의 이용율이 0.3, 0.7인 경우 워크로드에 따른 비주기 태스크들의 스케줄링 가능성 여부를 보여주고 있다. 주기 태스크의 이용율이 높은 경우에, 그리고 워크로드가 큰 경우에는 다중프로세서 시스템에서 TBS와 본 논문에서 제시한 방법의 스케줄링 가능성이 큰 차이를 보이지 않지만 주기 태스크의 이용율이 낮은 경우에는 워크로드가 커지더라도 본 논문에서 제시한 방법이 다중프로세서 시스템에서 TBS에 의한 알고리즘에 비해 약 20%정도 스케줄링 가능성이 향상된 것으로 나타난다. 그 이유는 다중프로세서 시스템에서 TBS 알고리즘의 경우 수행시간이 길고 우선순위가 낮은(종료시한이 긴) 태스크가 먼저 실행되고 수행시간이 짧고 우선순위가 높은(종료시한이 짧은) 태스크가 나중에 도착하여 실행되면 높은 우선순위의 태스크들이 거절당하는 경우가 발생하기 때문이다.

그림 5, 그림 6, 그림 7은 프로세서의 개수별로 다중 프로세서 시스템에서 합성이용율과 TBS 알고리즘의 워크로드별 변화로 워크로드별로 비교해 볼 때 합성이용율의 방법이 TBS 알고리즘 방법보다 더 많은 태스크 집합들을 스케줄링 가능하다는 것을 알 수 있다.

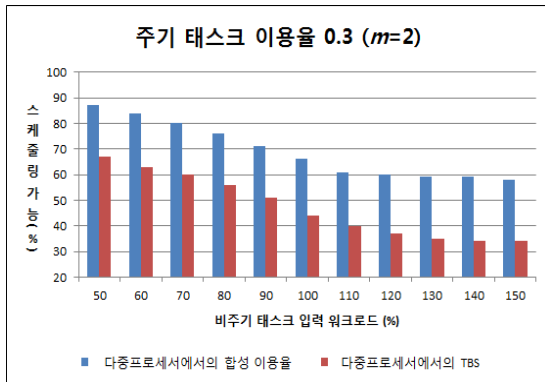


그림 3.  $T_p = 0.3$  인 경우 혼합 태스크 스케줄링  
Fig. 3 Mixed Task Scheduling in  $T_p = 0.3$

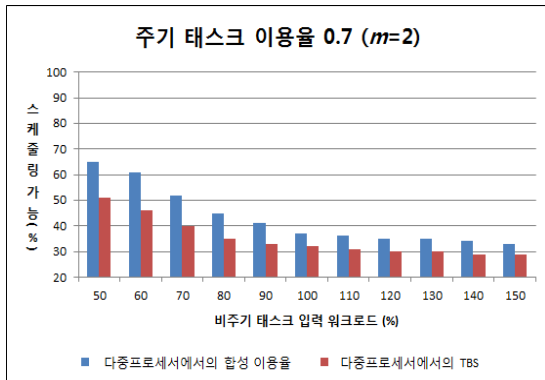


그림 4.  $T_p = 0.7$  인 경우 혼합 태스크 스케줄링  
Fig. 4 Mixed Task Scheduling in  $T_p = 0.7$

