실시간 시스템을 위한 실시간 메모리 교체 기법*

가진호[0] 김재훈
아주대학교 정보통신전문대학원 정보통신공학부
{signifie, jaikim}@madang.ajou.ac.kr

Real-Time Memory Swapping Policy for Real-Time System

Jin-Ho Ka[0] Jai-Hoon Kim
Graduate School of Information and Communication, Ajou University

요 약

실시간 시스템의 중요성이 높아짐에 따라 실시간 시스템의 성능 향상을 위한 다양한 방법들이 연구되어 왔다. 본 논문에서는 실시간 시스템에서 가상 페이지 시스템을 통해 메모리를 관리할 때 기존의 LRU 방식에 의한 메모리 교체 방법 대신 실시간 프로세스 스케쥴링과 유사한 실시간 메모리 교체 기법을 제안하였다. 실시간 메모리 교체 기법은 가장 오래 전에 사용된 페이지를 먼저 교체하는 LRU 방식의 메모리 교체 기법 대신 실시간 프로세스의 마감 시간 또는 주기를 기반으로 페이지를 교체함으로써 시간제약성을 준수할 가능성을 높인다. 시뮬레이션을 통해 성능을 평가한 결과 실시간 메모리 교체 기법을 통해 실시간 프로세스의 마감 시간 성공 가능성이 높아짐을 확인하였다.

1. 서론

실시간 시스템은 산업 시스템에서 많이 사용되어진 최근 네트워크의 발전과 멀티미디어를 비롯한 다양한 어플리케이션들의 발전으로 인해 실시간 시스템의 중요성이 점차 높아지고 있다. 실시간 시스템상에서 프로세스들의 실시간 제약성을 효과적으로 만족시키기 위해 기존의 운영 체제를 개선시키거나 전용의 실시간 운영체제를 개발하여 사용해왔다. 이러한 실시간 운영 체제들의 다양한 실시간 프로세스 스케쥴링 알고리즘을 비롯해서 인터럽트의 처리 방법, 타이머의 개선, 시스템 자원 사용시의 교착 상태 해결과 같은 여러 관점에서의 연구가 이루어져 왔다[1]. 많은 실시간 시스템의 메모리 관리는 메모리 사용을 위한 준비 시간을 줄이거나 예측 가능하도록 하기 위하여 가상 메모리를 사용하지 않고 프로세스를 항시 주기억 장치에 상주시키는 경우가 많다[2]. 그러나 실시간 시스템에서 다양한 처리가 요구됨으로써 프로세스의 크기가 증가하고 프로세스의 수가 증가할 경우 가상 메모리 사용과 메모리 관리가 필요하다. 기존의 운영체제에서 사용하던 메모리 관리 기법을 그대로 사용할 경우 실시간 시스템에는 적합하지 않을 수 있다. 본 논문에서는 가상 페이지 시스템을 통해 메모리를 관리하는 실시간 시스템에 보다 적합한 실시간 메모리 교체 방식을 제안하고 성능을 평가한다.

2. 관련연구

실시간 시스템은 주기적 또는 비주기적으로 발생하는 이벤트에 대해 일정한 시간 내에 처리를 완료해야 하는 시간 제약성을 갖는 프로세스를 수행한다. 이러한 시간적 제약성을 지키며 보다 많은 실시간 프로세스들의 성공적인 실행을 위해 다양한 실시간 프로세스 스케쥴링 방법들이 제안되었다. 주기적으로 발생하는 이벤트를 정해진 시간 내에 처리해야 하는 대표적인 실시간 프로세스 스케쥴링 방식으로는 RM(Rate Monotonic), DM(Deadline Monotonic), EDF(Earliest Deadline First), LST(Least Slack-time First)가 사용되며, 비주기적으로 발생하는 실시간 및 비실시간 이벤트를 처리하는 실시간 프로세스 스케쥴링 방식은 비주기 작업 서버를 이용하는 DS(Deferrable Server), SS(Sporadic Server), CUS(Constant Utilization Server), TBS(Total Bandwidth Server), CBS(Constant Bandwidth Server)가 대표적이다[3].

