운영체제 Project 3 Wiki

정상윤
2016025032

Implementing Double Indirect

• 디자인

명세에 따라 우선 param.h 의 FSSIZE를 20000 으로 수정했습니다.

그 다음 fs.h 에 정의된 매크로값들을 수정했습니다. 일단 Double Indirect가 추가됐기 때문에 NDIRECT를 1 감소시킨 11로 바꿔줬습니다. 그 후 Double Indirect의 크기 값을 추가해줬습니다. Double Indirect는 이중 Indirect이기 때문에 NINDIRECT x NINDIRECT 입니다. 바뀐 NDIRECT 값에 맞춰 dinode 구조체 속 addrs 배열의 크기를 [NDIRECT + 2]로 수정했습니다.

그 후 file.h 에 정의된 inode 구조체의 addrs 도 마찬가지로 [NDIRECT + 2]로 수정해줬습니다.

fs.c 파일 내의 bamp 함수를 수정해줬습니다. Double Indirect는 이미 구현된 Indirect를 이중으로 써준 것이기 때문에 기존 xv6의 Indirect 구현 코드를 참고하여서 구현했습니다. Double indirect에 필요한 dbp (buf) 와 d (uint) 변수를 추가해줬습니다. bn을 128로 나눠서 몫은 level-1 block number, 나머지는 level-2 block number 로 구현했습니다.

itrunc 함수 또한 마찬가지로 기존의 xv6 Indirect 코드를 참고해가며 구현했습니다. 이중 Indirect이기 때문에 for 문 또한 이중 for문이 필요합니다.

• 구현

```
13 #define FSSIZE       20000  // size of file system in blocks
```

param.h 변경점
• FSSIZE 값을 20000 이상으로 수정

```
24 #define NDIRECT 11 // double indirect 추가
25 #define NINDIRECT (BSIZE / sizeof(uint)) // indirect entry number
26 #define NDOUBLEINDIRECT (NINDIRECT * NINDIRECT) // double indirect entry number
27 #define MAXFILE (NDIRECT + NINDIRECT + NDOUBLEINDIRECT)
28
29 // On-disk inode structure
30 struct dinode {
31   short type;           // File type
32   short major;          // Major device number (T_DEV only)
33   short minor;          // Minor device number (T_DEV only)
34   short nlink;          // Number of links to inode in file system
35   uint size;            // Size of file (bytes)
36   uint addrs[NDIRECT+2];   // Data block addresses
37 };
```

fs.h 변경점
• 기존 NDIRECT 값인 12를 11로 수정
• NDOUBLEINDIRECT 값 추가
• MAXFILE은 NDIRECT + NINDIRECT + NDOUBLEINDIRECT
• NDIRECT가 1 감소했음에 따라 dinode의 addrs[NDIRECT+2] 수정

```
12 // in-memory copy of an inode
13 struct inode {
14   uint dev;           // Device number
15   uint inum;          // Inode number
16   int ref;            // Reference count
17   struct sleeplock lock; // protects everything below here
18   int valid;          // inode has been read from disk?
19
20   short type;         // copy of disk inode
21   short major;
22   short minor;
23   short nlink;
24   uint size;
25   uint addrs[NDIRECT+2]; //NDIRECT + INDIRECT(1) + DOUBLEINDIRECT(1)
26 };
```

file.h 변경점

- dinode와 마찬가지로 addrs[NDIRECT+2] 수정

```
373 static uint
374 bmap(struct inode *ip, uint bn)
375 {
376   uint addr, *a, *d; // d: double indirect
377   struct buf *bp, *dbp; // dbp: double block pointer
378
```

```
405   if(bn < NDOUBLEINDIRECT) {
406     // Load double indirect block, allocating if necessary.
407     if((addr = ip->addrs[NDIRECT+1]) == 0)
408         ip->addrs[NDIRECT+1] = addr = balloc(ip->dev);
409
410     bp = bread(ip->dev, addr); // level-1 block pointer
411     a = (uint*)bp->data; // level-1 block entries
412
413     // bn/INDIRECT: 몫 : level-1 entry number,  나머지 : level-2 entry number
414     if((addr = a[bn/NINDIRECT]) == 0){
415         a[bn/NINDIRECT] = addr = balloc(ip->dev);
416         log_write(bp);
417     }
418
419     dbp = bread(ip->dev, addr); // level-2 block pointer
420     d = (uint*)dbp->data;
421
422     bn = bn % NINDIRECT;
423
424     if((addr = d[bn]) == 0){
425         d[bn] = addr = balloc(ip->dev);
426         log_write(dbp);
427     }
428
429     brelse(dbp);
430     brelse(bp);
431
432     return addr;
433
434   }
```