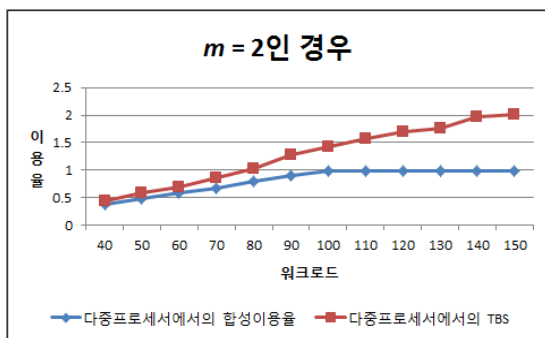


그림 5.  $m = 2$  인 경우 프로세서 이용률  
Fig. 5 Processor Utilization in  $m = 2$

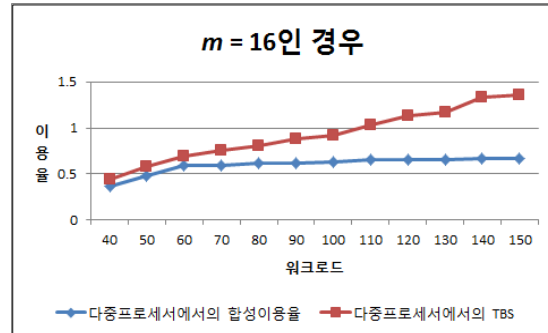


그림 6.  $m = 16$  인 경우 프로세서 이용률  
Fig. 6 Processor Utilization in  $m = 16$

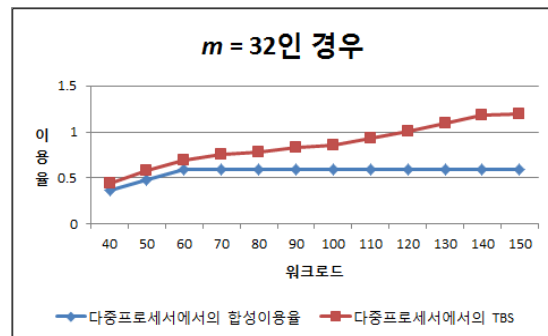


그림 7.  $m = 32$  인 경우 프로세서 이용률  
Fig. 7 Processor Utilization in  $m = 32$

#### IV. 결론

다중 프로세서 시스템에서 비주기 태스크들을 스케줄링하는 경우 대부분의 알고리즘이 주기 태스크를 먼저 스케줄링 한 후 나머지 대역폭을 비주기 태스크에 할당하여 스케줄링하는 방법을 사용한다. 본 논문에서는 EDF 스케줄링 방식을 이용하여 비주기 태스크만을 스케줄링하는 합성 이용률 방법을 주기 태스크에 적용할 수 있도록 변경하였고, 이를 이용하여 주기 태스크와 비주기 태스크가 혼합된 태스크 집합들을 스케줄링하는 방법을 제시하였다. 합성 이용률은 프로세서 이용률을 이용하기 때문에 스케줄링 가능성 여부를 빠르게 판단할 수 있다. 본 논문에서 제시한 다중 프로세서 시스템에서 주기 태스크와 비주기 태스크의 혼합 집합에 대한 스케줄링 가능성 여부는 주기 태스크의 이용률에

따라서 차이를 보이게 되는데 주기 태스크의 이용률이 높아지면 비주기 태스크가 사용할 수 있는 대역폭이 줄어들기 때문에 상대적으로 스케줄링 가능성이 줄어들게 된다. 주기 태스크의 이용률이 낮은 경우 다중프로세서 시스템에서 TBS 알고리즘과 본 논문에서 제시한 방법을 비교하면 약 20%정도 성능 차이를 보이며, 또한 워크로드에 따라 스케줄링 가능성이 다중프로세서 시스템에서 TBS 알고리즘보다 개선된 것을 볼 수 있었다. 본 논문에서는 다중 프로세서 환경에서 주기 태스크와 비주기 태스크의 혼합된 태스크 집합에서의 이용률 측정을 통한 스케줄링 가능성 여부를 판단하였지만 단일 코어, 다중 코어 시스템 환경에서 혼합 태스크 집합의 스케줄링 가능성 판단을 위한 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] 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.
- [4] 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.
- [5] T. F. Abdelzaher and V. Sharma, "A synthetic utilization bound for aperiodic tasks with resource requirements." in *15th Euromicro Conference on Real-Time Systems*, porto, Portugal, July. 2003.
- [6] Seok-Hwan Moon, In-Guk Kim, "A Study on Improved Synthetic Utilization for Real-Time Aperiodic Tasks Scheduling" *Journal of digital contents society* , v.9 no.3, pp.441-448, 2008.
- [7] Shinpei Kato and Nobuyuki Yamasaki, "Scheduling Periodic Tasks using Total Bandwidth Server on Multiprocessors", *IEEE/IFIP International Conference*, vol. 1, pp. 82-89, Dec 2008.
- [8] 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.
- [9] Baruah, Sanjoy, Marko Bertogna, and Giorgio Buttazzo. "Real-time Scheduling upon Heterogeneous Multiprocessors." *Multiprocessor Scheduling for Real-Time Systems*. Springer International Publishing, 2015. 205-211.



문석환(Seok-Hwan Moon)

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