데이터 페이지를 고려한 실시간 멀티 태스크 환경에서의 페이지 교체정책

이두희[1], 이창건[1], 김강희[2]
서울대학교 컴퓨터공학부[1], 숭실대학교 컴퓨터학부[2]
dhlee@rubis.snu.ac.kr, cglee@snu.ac.kr, kim.kanghee@gmail.com

Paging Replacement Policy for Multi Task based Real-time Application considering Data pages

Duhee Lee[1], Chang-Gun Lee[1], Kanghee Kim[2]
Seoul National University[1], Soongsil University[2]
dhlee@rubis.snu.ac.kr, cglee@snu.ac.kr, kim.kanghee@gmail.com

요 약

우리는 기존 연구에서 NAND Flash Memory의 실시간 멀티 태스크 환경에서 응용프로그램의 실시간성을 보장하기 위한 페이지 교체 방법인 "mRT-PLRU(Multi-tasking Real-Time constrained combination of Pinning and LRU)"를 완성하였다. 이 페이지 교체 정책은 응용프로그램이 목표치보다 높은 확률도 데드라인을 만족 할 수 있도록 하기 위해서 응용프로그램의 코드페이지 프로파일링 작업이 우선시 되어야 한다.

하지만 응용프로그램 안에는 코드페이지 뿐만 아니라 데이터 페이지도 존재하고 있다. 이 논문은 이 부분에 주목하여, 기존의 코드페이지 프로파일링을 데이터 페이지 프로파일링까지 확장하였고, 프로파일링 결과물을 이용하여 mRT-PLRU 페이지 교체 정책을 데이터 페이지까지 다룰 수 있도록 개선된 방식을 제시한다.

1. 서 론

실시간 응용 프로그램은 지정된 확률 이상으로 데드라인을 만족해야 의미가 있다. 경성 실시간 응용 프로그램일 경우, 그 값은 100%이고, 연성 실시간 응용 프로그램일 경우, 그 값은 사용자가 지정한다. 대표적인 연성 실시간 응용 프로그램으로는 미디어 플레이어가 있다. 미디어 플레이어는 사용자의 요구에 따라 1초에 15회에서 60회 정도의 미디어 파일 디코딩(decoding) 한다. 만일 디코딩이 사용자 요구시간 내로 끝나지 못 할 경우, 사용자는 끊겨서 재생되는 미디어를 시청해야 하는 불편함을 경험하게 된다.

사용자에게 이러한 불편함을 경험하지 않게 하기 위해서, 실시간 응용프로그램을 데드라인 안에 수행하기 위한 연구가 많이 진행되었다. 가장 고전적인 방식으로는 쉐도윙(shadowing)[5]방식이 있다. 컴퓨터 구조에서 상대적으로 실행 시간이 빠르고, 페이지 폴트가 발생하지 않는 저장장치인 메모리에 실시간 응용 프로그램을 모두 올려놓고 실행을 하는 쉐도윙 방식을 이용하면, 연성 실시간 응용 프로그램이 데드라인을 만족 확률이 가장 우수하겠지만, 현실적으로 모든 연성 실시간 응용프로그램을 메모리에 모두 올려놓기는 불가능 하다. 따라서 제 2 저장소에 응용 프로그램을 적재하고, 필요에 의해서 해당 응용프로그램의 해당 페이지를 메모리로 복사해야 하는데, 이 과정에서는 필연적으로 페이지 폴트가 발생한다.

우리는 기존 연구에서, 페이지 폴트가 발생하는 상황에서, 여러 개의 실시간 응용프로그램이 동시에 데드라인 만족 확률을 보장 받기 위한 페이지 교체 방식인 mRT-PLRU(Multi-tasking Real-Time constrained combination of Pinning and LRU)[3]를 완성하였다.

mRT-PLRU 페이지 교체 방식은 연성 실시간 응용 프로그램의 코드 페이지를 프로파일링 하여, 연성 실시간 응용 프로그램에서 많이 쓰이는 페이지를 메모리에 고정적으로 적재하고, 나머지를 LRU방식으로 교체하는 전략을 사용한다.

하지만 mRT-PLRU는 문제를 단순화하여 해결하기 위하여, 코드 페이지만을 프로파일링 하여 페이지 교체에 참고한다. 실제 연성 실시간 응용프로그램은 코드 페이지 뿐만 아니라 데이터 페이지도 동시에 접근하면서 수행되기 때문에, 데이터 페이지도 페이지 교체에 참고하여 동작하는 것은 필연적인 상황이다.

