

제한성 만족 확률에 기반한 (m, k)-준경성 실시간 태스크 스케줄링

김경훈[○] 김종 홍성재

포항공과대학교

{jysh[○], jkim, sjhong}@postech.ac.kr

Scheduling (m, k)-firm Real-time Tasks based on the (m, k)-firm Constraint Meeting Probability

Kyong Hoon Kim[○], Jong Kim, Sung Je Hong

Dept. of Computer Science & Engineering,

Pohang University of Science and Technology (POSTECH)

요 약

최근의 실시간 시스템에 관한 연구는 준경성(firm) 실시간 시스템에 많은 초점을 두고 있다. 준경성 실시간 시스템은 경성(hard)이지만, 가끔 종료시한을 못 맞추는 것을 허용하는 실시간 시스템이다. (m, k)-준경성 태스크 모델은 임의의 연속하는 k 개의 작업 중에서 적어도 m 개의 종료시한은 만족해야 하는 태스크 모델로, 대표적인 준경성 실시간 시스템이다. 기존의 (m, k)-준경성 태스크 스케줄링 알고리즘에서 사용한 우선순위 결정 방법은 크게 두 가지로 나뉘어지며, 첫번째는 연속해서 종료시한을 놓쳐도 되는 작업의 개수에 기반하는 것이고 두번째는 손실률에 기반하는 것이다. 두 방식 모두 (m, k)-준경성 제한성의 개념을 정확하게 반영하지 못 하여 성능이 낮게 나온다. 따라서 본 연구는 우선순위 결정에 사용할 새로운 제한성 만족 확률을 제안하고, 이에 기반한 스케줄링 알고리즘도 제시한다. 그리고 모의 실험을 통해서 제안하는 알고리즘들의 성능을 평가한다.

1. 서론

실시간 시스템은 크게 경성(hard)과 연성(soft)로 나누어진다. 경성 실시간 시스템은 태스크의 종료시한(deadline)을 못 맞추는 경우 시스템에 큰 손실을 가져다 주는 것이며, 연성 시스템은 종료시한을 넘겨서 실행을 완료해도 어느 정도 보상된 값을 받는 시스템이다. 하지만 최근에는 경성이지만 가끔 종료시한을 못 맞추는 것을 허용하는 준경성(firm) 실시간 시스템에 관한 많은 연구가 진행되고 있다. 이러한 준경성 실시간 시스템은 멀티미디어, 컨트롤 시스템, 서비스질(QoS) 관련 시스템 등에 응용이 된다.

준경성 실시간 시스템에 관해서 많은 연구가 진행되어 왔다. Koren과 Shasha는 skip factor s 를 제안하여 연속하는 종료시한의 놓치는 회수를 지정하였다[1]. Hamdaoui와 Ramanathan은 연속하는 임의의 k 개의 작업 중에서 적어도 m 개의 종료시한을 만족해야 하는 (m, k)-준경성 태스크 모델을 제안하였다[2]. Cho와 Shin은 연속해서 놓쳐도 괜찮은 종료시한의 최대 개수를 가지는 모델을 제안하였으며[3], Bernat 등은 이러한 여러 태스크 모델들을 정리하여 Weakly hard 실시간 시스템을 제안하였다[4]. 또한 West와 Poellabauer는 Window-constrained 시스템을 제안하여 DWCS라는 스케줄링 알고리즘을 제시하였다[5].

이렇듯 많은 준경성 실시간 시스템 모델에서 제시된 스케줄링 알고리즘은 고정 우선순위 기반의 알고리즘과 동적 우선순위 기반의

스케줄링 알고리즘으로 나누어진다. 고정 우선순위 기반의 알고리즘은 태스크 별로 우선순위를 고정하여 상위 우선순위 태스크의 스케줄링을 보장해준다. 반면에 동적 우선순위 기반 알고리즘은 전체 성능을 높이기 위해서 최선책(best-effort)으로 태스크들을 스케줄링한다.

