멀티프로세서 시스템에서 실시간 태스크들의 성공률을 개선한 스케줄링 알고리즘

강호석, 김용석
강원대학교 컴퓨터공학과

Real-Time Task Scheduling Algorithms to Enhance Success Ratio in Multiprocessor Systems

Ho-Seok Kang, Yong-Seok Kim,
Kangwon National University

요 약

효율적인 스케줄링 알고리즘은 적은 문맥교환 횟수를 가지면서 동시에 잘 정의된 임의의 태스크 집합에 대해 높은 스케줄링 성공률을 갖고 있어야 한다. 기존의 단일 프로세서 스케줄링 알고리즘들은 멀티프로세서 스케줄링 알고리즘들에 비해 시간복잡도가 낮지만 멀티프로세서 환경에서 그대로 적용시킬 경우 스케줄링 성공률이 많이 떨어진다. 본 논문에서는 비슷한 시간 복잡도를 가지면서도 멀티프로세서 환경에서 높은 성공률을 얻을 수 있는 EDF-ZLP와 LLF-RP 알고리즘을 제안하고 이 알고리즘들의 추가적인 성능 향상 방안을 제안한다.

1. 개 요

실시간 시스템은 군사, 멀티미디어, 자동화 시스템, 산업용 로봇, 항공 제어 등의 여러 분야에서 광범위하게 사용되고 있다. 이러한 시스템은 마감시간(deadline)이내에 올바른 정보를 산출해야 하며 시스템의 안정성을 위해서 적은 오버헤드를 가지면서도 임의의 잘 정의된 태스크 집합에 대해 높은 스케줄링 성공률을 갖는 스케줄링 알고리즘이 필요하다[3]. 스케줄링 과정에서의 오버헤드는 스케줄링에 소요되는 시간과 문맥교환에 소비되는 시간이 중요한 요소이며, 문맥교환에 소비되는 시간은 문맥교환 횟수에 비례하게 된다. 따라서 적은 오버헤드를 갖는 스케줄링 알고리즘은 문맥교환 횟수와 스케줄링에 소요되는 시간을 최소화해야한다. 그러나 기존의 대부분의 멀티프로세서 스케줄링 알고리즘은 높은 시간 복잡도를 갖고 있는 반면에 단일 프로세서 스케줄링 알고리즘은 비교적 낮은 시간 복잡도를 갖고 있으나 그대로 적용하기에는 현실적으로 스케줄링 성공률이 낮은 문제가 있다[2].

Earliest Deadline First(EDF)는 이른 마감시간을 갖는 태스크에게 높은 우선순위를 주는 방법으로서 적은 문맥교환 횟수를 갖는 스케줄을 만드는 특징이 있다. 이 알고리즘은 단일 프로세서 시스템에서 최적의 스케줄을 만들지만 멀티프로세서 시스템에서는 잘 정의된 임의의 태스크 집합에 대해 스케줄할 수 없는 경우가 많다[1]. 또한 Least Laxity First (LLF)는 적은 laxity를 갖는 태스크에게 높은 우선순위를 주는 방법으로서 스케줄링 오버헤드가 무시된다면 단일 프로세서 시스템에서는 최적이며 멀티프로세서 실시간 시스템의 경우에서 좋은 성능을 보인다. 그러나 스케줄러의 오버헤드를 고려할 경우 LLF로 생성된 스케줄은 많은 문맥교환 횟수를 갖고 있기 때문에 시스템의 오버헤드를 가중시킨다[1].

본 논문에서는 적은 오버헤드를 갖는 스케줄링 알고리즘인 EDF-ZLP과 LLF-RP을 제안한다. 태스크들은 수행준비완료시간(ready time), 실행시간(execution time), 마감시간(deadline), 및 우선순위(priority)를 갖는다. 본 논문에서는 제안된 알고리즘과 기존의 알고리즘을 비교하기 위해 모든 태스크들은 동일한 수행준비완료시간을 갖는다는 가정 하에 시뮬레이션을 수행하였나 임의의 수행준비완료시간을 갖는 태스크 집합에 대해서도 본 논문에서 제안한 알고리즘을 그대로 적용가능하다.

