실시간 시스템에서 이용율을 이용한 스케줄링 가능성 검사

임경현[0*], 손옥희[*], 박경우[**]
[*,**]목포대학교 컴퓨터공학과
e-mail:[0*]gylim00@nate.com

Real-Time System from Utilization Rate use Inspection of Schedulability

Kyung-Hyun Lim[0*], Og-Hee Son[*], Kyung-Woo Park[**]
[*,**]Dept of Computer Engineering, Mokpo National University

요 약

실시간 시스템에서 태스크의 스케줄링은 매우 중요한 역할을 담당한다. 실시간 시스템에서 각각의 태스크들은 제한된 시간이 주어져 있고, 이 태스크들은 제한된 시간 내에 수행되지 못하는 경우에는 큰 피해를 입을수 있으므로 반드시 지켜져야 한다. 이에 대한 사전에 태스크 예측 가능성에 관한 연구가 활발히 진행중이다. 본 논문에서는 각각의 태스크 이용율을 사용하여 예측가능성을 높이기 위한 알고리즘을 제안하며, 기존의 태스크 전체 이용율을 이용한 모델과 비교 분석 하였다.

1. 서론

일반적인 시스템에서는 태스크의 처리량과 컴퓨터 자원의 이용율을 높이기 위해 컴퓨터 자원을 관리해 왔으며 주어진 일들이 쌓이는 대기 행렬에서 모든 작업을 시간 제약없이 처리하였다. 실시간 시스템에서는 각각의 일에 우선 순위를 두어 일을 처리함으로써 시간 제약을 만족시키게 되며, 자원에 대한 소유권을 주어 먼저 처리해야 될 일들이 우선적으로 사용하게 된다. 기존의 컴퓨터 시스템과 달리 시스템 동작의 논리적 정확성 뿐만 아니라, 시간적 정확성에도 좌우되는 시스템을 말한다. 기존의 시분할 시스템이 높은 처리율에 중점을 두었다면 실시간 시스템은 스케줄 가능성(schedulability)에 중점을 둔다.

그리고, 시분할 시스템은 공평성에 그리고 실시간 시스템은 안정성 및 예측성에 그 목표를 두고 있다[1,6,7,10,11,13]. 본 논문에서는 Rate Monotonic과 Earliest Deadline First의 우선 순위 기반 스케줄링 알고리즘의 기반으로 태스크 집합을 제안하고자 하는 알고리즘으로 가능성 분석후 적합한 스케줄러를 선택하고 시스템의 안정된 스케줄링이 되도록 재구성한 선택 알고리즘 기법을 제시하고자 한다.

본 논문은 2장에서는 실시간에 관련된 관련 연구 분야와 RM(Rate Monotonic) 정적 우선 스케줄링 정책과 EDF(Earlist Deadline First) 동적 우선 스케줄링 정책에 대해서 논한다. 3장에서는 제안된 스케줄링 알고리즘에 대해서 제시하고, 4장에서는 Task Parameter 값을 기존에 전체 이용율 알고리즘과 제안한 알고리즘 태스크 이용율을 가지고 예측성에 대해서 평가하고자 한다. 5장에서는 연구의 결과를 통하여 결론 및 연구과제를 제시하여 끝을 맺는다.

2. 관련연구

2.1 스케줄링 가능성(Schedulability)

스케줄러에 의해 생성된 것으로, 시스템에 존재하는 모든 태스크들을 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 말할수 있다.

스케줄링 가능성 검사는 우선순위 구동방식의 알고리즘을 유용하게 사용할 수 있는 기법이다. 시스템에 적합한 스케줄링 기법을 선택하고 태스크의 주기, 실행시한, 마감시한등을 결정할 때 유용하게 판단 할수 있다. 이러한 이유로 시스템에 안정적인 스케줄링을 할수 있도록 다중 프로세서 시스템에서 우선 순위 구동 알고리즘을 많이 사용되고 있다[4,11].