준경성 실시간 시스템에서 최선책 스케줄링 알고리즘은 크게 두 가지 방식으로 태스크의 우선순위를 결정한다. 첫번째는 연속해서 종료시한을 못 맞춰도 되는 거리에 기반한 알고리즘으로, DBP(Distance Based Priority) 방식이다[2, 6]. 이 방식은 그 거리가 적은 태스크에게 우선순위를 더 준다. 두번째는 손실률에 기반한 알고리즘으로[5], 허용 가능한 손실률인 $(k-m)/k$ 이 낮은 태스크에게 높은 우선순위를 준다. 이 두 방식 모두 준경성의 제한성(constraint)을 정확히 반영하지 못하는 단점을 가지고 있다. 따라서 본 연구는 준경성의 제한성을 보다 잘 반영하는 확률을 제안하고, 이에 기반하는 스케줄링 알고리즘을 제시한다.

본 논문은 다음과 같은 순서로 이루어진다. 2 절에서는 연구 동기 및 시스템 모델을 제시하고 3 절에서는 제안하는 알고리즘을 설명한다. 4 절에서 모의실험을 통한 실험 결과를 분석하며 5 절에서 결론을 맺는다.

2. 연구 동기 및 시스템 모델

2.1 연구 동기

최근에 (m, k) -준경성 실시간 시스템에 관해 많은 연구가 이루어져, 스케줄링 알고리즘도 많이 제안이 되었다. 대부분의 스케줄링 알고리즘에서 TASK들의 우선 순위를 설정할 때, 크게 두 가지 방법을 사용한다. 첫번째는 DBP 방식으로 연속해서 종료시한을 못 맞춰도 되는 거리에 기반하는 것이며[2, 6], 두번째는 손실률에 기반하는 방법이다[5]. 하지만, 이 두 방법 모두 (m, k) -준경성 제한성(constraint)의 개념을 정확히 반영하지는 않는다.

예를 들어, (1, 3)과 (3, 9)의 준경성 제한성을 가지는 두 TASK가 주어졌다고 하자. (m, k) -준경성 제한성의 관점에서 보면, 이 두 TASK의 우선순위는 비슷하면서 (1, 3)-준경성 TASK가 약간 높은 우선순위를 가져야 한다. 왜냐하면, 긴 시간 동안에 서로 만족해야 하는 비율은 모두 1/3이지만 (1, 3)-준경성 제한성이 (3, 9)보다는 약간 더 어려운 제한성을 가지기 때문이다. 하지만, DBP의 경우에 전자의 우선순위가 2이며 후자의 우선순위가 60이 되어 3 배나 차이가 난다.(거리가 작을수록 높은 우선 순위) 그리고 손실률에 기반하는 방법은 두 TASK 모두 같은 우선순위인 2/3를 갖는다.

본 연구에서는 기존의 우선순위 결정 방법과는 달리, (m, k) -준경성 제한성의 개념을 보다 잘 반영하는 확률을 제안한다. 그리고 제안하는 확률을 기반으로 (m, k) -준경성 TASK들을 스케줄링하는 알고리즘을 제안한다.

2.2 시스템 모델

사용하는 TASK 집합은 n 개의 (m, k) -준경성 TASK들로 이루어지며, 각 TASK τ_i 는 (c_i, p_i, m_i, k_i) 의 네 파라미터로 정의가 된다. TASK τ_i 는 p_i 주기로 계산 시간 c_i 를 요구하는 작업을 생성하며, 각 작업의 상대 종료시한은 p_i 로 다음 주기의 시작 전에 맞춰야 한다. (m_i, k_i) 는 준경성 제한성을 나타내는 것으로, 연속하는 임의의 k_i 개의 작업 중에서 적어도 m_i 개는 종료시한을 만족해야 한다.

2.3 (m, k) -준경성 제한성 만족 확률

본 연구에서는 주어진 (m, k) -준경성 TASK에 대해서 그것의 제한성을 만족할 확률에 대한 개념을 새로이 제안한다. 먼저, TASK의 실행확률(execution probability)을 정의하고 그 다음에 제안하는 확률에 대해서 정의한다.

한 (m, k) -준경성 TASK가 주어질 때, 그것의 실행확률(execution probability)은 임의의 주기에 발생하는 작업이 종료시한을 만족할 확률로 정의한다. 만약 TASK 집합의 이용률($=\sum_{i=1}^n c_i/p_i$)이 1보다 같거나 작을 경우, 각 TASK들의 실행확률은 1이 된다. 왜냐하면, EDF(Earliest Deadline First) 알고리즘으로 스케줄링을 하면 항상 종료시한을 만족하기 때문이다. 하지만 이용률이 1보다 큰 과부하 TASK 집합에서는 각 TASK의 종료시한을 보장받지 못한다. 이 논문에서는 각 TASK τ_i 의 실행확률 μ_i 를 TASK 집합의 이용률의 역수로 정의하여 사용하며, 이것에 대해서는 5 절에서 간단히 언급할 것이다.

