실시간 시스템에서 태스크 이용율을 이용한 스케줄링 가능성 검사☆

Schedulability Test using task utilization in Real-Time system

임 경 현* 서 재 현** 박 경 우***
Kyung-Hyun Lim Jae-Hyeon Seo Kyung-Woo Park

요 약

실시간 스케줄링 알고리즘에서는 비율단조(RM) 스케줄링 알고리즘과 마감시한(EDF) 스케줄링 알고리즘이 가장 일반적으로 사용되고 있다. 이러한 알고리즘에서는 태스크 집합의 전체 이용율 값을 가지고 수행 가능성을 판별하였다. 그러나 임의의 태스크에서 이용율 값이 초과되면 개별 태스크의 한계성을 전혀 예측할 수 없는 문제점이 있었다. 본 논문에서 제안한 알고리즘은 이용율 값이 초과한 태스크를 예측하고, 개별 태스크의 이용율 값을 기반으로 스케줄링 가능성 여부를 판단하는 방법을 제시하였다. 또한, 실시간 시스템에서 스케줄링 가능성 검사의 한계성을 시뮬레이션을 통해 예측하고 결과를 분석하였다.

Abstract

The Rate Monotonic(RM) scheduling algorithm and Earliest Deadline First(EDF) scheduling algorithm are normally used in Real-Time scheduling algorithm. In those scheduling algorithm, we could predict the performance possibility with total utilization value of task group. But, it had problems with prediction of the boundedness in individual task when the utilization value was over in temporary task. In this paper, the suggested scheduling algorithm can predict task when the utilization value was over and it suggested the method of predicting scheduling possibility based on the utilization value of individual task as well. It predicted the boundedness of scheduling possibility test through simulation in Real-Time scheduling algorithm and analyzed the result.

☞ Keyword : Real Time System, Scheduling Algorithm, Schedulability analysis, Rate Monotonic (RM), Earlist Deadline First (EDF)

1. 서 론

실시간 시스템은 각각의 일에 우선 순위를 두어 일을 처리함으로써 시간 제약을 만족시키게 되며, 자원에 대한 소유권을 주어 먼저 처리해야 될 일들이 우선적으로 사용하게 된다. 즉, 실시간 시스템의 동작은 논리적 정확성뿐만 아니라, 시간적 정확성에도 좌우되는 시스템을 말한다[1,2].

실시간 시스템에서 자원 제어와 마감 시간의 문제에 대한 해결법을 제안하는데 있어서 요구되는 속성은 예측성(predictability)과 한계성(boundedness)을 들 수 있다[3,4]. 실시간 스케줄링 방법은 새로운 태스크의 실행을 허가하기 전에 새로운 태스크 집합의 스케줄 가능성(schedulability)을 분석함으로써 시스템 전체의 안정성을 유지할 수 있어야 한다[2,5].

본 논문에서는 타스크간의 의존성관계가 있는 실시간 태스크 집합이 존재함을 가정하고, 개별 태스크 이용율 값은 순차적으로 누적하면서 개별

* 준 회 원 : 목포대학교 대학원 컴퓨터공학과 석사
gylim00@nate.com(제 1저자)
** 정 회 원 : 목포대학교 정보공학부 정보보호전공 부교수
jhseo@mokpo.ac.kr(공동저자)
*** 정 회 원 : 목포대학교 정보공학부 컴퓨터공학전공 부교수
kwpark@mokpo.ac.kr(공동저자)
☆ 이 논문은 2003년도 목포대학교 학술연구비에 의하여 연구되었음.
[2004/03/08 투고 - 2004/03/16 심사 - 2004/09/16 심사완료]

태스크의 한계성을 예측하는 방법을 사용하였다. 성능 평가는 태스크를 순차적으로 누적하면서 이용율 값이 초과하는 태스크를 검사하는 방법을 제시하였다.

본 논문은 다음과 같은 순서로 구성되어 있다. 2장에서는 실시간 시스템, 실시간 스케줄링(정적·동적 스케줄링 알고리즘), 스케줄링 가능성에 대해서 살펴보고, 3장에서는 개별 태스크 이용율 알고리즘을 언급한 다음 4장에서는 시뮬레이션을 결과를 논한 후 5장에서 본 논문의 결과로서 알고리즘에 대한 평가와 연구 방향을 기술하였다.

2. 관련 연구

2.1 실시간 시스템

실시간 시스템이란 제한된 응답시간이나 오동작을 포함한 결과의 위험성에 관한 명확한 규정을 만족하는 시스템이라고 정의되고 있다[4].

일반적으로 실시간의 실행시간 제약조건에 따라 경성 실시간 시스템(Hard Real-Time), 준경성 실시간 시스템(Firm Real-Time), 연성 실시간 시스템(Soft Real-Time) 세 가지로 분류할 수 있다[6].

경성 실시간 시스템은 외부 이벤트에 대해 명시된 시간 내에 응답을 하지 못했을 경우 완전한 실패로 여겨지는 시스템이다. 준경성 실시간 시스템은 경성과 연성의 중간 형태로 마감 시간을 넘겨 수행을 마치는 것은 무의미한 경우를 의미하며 손실이 치명적이지 않는 경우를 말한다. 연성 실시간 시스템은 시간 제약조건을 만족시키지 못하더라도 경성의 경우처럼 치명적이지 않고 마감시간을 넘겨 수행하여도 계산의 결과가 의미가 있는 경우를 말한다.

