안드로이드 응용 프로그램의 우선순위를 고려한 메모리 중복 제거 스케줄러

홍경환O, 황인중, 신동군
성균관대학교 전자전기컴퓨터공학과
redc7328@skku.edu, sinban@skku.edu, dongkun@skku.edu

Priority-aware Memory Deduplication Scheduler for Android Applications

Gyeonghwan HongO, Injung Hwang, Dongkun Shin
Department of Electronical and Computer Engineering, Sungkyunkwan University

요 약

모바일 OS에서는 여유 메모리 공간을 확보하기 위해, 각 프로세스의 중복된 메모리 페이지를 탐색하고 공유 메모리로 전환하는 KSM(Kernel Samepage Merging)을 활용한다. 그러나 KSM은 응용 프로그램의 우선순위에 관계 없이 모든 응용 프로그램의 메모리 영역을 탐색하기 때문에, 메모리 중복 제거 가능성이 적은 프로세스의 메모리를 탐색하는 탐색 낭비 문제가 발생한다. 본 논문에서 제안하는 KSM 스케줄러는 모바일 OS에서 설정한 우선순위와 KSM의 메모리 중복 제거 속도를 고려하여, 탐색 프로세스 목록을 스케줄링한다. KSM 스케줄러는 기존 KSM에 비해 메모리 중복 제거량에 거의 변화가 없으면서 메모리 탐색 시간을 최대 52% 줄였다.

1. 서 론

스마트폰, 스마트 워치 등 모바일 장치에서는 다양한 응용 프로그램이 동작하고, 각 응용 프로그램들은 한정된 자원 내에서 빠른 반응 시간을 요구한다. 특히, 모바일 OS에서는 응용 프로그램 로딩 시간을 단축하기 위해 메모리 상에 응용 프로그램을 캐싱한다. 그러나, 모바일 장치의 메모리 자원이 한정되어 있기 때문에, Android 같은 모바일 OS에서는 메모리에 캐싱되는 응용 프로그램을 우선순위 기반으로 관리하고 있다.

Android에서는 모든 응용 프로그램들은 사용자 상호작용이 끝난 후, 백그라운드 동작도 하지 않게 되면 메모리에 캐싱되고 우선순위도 낮아진다. 여유 메모리가 부족한 경우에는 캐싱된 응용 프로그램 중 가장 우선순위가 낮은 응용 프로그램을 강제 종료한다.

우선순위를 기반으로 응용 프로그램을 강제 종료하더라도, 메모리 용량의 한계에 의해 캐싱되는 응용 프로그램의 수가 제한된다. 따라서, 한정된 메모리에 더 많은 응용 프로그램을 캐싱하기 위해, 각 프로세스의 메모리 영역을 지속적으로 탐색하여 중복된 물리 메모리 페이지를 제거하는 KSM(Kernel Samepage Merging)[1]을 적용하는 연구가 진행된 바가 있다[2][3].

그러나, KSM은 모바일 응용 프로그램의 우선순위를 반영하지 않고 모든 프로세스의 메모리 영역을 주기적으로 탐색한다. 특히, 우선순위가 낮은 프로세스들은 메모리 영역에 변화가 거의 없기 때문에, 중복된 메모리 영역에도 변화가 거의 없다. 우선순위가 높은 프로세스에 비해 우선순위가 낮은 프로세스에서 기대할 수 있는 메모리 중복 제거량이 매우 낮기 때문에, 이들을 골고루 탐색하는 것은 메모리 중복 효율을 떨어뜨린다.

따라서, 본 논문에서는 Android에서 설정한 우선순위에 따라 메모리 중복 제거의 대상이 되는 탐색 프로세스 목록을 조절하는 KSM 스케줄러를 제안한다. 본 논문에서는 Android 기반 스마트폰인 Nexus 5 상에서 KSM 스케줄러의 효과를 실험을 통해 증명하였다. KSM 스케줄러는 기존 KSM에 비해 메모리 중복 제거가 많이 일어날 것으로 예상되는 응용 프로그램의 메모리 영역 탐색을 먼저 수행하고, 메모리 중복 제거가 거의 일어나지 않을 것으로 예상되는 응용 프로그램의 메모리 영역 탐색을 수행하지 않음으로써 메모리 중복 제거량은 2~8% 내외로 줄어들면서 메모리 탐색 시간은 9~52% 줄였다.