한 (m, k) -준경성 TASK의 (m, k) -준경성 제한성 만족 확률((m, k) -firm constraint meeting probability)은 다음과 같이 정의한다. 즉, TASK의 실행확률 μ 를 알고 있다고 가정할 때, 임의의 주기에 생성된 그 TASK의 작업이 그것의 (m, k) -준경성 제한성을 만족할 확률로 정의한다. 그리고 그 확률은 식 1에 의해서 구할 수 있다.

$$\psi(m, k, \mu) = \sum_{i=m}^k \mu^i (1 - \mu^{k-i}) \quad \dots \text{식 1}$$

작업이 생성되어 그것의 종료시한을 만족할 확률은 μ 이며 놓칠 확률은 $(1 - \mu)$ 이다. k 개의 작업 중에서 적어도 m 개가 종료시한을 만족해야 하므로, 식 1과 같이 (m, k) -준경성 제한성 만족 확률이 정의된다.

예를 들어, (1, 3)-준경성 TASK, (2, 6)-준경성 TASK, (3, 9)-준경성 TASK가 주어지고, 각 TASK들의 실행확률이 모두 0.66이라고 가정하자. 이때, 각각의 준경성 제한성 만족 확률은 $0.9607(=\psi(1, 3, 0.66))$, $0.9805(=\psi(2, 6, 0.66))$, $0.9906(=\psi(3, 9, 0.66))$ 이 된다. 이 값들에 의해, 세 제한성은 서로 비슷한 정도의 어려움을 가지지만 약간의 차이가 있음을 알 수 있다. 즉, τ_1 이 τ_2 보다, 그리고 τ_2 는 τ_3 보다 조금씩 더 어려운 제한성을 가진다. 이처럼 (m, k) -준경성 제한성 만족 확률이 기존의 DBP나 손실률 기반 방법들 보다 (m, k) -준경성 제한성의 개념을 보다 잘 반영한다.

3. 제안하는 스케줄링 알고리즘

제안하는 스케줄링 알고리즘은 두 부분으로 구성된다. 첫번째는 주어진 제한성인 m 와 k 를 동적으로 관리하는 것이고, 두번째는 TASK의 우선순위를 결정하는 것이다.

3.1 m_i 와 k_i 의 관리

TASK τ_i 는 주기 p_i 마다 c_i 의 계산 시간을 요구하는 작업을 생성한다. 그 작업들의 종료시한을 만족하는지에 관한 기록을 관리하기 위해서 $(k_i - 1)$ -비트의 h_i 비트열을 관리한다. h_i 비트열에서 제일 오른쪽 비트가 바로 이전 주기의 상태를 의미하는 것으로, $h_i[j]$ 비트는 이전 $(k_i - j)$ 번째 주기의 작업이 종료시한을 만족했는지 여부를 나타낸다. 만약 만족했다면 1로 표시되며, 그렇지 않으면 0으로 표시된다. 시작 시점의 h_i 비트열은 모두 1로 표시된다.

주어진 m_i 와 k_i 는 TASK가 실행되면서 변하게 되는데, 현재 제한성을 나타내기 위해서 m_i' 와 k_i' 를 사용한다. m_i' 와 k_i' 는 시작 시점에 각각 m_i 와 k_i 로 설정이 되며, 매 주기가 끝날 때마다 다음과 같이 재설정한다.

- $k_i' \leftarrow h_i$ 비트열에서 제일 왼쪽 0의 위치; 만약 없으면 k_i
- $m_i' \leftarrow m_i - (h_i[k_i'] \text{과 } h_i[k_i'] \text{ 비트 사이의 1의 개수})$

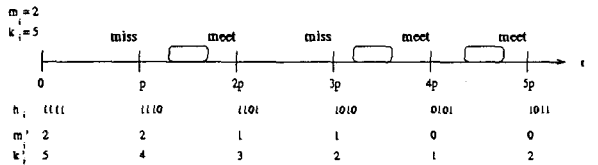


그림 1 m_i' 와 k_i' 의 관리 예제

그림 1은 (2, 5)의 (m, k) 대해서 실행 도중에 m_i' 와 k_i' 를 관리하는 예를 보여 준다. 제안하는 m_i' 와 k_i' 를 관리 방식은 다음과 같은 의미를 가진다. 즉, TASK가 주어진 (m, k) -준경성 제한성을 만족하기 위해서 향후 k_i' 주기 동안 적어도 m_i' 개의 종료시한을 만족해야 하는 것을 의미한다.