2.2 실시간 스케줄링

실시간 시스템에 의해 스케줄링 되고 수행되는 개별 단위 일(work)을 하나의 작업(job)이라 하고, 서로간 밀접하게 연관되어 하나의 시스템 기능을 제공하는 여러 작업들의 집합을 하나의 태스크(task)라 한다. 즉, 이벤트가 발생했을 때 실행되는 태스크의 실체(instance)를 해당 태스크의 작업이라 한다.

실시간 태스크는 시간 속성(attributes)으로 그 태스크를 정의할 수 있다. 시간 속성은 해당 태스크의 시간 제약 및 시간 행위를 의미한다. 태스크의 도착 시간 (arrival time)은 그 태스크가 프로세서에 의해 언제라도 실행 가능하게 준비되어진 시점을 의미한다. 태스크의 실행 시간(execution time)은 해당 태스크의 실행을 완료하기 위해 요구되어지는 최악의 경우 할당 시간을 의미한다. 태스크의 만기(deadline)는 그 태스크의 실행이 완료해야 하는 요구된 시점을 의미한다[6].

2.2.1 정적 스케줄링 알고리즘

정적 스케줄링 알고리즘은 스케줄작업 수행이전에 태스크의 우선순위가 정해져 있고, 새로운 태스크 작업을 위해 스케줄 우선순위를 할당할 수 없는 상태이다. 고정 우선순위 기반 스케줄링은 가장 널리 사용되고 있는 방법으로 비율단조(RM) 스케줄링 기법이며 Liu와 Layland[7]에 의해서 제안되었다. 이들은 우선 태스크가 상대적인 마감시간과 주기 시간이 같음($D_i = P_i$)을 가정하였다. RM은 개별 태스크의 주기의 역수인 빈도율(Rate)이 높을수록, 즉 주기가 짧을수록 더 높은 우선순위를 부여하는 방식이다. 따라서, 각 태스크의 주기가 주어지면 태스크들간의 우선순위는 정적으로 결정될 수 있으며 이 우선순위는 시스템이 수행되는 동안 고정된다[5,7,8].

단일 프로세서 환경에서 비율 단조 스케줄링 알고리즘을 사용하는 경우 태스크의 집합의 스케줄 가능성은 주어진 주기 태스크들의 집합이 프로세서 이용율(Utilization, U)이 $n(2^{1/n}-1)$ (n의 값은 태스크의 수) 보다 작거나 같으면 스케줄링이 가능하다고 확인되며, 이에 의해 확인되지 못하는

경우는 동시에 시작된 개별 태스크의 첫 번째 작업은 수행시간(Completion time)을 구한 값이 마감 시간보다 작은가의 여부에 따라 결정된다.

Liu와 Layland는 RM 태스크 스케줄링 주기를 P_i로, 수행시간은 C_i로 표기하면 태스크의 프로세서 이용율(U)은 C_i/P_i으로 표시될 수 있고, 태스크 집합의 프로세서 이용률은

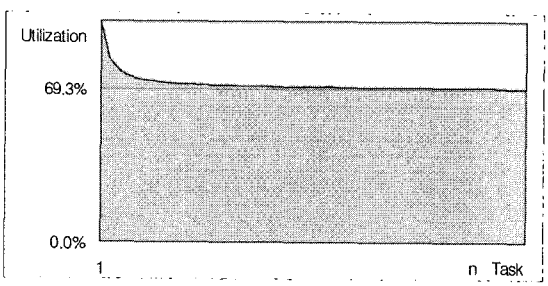
$$U=\frac{C1}{P1}+\frac{C2}{P2}+\frac{C3}{P3}+\cdots+\frac{Cn}{Pn} \quad (1)$$

으로 표시되며, 다음과 같은 조건이 성립되면 독립적인 주기 태스크들의 집합은 각 태스크의 마감 시간을 만족시킬 수 있다.

$$U \leq U(n) = n(2^{1/n}-1) \quad (2)$$

이 식(2)에서 값은 태스크의 개수 n이 증가함에 따라 그림 1과 같이 점점 감소하여 69.3%에 수렴한다.

RM 스케줄링 기법은 69.3%보다 작은 태스크 집합이 스케줄 가능함을 의미한다. 그러나 식(2)에 의한 스케줄 가능 분석 방법은 정확한 것이 아니다. 이 조건식을 만족시키지 못하는 태스크 집합이라 할지라도 RM 스케줄링 기법에 의해서 타당한 스케줄이 생성될 수 있다[2].



