주기 태스크의 종료시간을 보장하기 위한 확장된 혼합 실시간 스케줄링 알고리즘

한대만, 최만억, 구용완
수원대학교 전자계산학과

Expended Hybrid Real-Time Scheduling Algorithm for guard Deadline of An Periodic Task

Dae Man Han
Department of Computer Science ,University of Suwon

요 약

본 논문은 고정 우선순위를 가지는 주기 태스크와 동적으로 발생하는 비주기 태스크를 스케줄링하는 방법으로 슬랙 스틸링(slack stealing) 알고리즘을 소개하고 발생되는 문제점으로부터 새로운 알고리즘을 제시한다. 기존의 실시간 스케줄링 알고리즘은 RM 스케줄링에 의해 할당된 주기 태스크의 우선순위가 동적으로 발생하는 비주기 태스크의 수행으로 인한 주기 태스크들간의 긴급함에 적절하게 대응하지 못하고 주기 태스크들이 자신의 제한시간을 넘길 수 있다. 이는 슬랙 스틸링(slack stealing)알고리즘에서 슬랙 타임을 구하는 함수의 한 요소로서 주기 태스크의 필요 계산시간 값이 긴급하지 않은 상위 우선 순위 태스크 의 계산시간 까지 포함하는 데 있다. 따라서 제안한 확장된 실시간 스케줄링 알고리즘은 RM스케줄링에 의한 스케줄링 가능성을 위배하지 않으면서 동적으로 발생하는 비주기적 태스크로부터 긴급도의 변화에 적용할 수 있는 우선순위 체제의 알고리즘을 제시하고, 제안한 알고리즘이 다른 실시간 알고리즘보다 성능 개선이 있음을 모의 실험을 통하여 증명한다.

1. 서 론

1.1.실시간 시스템의 개념.

실시간 시스템은 범용 컴퓨터 시스템과는 달리 시스템의 시간적 정확성에 유의해야 하는 시스템이다. 실시간 시스템의 예로서는 항공기 제어, 무기관리, 핵발전소등에서 넓게 사용되고 있다. 실시간 시스템에서 시간의 제약조건을 만족한다는 것은 단순히 시스템의 성능을 향상시키는 것을 의미하지 않는다. . 실시간 시스템을 개발하는데 드는 비용이 매우 높기 때문에 시스템 설계시에 시스템이 시간 제약조건을 만족할 수 있는지를 알 수 있어야 비용이 적게들고, 적합한 실시간 시스템을 만들 수 있다. 따라서, 실시간 시스템에서는 다음의 세가지 특성을 가진다. 첫째, 예측성(Predictability)이 높은 시스템을 개발하는 것이 매우 과제이다. 이와 마찬가지로 실시간 스케줄링에서도 실제 스케줄링을 해보지 않고도 주어진 주기적 태스크 집합에 대하여 모든 태스크의 마감시간을 만족할 수 있는지를 예측할 수 있어야한다. 둘째, 스케줄링 가능성 검사(Schedulability)는 스케줄링 측면에서 주어진 주기적 태스크 집합에 대하여 마감시간을 만족할 수 있는지의 여부를 조사하는 것이다. 셋째, 적합성(Feasibility)은 주어진 태스크 집합에 스케줄링 알고리즘이 적용 가능한지를 판단하는 것을 말한다. 이처럼 실시간 시스템에서는 예측성, 스케줄링 가능성, 적합성에 관한 검증을 가짐으로서 최적의 실시간 시스템을 구축할 수 있다.

2. 실시간 태스크 스케줄링.

2.1.슬랙 스틸링 알고리즘.

슬랙(slack)은 주기적인 태스크가 실행을 종료했을 때 종료시한까지 사용되지 않는 부분을 말한다. 슬랙 스틸링 알고리즘에는 정적 슬랙 스틸링 알고리즘와 동적 슬랙 스틸링 알고리즘이 있다. 두 알고리즘은 모든 종류의 비주기 태스크에 대해 최소 응답시간(response time)을 제공한다는 관점에서 최적이라고 증명되었다. 정적 슬랙 스틸링 알고리즘은 연성 종료시한을 갖는 비주기 태스크와 경성 종료시한을 갖는 비주기 태스크에 대해 적용하여 사용되는 실시간 스케줄링 알고리즘이다. 이 알고리즘은 주기 태스크에

종료시한 단조형으로 우선순위를 할당하고 비주기 태스크를 위한 주기적인 서버를 두지 않고 비주기 태스크가 도착하면 도착 시간에 해당하는 오프라인 시 구해놓은 주기 태스크의 각 우선순위 레벨에서의 가능한 슬랙(slack)들의 최소값을 비주기 태스크의 실행을 위하여 활용한다.

3. 제안된 실시간 스케줄링.

3.1.슬랙 스틸링(slack stealing)의 문제점.