기존의 UNIX계열 운영체제, Linux등은 가상메모리 시스템을 통해 시스템의 메모리를 관리하며 이는 페이징 시스템과 스왑핑 시스템으로 구성된다. 실행중인 프로세스의 메모리는 가상 페이지들로 구성되며 스와핑 시스템에 의해 물리적 메모리에 적재되거나 제거되어진다. 실행중인 프로세스가 사용하려는 가상 페이지를 물리적 메모리에 적재되어 있지 않다면 디스크등에 위치하는 스왑 구역에서 가상 페이지를 가져와 물리적 메모리에 적재하여야 하는데, 물리적 메모리에 사용 가능한 페이지가 없으면 운영체제는 물리적 메모리의 페이지에서 다른 페이지를 제거하여 가져올 페이지를 위한 공간을 마련하여야 한다[4][5].

실시간 시스템에서 실시간 프로세스만의 충분한 메모리 공간을 확보하여 가상 메모리를 사용하지 않으며 페이지의 교체가 일어나지 않게 하는 방식이 가장 좋은 방법이겠지만 실시간 프로세스들의 모든 페이지들을 적재할 메모리 공간이 부족할 경우 페이지 교체 시간의 오버헤드를 줄일 수 있는 방법이 필요하다. 디스크 접근이 많은 실시간 멀티미디어 시스템의 경우, 디스크 접근 횟수를 줄이기 위해 LRU와 같은 전통적인 디스크 버퍼 교체 알고리즘 대신 멀티미디어 시스템에 적합한 새로운 버퍼 교체 알고리즘이 제안되기도 했다[6].

3. 실시간 메모리 교체 알고리즘

실시간 메모리 교체는 가상 페이지 시스템을 사용할 때 실시간 프로세스의 스케쥴링 방식과 유사한 방법으로 메모리 내의 페이지를 교체한다. 즉 시간 제약성을 중요시하며 수행될 프로세스를 결정하는 것과 마찬가지로 시간 제약성을 중요시하여 주기억 장치에서 교체될 프로세스의 페이지를 교체한다.

* 본 연구는 한국과학재단 목적기초연구(2001-1-30300-016-2) 지원으로 수행되었음.

본 논문에서는 우선 순위 기반의 프로세스 스케쥴링 방식을 사용하는 실시간 시스템에서의 페이지 교체 알고리즘을 제안한다.

기존의 페이지 교체 방식인 LRU 방식을 사용한 RM 프로세스 스케쥴링의 프로세스 실행과 메모리 교체를 살펴보면 그림 1과 같다.

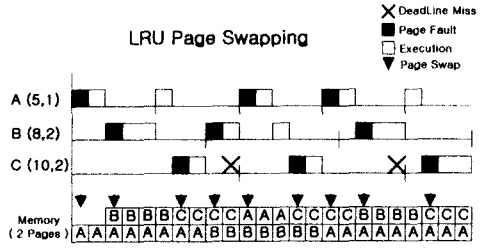

그림1. RM 프로세스 스케쥴링, LRU 페이지 교체시

그림 1에서 프로세스 A, B, C의 주기와 실행시간은 (5,1), (8,2), (10,2)이며 각 프로세스의 크기는 1 page이고 주기억 장치의 크기는 2 page라고 가정하였고 페이지 교체에 1 단위시간이 소요된다고 가정하였다. 프로세스 스케쥴링 알고리즘은 RM을 가정하여 A, B, C의 순서대로 우선권을 갖는다. LRU 알고리즘이 적용되는 페이지 교체는 프로세스의 실행 전에 필요한 페이지가 실제 메모리에 존재하지 않을 경우 가장 오래 전에 사용되었던 페이지가 스왑 구역으로 스왑 아웃되고 필요한 페이지가 실제 메모리로 스왑 인된다.

그림 2는 페이지 교체 방식을 프로세스 스케쥴링 방식과 동일한 RM 알고리즘을 적용한 경우의 프로세스 실행과 페이지 교체의 발생을 나타낸다.