〈그림 1〉 RM 스케줄링 필요충분조건

2.2.2 동적 스케줄링 알고리즘

마감시간 우선(EDF) 스케줄링 알고리즘은 동적 우선 스케줄링 정책에서 가장 흔히 사용되고 있는 스케줄링 방법으로 Liu와 Layland[7]에 의해서 제안되었다. 태스크의 우선순위는 마감 시간에 따라서 할당된다. 즉, 마감시간이 짧을수록 높은 우선 순위가 할당되므로, 임의의 순간에 실행되는 태스크는 실행이 완료되지 않은 태스크들 중에서 마감 시간이 가까운 것을 선택한다. 정적 스케줄링 알고리즘과 달리 태스크의 우선순위가 시간에 따라 변하게 된다[2,9].

마감 시간 스케줄링 알고리즘에서 다음의 조건이 만족되면 주기적으로 발생하는 독립적인 n개의 태스크들은 스케줄 가능하다고 할 수 있다. 이때, 태스크의 주기는 P_i로 수행 시간은 C_i로 표기한다.

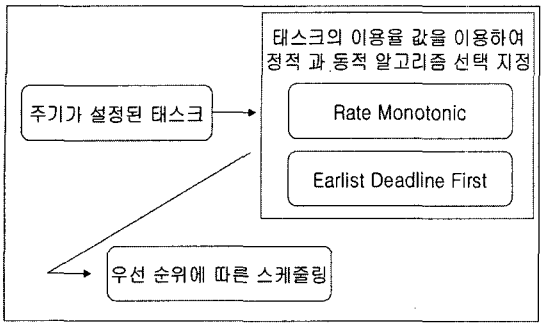
$$U=\frac{C1}{P1}+\frac{C2}{P2}+\cdots+\frac{Cn}{Pn} \leq 1 \quad (3)$$

즉, EDF 스케줄링 방법은 태스크 집합이 수행 가능(feasible) 집합이라면 이용율에서 1까지 사용할 수 있다는 점이다. 만일 1을 초과하는 태스크 집합이 있다면 다른 어떤 스케줄링 방법을 사용하더라도 스케줄 될 수 없다.

2.3 스케줄링 가능성(Schedulability)

스케줄링 중에 새로운 태스크로 인해 시스템이 과부하 상태일 경우 그 태스크와 새로운 태스크뿐만 아니라 기존의 태스크들 역시 마감 시간을 지키지 못할 수 있다[2,10,11]. 그러므로 태스크의 실행을 허가하는 것이 시스템 전체의 안정성을 손상시키는지 검사하기 위한 방법이 필요하다. 보통 스케줄링 가능성 분석(schedulability analysis)방법은 주어진 스케줄링 방법으로 스케줄 가능한 모든 태스크 집합에 대해서는 "예"라고 지시할 수 있고, 그렇지 않은 태스크 집합에 대해서는 "아니오"라고 지시 할 수 있는 스케줄링 가능성 분석 방법이 있다면 그러한 방법을 정확한(exact) 스케줄 가능성 분석 방법이라 한다[2].

스케줄러는 시스템에 존재하는 모든 태스크를



〈그림 2〉 태스크의 이용율 값을 이용한 스케줄링

사전에 정의된 기준에 의해 현재 이용 가능한 프로세서들에게 할당하는 것을 의미한다. 각 태스크가 만기 전에 실행을 완료하였을 경우, 해당 스케줄이 가능하다면 해당 태스크들의 집합은 스케줄링 가능성 알고리즘에 의해 스케줄 가능하다라고 말할 수 있다[10]. 스케줄링 가능성에서 예측성이란 시스템에 정의된 고장이나 작업 부하 조건에 태스크 종료 시간의 만족을 보장하는 것을 의미한다[6].

스케줄이 정적으로 결정되어 있기 때문에 이러한 스케줄링 방법을 오프라인(off-line) 스케줄링이라고도 한다. 그리고 실시간 스케줄링 방식은 시분할 스케줄링 방식에서와 마찬가지로 선점(Preemption) 가능한 방식과 불가능한 방식으로 구분되는데 대부분의 우선 순위 기반 스케줄링 방법은 선점형 방식에 따른다.

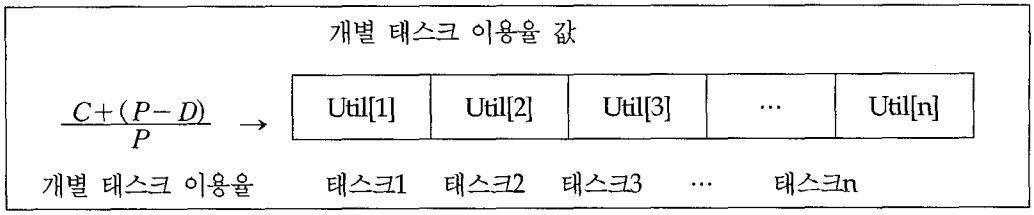
3. 개별 태스크 이용율 알고리즘

기존에 있는 태스크 전체 이용율 알고리즘[8,12,13]과 본 논문에서 제안한 태스크 개별 이용율 알고리즘은 그림 2에서와 같이 우선 순위에 의해서 태스크의 실행 시간, 마감 시간, 주기 시간 등의 주기 값이 설정되어 있다. 스케줄링 가능성 분석을 위해서 이용율 공식을 이용하여 가장 적절한 스케줄러의 우선 순위를 설정한 것은 태스크가 사전에 한계성에 대한 문제가 발생할 수 있는 점을 고려하여 설계되어졌기 때문이다. 이용율 값으로 스케줄링 가능성을 판단하며 RM 우선 순위 스케줄러가 불가판정을 받게되면 EDF에서 이용율 값을 판단하여 문제가 발생하면 스케줄러는 최종적으로 정적 및 동적 스케줄링을 할 수 없다. 그림 2에서는 태스크 이용율값을 가지고 스케줄링 필요충분조건에 의해 가장 적절한 스케줄링을 선택 한다.