2.2 비율 단조 스케줄링 정책

고정 우선 순위 기반 스케줄링으로서 가장 널리 사용되고 있는 방법으로 RM 스케줄링 기법으로 Liu와 Layland에 의해서 제안되었다[1]. 이들은 우선 태스크를 상대적인 마감시한이 주기와 같음($D_i = T_i$)을 가정하였다. RM은 각 태스크의 주기의 역수인 빈도율(Rate)이 높을수록, 즉 주기가 짧을수록 더 높은 우선 순위를 부여하는 방식이다. 따라서, 각 태스크의 주기가 주어지면 태스크들간의 우선 순위는 정적으로 결정될 수 있으며 이 우선 순위는 시스템이 수행되는 동안 고정된다[2,11].

단일 프로세서 환경에서 비율 단조 스케줄링 알고

리즘을 사용하는 경우 태스크의 집합의 스케줄 가능성은 주어진 주기 태스크들의 집합의 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 각 태스크의 첫 번째의 작업을 수행시간(Completion time)을 구한 값이 마감시간보다 작은가의 여부를 확인될 수 있다.

단일 프로세서 상에서 고정 우선 순위 기반형 스케줄링(Preemptive scheduling based on fixed priority)방식. 다음과 같은 가정을 하고 있다[1].

① 태스크들은 주기적이고, 주기와 종료시한은 같으며, 실행중에 스스로 중지(suspend)되지 않는다.

② 태스크들은 선점될 수 있고, 문맥교환(context switching)과 태스크 스케줄링에 드는 비용은 무시한다.

③ 모든 태스크들은 서로 독립적이다. 즉, 선행 제약(precedence constraints)은 존재하지 않는다.

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

$$U=\frac{C1}{T1}+\frac{C2}{T2}+\frac{C3}{T3}+\frac{Ci}{Ti}+\cdots+\frac{Cn}{Tn} \quad (1)$$

으로 표시되며, 다음과 같은 조건이 성립되면 독립적인 주기 태스크들의 집합이 각 태스크의 마감시간을 만족시킬수 있다.

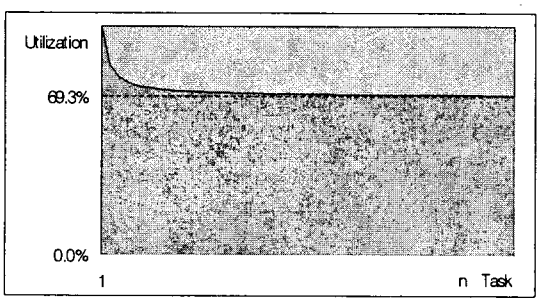
$$U \leq U(n) = n(2^{1/n}-1) \quad (2)$$

이 식에서 $n(2^{1/n}-1)$ 값은 태스크의 개수 n이 증가함에 따라 (표1)과 (그림1)같이 점점 감소하여 69.3%에 수렴한다.

RM 스케줄링 기법은 69.3%보다 작은 태스크 집합을 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 생성될수 있다[8].

(표1) RM 스케줄링 필요충분조건

Task	Utilization
1	100%
2	81.3%
3	77.9%
4	75.6%
5	74.3%
...	...
n	69.3%



(그림1) RM 스케줄링 필요충분조건

2.3 마감시간 우선 스케줄링 정책

EDF 알고리즘은 동적 우선 스케줄링정책에서 가장 흔히 사용되고 있는 스케줄링 방법으로 Liu와 Layland에 의해서 제안되었다[1,3]. 태스크의 우선순위는 마감시간에 따라서 할당된다. 즉, 마감시간이 짧을수록 높은 우선 순위가 할당되므로 임의의 순간에 실행되는 태스크는 실행이 완료되지 않은 태스크들 중에서 마감시간이 가까운 것이 선택된다. RM의 정적 스케줄링 알고리즘과 달리 태스크의 우선순위가 시간에 따라 변하게 된다[9].