데이터 페이지는 코드 페이지와 달리 쓰기 동작이 동반되기 때문에, 해결해야 할 문제가 증가한다. NAND

Flash 쓰기 동작은 읽기 동작보다 느리기 때문에, 페이지 폴트가 발생 했을 때, 쓰기와 읽기는 분리되어 다뤄져야 한다.

이 논문은 NAND Flash 메모리의 페이지 폴트가 발생하는 모바일 임베디드 상황에서, 데이터 페이지를 포함하는 여러 개의 연성 실시간 응용 프로그램을 수행하기 위한 연구를 다루고 있다.

2. 배 경 (mRT-PLRU)

mRT-PLRU는 연성 실시간 응용프로그램이 지정된 데드라인 만족 확률을 보장 받도록 하기 위해서, 태스크별 분석 단계와 stochastic analysis in loop optimization 단계로 구성되어 있다.

2.1 태스크별 분석 단계

태스크별 분석 단계는 다음과 같은 루프를 수행한다.

- 미디어 플레이어가 x개의 페이지만 사용 가능하도록 세팅 한다.
- x개의 페이지 중, 조합 가능한 (pinning page, LRU page)에 대해서, 각각의 미디어 플레이어의 평균 실행시간을 기록한다.
- x+1 개의 페이지가 사용가능하도록 세팅 한 다음에 처음으로 돌아간다.
- 이 작업은 n값이 쉐도잉 크기와 같을 때 까지 한다.

이렇게 태스크별 분석을 하면 메모리 사이즈가 x인 경우 그림 [1]과 같은 그래프를 얻을 수 있다.

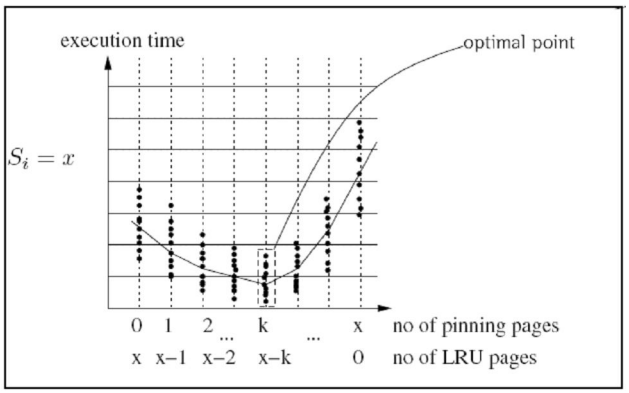

그림 [1] 메모리 사이즈 x인 경우의 실행시간

이 경우에서 평균 미디어 플레이어의 실행시간이 가장 작은 부분이, 메모리 사이즈 x인 경우의 최적 (pinning page, LRU page) 조합이다. 이 조합을 모든 메모리 사이즈에 대해서 조사하면, 그림 2[A]와 같은 최적 평균 실행 시간 그래프를 얻을 수 있다.