3.2 동적 TASK 우선순위 결정 방법

제안하는 우선순위 결정 방법들은 위에서 제시한 (m, k) -준경성 제한성 만족 확률에 기반하며, 다음의 여러 값들을 참조하여 TASK들의 우선순위를 정한다.

- $\psi_c(\tau_i)$: 타스크 τ_i 가 현재 시점에서 자신의 (m_i, k_i) -준경성 제한성을 만족할 확률
- $\psi_i(\tau_i)$: 타스크 τ_i 가 현재 주기에서 종료시한을 만족할 때 자신의 (m_i, k_i) -준경성 제한성을 만족할 조건부 확률
- $\psi_o(\tau_i)$: 타스크 τ_i 가 현재 주기에서 종료시한을 놓칠 때 자신의 (m_i, k_i) -준경성 제한성을 만족할 조건부 확률
- $\psi_e(\tau_i)$: $\psi_i(\tau_i)$ 와 $\psi_o(\tau_i)$ 의 차이

3.2.1 MPBP- ψ_c

이 방식에서는 각 타스크의 $\psi_c(\tau_i)$ 값을 우선순위로 정의한다. $\psi_c(\tau_i)$ 는 현재 시점에서 타스크 τ_i 가 자신의 제한성을 만족할 확률이므로, 식 1에 의해 $\psi_c(\tau_i)$ 는 $\psi(m_i', k_i', \mu_i)$ 가 된다. 왜냐하면 현재 시점에서 향후 k_i' 개의 주기 중에 적어도 m_i' 개의 종료시한을 만족해야 하기 때문이다.

MPBP- ψ_c 은 타스크의 $\psi_c(\tau_i)$ 값이 작을수록 높은 우선순위를 가진다. 따라서 스케줄러는 타스크들의 $\psi_c(\tau_i)$ 값을 중에서 가장 작은 것을 택해서 그 타스크를 실행한다. 그림 2는 타스크 집합 $\{\tau_i(2, 3, 2, 5), \tau_2(2, 4, 3, 4), \tau_3(1, 2, 1, 3)\}$ 에 대해서 MPBP- ψ_c 기반으로 스케줄링을 하는 예를 보여 준다.

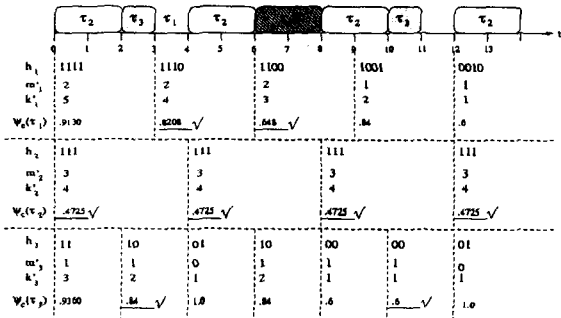


그림 2 MPBP- ψ_c 기반 스케줄링 예제

3.2.2 MPBP- ψ_o

이 방식에서는 각 타스크의 $\psi_o(\tau_i)$ 값을 우선순위로 정한다. $\psi_o(\tau_i)$ 를 구하기 위해서 현재 주기의 종료시한을 놓쳤을 때의 h_i^o 를 먼저 구한다. h_i^o 는 현재의 h_i 비트열을 왼쪽으로 1-비트 이동하고 맨 오른쪽 비트를 0으로 채운다. 그 다음, h_i^o 로부터 m_i^o 와 k_i^o 를 구하여 $\psi(m_i^o, k_i^o, \mu_i)$ 에 의해서 얻어진다. 그리고, $\psi_o(\tau_i)$ 값이 낮은 타스크가 높은 우선순위를 가진다. 이 방식은 제한성 만족 확률 뿐만 아니라 실패 상태로 가는 것을 미리 방지한다.