마감시간 스케줄링 알고리즘에서 다음의 조건이 만족되면 주기적으로 발생하는 독립적인 n개의 태스크들은 스케줄 가능하다고 할 수 있다. 이 때, 태스크의 주기는 Ti로 수행 시간은 Ci로 표기한다.

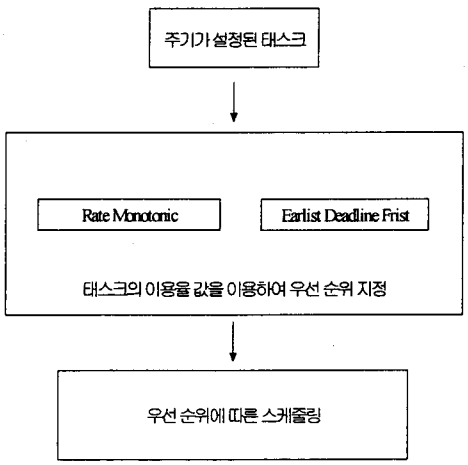
$$U=\frac{C1}{T1}+\frac{C2}{T2}+\frac{Ci}{Ti}+\cdots+\frac{Cn}{Tn} \leq 1 \quad (3)$$

3. 태스크 개별 이용율 알고리즘

기존에 있는 태스크 전체 이용율 알고리즘[5,6,12]과 본 논문에서 제안된 태스크 개별 이용율은 (순서도1) 같이 공통사항이 있다. 스케줄링 가능성 검사 및 스케줄러 선택 알고리즘이다. 또한, 태스크를 실시간 시스템에서 수행이전에 수행할수 있는지 없는지 하는 점이다. 본 논문에서 제안된 알고리즘이 다른점은 각각의 태스크의 이용율 값을 두어 어떤 태스크에서 문제가 생기는지 예측 할수 있게 알고리즘을 구성해 보았다.

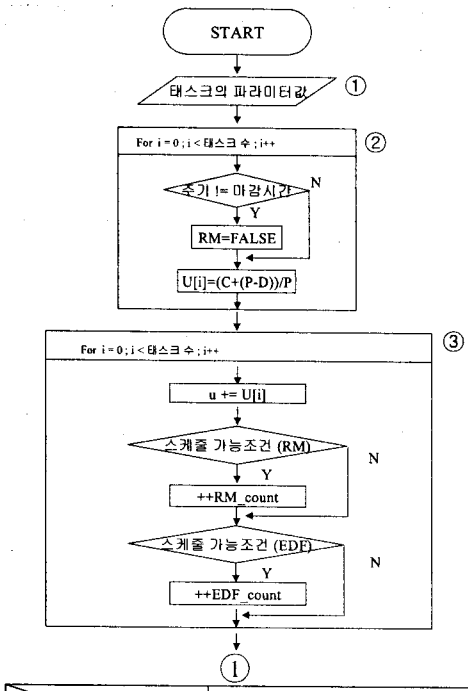
(모델1)에서 우선 주기가 설정된 태스크값이 설정되어 있고, 스케줄링 가능성 분석을 위해서 이용율 공식을 이용하여 가장 적절한 스케줄러의 우선순위를 설정한다. 이렇게한 이유는 태스크가 사전에 문제가 발생할수 있는점을 고려하여 설계되어 졌다. 이용율 값으로 가능성을 판단하며 RM 우선 순위 스케줄러가 불가판정을 받게 되면, EDF 이용율 값을 판별하여 문제가 발생하면 스케줄러는 수행 불가판정을 내린다. 여기에서 가장 적절한 스케줄링이 선택 된다는 점이다.