2. 배 경

2.1. KSM (Kernel Samepage Merging)

메모리 중복 제거는 내용이 같은 물리 메모리 페이지들 중 한 페이지만 남기고 모두 할당 해제한 뒤, 한 페이지를 공유하여 사용하도록 하는 기술이다. 리눅스에서는 메모리 중복 제거하는 커널 모듈로 KSM(Kernel Samepage Merging)을 사용한다.

KSM은 모든 프로세스의 MADV_MERGEABLE 플래그가 설정된 메모리 영역을 페이지 단위로 탐색한다. 각 페이지의 다른 가상 페이지와 내용이 같은 중복 페이지가 발견되면, 두 가상 페이지 중 하나가 가리키는 물리 페이지를 할당 해제하고, 두 가상 페이지가 한 물리 페이지를 가리키도록 한다. 이를 통해

이 논문은 2016년 정부(미래창조과학부)의 재원으로 (재)스마트IT융합시스템 연구단(글로벌프론티어사업)의 지원을 받아 수행된 연구임 ((재)스마트IT융합시스템 연구단-2011-0031863)

메모리 중복을 제거하고, 여유 메모리를 확보한다.

KSM은 커널 데몬으로 작동하며, 한 스캔 주기 동안 모든 응용 프로그램의 메모리 영역을 탐색한다. KSM은 일정량의 가상 메모리 페이지를 탐색하여 다른 가상 메모리 페이지와의 중복 여부를 확인하고, 일정 시간 동안 슬립 상태로 전환하기를 반복한다. 중복 가상 메모리 페이지가 확인되면 중복 제거 과정을 거친다. KSM의 1회 탐색량과 슬립 시간은 pages_to_scan과 sleep_millisecs이라는 매개 변수로 설정할 수 있다.

탐색할 메모리 영역이 동일할 경우에는, sleep_millisecs를 줄이거나 pages_to_scan을 늘리면 적극적으로 메모리 중복 제거 속도를 높일 수 있으나, CPU 유휴 시간이 줄어들어서 전력 소모가 증가하게 된다. 따라서, KSM의 메모리 중복 제거 속도를 높이면서 전력 소모도 줄이려면, 추가 중복 메모리 페이지를 발견할 수 있는 메모리 영역만을 탐색 대상으로 설정해야 한다.

2.2. 우선순위 기반 모바일 응용 프로그램 관리

Android에서는 모든 프로세스들은 중요성과 상태에 따라 각자 다른 우선순위 값(oom_adj)를 부여받는다. oom_adj가 낮을 수록 높은 우선순위임을 의미한다. Service Manager, Zygote 등 시스템 동작에 반드시 필요한 프로세스들은 -16이라는 가장 높은 우선순위를 부여받고, System UI와 Phone과 같이 사용자 경험에 항상 영향을 미치는 프로세스들은 -12라는 다음으로 높은 우선순위를 부여받는다.

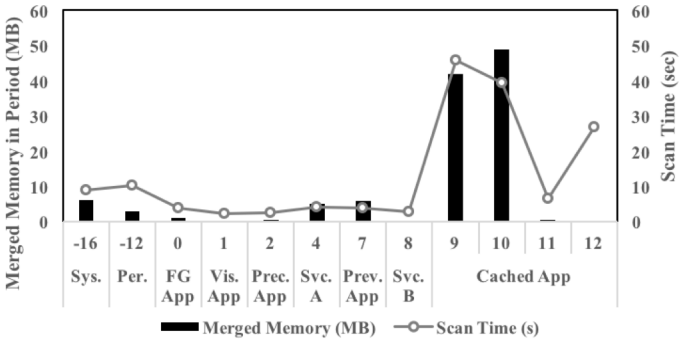
응용 프로그램들은 사용자 상호작용 여부, UI 표시 여부, 백그라운드 동작 여부 등에 따라 우선순위를 부여받는다. 현재 사용자와 상호작용하고 있는 응용 프로그램은 응용 프로그램 중 가장 높은 우선순위인 0을 부여받는다. UI도 표시하지 않고 백그라운드 동작도 하지 않아 메모리에 캐싱되는 응용 프로그램들은 가장 낮은 우선순위인 9~13을 부여받는다. 캐싱된 응용 프로그램들은 캐싱된 지 오래된 프로세스일 수록 낮은 우선순위를 받는다. Android 모바일 장치의 메모리 공간이 부족해지면, LMK(Low Memory Killer)는 우선순위가 가장 낮은 프로세스부터 강제 종료한다.

