리눅스 기반 스마트폰의 페이지 폴트 감소를 위한 익명 페이지의 지연된 사상 해제 기법

김정호 [1)O], 유종훈 [2)], 허승주 [1)], 홍성수 [1),2)]
[1)]서울대학교 융합과학기술대학원 융합과학부
[2)]서울대학교 전기정보공학부
{jhkim, jhyoo, sshong}@redwood.snu.ac.kr

Lazy Unmapping of Anonymous Pages for Reducing Page Faults in Linux-based Smartphones

Jeongho Kim[1)O], Jonghun Yoo[2)], Sungju Huh[1)], Seongsoo Hong[1),2)]
[1)] Department of Transdisciplinary Studies, GSCST, SNU
[2)]Department of Electrical and Computer Engineering, SNU

요 약

스마트폰 웹 브라우저의 높은 응답성은 우수한 사용자 경험을 제공하기 위해 중요한 요소이다. 이를 개선하기 위한 단말 제조사들의 노력에도 불구하고 여전히 웹 브라우저 응답성에 대한 부정적 경험이 보고되고 있다. 이 논문은 이 같은 긴 응답시간의 주요 원인이 익명 mmap 기반 동적 메모리 할당에 의해 유발되는 페이지 폴트임을 밝힌다. 익명 mmap은 통상적인 힙과 달리 할당이 해제될 때 물리 메모리로의 사상이 즉시 제거되기 때문에 새롭게 할당된 영역이 접근될 때 항상 페이지 폴트가 발생한다. 이 논문은 과도한 페이지 폴트 발생을 감소시키기 위해 익명 페이지의 지연된 사상 해제 기법을 제안한다. 이는 자유 메모리가 고갈되지 않는 한 익명 페이지의 사상을 유지시킴으로써 페이지 폴트 없이 접근되도록 지원하는 기법이다. 실험 결과 제안된 기법은 기존 시스템에 비해 웹 브라우저의 페이지 폴트 횟수를 46% 감소시키고, 응답 시간을 6% 단축할 수 있었다.

1. 서론

스마트폰 제조사들은 우수한 사용자 경험과 서비스 품질을 제공하기 위해 스마트폰의 대표적인 킬러 응용인 웹 브라우저의 응답성을 향상시키려는 노력을 기울여 왔다. 사용자는 인지하지 못할 만큼 짧은 지연시간 안에 쓸기(swipe)나 스크롤(scroll)과 같은 제스처에 대한 응답이 이루어지기를 기대한다. 통상적인 데스크탑 컴퓨팅 환경에서 이런 제스처에 대한 응답시간은 웹 페이지 로딩과 렌더링 연산에 의해 대부분 결정된다. 그런데 메모리 제약이 심한 스마트폰 환경에서는 응답시간 중 페이지 폴트 처리기의 수행 시간이 무시하지 못할 정도의 비중을 차지한다. 실제로 본 연구의 측정을 통해서 우리는 안드로이드 스마트폰에서 포털 사이트[1]에서 쓸기 동작을 수행할 때, 총 7,081번의 페이지 폴트가 발생하고, 이를 처리하는 데 걸리는 시간이 전체 응답시간 중 15%인 것을 확인하였다.

이와 같이 많은 수의 페이지 폴트가 발생하는 것은 웹 브라우저가 요청하는 많은 수의 동적 메모리 할당에 기인한다. 웹 브라우저는 렌더링 수행 중에 수많은 자바와 C++ 객체들을 사용한다. 각 객체는 생성될 때 LibC의 malloc()을 통해 가상 메모리 상에 할당되며, 소멸될 때에는 free()를 통해 할당이 해제된다. 이때 malloc()은 요청 크기에 따라 가상 메모리의 힙(heap) 또는 익명(anonymous) mmap 영역 중 한 곳에 객체를 할당한다. 할당된 객체는 리눅스 커널의 요구 사상(demand mapping) 기법에 의해 실제 물리 메모리에 사상된다.

이때 힙과 익명 mmap 영역 내의 할당은 다음과 같은 차이가 존재한다. 힙 영역의 페이지들은 할당된 객체가 free()되더라도 물리 메모리로의 사상이 해제되지 않는다. 그리고 이들 페이지는 나중에 힙 영역에 할당되는 객체를 위해 재사용된다. 따라서 힙 영역의 각 페이지들은 프로세스의 생명주기 중 최대 한번의 페이지 폴트를 발생시킬 수 있다. 한편 힙 할당의 이와 같은 특징은 외부 단편화(external fragmentation) 현상을 유발시키는 단점이 있다.