그림 2에서는 주기가 설정된 태스크가 있다고 가정하고 태스크의 이용율 값을 이용하여 RM 스케줄링 가능조건이 충족하면 RM 스케줄링 우선 순위에 따라 스케줄링을 선택한다. 다른 하나는 EDF 스케줄링 가능조건이 충족하면 EDF 스케줄링 우선 순위에 따라 스케줄링을 선택한다. RM과 EDF 스케줄링 가능조건이 충족하지 못하여 문제가 발생하는 경우가 있다. 기존 실시간 시스템의 스케줄 가능성 검사는 스케줄링에서 임의의 태스크가 주어진 마감 시간과 실행 시간을 수행하면 작업 가능과 불가능에 대해서 예측할 수 있었지만, 태스크의 마감 시간과 주기 시간을 초과하는 한계성을 예측 할 수는 없었다. 본 장에서 제안 하고자 하는 부분은 태스크를 가지고 이용율 값을 구한후 스케줄링 필요충분조건에서 임의의 태스크를 초과하는지를 알 수 있는 스케줄 가능성 검사이다.

```
.....
RMS='Y';
for( i=0 ; i < 태스크 수 ; i++) {
 if( 주기시간(P) != 마감시간(D) )
  RMS='N';
Util[i] =  ( 실행시간(C) + ( 주기시간(P) - 마감시간(D) )) / 주기시간(P)
.....
```

〈그림 3〉 개별 태스크 이용율 값 처리



〈그림 4〉 제안한 스케줄 가능성에서 이용율 값 처리

3.1 주기 시간과 마감 시간을 비교하여 처리

기존 알고리즘[12]은 처리과정에서 마감시간과 주기시간을 비교하여 두 정적·동적 이용율 공식을 사용하였으며, 그림 4에서는 동적 이용율 공식을 이용하여도 동일한 과정을 수행할 수 있으므로 공식 이용율을 간소화 하였다. 개별 태스크 이용율을 사용하여 개별 태스크의 이용율 값을 기억해 두는 과정이다. 그림 3에서는 단 하나 이상의 태스크가 마감 시간과 주기 시간이 다른 경우 RMS 변수 값에 'N'이라고 설정하고, 마감 시간과 주기 시간이 같다면 RMS변수 값이 'Y'값을 설정한다. 주어진 태스크의 집합을 스케줄할 수 있다면 최적의 스케줄링 알고리즘은 항상 실행 가능한 스케줄을 생성할 수 있다는 것을 의미한다. 스케줄 될 수 없는 태스크 집합에 대해서는 다른 어떤 정적 스케줄링으로도 스케줄할 수 없다는 것을 의미한다.

3.2 개별 태스크 한계성 예측 처리

대부분의 실시간 시스템에서는 RM 스케줄링 방법을 사용한다. RM 스케줄링 방법은 필요충분 조건 그림 1에서처럼 이용율을 제한하고 있다. 이에 반하여 EDF 스케줄링 알고리즘은 실시간 시스템에 적용하게 되면 이용율을 100% 사용할 수 있다. 그림 5에 있는 스케줄링 가능성 검사 기법은 아래와 같다. RM 가능성 검사 조건 식(2)와 EDF 가능성 검사 조건 식(3)이다.

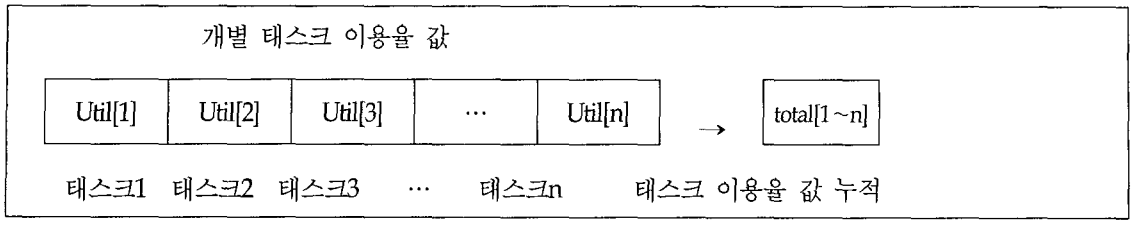
RM 식 (2) ≥ total [1], EDF 식 (3) ≥ total [1]
RM 식 (2) ≥ total [2], EDF 식 (3) ≥ total [2]
.
.
.
RM 식 (2) ≥ total [n], EDF 식 (3) ≥ total [n]