(모델1) 이용율 값을 이용한 스케줄링

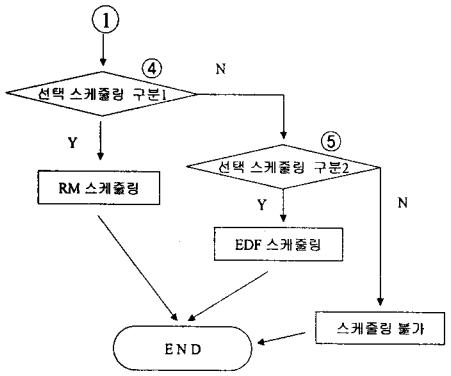


- 태스크 이용율 값을 이용하여 적절한 스케줄러 선택 -

(순서도1) 태스크 개별 이용율 알고리즘



구분 \ 조건	수행 내용
스케줄 가능조건 (RM)	U > 0 and U <= 태스크수*(2^(1/태스크수)-1)
스케줄 가능조건 (EDF)	U > 0 and U <= 1



구분 \ 조건	수행 내용
선택 스케줄링 구분1	RM != FALSE and RM_count == 태스크수
선택 스케줄링 구분2	EDF_count == 태스크수

- 제안하고자 하는 스케줄 가능성 결과에 따른 스케줄러 선택 알고리즘 -

① 파라미터의 값을 입력을 기다린다.
② 파라미터의 값이 입력된 태스크 개수만큼 Deadline과 Period값을 서로 비교하여 틀리면 정적 우선 순위 스케줄링 알고리즘을 수행할 수 없게 RM 변수값을 FALSE로 설정하여 둔다. 여기에서 중요한 점은 기존의 알고리즘에 정적 및 동적으로 나누어서 각기 이용율 공식을 대입하였으나, 여기에서는 U[i] = $\sum_{i=0}^{n} \frac{C+(P-D)}{P}$ 공식으로 한가지로 통일하였고 U[i] 변수란 값에 각기 계산된 이용율 저장해 둔다.
③ 태스크 개수 만큼 수행하면서 이용율 공식에 의해서 각기 저장된 고유의 변수값을 점차적으로 누적하면서 태스크 이용율이 스케줄 가능조건에 있는 정적 스케줄링 (RM)을 보면은 이용율 조건이 U > 0 과 U < N * ($2^{1/N}$ - 1) 공식을 대입하여 가능하면 RM_count변수값을 하나를 증가시킨다. 마찬가지로, 동적 스케줄링 (EDF) 가능조건을 보면 U > 0 과 U < 1까지 태스크가 수행가능하다고 판단하여 가능하다고 하면 EDF_count변수값을 하나증가시킨다.
④ 스케줄링 구분1에서는 RM_count 변수값을 FASLE이고 RM_count 변수값이 태스크 개수와 동일하고 판단되어지면 정적 우선 순위 스케줄링인 비율 단조(Rate Monotonic) 스케줄링 수행한다.
⑤ 스케줄링 구분2에서는 EDF_count 변수값이 태스크 개수와 동일하다고 판단되어지면 동적 우선순위 스케줄링인 마감 시간 우선(Earlist Deadline First) 스케줄링을 수행한다. 만일 이 조건 마저 틀리다면 정적 및 동적 스케줄링의 알고리즘에서 수행할수 없으므로 스케줄링이 불가능하다고 판단하게 되어진다.

기존에 제안한 알고리즘은 단지 이용율 집합의 전체를 가지고 수행가능한지를 판별만 했으나 본 논문에서 제안하고자 하는 부분은 RM과 EDF 스케줄링을 이용율 조건에서 각각 몇 번째 태스크에서 문제가 발생할 수 있는지를 사전에 예측할 수 있다는 점이다.