3. 기존 KSM의 메모리 탐색 낭비 문제

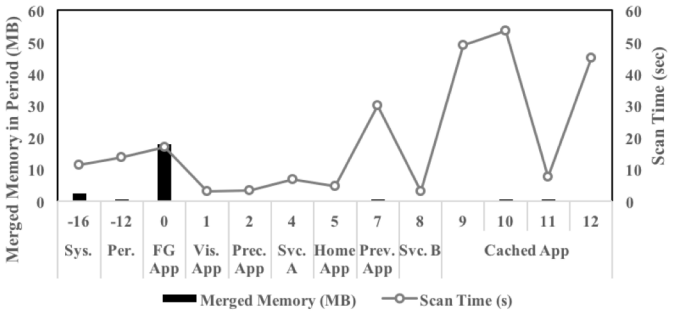
Android에서는 응용 프로그램의 중요성과 상태에 따라 우선순위를 부여하고, 우선순위를 기반으로 응용 프로그램을 관리한다. 그러나, KSM은 우선순위에 관계 없이 모든 응용 프로그램을 탐색하기 때문에, 메모리 중복 제거가 거의 일어나지 않는 응용 프로그램에 대해서도 필요 없는 탐색을 수행한다.

캐싱된 응용 프로그램들은 사용자 상호작용이 거의 없고 백그라운드 처리를 수행하지 않기 때문에, 메모리 내용에 변화가 거의 없다. 따라서, 우선순위가 낮은 응용 프로그램의 메모리 영역을 탐색하여도 새로운 중복된 메모리 영역이 발견되기 어렵다.

스마트폰 상에서 실제 응용 프로그램을 수행하여, 이러한 메모리 탐색 낭비 문제가 발생함을 확인하였다. Android 스마트폰 Nexus 5 상에서 28개의 응용 프로그램을 순차적으로 로드하여 메모리에 캐싱한 후, Minecraft PE라는 게임 응용



(a) 28개의 응용 프로그램을 순차적으로 실행한 직후



(b) 28개의 응용 프로그램 로드 후, 2분 간 Minecraft PE 실행

그림 1. 기존 KSM 적용하여 실제 모바일 응용 프로그램 실행 시, 프로세스 별 메모리 중복 제거량과 탐색 시간 비교

프로그램을 2분 간 실행하여 사용자와 상호작용하였다. 응용 프로그램을 순차적으로 로드한 직후의 한 KSM 스캔 주기 동안의 프로세스 종류 별 메모리 중복 제거량과 스캔 시간을 그림 1의 (a)와 같이, Minecraft PE을 실행하는 도중의 한 KSM 스캔 주기 동안은 그림 1의 (b)와 같이 조사하였다.

그림 1의 (a)에서는 포그라운드 응용 프로그램(FG App)인 런처에서 사용자와 상호작용을 전혀 수행하지 않기 때문에, 약 1MB의 중복 메모리 영역을 제거한다. 그림 1의 (b)의 포그라운드 응용 프로그램인 Minecraft PE 메모리 영역에 지속적인 변화가 있기 때문에 18MB의 중복 메모리 영역을 제거하였다.

캐싱된 응용 프로그램(Prev. App + Cached App)의 경우, 시간이 지남에 따라 새로운 중복 메모리 영역을 발견하지 못하는 현상이 관측된다. 그림 1의 (a)에서는 최근에 로드했던 응용 프로그램에 대한 메모리 중복 제거가 완전히 끝나지 않은 상태이기 때문에, oom_adj가 9~12인 응용 프로그램에서 총 90.8MB의 메모리 중복을 제거하였다. 반면, 그림 1의 (b)에서는 1.7MB에 불과하다. 캐싱된 응용 프로그램의 메모리 중복 제거량이 포그라운드 응용 프로그램과 시스템 프로세스에 비해 현저하게 적은데도 불구하고, KSM 스캔 주기의 약 74.7%에 해당하는 185초가 캐시된 응용 프로그램의 메모리 탐색에 사용된다.