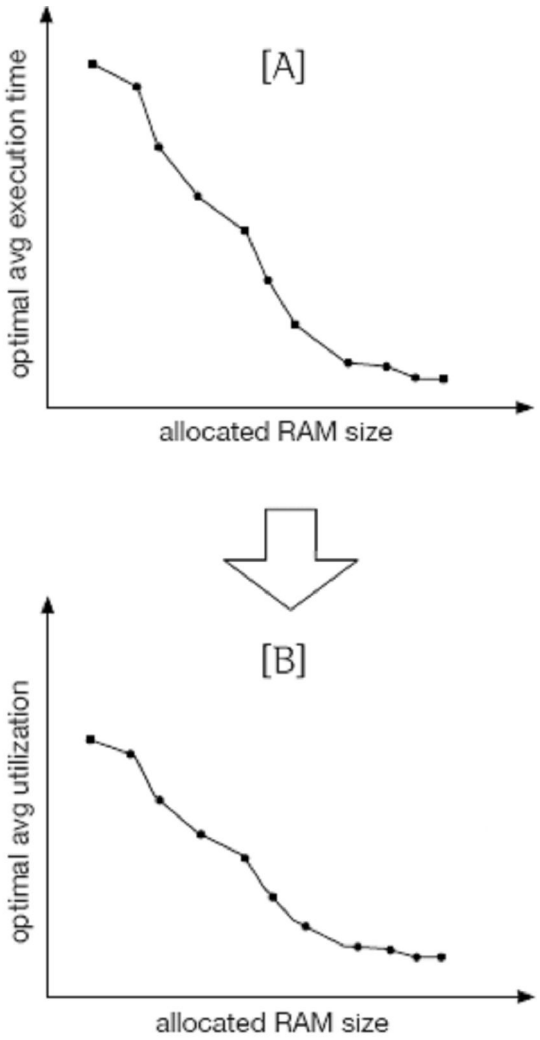

그림 [2] 최적 평균 실행 시간 그래프와
최적 평균 이용률 그래프

태스크별로 주기가 다를 경우에는 평균 실행 그래프보다 평균 이용률 그래프가 더 의미 있기에, 마지막으로 최적 평균 실행 그래프를 이용률 그래프로 변환하면, 태스크별 분석 단계는 마무리 된다.

2.2 stochastic analysis in loop optimization

이 단계에서는 태스크별 분석에서 얻어진 최적 평균 이용률 그래프를 이용해서 태스크에게 메모리를 할당하는 작업을 한다.

초기 세팅으로는 모든 태스크에 대해서 메모리를 한개씩만 할당한 후, 모든 태스크를 수행한다. 태스크에 메모리가 한개씩 할당된 경우엔 모든 태스크가 데드라인을 만족하지 못한다고 가정을 하자. 그럼 이 상황에서 최적 평균 이용률 그래프의 기울기가 가장 높은 태스크에게 메모리를 한개 더 할당한다. 이 말은 해당 태스크에 메모리를 줄 경우, 시스템 전반적으로 이용률이 낮아져서, 데드라인을 만족 할 확률이 높아짐을 의미하기 때문이다.