제안한 알고리즘 구조는 그림 5이며 누적 과정은

```
.....
for( i=1 ; i <= 태스크 수 ; i++) {
 for( j=1 ; j <= i ; j++)
   total[i] = total[i] + Util[j]; // 개별 태스크 값을 total[1 ~ n] 변수에 누적

 if( total[i] ≥ 0 and total[i] ≤ 태스크수 * ( ( 2 1/태스크수 ) - 1 ) )
  RM_count = RM_count + 1;   //RM 수행가능 태스크 수
 if( total[i] ≥ 0 and total[i] ≤ 1)
  EDF_count = EDF_count + 1; //EDF 수행가능 태스크 수
}
.....
```

〈그림 5〉 스케줄링 가능성 조건 처리



〈그림 6〉 스케줄링 가능성에서 태스크의 한계성 예측 처리

아래와 같다. 태스크 1의 이용율 값을 total[1]은 누적된 태스크 작업(이용율 값)을 의미한다. 그림 6은 개별 태스크의 이용율 값을 순차적으로 태스크 이용율 값에 누적하는 과정을 의미한다.

```
total [1]=태스크1 이용율 값
total [2]=태스크1+태스크2 이용율 값
total [3]=태스크1+태스크2+태스크3 이용율 값
                    .
                    .
                    .
total [n]=태스크1+태스크2+태스크3+…
    +태스크n 이용율 값
```

RM 스케줄링 가능성 조건 검사 total[1~n]부분에서 문제가 충족되면 RM 수행가능 태스크 수를 증가하고, 문제가 발생하면 RM 수행가능 태스크 수를 증가하지 않는다. 그리고 EDF 스케줄링 가능성 조건 검사 total[1~n]부분에서 문제가 충족되면 EDF 수행가능 태스크 수를 증가하고, 문제가 발생하면 EDF 태스크 수를 증가하지 않는다. 그림 5와 그림 6은 스케줄링 가능성의 부분 중에서 주기 시간이내 수행해야할 마감 시간이 초과(한계성)되는 태스크를 예측하는 처리 과정이다.

3.3 이용율 값을 이용하여 스케줄링 판정 처리

그림 7에서는 RM과 EDF 스케줄링의 수행이 적합한지를 확인하는곳이며, 그림 7에 대한 RM 스케줄링이 수행하려면 total[1~n]값이 RM 식(2)을 이용율 값을 초과하지 않아야 한다. 즉, RM_count 수와 태스크 수가 같을때 RM 스케줄링을 수행할 수 있도록 설정되어 있다.

RM 식 (2) ≥ total [1]	EDF 식 (3) ≥ total [1]
RM 식 (2) ≥ total [2]	EDF 식 (3) ≥ total [2]
:	:
RM 식 (2) ≥ total [n]	EDF 식 (3) ≥ total [n]

태스크들은 RM에서 수행 불가능한 태스크 집

```
.....
if( RMS == 'Y' and RM_count == 태스크 수)
   RM schedule;          // RM 스케줄링 실행
 else if( EDF_count == 태스크 수 )
        EDF schedule;  // EDF 스케줄링 실행
      else
         return -1;      // RM, EDF 스케줄링 불가능
.....
```

〈그림 7〉 재구성한 선택 알고리즘에서 적합한 스케줄링 판정 처리

합이 있다면 EDF 스케줄링 가능성을 검사하며 위 식에서 모든 이용율 값이 충족할 때 EDF 스케줄링을 수행한다. 그림 7에서 EDF_count와 태스크 수가 같을때 EDF 스케줄링을 수행할 수 있도록 설정되어 있다. RM 스케줄링에서는 보통 실행시간과 주기시간이 있다. EDF 스케줄링은 주기시간과 마감시간이 동일하다고 설정하는 방법과 주기시간과 마감시간이 틀린 설정 두가지 방법이 있다. 즉, 주기시간과 마감시간이 틀리다면 RM 스케줄링에서 수행 불가능하므로 EDF 스케줄링을 설정하였다[14].

4. 시뮬레이션

4.1 시스템 구성

본 장에서는 기존 실시간 시스템의 태스크 전체 이용율 알고리즘과 제안한 태스크 개별 이용율 알고리즘의 성능을 비교 분석하기 위한 시뮬레이션을 하였다. 시뮬레이션 환경은 시분할 운영체제인 리눅스 커널 2.4.18과 GCC 컴파일러의 환경에서 실시간 환경에 맞도록 설정한 태스크 파라미터 값을 사용한다.

4.2 변수 요인별 실험 및 성능 평가

태스크 파라미터 값이 미리 설정되어 있으며 본 논문에서 제안한 개별 이용율 알고리즘을 스케줄 가능성 검사 조건으로 판별하여 태스크가 마감시간이내에 수행가능하면 마감 시간과 주기 시간에 맞는 우선순위 스케줄링을 수행할 수 있도록 구성하였으며, 스케줄링이 불가능하다고 판단 시에는 개별 태스크의 이용율 값을 어떤 변수 값에 하나씩 누적할 때 마다 몇 번째 태스크에서 문제가 생기는지 사전에 한계성을 예측할 수 있는 알고리즘을 구성하였다.