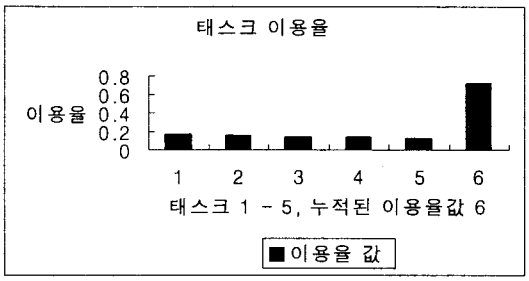
4. 시뮬레이션 및 성능 평가

본 논문에서 제안된 태스크 개별 이용율 알고리즘과 태스크 전체 이용율 알고리즘을 시분할 운영체제인 리눅스 커널 2.4.18과 GCC 컴파일러의 환경에서 실시간 환경에 맞는 다음과 같은 제한된 조건을 (표2), (표3) 가지고 평가를 하고자 한다.

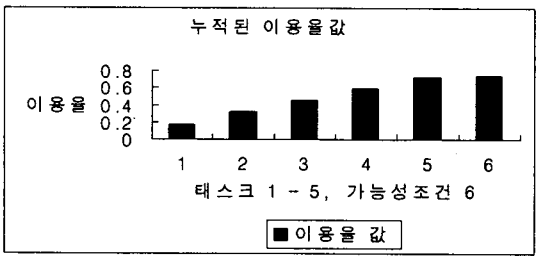
(표2) 마감시간과 주기가 같은 경우

T	1	2	3	4	5
C	50	60	70	80	90
P	300	400	500	600	700

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



(그림2) 마감시간과 주기가 같은 태스크 이용율 값



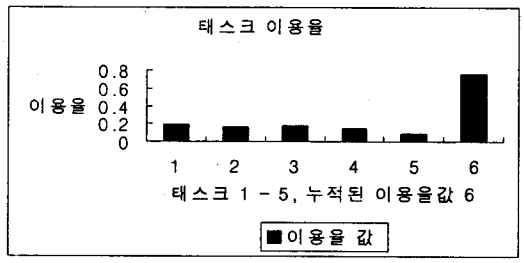
(그림3) 마감시간과 주기가 같은 제안 태스크 이용율 값

(그림2)에서 태스크 이용율을 보면 누적된 이용율 값이 0.718571임을 알 수 있다. RM과 EDF를 동일한 주기로 설정되어져 있다. 그래서 EDF가 RM의 주기적인 특성을 갖는다. 태스크 집합을 Liu 와 Layland의 $n(2^{1/n}-1)$의 스케줄링 가능성 조건이 맞는지 확인하면 이용율이 (그림3)에서 가능성 조건이 0.743492이기 때문에 RM과 EDF 알고리즘을 수행할 수 있는 조건에 충족됨을 알 수 있지만 선택 알고리즘에서 RM 알고리즘을 수행할 수 있게 된다. 여기에서는 전체이용율을 이용할 경우와 제안한 알고리즘을 사용시 동일하게 동작함을 알 수 있었다.

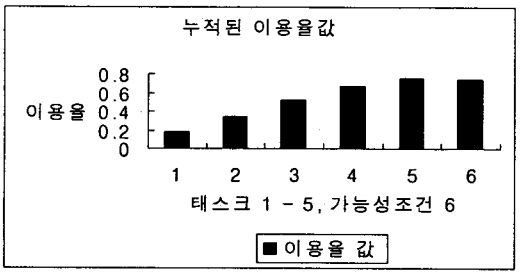
(표3) 마감시간과 주기가 틀릴 경우

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



(그림 4) 마감시간과 주기가 틀린 태스크 이용율 값



(그림 5) 마감시간과 주기가 틀린 제안 태스크 이용율 값

(그림4)에서 스케줄링한 결과 누적된 이용율 값이 0.759048인 경우임을 알수 있었고, RM의 (그림5)가능성 조건을 보면 0.743492이므로, 태스크의 5번에서 이용율이 초과 되었다는 것을 확인할 수 있다. RM 스케줄링은 수행조건에 위배되므로서 본 논문에서 제안한 알고리즘에 의해 EDF의 수행 가능 조건 구간은 $(0 \le i \le 1)$이므로 이 조건에 충족됨을 알 수 있고, 종료시한과 주기가 각기 틀리게 부여 되었으므로 선택 알고리즘에서 EDF를 스케줄링하게 된다. (그림4)와(그림5)를 보면 전체이용율을 가지고 판단할 경우에는 어떤 태스크에서 문제가 발생하는지를 예측할 수 없지만 제안한 알고리즘에서는 정확히 태스크 5번째에서 이용율이 초과되었음을 확인할 수 있었으며 사전 예측성이 훨씬 좋았다.