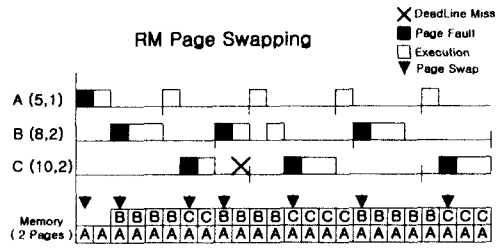

그림2. RM 프로세스 스케쥴링, RM 페이지 교체시

프로세스 스케쥴러에 의해 실행 될 프로세스가 선택되면 해당 프로세스의 가상 페이지가 메모리에 존재하는지를 검사한다. 만약 메모리에 프로세스의 페이지가 존재하지 않으며 비어있는 페이지가 존재하지 않는 경우, 페이지를 스왑 인 하기위한 공간을 확보하기 위해 메모리 교체 알고리즘에 의해 가장 우선 순위가 떨어지는 프로세스의 페이지를 스왑 아웃한다. 이 때 사용되는 메모리 교체 알고리즘의 프로세스 스케쥴링 알고리즘과 동일한 알고리즘을 적용한다. 즉 교체될 페이지를 선택할 때 주기가 가장 긴 프로세스의 페이지를 선택한다.

그림 2에서 프로세스 스케쥴링 알고리즘은 RM이므로 제안된 실시간 메모리 교체 방식을 적용시키면 페이지의 교체 알고리즘 역시 RM방식으로 교체가 이루어진다. 실시간 프로세스 스케쥴링 및 실시간 메모리 교체 알고리즘으로 RM을 예로 설명하였는데 EDF, LST등도 선택이 가능하다.

4. 시뮬레이션 및 성능 평가

실시간 메모리 교체 알고리즘의 성능을 평가하기 위해 시뮬레이션 프로그램을 작성하였다. 시뮬레이션에서 가정한 시스템 모델은 주기적 프로세스 스케쥴링 알고리즘 중 우선 순위 기반의 RM, EDF 방식 프로세스 스케쥴링을 사용하는 실시간 시스템으로, 여기에 기존의 LRU 방식의 페이지 교체 시와 본 논문에서 제안하는 실시간 메모리 교체 방식(RM과 EDF)을 적용시켰을 때의 성능을 비교해 본다. 실시간 프로세스 실행의 성능 평가 기준으로 시뮬레이션 시간 동안 발생한 전체 프로세스들의 주기 중에 마감 시간을 만족하는 주기들의 비율인 마감 시간 성공 비율(Success)을 사용하였다.

그림 3은 실시간 프로세스들의 총 프로세서(CPU) 이용율(U) 변화에 따른 성능 비교로 5개의 주기적 실시간 프로세스가 실행 되고 있으며 프로세스들의 상대적 마감 시간(D_i)과 주기(P_i)는 동일하다고 가정했으며 5개의 프로세스의 주기는 표 1에 나타내었다.

표1. 프로세스들의 주기(P_i)

P_1	P_2	P_3	P_4	P_5
16	20	22	24	26

프로세스 실행시의 필요한 페이지 크기는 모두 1로 동일하며 시스템 메모리에 적재 가능한 페이지 크기는 3으로 가정하였다. 페이지 교체 시간을 제외한 실시간 프로세스들의 총 프로세서 이용율(U)은 아래와 같이 계산된다.

$$U = \sum_{i=1}^{N} (C_i / P_i)$$

C_i: 프로세스 i의 실행시간, P_i: 프로세스 i의 주기, N: 프로세스 수

그림 3은 사용율 U의 변화에 따른 RM, EDF 방식 프로세스 스케쥴링에서 기존의 LRU 방식과 RM, EDF 방식의 메모리 교체 시의 성능 비교이다.