태스크의 스케줄링 가능성 검사 예측성에 대한 다양한 모델이 존재할 수 있고 고려해야 할 많은 변수들이 있기 때문에 시뮬레이션의 복잡성을 피하고, 본 논문에서 도출하고자 하는 알고리즘의 성능 평가를 위해 표 1과 같은 요인으로 정의한다. 태스크의 파라미터 값은 참고문헌[12,13] 자료를 토대로 분석하였다.

4.2.1 마감시간과 주기가 같은 경우

기존 알고리즘에서는 RM 스케줄 가능 검사시 무조건 수행 불가만 내렸지만 개별 태스크에 대한 한계성이 없다. 본 논문에서 제안한 알고리즘에서 RM 스케줄링 가능성 검사시 5번째 태스크에서 이용율을 초과함을 알 수 있었으며, 기존의 EDF 스케줄링 가능성 검사는 개별 검사가 불가능한 점을 확인할 수 있었다. 제안한 EDF 스케줄링방법

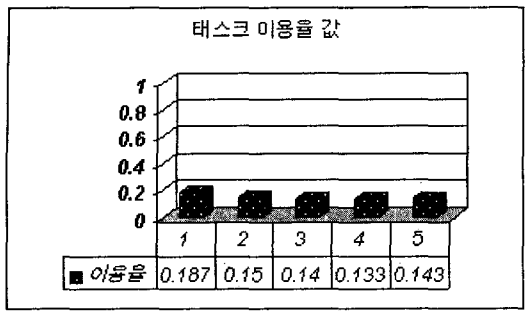
〈표 2〉 마감시간과 주기가 같은 경우의 매개변수

T	1	2	3	4	5
C	56	60	70	80	100
P	300	400	500	600	700

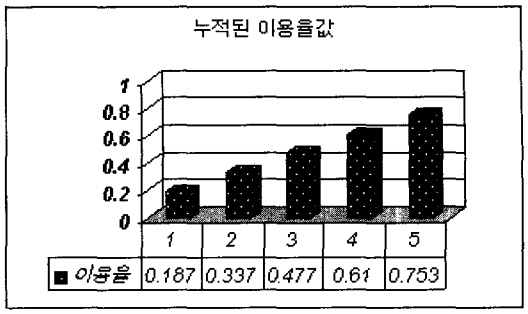
* T : Task, C : Completion Time, P : Periodic Time

〈표 1〉 시뮬레이션에서 사용되는 기본 설정값

적용 변수	변수 값 의미
태스크 ID	오름차순으로 우선순위 설정
실행 시간 (C)	태스크의 마감 시간 이전에 작업을 완료할 수 있는 변수 값
마감 시간 (D)	태스크의 주기 시간과 동일하거나 그 보다 작은 변수 값
주기 시간 (P)	태스크를 주기적으로 작업을 처리할 수 있는 변수 값
이용율 값 (Util[1~n])	$=\frac{C+(P-D)}{P}$ 변수 값



〈그림 8〉 개별 이용율 값 실험 #1



〈그림 9〉 제안한 알고리즘 수행시 이용율 값 실험 #1

에서는 개별 및 전체 수행 동안 스케줄링이 실패하지 않았다. 제안한 RM 스케줄링 가능성 검사에서는 태스크의 한계성을 예측할 수 있으며, 전체 이용율을 이용할 경우 EDF 스케줄링만 선택한다.

이 태스크 집합의 스케줄링 가능성 검사는 각 개별 태스크의 스케줄링 가능성 검사를 위하여 RM 가능성 검사는 식(2)와 EDF 가능성 검사는 식(3)을 충족하여야 하며, 조건 식(2) · 식(3)에 의해 전체 태스크 집합 스케줄링 검사를 통과하기 때문이다.

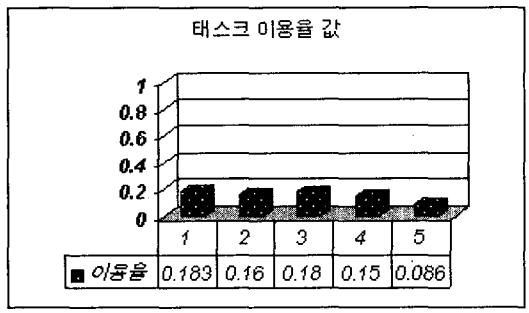
식(2)을 사용하여 RM의 개별 스케줄링 가능성 검사하면 태스크 5번째에서 아래와 같이 문제가 발생한다.

Task 1의 total[1] = 0.187 $\leq$ 0.743 = $5(^{1/5}-1)$
Task 2의 total[2] = 0.337 $\leq$ 0.743 = $5(^{1/5}-1)$
Task 3의 total[3] = 0.477 $\leq$ 0.743 = $5(^{1/5}-1)$
Task 4의 total[4] = 0.61 $\leq$ 0.743 = $5(^{1/5}-1)$
Task 5의 total[5] = 0.753 $>$ 0.743 = $5(^{1/5}-1)$

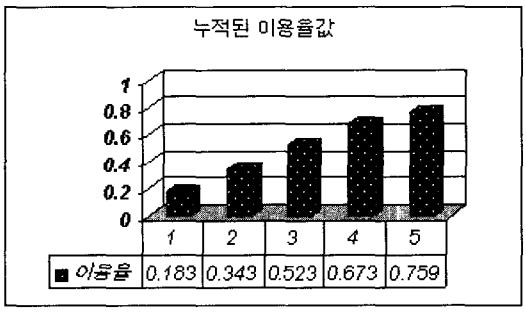
식(3)을 사용하여 EDF의 개별 가능성 검사하면 아래와 같이 별 문제 없이 수행한다.