위의 실험 결과를 비추어보면, 포그라운드 응용 프로그램은 사용자 상호작용의 여부에 따라 메모리 변화가 일어나기 때문에, 메모리 중복 제거량이 매번 달라지게 된다. 반면, 캐싱된 응용 프로그램의 경우, 사용자 상호작용이 끝난 직후에는 변화된

메모리 내용을 반영해야 하기 때문에 메모리 중복 제거가 많이 일어난다. 메모리에 캐싱된 지 오래되어 우선순위가 낮아지게 되면, 메모리 내용에 변화가 거의 없어서 메모리 중복 제거가 거의 일어나지 않는다. 따라서, 우선순위가 높은 프로세스에 대해 먼저 탐색을 수행하여 지속적으로 변화하는 메모리 내용을 빠르게 반영하고, 우선순위가 낮은 프로세스 중에서는 메모리 중복 제거량의 변화에 따라 탐색 대상에서 제외하도록 해야 한다.

4. KSM 스케줄러

본 논문에서는 모바일 응용 프로그램의 우선순위를 KSM 탐색 동작에 적용하는 KSM 스케줄러를 제안한다. KSM은 스캔 주기가 시작할 때 KSM 스케줄러를 호출하여 해당 스캔 주기에 탐색할 프로세스의 목록을 결정한다. KSM은 KSM 스케줄러에 의해 결정된 탐색 프로세스 목록을 바탕으로 메모리 탐색 및 중복 제거 동작을 수행한다.

먼저, KSM 스케줄러는 탐색 프로세스 목록에 포함될 프로세스를 결정한다. 탐색 프로세스 목록에는 기본적으로 MADV_MERGEABLE 플래그가 설정된 페이지를 포함한 프로세스들로 구성한다. 캐싱된 응용 프로그램 중에서 메모리 중복 제거량이 줄어들고 있으면서 512KB 이하로 떨어진 프로세스를 앞으로 메모리 중복 제거가 거의 일어나지 않을 것이라고 추측하고, 이를 탐색 프로세스 목록에서 배제한다. 또한, 배제된 프로세스 중에서는 탐색에서 배제된 기간 동안 oom_adj가 낮아진 적이 있는 경우, 프로세스의 우선순위가 상승해서 메모리 영역 내에 변화가 있었을 것으로 추측하여 탐색 프로세스 목록에 삽입한다.

탐색할 프로세스의 순서는 oom_adj에 대해 오름차순으로 정렬하여, 우선순위가 높은 프로세스를 먼저 탐색하도록 한다. 먼저 탐색함으로써, 변화한 메모리 영역 중 중복 메모리 영역을 빠르게 반영할 수 있도록 한다.

5. 실 험

본 논문에서는 기존 KSM 대비 KSM 스케줄러의 효과를 증명하기 위해, 2GB의 DRAM을 탑재한 실제 Android 스마트폰인 Nexus 5 상에서 실험하였다. 해당 스마트폰 상에서는 3종의 Game, 3종의 Web Browser를 포함한 총 28개의 응용 프로그램을 차례로 실행하였다. 응용 프로그램 실행이 끝나고 나면, KSM이 중복 메모리 영역을 충분히 제거할 수 있도록 워크로드를 수행하지 않는 유휴 기간을 두었다. 본 실험에서는 3개의 KSM 탐색 주기 동안 유휴 기간을 두었다.

KSM 스케줄러를 적용하면, 그림 2와 같이 기존 KSM에 비해 전반적으로 메모리 중복 제거량의 감소폭은 2~8% 내외로 적은 반면, KSM 스캔 시간은 9~52% 만큼 줄어들게 된다. 워크로드가 끝나고 3번째 유휴 기간이 지난 후에는 그림 3과 같이 메모리 중복 제거량과 탐색 시간이 프로세스 별로 분포하게 된다. 기존 KSM이 동작할 때의 분포인 그림 1의 (a)에 비해, 캐싱된 응용 프로그램의 탐색 시간이 크게 줄어들었다.