※ (그림2)와 (그림4)에서 태스크 1 - 5까지는 개별 태스크 이용율 값이며, 기존에 스케줄링 가능성 검사 시에는 (그림2),(그림4) 6번째 누적된 값을 (그림3),(그림5)와 같이 가능성 조건을 가지고 판단하였다.

5. 결론

본 논문에서는 제안된 스케줄링 가능성 검사 알고리즘과 선택 알고리즘을 이용한 파라미터 값을 가지고 이용율을 사용한 스케줄링의 가능·불가능 성을 검사해 보았다. 기존에 제시된 알고리즘은 미리 태스크 전체의 이용율 값을 가지고 있어 전체가 가능한지 불가능한지에 대해서만 다루었지만, 본 논문에서 제안된 알고리즘에서는 각각의 태스크에 이용율을 부여함으로써 어떤 태스크에서 수행가능하고 불가능한지를 미리 예측할수 있게 되었다. 정적인 RM 알고리즘과 동적인 EDF 스케줄링 알고리즘 중 적절한 태스크를 수행할 수 있게 한 선택 알고리즘이다. 여기에선 한정적인 상태에서 시뮬레이션을 해 보았으나 폭 넓게, 예측할 수 있는 시스템 구축이 요구된다.

참고문헌

[1] 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.
[2] J. Lehoczky, L. Sha, and Y. Ding, "The Rate Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behavior", In Proceedings of the 10the Real-Time System Symposium, pp.166-171, 1989.
[3] 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.
[4] An Fredette and Rcleaveland, "A Generalized to Real-Time Schedulability Analysis", 10th workshop on real-time operating system and software, 1993.
[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] 박정훈, "스케줄링 정책의 다양성을 지원하는 실시간 커널의 설계및 구현", 서울대학교, 석사학위논문, 1997.
[7] 신형식, 김태웅, "실시간 시스템의 개관", 한국 정보처리학회 논문지 제5권 제4호, pp. 2-11, 1998. 7.
[8] 김성관, 하란, "실시간 스케줄링", 한국 정보처리학회 논문지 제5권 제4호, p.12-21, 1998. 7.
[9] 류진열, 김광, 허신 "Mach 커널의 재구성을 위한 확장된 스케줄 가능성 검사를 수행하는 실시간 스케줄러" 정보처리학회 논문지 제7권 2호, pp.507-519, 2000. 2.
[10] 김창배, "실시간 운영체제의 동일 우선 순위에 대한 다중 작업 지원의 설계및 구현", 부경대학교, 석사학위논문, 2001.
[11] 심재홍, "다양한 실시간 스케줄러를 지원하기 위한 커널 구조화 및 재구성 방안", 아주대학교, 박사학위논문, 2001.
[12] 최정훈, 김경화, 김두상, 최대수, 임종규, 박한규, 구용완, "실시간 리눅스에서 선택 알고리즘을 이용한 스케줄링 성능평가", 한국정보과학회 가을학술발표논문집 Vol. 29, No.2, pp.430-432, 2002.
[13] 신동헌, 이준택, 조수현, 김영학, "경성 실시간 태스크를 위한 확장된 가능성 검사를 통한 비율단조 기반 스케줄링 기법", 한국정보과학회 봄학술발표논문집 Vol.30, No. 1, pp.202-204, 2003.