졸업논문 진행상황

1주차 3.14~3.20

2013313710 정수진

**논문 제출 전까지의 시간계획**

월 - 일

1주차: 3.13 - 3.19

실험환경구축, 자료공부&코드분석

2주차: 3.20 - 3.26

졸업논문 계획서 작성, 알고리즘 짜기(+차트 그리기)

3주차: 3.27 - 4.2

코드수정 1차, 코드수정 2차, 안드로이드 실험

4주차: 4.3 - 4.9

코드수정 완료(했길 바라고 있음), 실험완료

5주차: 4.10 - 4.14

논문작성

**1주차 진행**

1. 개인 공부

* NAND 플래시메모리에 대한 이해
* 파일시스템 ext2,3과 ext4의 차이. Ext4의 구조
* Ext4에서의 defragmentation기법

1. 논문전개방향 잡기

* 논문아이디어: 플래시메모리 수명을 고려하여 최소한의 defragmentation만 수행
* 논문전개구상: 특정 크기(이하 기점T) 이상의 fragment를 defragmentation에 포함시키는 것은 defragmentation으로 인한 성능향상에 큰 기여를 하지 못할 뿐더러 플래시 메모리 수명을 단축시킨다. 따라서 기점 T 이상으로는 defragmentation을 하지 않는 기법을 제안한다.

1. 코드수정을 위한 준비

* e4defrag.c 코드 분석
* e4defrag.c를 수정한 후 빌드하여 적용시키기만 하면 되도록 실험환경 세팅

**2주차 진행**

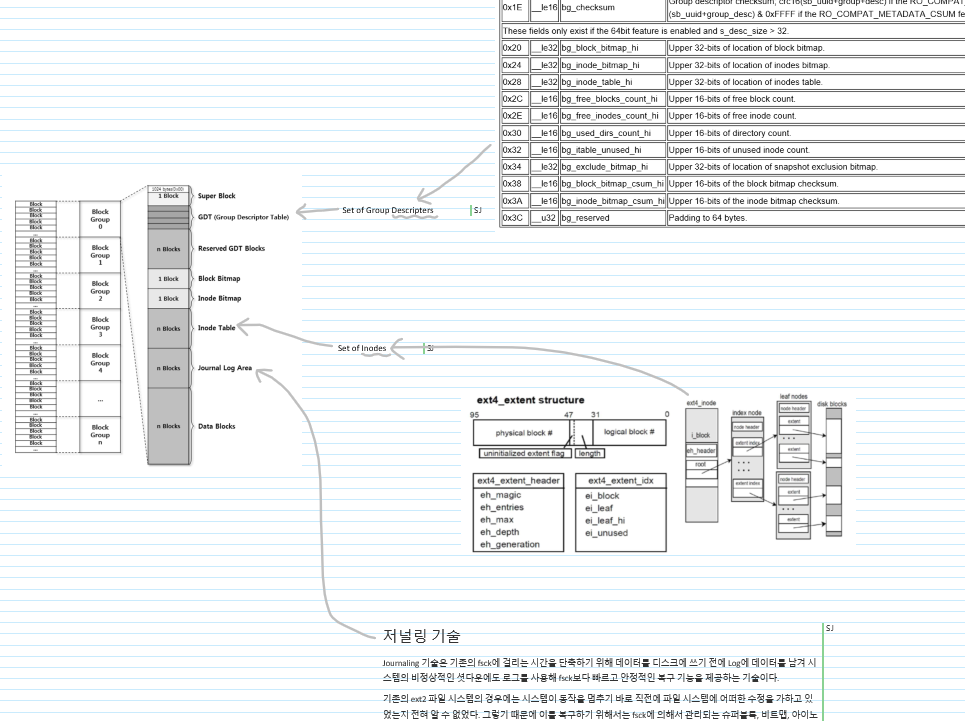
1. 제안서 작성

* 논문 실험에 참고할 다른 논문 분석, 구글링, 필요 정보 추출

“안드로이드 운영체제의 Ext4 파일 시스템에서 삭제 파일 카빙 기법\*”

“ext4 online defragmentation”

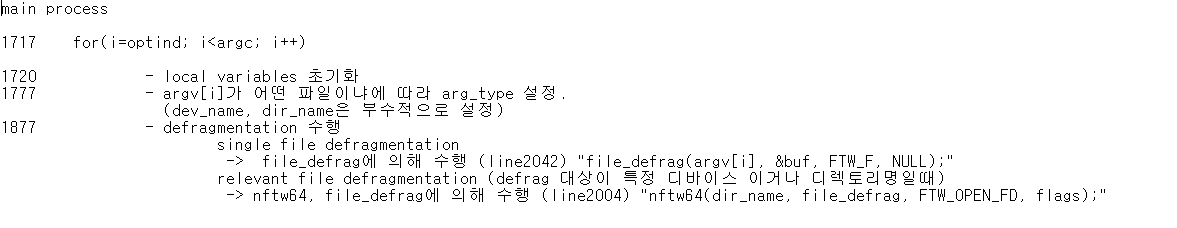
[**https://ext4.wiki.kernel.org/index.php/Ext4\_Disk\_Layout#Blocks**](https://ext4.wiki.kernel.org/index.php/Ext4_Disk_Layout#Blocks)



<ext4 구조>

1. 코드 분석(거의)

* e4defrag main 함수 레이아웃 작성
* e4defrag file\_defrag 는 보는 중



<e4defrag의 main 함수>

3주차 진행

1. 논문 아이디어 구체화

* ‘e4defrag.c’내에서 defragmentation을 수행하기 위해 get\_file\_extents로 physical extents를 가져오는 것을 참고, get\_file\_extents가 크기 k (block count)의 file\_extent만 가져오도록 수정
* 코드 수정 전 피드백이 필요한 부분 발견

(1) file\_defrag 수행 후 해당 파일의 모든 extent가 한 곳에 모여있다고 파일시스템이 인식했었다면 그것 또한 고쳐야 한다

(2) file extents들이 논리주소로 정렬된 후 큰 extent와 작은 extent들이 교차적으로 나타난다면 defrag후 성능향상이 미미할 가능성이 있다.

1. 피드백을 받은 후 코드 & 컴파일 관련 search, 수정방향 작성

* arm cross compile로 안드로이드 목적파일 만들어 내는 방법 알아내기
* 버전A.1 코딩 & 컴파일

defrag하는 도중 defrag에 참여한 모든 extent가 논리주소로 정렬된 상태일때

정렬순서대로 각 extent의 크기 프린트하도록 원본파일 수정

* 버전A.2 코딩 & 컴파일

지정크기 k 이하의 extent만 defrag에 참여하도록 원본파일 수정

1. 진행사항

* E4defrag.c 단독 컴파일은 하지 않기로 하고, make유틸리티 사용시 옵션을 주거나 makefile을 (arm-linux-gnueabi-gcc가 사용되도록) 수정하여 사용하기로 하였다.
* 버전 A.1는 이미 옵션 -cv에 대하여 구현되어 있어서, 옵션 -c만 설정해도 원하는 정보를 출력하도록 수정하였다.
* get\_file\_extents가 arg로 k(blk\_limit)을 받도록 수정&컴파일 완료