2. EDF-ZLP 와 LLF-RP 알고리즘

EDF-ZLP (EDF with Zero Laxity Preemption)의 기본 아이디어는 EDF의 문맥교환 횟수를 유지하면서 잘 정의된 임의의 태스크 집합에 대해 스케줄링 성공률을 높이데 있다. EDF는 늦은 마감시간을 가지면서 laxity가 적은 태스크가 존재할 경우 그 태스크의 마감시간을 놓치는 경우가 발생하게 되는데, 이러한 점을 보안하기 위해서는 태스크들의 laxity를

고려해야한다.

EDF-ZLP의 우선순위 할당방법은 laxity가 0인 테스크들에게 가장 높은 우선순위를 부여하고 나머지 태스크들에 대해서는 EDF방식으로 우선순위를 부여하되 같은 마감시간을 갖는 태스크들에 대해서는 laxity가 적은 태스크에 더 높은 우선순위를 부여하는 방식을 사용한다.

LLF-RP (LLF with Restricted Preemption)의 기본 아이디어는 LLF로 스케줄할 수 있는 태스크 집합의 수를 유지하면서 문맥교환 횟수를 줄이는데 있다. 대기중인 태스크의 laxity는 매 단위 시간마다 감소하므로 LLF는 단위시간에 따라 우선순위가 변경될 수 있으며 이것은 문맥교환 횟수에 큰 영향을 준다.

LLF-RP의 우선순위 할당 방법은 laxity가 0인 태스크들에게 가장 높은 우선순위를 부여하고 나머지 태스크들에 대해서는 프로세서에 유휴시간이 발견될 때만 LLF방식으로 우선순위를 부여하되 만일 같은 laxity를 갖는 태스크들에 대해서는 마감시간이 빠른 태스크에 더 높은 우선순위를 부여하는 방식이다.

3. 제안된 알고리즘의 개선

EDF-ZLP-E(EDF-ZLP Enhanced)와 LLF-RP-E(LLF-RP Enhanced)의 기본 아이디어는 현재의 스케줄링 알고리즘으로 스케줄할 수 없는 패턴이 발견되면 LLF로 스케줄을 하고 문제의 상황에서 벗어나면 다시 원래의 스케줄링 알고리즘으로 전환하여 스케줄 가능한 태스크 집합을 확대시키는 것이다. 다음은 스케줄할 수 없는 패턴에 대한 설명이다.

패턴 1. CPU의 개수를 k개라고하고, CPU에서 laxity가 0인 상태로 수행중인 태스크의 개수가 m이고, 대기 큐에는 이들의 최소 나머지 수행시간보다 작은 laxity를 갖는 태스크가 n개가 있다고 하자. 만약 $k-m < n$ 이면 EDF-ZLP와 LLF-RP로 스케줄이 안될 가능성이 높으나 LLF로는 스케줄이 가능하다.

프로세서가 3개일 경우 LLF로는 스케줄 가능하지만 EDF-ZLP와 LLF-RP로 스케줄 할 수 없는 태스크 집합을 표1에 나타나 있다 그림 1, 2, 3는 표1의 태스크 집합을 LLF, EDF-ZLP, LLF-RP로 스케줄하는 모습을 도식으로 나타낸 것이다. 그림에서 세로축은 태스크들을 나타내고 가로축은 단위 시간을 나타낸다. 프로세서에서 수행중인 경우는 검은색으로 나타내었고 각각의 값들은 해당 태스크의 laxity을 나타낸다.