슬랙 스틸링 방식의 문제점은 RM 스케줄링에 의해 할당된 주기 태스크의 우선순위가 동적으로 발생하는 비주기 태스크의 수행으로 인한 주기 태스크들간의 긴급도 변화에 대응하지 못한다는 것이다. 이는 RM 스케줄링으로 주기 태스크를 스케줄링 하는 도중에 비주기 태스크가 도착하면 이에 대해서도 스케줄링을 하고 비주기 태스크의 스케줄링된 이후에 RM은 선점형이기 때문에 주기가 작은 주기 태스크가 도착하면 선점을 해야한다. 그러나 비주기적 태스크에 의해서 선점되는 주기에 존재하는 주기적 태스크의 크기는 달라지기 때문에 자신의 주기태에서 스케줄링되지 못하고 종료시한을 넘기게 된다.

이러한 문제점을 해결하기 위해서 비주기 태스크를 대기시간을 최소화하면서 스케줄링하기 위해서는 RM 스케줄링에 의한 스케줄링 가능성을 위배하지 않으면서 동적으로 발생하는 긴급도의 변화에 적응할 수 있는 우선순위 체계가 개발되어야 하며, 이러한 우선순위 체계하에서 비주기 태스크를 위한 계산시간을 할당할 수 있는 서버의 개발이 필요하다.

3.2. 제안된 알고리즘의 적용.

Liu와 Layland는 RM 스케줄링 기법을 위하여 다음과 같은 스케줄링 가능성 분석 조건을 제시 하였는데, 전체 프로세스 이용율(total utilization) U를 이용하고 있다. $U=\sum_{i=1}^{N} U_i=\sum_{i=1}^{N}\le N(2^{\frac{1}{n}}-1)$ ------①

이 식에서 $N(2^{\frac{1}{n}}-1)$값은 태스크의 개수 N이 증가함에 따라 점점 감소하여 0.69에 수렴하게 된다. 이는 프로세스 이용율 U의 값이 0.69보다 작은 태스크 집합은 스케줄링이 가능하다.따라서, 주기적 태스크를 스케줄링 하고 남아있는 여유공간(슬랙)을 찾아내어 비주기적 태스크에 할당한다면 선점에 의하여 우선순위가 높은 다른 주기 태스크가 도착한 경우 적절한

$U_i(t)$	구간 [0,t]에서의 전체 계산량
$A_i(t)$	비주기 태스크에 서비스를 위한 계산량
$P_i(t)$	우선순위 i의 태스크와 우선순위가 i이상인 주기태스크의 계산 시간
A_{ij}	구간에서 $[C_{ij-1}, C_{ij}]$ 우선순위 i로 비주기 태스크에 할당 가능한 처리기 시간
$I_i(t)$	i-레벨에서의 활용되지 못하는 구간과 유휴시간

$$U_i(t)=A_i(t)+P_i(t)-I_i(t)$$

대응을 하기 위해 τ_3의 주기적 태스크가 자신의 주기를 비주기적 태스크를 위해 슬랙을 검색하는 구간인 $C_{ij-1}\le t\le C_{ij}$ 에서 각각의 우선순위 레벨에 대하여 주기적 태스크가 자신의 우선순위에서 D_{ij}(데드라인)을 넘기는 실행이 발생하면 비주기적 주기적 태스크를 위해 가용슬랙을 구했던 $C_{ij-1}\le t\le C_{ij}$에서 실행되는 비주기적 태스크 즉, 종료시한을 넘긴 T_3보다 우선순위(i)가 높은 비주기적 태스크로 인하여 T_3(주기적 태스크)가 종료시한을 넘겼기 때문에 실행크기는 비주기 태스크와 T_3가 동일하다. 따라서, T_3(주기적 태스크)가 자신의 종료시한을 넘어서지 않기 위하여 T_3와 T_3가 스케줄링 되는 하이퍼주기(H)에서 높은 우선순위로 인하여 실행을 위해 스케줄링 되었던 비주기적 태스크와 우선순위를 교환한다. 즉, 비주기적 태스크는 자신의 데드라인을 넘어서지 않는 범위에서 발생되는 종료시한을 넘어선 주기적 태스크에게 자신의 우선순위와 교환한다

$A_j(s,t)$는 구간 $[s,t]$에서 j가 가장 작은값 즉, 제일 우선순위가 높은값을 찾아낸다.