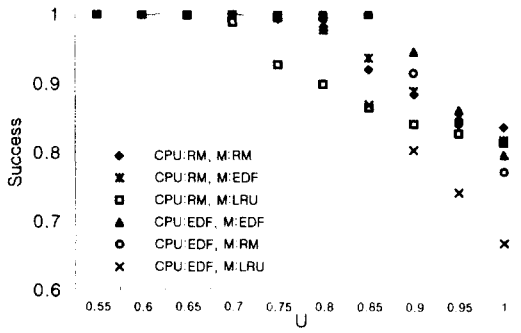

그림3. 프로세서 이용율 변화에 따른 성능 비교, RM/EDF 프로세스 스케쥴링

RM 방식 프로세스 스케쥴링에서 LRU의 경우 프로세서 이용율이 0.6이상일 때부터 마감 시간 성공 비율(Success)이 1.0이하로 떨어지기 시작하지만 RM의 경우 0.7이상일 때부터 마감 시간 성공 비율이 1.0이하로 떨어지며 EDF는 RM과 유사한 결과를 보인다. 전체적으로 프로세서 이용율이 증가하면서 RM과 EDF는 LRU보다 대략 3-8%정도 높은 마감 시간 성공 비율을 나타낸다.

EDF 방식 프로세스 스케쥴링에서 LRU는 프로세서 이용율 0.75이상일 때부터 마감 시간 성공 비율이 떨어지기 시작하며, EDF는 0.85이상부터 떨어지기 시작한다. RM은 전체적으로 EDF보다 2-4% 낮은 마감 시간 성공 비율을 보이면서 EDF와 비슷한 결과를 보여준다. EDF는 프로세서 사용율이 0.95일때 LRU보다 최대 10% 높은 마감 시간 성공 비율을 보여준다. 이러한 그림 3의 결과는 LRU방식의 페이지 교체보다 RM, EDF방식의 페이지 교체가 모든 프로세서 사용율 변화 영역에서 보다 높은 마감 시간 성공 비율을 나타낸

다고 볼 수 있다.

그림 4는 메모리의 물리적 크기 변화에 따른 성능의 변화를 나타낸다. 10개 주기적 실시간 프로세스가 존재한다고 가정하였으며 프로세스들의 실행시간과 크기는 1로 하였고 주기(마감시간)는 표 2에 나타내었다.

표2. 프로세스들의 주기(마감시간)

프로세스	주기	프로세스	주기
1	10	6	15
2	11	7	16
3	12	8	17
4	13	9	18
5	14	10	19

전체 물리적 메모리의 페이지 크기를 10부터 1로 변화 시켰을 때의 결과를 시뮬레이션하였다.

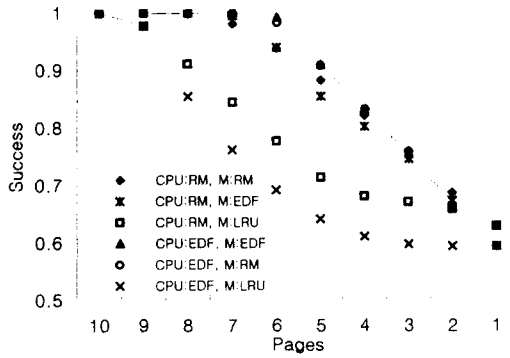

그림4. 페이지 크기 변화에 따른 성능 비교, RM/EDF 프로세스 스케쥴링

모든 프로세스들의 페이지를 저장할 수 있는 충분한 크기의 메모리 공간을 갖지 못해 페이지 교체가 발생하면서 마감 시간 성공 비율이 변하는 결과를 나타낸다. RM 방식 프로세스 스케쥴링과 EDF 방식 프로세스 스케쥴링의 경우 모두 페이지 교체가 발생하지 않는 page 크기가 10인 때와, 페이지 교체가 가장 빈번하게 일어나는 page 크기 1인 때 LRU와 RM/EDF는 동일한 마감 시간 성공 비율를 나타내지만 그 외의 구간에서는 RM 프로세스 스케쥴링시 RM/EDF는 LRU보다 최대 17%, EDF 프로세스 스케쥴링시 RM/EDF는 LRU보다 최대 30% 높은 마감 시간 성공 비율을 나타낸다.

그림 4는 페이지 크기가 1일 때 모든 경우 페이지 교체가 이루어 지는 경우를 제외하고는 모든 프로세스들의 총 필요 페이지 수보다 적을 때 RM, EDF방식의 실시간 메모리 교체 방식의 성능이 기존의 LRU 방식보다 우수함을 나타낸다.