6. 결 론

본 논문에서는 Android에서 KSM을 이용하여 메모리 중복 제거를 수행할 때, 응용 프로그램의 우선순위를 고려하지 않아서 발생하는 탐색 낭비 문제를 지적하였다. 따라서, 본 논문에서는 응용 프로그램의 우선순위를 고려하여 탐색할 프로세스를 선정하는 KSM 스케줄러를 제안하였다. 실제 장치에서의 실험을 통해, KSM 스케줄러가 기존 KSM에 비해 메모리 중복 제거량에 거의 변화가 없으면서 메모리 탐색 시간을 최대 52% 줄임을 보였다. 본 논문에서는 메모리 탐색 시간 단축까지 다루었으나, sleep_millisecs나 pages_to_scan과 같은 매개 변수를 설정하면 모바일 장치의 전력 소모를 줄일 수 있다. 차후 연구에서는 KSM 스케줄러의 탐색 프로세스 목록 결정 알고리즘을 개선하여, 모바일 장치의 전력 소모에 미치는 영향을 다룰 예정이다. 또한, Tizen 등 다른 모바일 OS의 응용 프로그램 관리 모듈과 연계하여 응용 폭을 넓힐 예정이다.

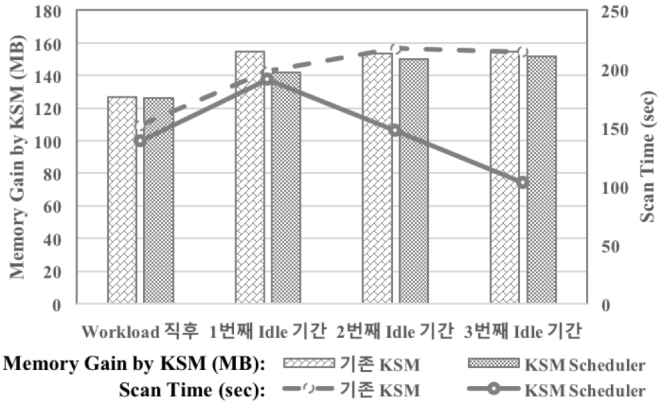


그림 2. 기존 KSM 대비 KSM Scheduler 적용 시의 메모리 중복 제거량 및 탐색 시간 비교

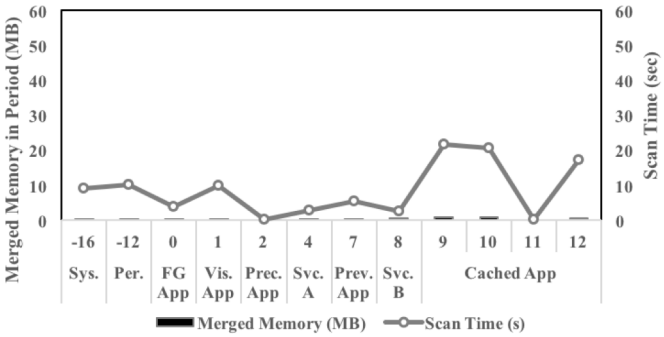


그림 3. KSM Scheduler 적용 시 3번째 유휴 기간이 지난 후의 프로세스 별 메모리 중복 제거량과 탐색 시간 비교

참고문헌

[1] A. Arcangeli, I. Eidus, and C. Wright. "Increasing Memory Density by Using KSM." Proceedings of the Linux Symposium. 2009.

[2] S. Kim, J. Jeong, and J. Lee. "Selective Memory Deduplication for Cost Efficiency in Mobile Smart Devices." IEEE Transactions on Consumer Electronics (TCE). Vol.60, No. 2, pp. 276-284, 2014.

[3] J. Ahn, and D. Shin. "Optimizing Power Consumption of Memory Deduplication Scheme." The 18th International Symposium on Consumer Electronics (ISCE 2014). IEEE, 2014.