반면, 익명 mmap 영역의 페이지들은 할당된 객체가 free()될 때 물리 메모리로의 사상이 해제된다. 따라서 새롭게 할당된 객체의 각 페이지에 최초로 접근할 때마다 페이지 폴트가 한 번씩 발생한다. 한편 사상이 해제된 물리 메모리는 커널에 즉시 반납되므로 외부 단편화 현상을 피할 수 있다.

스마트폰 환경에서 웹 브라우저가 요청하는 객체들은 대부분 익명 mmap 영역에 할당되도록 설계된다. 왜냐하면 메모리 제약이 심한 스마트폰 환경에서 단편화 현상은 종종 메모리 고갈을 유발하여 응용 강제 종료와 같이 매우 심각한 문제를 초래하기

때문이다. 그러나 이런 선택은 결과적으로 과도한 페이지 폴트를 유발하여 느린 응답시간의 주요 원인으로 작용한다.

이 논문은 익명 페이지가 페이지 폴트 없이 접근될 수 있도록 지원하는 지연된 사상 해제 기법을 제안한다. 이 기법은 익명 mmap 영역이 free()될 때 물리 메모리로의 사상을 바로 해제시키지 않고 임시로 저장해 두었다가, 향후 malloc()이 요청될 때 미리 사상된 페이지들을 제공한다. 이 페이지들은 물리 메모리가 소진되었을 때 커널의 동적 메모리 할당자에 의해 회수된다.

우리는 제안된 기법을 안드로이드 스마트폰인 갤럭시 넥서스 상에 구현하고, 실험을 통해 웹 브라우저의 성능 향상을 측정하였다. 실험 결과 제안된 기법은 쓸기 동작 중에 발생하는 페이지 폴트 횟수를 46% 감소 시키고, 응답시간을 6% 단축시켰다.

2. 대상 시스템 모델과 문제 정의

이 논문에서 제안된 기법의 대상 시스템 모델은 그림 1에 나타난 것과 같다. 대상 시스템은 응용, Bionic LibC, 그리고 리눅스 커널의 세 계층으로 구성된다. 웹 브라우저는 응용 계층에서 수행되며, 동적 메모리 할당을 위해 LibC가 제공하는 malloc()과 free() API가 호출된다. malloc()은 할당 크기가 시스템 환경 변수인 MAP_THRESHOLD 이하인 경우 힙 영역에, 초과인 경우 익명 mmap 영역에 메모리를 할당한다. 익명 mmap 영역의 할당과 해제는 각각 리눅스 커널이 제공하는 mmap()과 munmap() 시스템콜에 의해 수행된다.

위와 같은 시스템에서 우리는 웹 브라우저가 페이지 폴트 없이 익명 페이지에 접근할 수 있게 하는 커널 기법을 개발하고자 한다. 이때 다음 두 가지 제약사항이 함께 지켜져야 한다. 첫째, 웹 브라우저에 의해 free()된 익명 페이지는 커널이 필요할 때 언제든지 회수될 수 있도록 허용함으로써 메모리 오버헤드를 피해야 한다. 둘째, 기존 응용과 라이브러리와의 호환성을 유지하기 위해 LibC API와 시스템콜 인터페이스에 변경이 없어야 한다.

3. 지연된 사상 해제 기법

이 장에서는 앞 장에서 정의된 문제를 해결하기 위한 지연된 사상 해제 기법에 대해 자세히 설명한다. 제안된 기법은 익명 mmap 페이지의 할당 해제를 위해 munmap() 시스템콜이 호출되면, 해당 페이지를 미리 사상된 페이지 저장소(pool of premapped pages)에 보관한다. 이 페이지들은 사용자 수준 접근권한이 비활성화되어 응용이 더 이상 접근할 수 없으나, 물리 메모리로의 사상은 여전히 남겨진 채로 지속된다. 이후 mmap() 시스템콜을 통해 익명 페이지의 할당이 요청되면 이 저장소에 있는 페이지 중 하나를 반환한다.

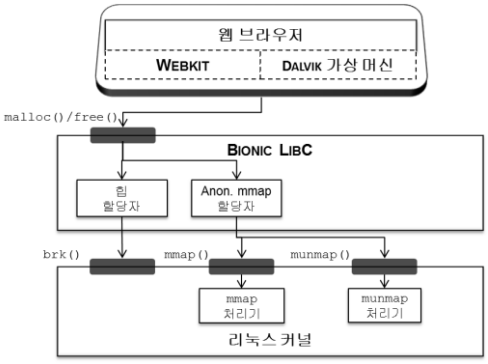

그림 1. 대상 시스템 모델.