수식을 세워보면

$A_i(t)=A_{ij}, C_{ij-1}\le t\le C_{ij}, j\ge 1$에서

$\min_{(C_{ij-1}\le t\le D_{ij})}\left\{\frac{A_{ij}+P_i(t)}{t}\right\}\le 1$ ---②을 검토하여 ②>1이 발생하면 스케줄링 되지 못하지만 만일 원인이 주기적 태스크가 종료시한을 넘기는 원인으로 인하여 일정 구간에서 태스크들이 스케줄링 되지 못하는 것을 방지하기 위하여 전체 스케줄링 구간에서 D_{ij}(제한시간)을 지난 주기태스크(T_3)의 나머지 수행부분 (C_i(τ_3의 최악수행시간) - D_{ij}(우선순위 i에서

위 종료시한))에 현재의 우선순위를 부여하고 비주기 태스크의 우선순위 와 교환한다. 이 경우 우선순위 i의 비주기 태스크 A는 자신의 D_{ij}를 넘어서지 않음을 보장해야 한다. 그리고, 우선순위를 경과한 주기 태스크와 이 주기 태스크 보다 우선순위가 높은 비주기 태스크의 우선순위를 교환한 경우 응답시간의 단축에 우선을 두고있는 비주기 태스크 성능향상에 영향을 주지 않는다.

주기 태스크와 비주기 태스크의 우선순위 교체 알고리즘을 보면 다음과 같다.

(1) 위의 식②을 검토하여 주기적 태스크에게 스케줄링이 가능한지를 검사하고 스케줄링을 한다. 만일 스케줄링이 가능성 검사에서 비주기 태스크와 주기태스크를 합한 값이 구간의 주기를 넘어서 식 ②>1이 되는 경우에는 주기적 태스크가 종료시한을 넘긴 경우가 있다. 주기적 태스크로 인한 경우 중 주기적 태스크가 비주기적 태스크로 인하여 종료시한이 넘어서게 됐는지를 검사한다.

이는 다음과 같은 수식으로 검사한다

A_i 비주기 태스크의 도착시간

T_i 주기태스크의 주기

C_i 최악실행시간(worst case execution)

I_i 위상(phase)

D_i 마감시한(deadline)

수식②에서 $P_i(t)$의 값은 비주기적 태스크와 주기적 태스크를 모두 계산하기 때문에 슬랙 스틸링에서 문제점이 발생한다.

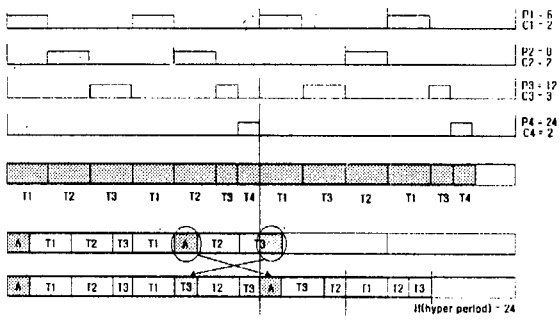
$$P_i(t) = \sum_{j=1}^{i} C_j \times \lceil \frac{t}{T_j} \rceil \quad \text{-------③}$$

D_{ij-1}은 우선순위 i에서 임의의 시점(s)가 속한 태스크 주기의 이전 종료시한이다.

(2) D_{ij}를 넘긴 주기 태스크를 검사하여 수행되지 못한 부분의 크기를 알아낸다. (만일, D_{ij}를 넘긴 주기 태스크가 τ_3라면 τ_3의 구간당 실행시간 - 현재 수행된 실행시간)

(3) 스케줄링된 비주기 태스크에서 현재 스케줄링되는 구역 $C_{ij-1} \le t \le C_{ij}$으로부터 (2)에서 찾아낸 주기태스크의 크기와 동일한 비주기 태스크를 검색.

(4) 비주기 태스크의 우선순위와 주기 태스크의 스케줄링 되지 못한 부분의 우선순위 (i)를 교환.



[그림3-1] 우선순위 교환 방법

이와 같은 우선순위 할당방법으로 주기태스크의 실행주기가 자신의 종료시한 이내에서 스케줄링 될 수 있도록 한다.

4.결론 및 향후연구 과제.

제안한 확장된 스케줄링을 적용한 결과 주기적 태스크의 종료시한을 만족하고 슬랙스틸링을 사용하여 비주기 태스크의 응답시간을 단축할 수 있다. 향후 연구과제로는 제안한 알고리즘을 실시간시스템에 적용하고 자원을 가진 인스턴스간의 병렬성이나 자원을 발생하지 않도록 하는 동기화 방법을 연구하고 이로부터 자원할당을 알맞게 할 수 있는 방법을 연구할 예정이다.

5. 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 10th Real-Time Systems Symposium, pp.166-171,1989.

[3] K. W. Tindell, A. Bums, and A. J. Wellings, "An Extendible Approach for Analyzing Fixed Priority Hard Real-Time Tasks", Real-Time Systems,Vol. 6, no. 2, pp,133-151,1994.

[4] R. I Davis "Scheduling Slack Time in Fixed Priority Pre-emptive Systems"

[5]N. Audsley A. Burns "Real-time System Scheduling"