이렇게 해당 태스크에 메모리를 하나씩 배정하는 작업

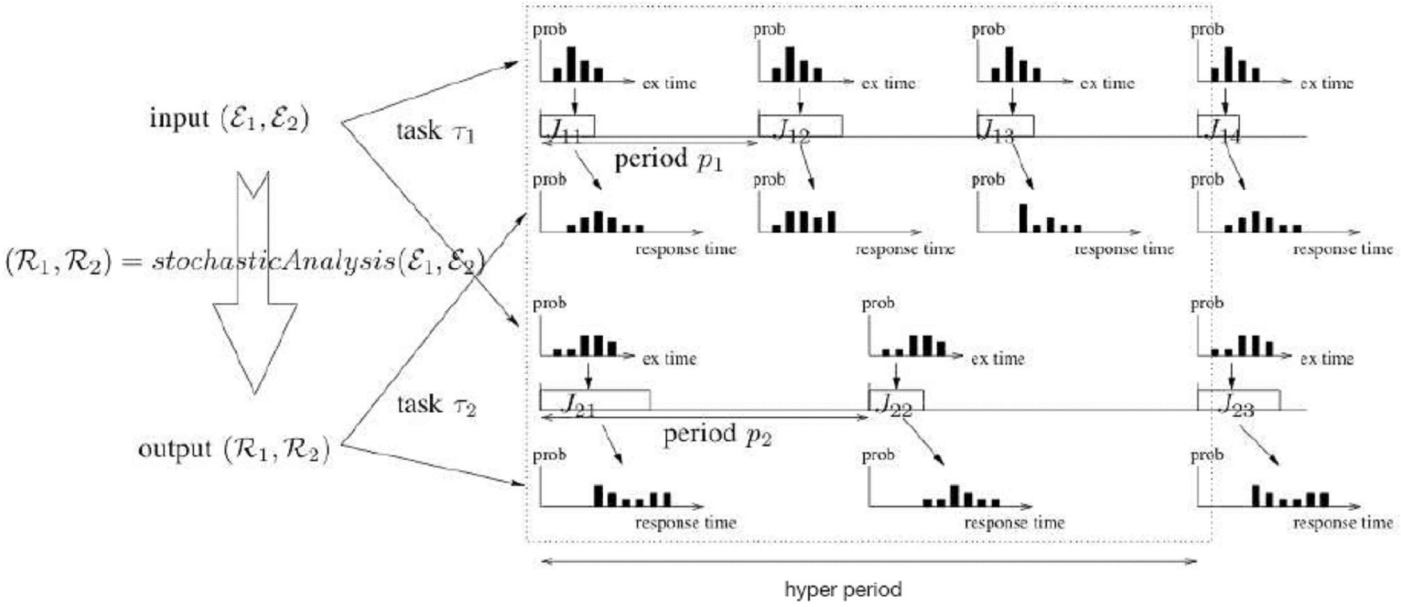

그림 [3] stochastic analysis

을 한 후, 전체 시스템이 데드라인을 만족하는지 확인을 해야 한다. 이 작업은 stochastic analysis[4]를 이용한다. 만일 전체 시스템이 데드라인을 만족하지 못 할 경우, 다시 평균 이용률 그래프의 기울기가 가장 높은 태스크에 메모리를 한개 더 배정하는 루프를 데드라인이 만족될 때 까지 반복한다. 루프가 종료되면, 태스크별로 할당된 메모리의 크기가 결정된다.

3. 데이터 페이지로 확장

연성 실시간 응용 프로그램의 대표적인 미디어 플레이어의 ELF(Executable and Linking Format) 파일에는 15.31%의 읽기전용 데이터 페이지 구간과 3.68%의 읽기-쓰기 데이터 페이지 구간이 포함되어 있다.

mRT-PLRU 페이지 교체 정책이 위 데이터 페이지들까지 다루기 위해서, 데이터 페이지 구간들의 특성을 알아보도록 하자.

3.1 데이터 페이지의 특성

미디어 플레이어의 ELF에는 15.31%의 읽기전용 데이터가 있다. 이 읽기전용 데이터 구간은 이름에서 시사하는 바와 같이 오직 읽히기 위한 데이터가 위치한다. 따라서 기본적으로 코드페이지와 하는 역할이 다르지만, mRT-PLRU 페이지 교체 정책의 입장에선 오직 읽기만 발생하는 코드 페이지 구간과 동일하게 처리 할 수 있다.

하지만 읽기-쓰기 데이터 구간은 읽기전용 데이터 구간과 상황이 다르다. 운영체제가 이 구간을 접근 하려고 할 때, 해당 페이지가 메모리에 올라와 있지 않으면, 페이지 폴트가 발생하는데, 이 때 발생하는 페이지 폴트의 타입이 코드 페이지나 읽기전용 페이지에서 발생하는 읽기(reading) 폴트와는 다른, 쓰기(writing) 페이지 폴트가 일어난다. NAND Flash Memory 에서는 쓰기위해서 걸리는 시간이 읽기 위해서 걸리는 시간보다 길기 때문에, 쓰기 페이지 폴트가 읽기 페이지 폴트보다 오래 걸린다. 따라서 mRT-PLRU가 읽기-쓰기 데이터 구간을 다루기 위해선 두 가지 방식의 폴트를 모두 처리해야 한다.

또한 NAND Flash Memory 에서는 가비지 컬렉션이 일어나면서 쓰기 효유를 급격히 떨어뜨리는 현상이 있지만, 이 논문에서는 가비지 컬렉션이 일어나지 않을 만큼 충분히 많은 페이지가 있다고 가정하였다.

3.3 mRT-PLRU로의 적용 방법

데이터 페이지를 mRT-PLRU로 확장하기 위해서는 어느 페이지를 메모리에 항상 적재할지 결정하는 문제를 해결해야 한다. 이 논문에서는 다음과 같은 두 가지 방법을 제시한다.

모든 페이지에 대해서,

- 미디어 플레이어가 재생되면서 읽기 위해 접근되는 횟수와 읽기 페이지 폴트 비용의 곱 + 미디어 플레이어가 재생되면서 쓰기 위해 접근되는 횟수와 쓰기 페이지 폴트 비용의 곱

- 각각의 프레임에서 접근되는 횟수의 최소값

첫째 방식은 미디어 플레이어의 평균적인 관점을 반영하기 위한 계획이다. 첫째 방식에 의해서 상위 랭크된 페이지는 시스템에서 평균적으로 많이 쓰이는 페이지다. 하지만 특정 프레임에서 집중적으로 사용되는 경우에 대

한 보상은 해주지 못한다.
둘째 방식은 최소값을 보장하기 위한 방식이다. 둘째 방식을 통해서 상위 랭크된 페이지는 어느 프래임이든 최소한 그 정도의 페이지 접근이 발생하는 것이다. 따라서 특정 프레임에서 집중적으로 접근이 되더라도, 이후에 접근이 잘 안되는 페이지라면 높은 점수를 받을 수 없다.