태스크	수행준비 완료시간	요구시간	마감시간	laxity
T_0	0	3	3	0
T_1	0	3	3	0
T_2	0	2	4	2
T_3	0	2	4	2
T_4	0	2	4	2

표 1 태스크 집합

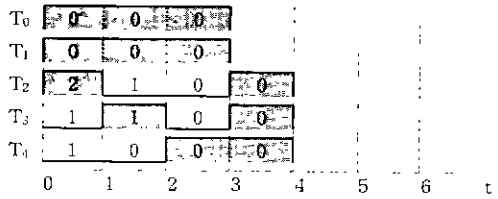


그림 1. LLF 스케줄링

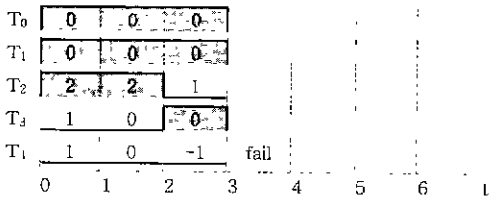


그림 2. EDF-ZLP 스케줄링

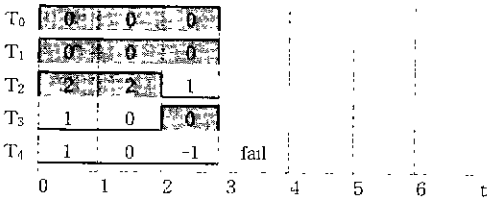


그림 3 LLF-RP 스케줄링

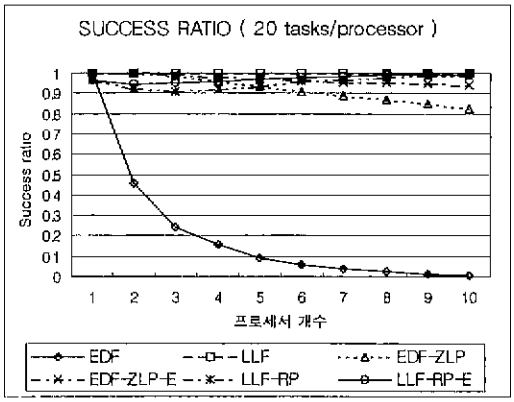
4. 성능 분석

시뮬레이션을 위한 테스크 집합의 생성 방법은 처음에 각각의 태스크마다의 수행준비완료시간과 실행시간을 설정하고, 우선순위를 매 단위시간마다 바꾸면서 가장 높은 우선순위의 태스크를 수행시킨다. 만일 실행시간만큼 수행된 태스크가 있으면 그 태스크의 마감시간을 현재의 단위시간으로 설정한다.

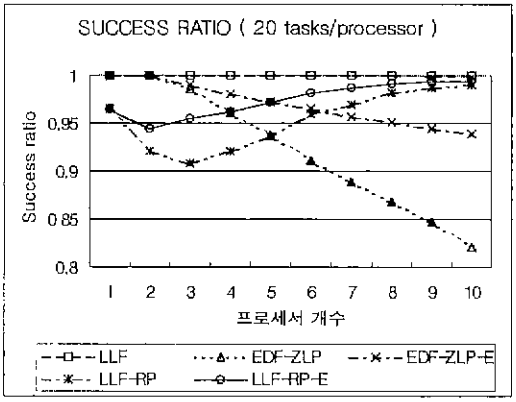
프로세서 개수를 1개부터 10개까지 증가시키되 프로세서당 태스크의 개수는 20개로 일정하게 실험을 하였고, 10000번 수행시킨 후 평균값을 취하였다. 우리의 관심값들은 스케줄링 성공률과 문맥교환 횟수이며 시뮬레이션시에는 알고리즘 자체를 판단하기 위해 수행준비완료시간을 0이라고 가정하였다. 여기서 스케줄링 성공률은 절대적인 수치가 아니라 단지 다른 알고리즘과의 비교를 위한 것이다.

그림 4과 그림 5는 프로세서 개수의 변화에 따른 스케줄링 성공률의 변화와 문맥 교환의 변화를 나타낸다 스케줄링 성공률면에서 보면 EDF는 프로세서 개수가 증가할수록 본 논문에서 제안한 알고리즘들보다 스케줄링 성공률이 급격하게 떨어지는 반면에 문맥 교환 횟수면에서는 별 차이가 없다. LLF는 스케줄링 성공률면에서 약간 높은 결과를 내지만 문맥교환 횟수가 프로세서 개수의 증가에 따라 급격히 증가하므로 본 논문에서 제안한 알고리즘들이 더 좋은 결과가 나왔다.