이때 이 페이지의 사용자 수준 접근권한이 다시 활성화되며, 물리 메모리로의 사상이 이미 존재하므로 페이지 폴트 없이 접근 가능하다. 미리 사상된 페이지 저장소 내의 페이지들은 자유 메모리가 고갈되면 커널 동적 메모리 할당자에 의해 회수된다.

그림 2는 제안된 기법의 구성요소, 제어와 데이터 흐름을 보인다. 먼저 미리 사상된 페이지 저장소가 각 프로세스마다 하나씩 추가된다. 그리고 mmap() 처리기, munmap() 처리기, 커널 동적 메모리 할당자가 제안된 기법을 위해 확장된다.

mmap() 처리기는 익명 페이지 할당 시 이미 사상된 페이지들을 반환하도록 변경된다. mmap() 처리기는 익명 페이지를 할당할 때 우선 미리 사상된 페이지 저장소에 요청된 크기를 만족하는 연속적인 페이지들이 있는지 판단한다. 만약 있다면 mmap() 처리기는 그 페이지들을 반환한다. 그렇지 않다면 새로 연속적인 페이지들을 할당하고 이들을 미리 물리 메모리에 사상하여 반환한다. 미리 사상될 물리 메모리는 자유 메모리 저장소로부터 공급된다. 이를 통해, mmap() 처리기로 할당된 익명 페이지들에서 페이지 폴트가 발생하지 않도록 한다.

munmap() 처리기는 익명 페이지 해제 시 물리 메모리로의 사상이 즉시 제거되지 않도록 변경된다. munmap() 처리기는 먼저 사상 해제를 요청 받은 페이지가 익명 페이지인지 확인하고, 이를 미리 사상된 페이지 저장소에 임시로 저장한다. 이 저장된

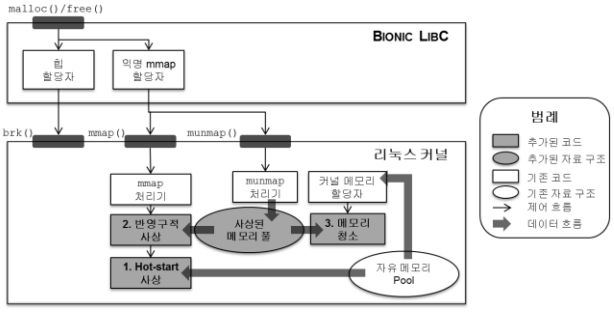

그림 2. 지연된 사상 해제 기법을 위한 커널 구성요소와 제어 및 데이터 흐름.

페이지들은 추후 mmap() 처리기 또는 커널 메모리 할당자에 의해 사용되거나 해제된다.

커널 메모리 할당자는 자유 물리 메모리가 고갈되었을 때 미리 사상된 저장소의 페이지들을 우선 회수하여 자유 물리 메모리를 확보하도록 확장된다. 기존의 커널 메모리 할당자는 물리 메모리 할당 도중 자유 메모리 소진으로 인해 실패한 경우, 메모리 압축(compaction), 메모리 반환(reclamation) 그리고 OOM(out of memory) killer 순으로 자유 물리 메모리를 확보를 위한 동작을 수행한다. 이 기법들이 수행되는 도중에 충분한 자유 물리 메모리가 확보하면 다음 순서의 기법은 수행되지 않는다. 확장된 커널 메모리 할당자는 메모리 반환 기법 수행 도중에 미리 사상된 페이지 저장소의 페이지들을 반환한다. 따라서 응용 강제 종료와 같은 심각한 문제는 적어도 미리 사상된 페이지들로 인해 발생하지 않음이 보장된다.

4. 실험 평가

이 장은 제안된 기법의 실험적 검증 결과를 기술한다. 우리는 제안된 기법을 안드로이드 4.1과 리눅스 커널 3.0.31이 탑재된 갤럭시 넥서스 스마트폰 상에 구현하였다. 그리고 웹 브라우저의 쓸기 제스처 수행 시 발생하는 페이지 폴트 횟수와 응답시간을 측정하였다. 구체적으로, 웹 브라우저를 통해 모바일 포털 사이트[1]의 로딩을 마친 후 메인 뉴스 기사 프레임 상에서 좌우로 쓸기 제스처를 수행하였다. 이때 안드로이드 입력 이벤트 서버에 제스처 동작이 도달한 시간부터 렌더링이 완료되는 시간까지를 응답시간으로 정의하고 이를 측정하였다.