4. 실험

실험은 삼성 Q1 울트라 모바일 컴퓨터를 사용하였고, 리눅스 커널은 2.6.24.1 을 사용하였다. 연성 실시간 응용프로그램으로는 MPlayer[1]를 사용하였고, 저장소로는 Mtron SSD를 사용하였다.

4.1 데이터 페이지 프로파일링

데이터 페이지를 프로파일링 하기 위해서 리눅스의 페이지 폴트 핸들러 와 가상 메모리 시스템을 수정[2]하였다. 기본적인 전략은 다음과 같다.

1. 미디어 플레이어를 위한 가상 메모리를 1page 로 강제 할당 한다.
2. 미디어 플레이어가 수행되면서 발생하는 페이지 폴트의 주소를 수정된 페이지 폴트 핸들러에서 기록한다.

하지만 실험에 사용된 Intel 기반 시스템에서는 미디어 플레이어가 수행되기 위해서 동시에 두개 이상의 페이지가 필요한 경우가 있다. 대표적으로는 load 인스트럭션을 수행 할 때, 두개의 페이지가 동시에 메모리에 있어야 정상적으로 프로파일링이 가능하다. 동시에 있지 않을경우 두개의 페이지를 반복적으로 메모리에 올리는 무한반복 상황이 일어난다.
이 문제를 해결하기 위해서, 우리는 두 페이지가 임계치 이상 반복적으로 폴트를 일으킬 경우, 그 두 페이지를 임시적으로 메모리에 올림으로써 해결하였다.

4.1 데이터 페이지를 고려한 mRT-PLRU의 두가지 방식 비교.

섹션 3.3에서 언급된 두가지 방식중 전자를 Type 1, 후자를 Type 2로 명명하였고, 1초에 30번 프레임을 재생하는 동영상 중, 많이 알려진 동영상을 바탕으로 실험을 하면 다음과 같은 결과가 나온다.

	Type 1	Type 2
starwars 2	21	20
jurassic park	19	17

표[1] 방법에 따른 필요한 메모리의 크기

미디어 플레이어의 데이터 페이지 중, 쉐도윙에 필요한 데이터 페이지는 총 38페이지다. Type1과 Type2 모두 쉐도윙과 비교하였을 때, 절반정도의 메모리만으로 데드라인을 만족함을 알 수 있으며, Type2가 근소하게 앞서고 있음을 알 수 있다.

5. 결론

mRT-PLRU는 멀티 태스크 환경에서 실시간 응용프로그램을 수행하기 위한 좋은 방법이지만, 데이터 섹션은 다루지 못하고 있다. 이 논문은 mRT-PLRU를 데이터 섹션까지 다루도록 하는 새로운 방법을 제시하였고, 영화 두 편에 대해서 검증을 하였다.
데이터 섹션을 다루는 방법에 대해서는 이 논문에서 제시된 방법 외에도 여러 가지 방법이 연구 될 수 있다. 더 효율적인 방법은 다음 연구로 남겨둔다.

6. 사사

이 연구를 위해 연구 장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에 감사드립니다. 이 연구는 지식경제부 및 한국산업기술평가관리원의 산업원천기술개발사업(정보통신)의 일환으로 수행하였습니다.
[10035243 , CPS를 위한 콤포넌트 기반 설계이론 및 제어커널 개발]

7. 참고문선

[1] MPlayer V1.0rc2. http://mplayerhq.hu
[2] Jong-Chan Kim, Chang-Gun Lee, Eun-Yong Ha. Page Replacement Policy for Virtual-memory based Real-time Embedded Systems, in KCC2008. Jun. 2008
[3] D Lee, C.-G Lee, K Kim, A Generic Framework for Real-Time Program Executions on NAND Flash Memory in Multi-Tasking Embedded Systems, in RTSS09, Dec. 2009
[4] KangHee Kim, Chang-Gun Lee, A Safe Stochastic Analysis with Relaxed Limitations on the Periodic Task Model, in IEEE Transactions on Computer, May. 2009
[5] C.Park, J.Seo, S.Bae, H.Kim, S.Kim, and B.Kim. A Low-cost Memory Architecture with NAND XIP for Mobile Embedded Systems. In Proc. of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, Oct. 2003.
[6]Rockafellar, R.T. (1970). Convex analysis. Princeton, Princeton University Press.