Task 1의 total[1] = 0.187 $\leq$ 1
Task 2의 total[2] = 0.337 $\leq$ 1
Task 3의 total[3] = 0.477 $\leq$ 1
Task 4의 total[4] = 0.61 $\leq$ 1
Task 5의 total[5] = 0.753 $\leq$ 1

RM, EDF 알고리즘에서도 각 개별 태스크의 스케줄링 가능성 검사를 수행함을 확인할 수 있다. 그림 9에서 태스크 이용율을 보면 누적(태스크 5 : 전체 태스크 이용율 값)된 이용율 값이 0.753임을 알 수 있다. RM과 EDF를 동일한 주기로 설정되어있어 EDF가 RM의 주기적인 특성을 갖는다. 표 3의 기존의 RM 검사 수식 $n(^{1/n}-1)$의 조건값이 0.743이고 제안한 알고리즘에 의한 누적된 이용율 값 0.753과 같지 않아 RM 알고리즘 수행은 불가능하다. 따라서, EDF 알고리즘을



〈그림 10〉 개별 이용율 값 실험 #2



〈그림 11〉 제안 알고리즘 수행시 이용율 값 실험 #2

수행할 수 있는 조건값에 충족됨을 알 수 있고 선택 알고리즘에서는 EDF 알고리즘을 수행할 수 있게 된다.

4.2.2 마감시간과 주기가 다른 경우

주기시간과 마감시간이 다른 경우에는 RM 스케줄링을 수행할 수 없다. 기존 EDF 스케줄링 가능성 검사 알고리즘은 전체적으로만 태스크가 수행을 판별하고, 제안한 EDF 스케줄링 가능성 검사 알고리즘에서는 개별 및 전체 검사를 수행한다.

식(2)을 사용하여 RM의 개별 스케줄링 가능성 검사하기 위하여 주기와 마감시간이 같은 조건이 필요하므로 RM 스케줄링 가능성 검사를 수행 할 수 없다.

〈표 3〉 마감시간과 주기가 다른 경우의 매개변수

T	1	2	3	4	5
C	20	30	40	50	60
P	300	400	500	600	700
D	265	366	450	560	700

* T : Task, C : Completion Time, P : Periodic Time, D : Deadline Time

식(3)을 사용하여 EDF의 개별 가능성 검사하면 아래와 같이 별 문제 없이 수행한다.

Task 1의 total[1] = 0.183 ≤ 1
Task 2의 total[2] = 0.343 ≤ 1
Task 3의 total[3] = 0.523 ≤ 1
Task 4의 total[4] = 0.673 ≤ 1
Task 5의 total[5] = 0.759 ≤ 1

그림 11에서 전체 이용율을 가지고 스케줄링 가능성 검사를 판단할 경우에는 기존의 알고리즘에서 어떤 태스크에서 문제가 발생하는지를 예측할 수 없지만 제안한 알고리즘에서는 정확히 태스크 5번째에서 RM 알고리즘 이용율 필요조건이 초과되었음을 확인할 수 있다. 즉, 그림 11에서 스케줄링한 결과 누적(전체 태스크 이용율 값)한 이용율 값이 0.759임을 알 수 있었고, 만일 RM의 가능성 조건을 수행 할 수 있다면 이용율 상한선이 0.743이므로, 태스크의 5번에서 이용율이 초과 되었다는 것을 확인할 수 있다. 하지만, 기존의 RM 스케줄링은 수행 조건에 위배되므로서 본 논문에서 제안한 알고리즘에 의해 EDF의 수행 가능 조건에 충족됨을 알 수 있고, 종료 시간과 주기가 각기 다르게 부여되었으므로 선택 알고리즘에서 EDF를 스케줄링하게 된다.

5. 결 론

실시간 시스템의 기존 태스크 전체 이용율 알고리즘을 이용하여 주기가 설정된 값을 가지고 스케줄링 가능성 검사 판정시 전체 이용율에서는 전체의 집합의 이용율 값에 대해서 판단하는 것은 오직 실행 가능성과 실행 불가능성의 방법 두 가지만 존재한다. 하지만 본 논문에서 제안한 알고리즘은 개별 이용율이며 개별 집합의 이용율 값을 두어 실행 가능한 조건을 판별시 사전에 몇 번째 태스크에서 문제가 발생하는지 개별 태스크 이용율 값을 사용하여 한계성에 대해서 예측 가능하게 되었다.

제안한 알고리즘을 수행하고 마감 시간과 주기 시간에 따라 스케줄링을 판단하는 조건에 의해서 재구성된 선택 알고리즘은 정적 알고리즘과 동적 알고리즘으로 선택될 수 있도록 설정되어져 있다. 알고리즘에 대한 평가는 다음과 같다. 스케줄링을 수행하기 이전에 마감 시간, 실행 시간, 주기 시간의 태스크 값을 가지고 스케줄링 가능성 검사의 한계성을 예측하여 실시간 시스템에서 사전에 안정적인 시스템을 유지하는 것이다.