그림 3은 실험 결과를 나타낸다. 기존 시스템 대비 페이지 폴트 발생 횟수가 약 46% 감소되었음을 알 수 있다. 결과적으로 쓸기 제스처의 응답 시간은 약 6% 감소되었다.

5. 관련 연구

페이지 폴트 횟수 감소를 위한 기존 접근 방법들은 하드웨어, 응용, 커널 레벨의 세 가지로 분류할 수 있다. 하드웨어 레벨의 접근 방법에는 페이지

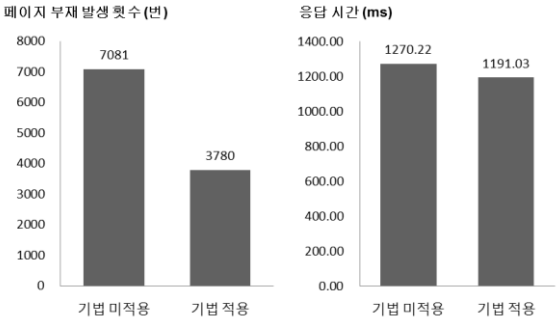

그림 3. 제안된 기법에 의한 웹 브라우저 쓸기 동작의 페이지 폴트 감소와 응답시간 감축 결과.

크기를 증가시키는 Huge TLB 기법이 있다. 이를 통해 적은 수의 페이지 폴트로 넓은 메모리 영역을 사상할 수 있다. 이를 위해 x86 아키텍처는 최대 4MB의 페이지 크기를 지원한다. 하지만 페이지 크기가 클수록 내부 메모리 단편화 문제가 심해져 메모리 제약이 덜한 서버 환경에서 주로 사용된다 [5].

응용 레벨의 대표적인 접근 방법은 MAP_THRESHOLD 환경 변수를 증가시켜 주로 힙 영역에서 동적 할당을 수행하는 것이다. [2]. LibC는 이를 지원하는 mallopt() API를 제공한다. 그러나 이는 결국 외부 메모리 단편화를 심화시키는 단점을 가진다.

커널 레벨의 접근 방법에는 페이지 폴트가 발생할 페이지들을 예측하여 미리 사상하는 기법들이 존재한다. 그러나 미래에 접근될 페이지를 예측하기 위해 메모리 참조 기록[3]과 과거의 메모리 접근 패턴[4]을 분석하는 과정이 필요하기 때문에 무시하지 못할 만한 런타임 오버헤드를 야기할 수 있다.

6. 결론

이 논문은 안드로이드 스마트폰의 웹 브라우저 응답성 향상을 위한 지연 사상 해제 기법을 제안하였다. 제안된 기법에서 커널은 임시적 사상을 갖는 페이지들의 물리 메모리로의 사상을 지연 해제한다. 이를 통해 웹 브라우저의 쓸기 제스처의 페이지 폴트 횟수를 46%, 응답시간을 6% 감소할 수 있었다.

Acknowledgement

본 연구는 지식경제부 및 한국산업기술평가관리원의 산업융합원천기술개발사업(No. 10043227, 장거리 스캔을 탑재하고 IP68 기준을 따르는 AIDC system과 10개국 호환가능한 EMV를 탑재한 페이먼트 system 개발)과 LG 전자(No. 0421-20130020, LG webOS의 Cross-Layer 최적화)의 지원을 받아 수행되었음.

7. 참고 문헌

[1] Mobile Naver, http://m.naver.com.

[2] P. Ezolt, "A study in malloc: a case of excessive minor faults," in Proc. of the 5th USENIX Linux Showcase & Conference, pp. 17, 2001.

[3] S. Cho and Y. Cho, "Page Fault Behavior and Two Prepaging Schemes," In Proc. of the 15th International Phoenix Conference on Computers and Communications, pp.15-21, 1996.

[4] H. Ahn, S. Cho, H. Na, H. Han. “Access pattern based stream buffer management scheme for portable media players,” IEEE Transactions on Consumer Electronics, pp.1522-1529, 2009.

[5] HugeTLB - Large Page Support in the Linux Kernel, http://linuxgazette.net/155/krishnakumar.html.