EDF-ZLP와 EDF-ZLP-E를 비교해 보면 EDF-ZLP-E가 문맥교환 횟수는 2배정도 늘어나지만 프로세서 개수의 증가

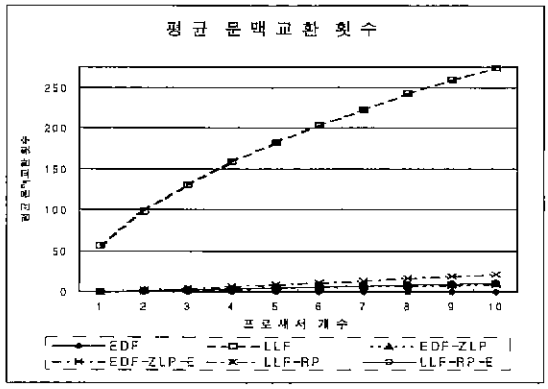


(a) EDF 포함

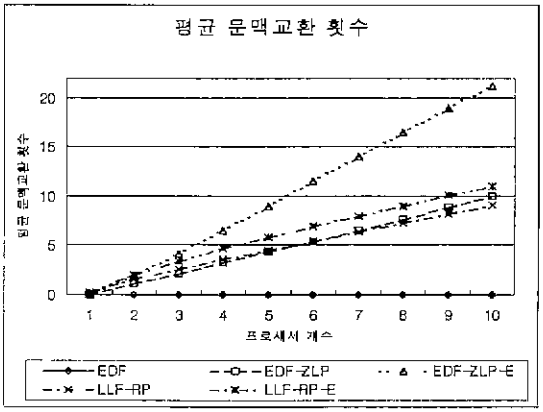


(b) EDF 제외

그림 4. 스케줄링 성공률



(a) LLF 포함



(b) LLF 제외

그림 5 평균 문맥 교환 횟수

에 따라 더 좋은 성능을 보인다. LLF-RP와 LLF-RP-E 역시 같은 결과를 보이고 있다. 특이한 사항으로는 프로세서의 개수가 적을 때에는(본 시뮬레이션 환경에서는 5개이하) EDF-ZLP가 우수하지만 많을 때에는 LLF_RP가 더 높은 성공률을 보이고 있다.

5. 결 론

EDF를 그대로 멀티프로세서 환경에 적용할 경우 적은 문맥교환 횟수로 스케줄링 오버헤드는 적으나 스케줄링 성공률이 프로세서 개수의 증가에따라 급격히 감소하기 때문에 안정적인 스케줄을 만들 수 없다. LLF도 그대로 멀티프로세서 환경에 적용할 경우 높은 스케줄링 성공률을 얻을 수 있으나 문맥교환 횟수가 너무 많아 스케줄링 오버헤드가 발생하게 된다. 본 논문에서 제안한 알고리즘들의 특징은 다음과 같다.

1. EDF와 거의 차이없는 적은 문맥교환 횟수.
2. LLF와 비슷한 높은 스케줄링 성공률.
3. EDF, LLF와 비슷한 스케줄링 소요시간.

그러므로 제안된 알고리즘은 실시간 멀티프로세서 환경에서 보다 높은 안정성을 제공할 수 있게 된다.

6. 참 고 문 헌

[1] M. L. Dertouzos, A. K. Mok, "Multiprocessor On-Line Scheduling of Hard-Real-Time Tasks," IEEE Transactions on software engineering, VOL. 15, NO. 12, December 1989.

[2] J. A. Stankovic, M. Spuri, M. D. Natale, G. Buttazzo, "Implications of Classical Scheduling Results For Real-Time Systems," IEEE computer, vol.28, No. 6, June 1995

[3] J. A. Stankovic, K. Ramamritham, "Real-time computing systems : The next generation," Tutorial on Hard Real-Time Systems, IEEE Press, New York, 1998.