그림 5는 페이지 교체 시간 (Swap time)의 변화에 따른 성능 비교로 10개의 주기적 실시간 프로세스들은 표 3과 같이 1000에서 1900의 주기(마감시간)를 갖고 실행 시간은 50에서 150이며 프로세스의 크기는 1로 가정하였다.

표3. 프로세스들의 입력 파라메터

프로세스	주기	실행시간	프로세스	주기	실행시간
1	1000	90	6	1500	130
2	1100	110	7	1600	60
3	1200	80	8	1700	140
4	1300	120	9	1800	50
5	1400	70	10	1900	150

전체 프로세서 이용율은 0.71로 고정되어 있으며 물리적 메모리의 크기는 7로 가정하여 페이지 교체 시간이 10일 때부터 200일 때까지의 결과를 나타낸다.

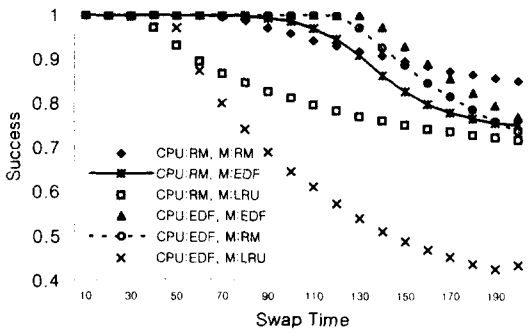

그림5. 페이지 교체 시간 변화에 따른 성능 비교, RM/EDF 프로세스 스케쥴링

RM 방식 프로세스 스케쥴링일 때 페이지 교체 시간이 증가함에 따라 LRU보다 RM이 최대 15%, EDF는 최대 17% 높은 마감 시간 성공 비율을 나타낸다. EDF는 페이지 교체 시간이 120이하일 때까지는 RM보다 1-3% 높은 마감 시간 성공 비율을 나타내지만 그 이후로는 최대 10% 낮은 마감시간 성공 비율을 나타낸다.

EDF 방식의 프로세스 스케쥴링일 때는 페이지 교체 시간이 50일 때부터 LRU의 마감 시간 성공 비율이 1.0 이하로 떨어지는 반면 EDF는 페이지 교체 시간이 140이상일 때부터, RM은 130 이상일 때부터 마감 시간 성공 비율이 1.0 이하로 떨어진다. EDF는 페이지 교체 시간 130이후 140에서 LRU보다 46%나 높은 마감 시간 성공 비율을 나타낸다. 그림 5는 메모리 교체 시간이 증가 하면서 LRU방식의 메모리 교체보다 실시간 메모리 교체가 보다 높은 마감 시간 성공 비율을 갖는 것을 나타낸다.

5. 결론 및 향후 과제

본 논문에서는 실시간 시스템에서의 실시간 메모리 교체 기법을 제안하여 시뮬레이션을 통해 성능을 평가했다. 기존의 LRU 방식 메모리 교체 방식대신 실시간 프로세스의 스케쥴링과 유사한 실시간 메모리 교체 방식을 사용함으로써 실시간 프로세스의 마감 시간 성공 비율을 높일 수 있음을 알 수 있었다. 향후 연구 과제로는 비주기적 실시간 프로세스들의 실시간 메모리 교체 방식 적용에 대한 성능 평가와 실제 시스템으로의 적용에 대한 연구가 있다.

참고 문헌

[1] Jane W. S. Liu, " Real-Time Systems" , Prentice Hall, 2000.

[2] Michael Barabanov, Victor Yodaiken, " Real-Time Linux" , *Linux Journal*, Mar. 1996.

[3] Maurice J. Bach, " Design of the Unix Operating System" , Prentice Hall, 1987

[4] Daniel P. Bovet, Marco Cesati, " Understanding the Linux Kernel" , *O' Relly*, 2001.

[5] 심재홍, " 다양한 실시간 스케줄러를 지원하기 위한 커널 구조화 및 재구성 방안" , 아주대학교 대학원 컴퓨터공학과 공학 박사학위 논문, 2001.

[6] Banu Ozden, Rajeev Rastogi, Avi Silberschatz, " Buffer Replacement Algorithms for Multimedia Storage Systems" , In *Proceedings of the International Conference on Multimedia Computing and Systems*, Hiroshima, Japan, June, 1996