본 논문에서는 주기적인 태스크를 이용하여 스케줄링 가능성 검사를 하였지만 추후 연구 방향으로는 주기, 비주기 태스크외 여러가지 요인을 고려한 다양한 스케줄링 가능성 검사 알고리즘의 연구가 필요하다.

참 고 문 헌

[1] An Fredette and Rcleaveland, "A Generalized to Real-Time Schedulability Analysis", 10th workshop on real-time operating system and software, 1993.

[2] 김성관, 하란, "실시간 스케줄링", 한국 정보처리학회 논문지 제5권, 제4호, pp.12-21. 1998. 7.

[3] Krithi Ramamritham, John A. Stankovic, "Scheduling Algorithms and Operating System Supports for Real-Time Systems," Proceedings of the IEEE. Vol. 82, No.1, January pp. 55-67, 1994.

[4] 이은미, 허신, "분산 실시간 시스템에서의 자원 제어기법", 한국 정보과학회 논문지 제28권, 제3호, pp.161-172, 2001. 4.

[5] Manabe, Y. and Aoyagi, S., "A Feasibility Decision Algorithm for Rate Monotonic and Deadline Monotonic Scheduling", RT systems, v.14, no.2, pp.171-182, 1998.

[6] 심재홍, "다양한 실시간 스케줄러를 지원하기 위한 커널 구조화 및 재구성 방안", 아주대학교, 박사학위논문, 2001.

[7] C. L. Liu and J. W. Layland, "Scheduling AlgorIthms for Multiprogramming in a Hard Real-Time Environment", Journal of the ACM, vol. 20, no.1, pp. 46-61, 1973.

[8] John Lehoczky, Lui Sha and Ye Ding, "The Rate Monotonic Scheduling Algorithm : Exact Characterization And Average Case Behavior", Proc. of IEEE Real-Time Systems Symposium, pp.166-171, Dec. 1989.

[9] Chetto. H and Chetto. M, "Some Results of the Earliest Deadline scheduling Algorithm", IEEE Transactions on Software Engineering vol.15, no.10 , pp.1261-1269, Oct. 1989.

[10] 신형식, 김태웅, "실시간 시스템의 개관", 한국 정보처리학회 논문지 제5권, 제4호, pp. 2-11, 1998. 7.

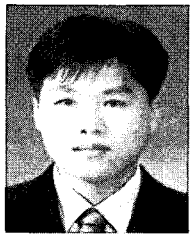
[11] 김창배, "실시간 운영체제의 동일 우선 순위에 대한 다중 작업 지원의 설계및 구현", 부경대학교, 석사학위논문, 2001.

[12] 최정훈, 김경화, 김두상, 최대수, 임종규, 박한규, 구용완, "실시간 리눅스에서 선택 알고리즘을 이용한 스케줄링 성능평가", 한국 정보과학회 가을 학술발표 논문집 Vol. 29, No.2, pp.430-432, 2002.

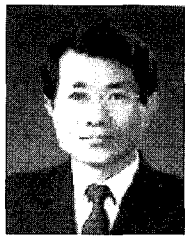
[13] 류진열, 김광, 허신 "Mach 커널의 재구성을 위한 확장된 스케줄 가능성 검사를 수행하는 실시간 스케줄러", 한국 정보처 리학회 논문지 제7권, 제2호, pp.507-519, 2000. 2.

[14] 심재홍, 송재신, 최경희, 박승규, 정기현 "다양한 실시간 스케줄링 알고리즘들을 지원하기 위한 재구성 가능한 스케줄러 모델", 정보과학회논문지 제29권 4호, pp.201-212, 2002. 4.

◐ 저 자 소 개 ◑



임 경 현
2001년 초당대학교 컴퓨터과학과 졸업(학사)
2004년 목포대학교 대학원 컴퓨터공학과 졸업(석사)
관심분야 : 실시간 시스템, 분산 시스템, 데이터베이스, etc.
E-mail : gylim00@nate.com



서 재 현
1985년 전남대학교 계산통계학과 졸업(학사)
1988년 중앙대학교 대학원 전자계산학과 졸업(석사)
1996년 전남대학교 대학원 전산통계학과 졸업(박사)
1996년 9월~현재 목포대학교 정보공학부 정보보호 전공 부교수
관심분야 : 시스템 및 네트워크 보안, 컴퓨터 네트워크, 네트워크 관리 etc.
E-mail : jhseo@mokpo.ac.kr

박 경 우
1986년 전남대학교 계산통계학과 졸업(학사)
1988년 전남대학교 대학원 전산통계학과 졸업(석사)
1994년 전남대학교 대학원 전산통계학과 졸업(박사)
1995년 3월~현재 목포대학교 정보공학부 컴퓨터공학 전공 부교수
관심분야 : 분산시스템, 시스템 소프트웨어, 정보보호 etc.
E-mail : kwpark@mokpo.ac.kr