fs.c bmap 함수 변경점

- Double Indirect를 위한 변수 추가 (d, dbp)
- Indirect 보다 큰 bn이 들어왔을 때 Double Indirect 코드 추가
- 전체적인 로직은 기존 Indirect 와 거의 같음.

```
444 static void
445 itrunc(struct inode *ip)
446 {
447   int i, j;
448   struct buf *bp, *dbp;
449   uint *a, *d;
```

```
470   if(ip->addrs[NDIRECT+1]){
471     bp = bread(ip->dev, ip->addrs[NDIRECT+1]);
472     a = (uint*)bp->data;
473
474     for(i=0; i<NINDIRECT; i++){
475         if(a[i]){
476             dbp = bread(ip->dev, a[i]);
477             d = (uint*)dbp->data;
478             for(j=0; j<NINDIRECT; j++){
479                 if(d[j])
480                     bfree(ip->dev, d[j]);
481             }
482             brelse(dbp);
483             bfree(ip->dev, a[i]);
484         }
485     }
486
487     brelse(bp);
488     bfree(ip->dev, ip->addrs[NDIRECT+1]);
489     ip->addrs[NDIRECT+1] = 0;
490
491   }
492
493   ip->size = 0;
494   iupdate(ip);
495 }
```

fs.c itrunc 함수 변경점

- bmap과 마찬가지로 Double Indirect를 위한 변수 추가 (d, dbp)
- 전체적인 로직은 기존 Indirect 와 거의 같음.

• 실행결과

```
xv6...
cpu0: starting 0
sb: size 20000 nblocks 19937 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ file_test
Test 1: Write 8388608 bytes
Test 1 passed

Test 2: Read 8388608 bytes
Test 2 passed

Test 3: repeating test 1 & 2
Loop 1: 1.. 2.. ok
Loop 2: 1.. 2.. ok
Loop 3: 1.. 2.. ok
Loop 4: 1.. 2.. ok
Loop 5: 1.. 2.. ok
Loop 6: 1.. 2.. ok
Loop 7: 1.. 2.. ok
Loop 8: 1.. 2.. ok
Loop 9: 1.. 2.. ok
Loop 10: 1.. 2.. ok
Test 3 passed
All tests passed!!
```

• 트러블슈팅

```
xv6...
cpu0: starting 0
sb: size 20000 nblocks 19937 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
$ file_test
Test 1: Write 8388608 bytes
Test 1 passed

Test 2: Read 8388608 bytes
Test 2 passed

Test 3: repeating test 1 & 2
Loop 1: 1.. main-loop: WARNING: I/O thread spun for 1000 iterations
2.. ok
Loop 2: 1.. 2.. ok
Loop 3: 1.. 2.. ok
Loop 4: 1.. 2.. ok
Loop 5: 1.. 2.. ok
Loop 6: 1.. 2.. ok
Loop 7: 1.. 2.. ok
Loop 8: 1.. 2.. ok
Loop 9: 1.. 2.. ok
Loop 10: 1.. 2.. ok
Test 3 passed
All tests passed!!
```

큰 어려움은 없었습니다. 단 테스트 도중 Warning이 떠서 어딘가 잘못된 것인가 싶어서 추가적으로 테스트를 반복해본 결과 무작위하게 Warning이 뜨고, 결과는 항상 끝까지 잘 나와서 넘어갔습니다.