3.2.3 MPBP- ψ_e

이 방식에서는 각 타스크의 $\psi_e(\tau_i)$ 값에 의해서 우선순위를 정한다. $\psi_e(\tau_i)$ 는 $\psi_i(\tau_i) - \psi_o(\tau_i)$ 로 정의되어 현재 주기에서 종료시한을 만족했을 때의 확률과 놓쳤을 때의 확률의 차이값으로 일종의 보상값(reward)으로 작용한다. 따라서 이 값이 높은 타스크에게 우선순위를 준다.

$\psi_i(\tau_i)$ 를 구하기 위해서 현재 주기의 종료시한을 만족했을 때의 h_i^i 를 먼저 구한다. h_i^i 는 현재의 h_i 비트열을 왼쪽으로 1-비트 이동하고 맨 오른쪽 비트를 1로 채운다. 그 다음, h_i^i 로부터 m_i^i 와 k_i^i 를 구하여 $\psi(m_i^i, k_i^i, \mu_i)$ 에 의해서 얻어진다.

4. 모의실험 결과

제한하는 알고리즘들의 성능을 모의실험을 통해서 평가하였다. 모의실험은 각 이용률별 타스크 집합을 500 개 생성하여 그 값들의 평균을 사용하였다. 비교 척도는 각 타스크 별로 모의실험 기간 동안 자신의 (m, k) -준경성 제한성을 만족한 비율을 이용하였다.

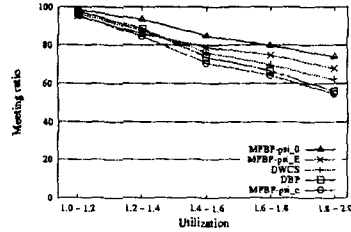


그림 3 모의실험 결과

그림 3은 모의실험 결과로 이전 연구 DBP [2]와 DWCS [3]와 성능을 비교하였다. 실험 결과로, MPBP- ψ_o 가 가장 좋은 성능을 보인다. 두번째 좋은 성능을 보이는 것은 낮은 과부하 상태에서는 DBP이며, 높은 과부하 상태에서는 MPBP- ψ_e 이다. 지면상 다른 실험 결과는 생략한다.

5. 결론

본 연구는 (m, k) -준경성 타스크 스케줄링 알고리즘에서 새로운 제한성 만족 확률을 제안하고, 이를 기반으로 우선순위를 부여하는 스케줄링 알고리즘을 제안하였다. 제안하는 확률은 각 타스크가 자신의 (m, k) -준경성 제한성을 만족하는 확률을 의미하는 것으로, 이전 우선순위 결정 방법에서 사용하던 방식보다 더 (m, k) -준경성 제한성 개념을 잘 반영한다.

본 논문에서는 모든 타스크들간 동일한 실행확률을 가정하였다. 이는 미리 스케줄링하여 조정할 수도 있으며, 또는 각 타스크 별로 우선순위를 부여하는 데에도 사용할 수 있다. 예를 들어, 타스크의 실행확률을 낮추어서 실행 도중에 높은 우선순위를 부여 받을 수 있다. 향후 연구에서 이에 대해서 자세히 분석할 것이다.

참고문헌

- [1] G. Koren and D. Shasha, "Skip-Over: Algorithms and Complexity for Overloaded Systems that Allow Skips," *Proc. IEEE Real-Time Systems Symp.*, pp. 110-117, December 1995.
- [2] M. Hamdaoui and P. Ramanathan, "A Dynamic Priority Assignment Technique for Streams with (m, k) -Firm Deadlines," *IEEE Trans. On Computers*, vol. 44, no. 12, pp. 1443-1451, December 1995.
- [3] J. Cho and H. Shin, "Heuristic Scheduling for Multimedia Streams with Firm Deadlines," *Proc. IEEE Conf. On Real-Time Computing Systems and Applications*, pp. 67-72, October 1997.
- [4] G. Bernat, A. Burns, and A. Llamasi, "Weakly Hard Real-Time Systems," *IEEE Trans. on Computers*, vol. 50, no. 4, pp. 308-321, April 2001.
- [5] R. West and C. Poellabauer, "Analysis of a Window-Constrained Scheduler for Real-Time and Best-Effort Packet Streams," *Proc. IEEE Real-Time Systems Symp.*, pp. 239-248, November 2000.
- [6] M. Hamdaoui and P. Ramanathan, "Evaluating Dynamic Failure Probability for Streams with (m, k) -Firm Deadlines," *IEEE Trans. on Computers*, vol. 46, no. 12, pp. 1325-1337, December 1997.