파일처리 과제: **LRU-K 구현**

김현준 (**2012003954** 컴퓨터전공, 한양대학교)

구현내용:

`PF_BufPageDesc` 및 **`PF_BufferMgr::InitPageDesc`**, **`PF_BufferMgr::GetPage`**, **`PF_BufferMgr::UnpinPage`** 을 패치하여 기존 LRU Page Replacement Algorithm을 LRU-K로 변경했습니다.

LRU-K의 상수 “K”는 **`src/pf_buffermgr.h`** 의 **`K_FOR_LRU_K`** 정의를 수정하여 설정이 가능합니다.

코드 수정사항:

(GitHub에서 코드 변경사항을 확인하기 위해 생성한 Pull Request(https://github.com/yoloseem/edubase/pull/1) 의 commit logs (https://github.com/yoloseem/edubase/pull/1/commits) 또는 files changed (https://github.com/yoloseem/edubase/pull/1/files) 페이지를 통해 확인하실 수도 있습니다.)

`src/pf_buffermgr.h`

7 ■■■■■ src/pf_buffermgr.h — View

```
@@ -15,6 +15,9 @@
 #ifndef PF_BUFFERMGR_H
 #define PF_BUFFERMGR_H

+#include <time.h>
+#include <deque>
+
 #include "pf_internal.h"
 #include "pf_hashtable.h"

@@ -28,6 +31,9 @@
 // next.
 #define INVALID_SLOT  (-1)

+// K_FOR_LRU_K is configurable value for LRU-K algorithm
+#define K_FOR_LRU_K 2
+
 //
 // PF_BufPageDesc - struct containing data about a page in the buffer
 //
@@ -37,7 +43,7 @@ struct PF_BufPageDesc {
     int         prev;       // prev in the linked list of buffer pages
     int         bDirty;     // TRUE if page is dirty
     short int   pinCount;   // pin count
+    std::deque<time_t> hists;    // access time histories
     PageNum     pageNum;    // page number for this page
     int         fd;         // OS file descriptor of this page
 };
```

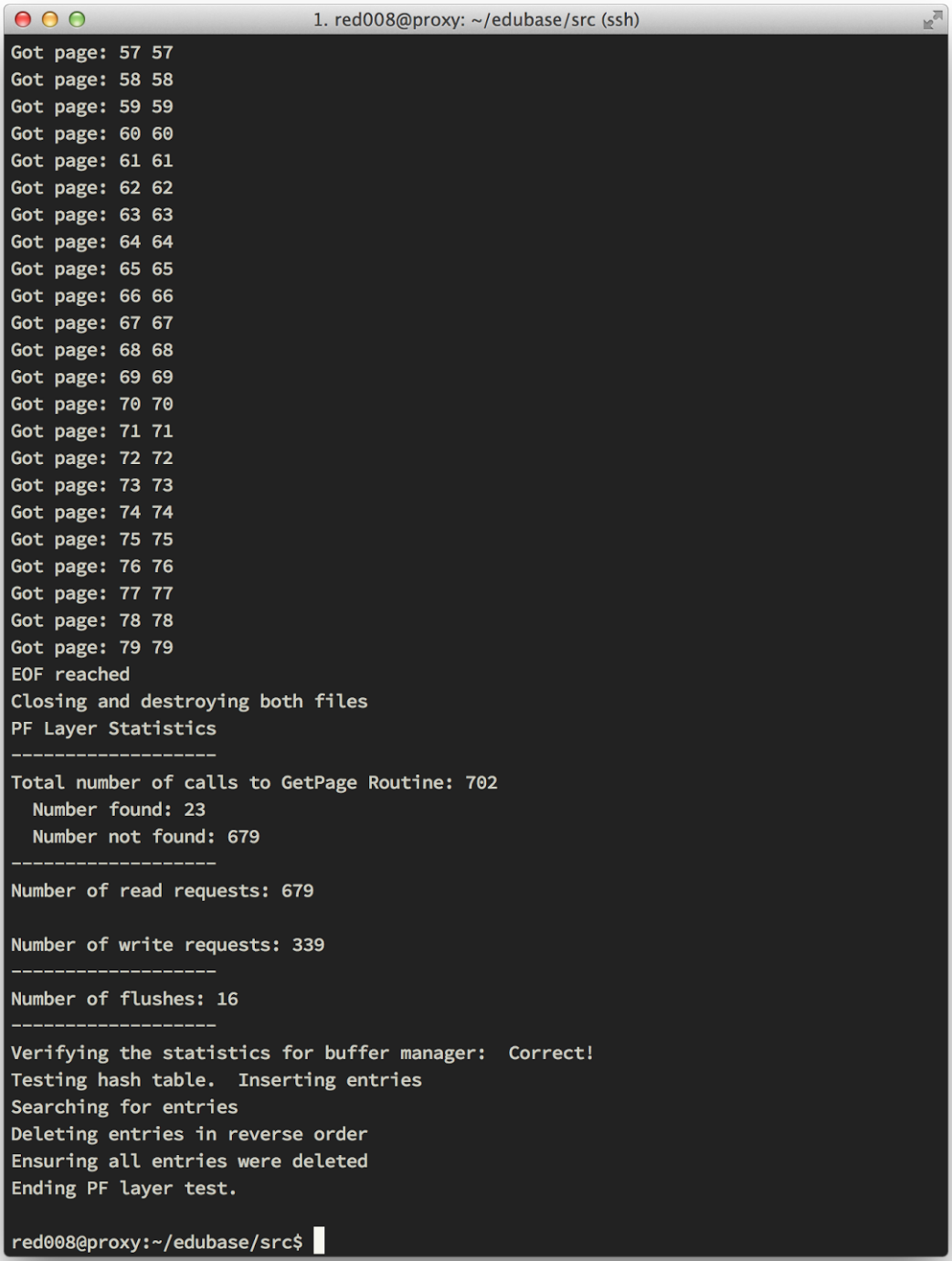
src/pf_buffermgr.cc

24 ■■■■ src/pf_buffermgr.cc View

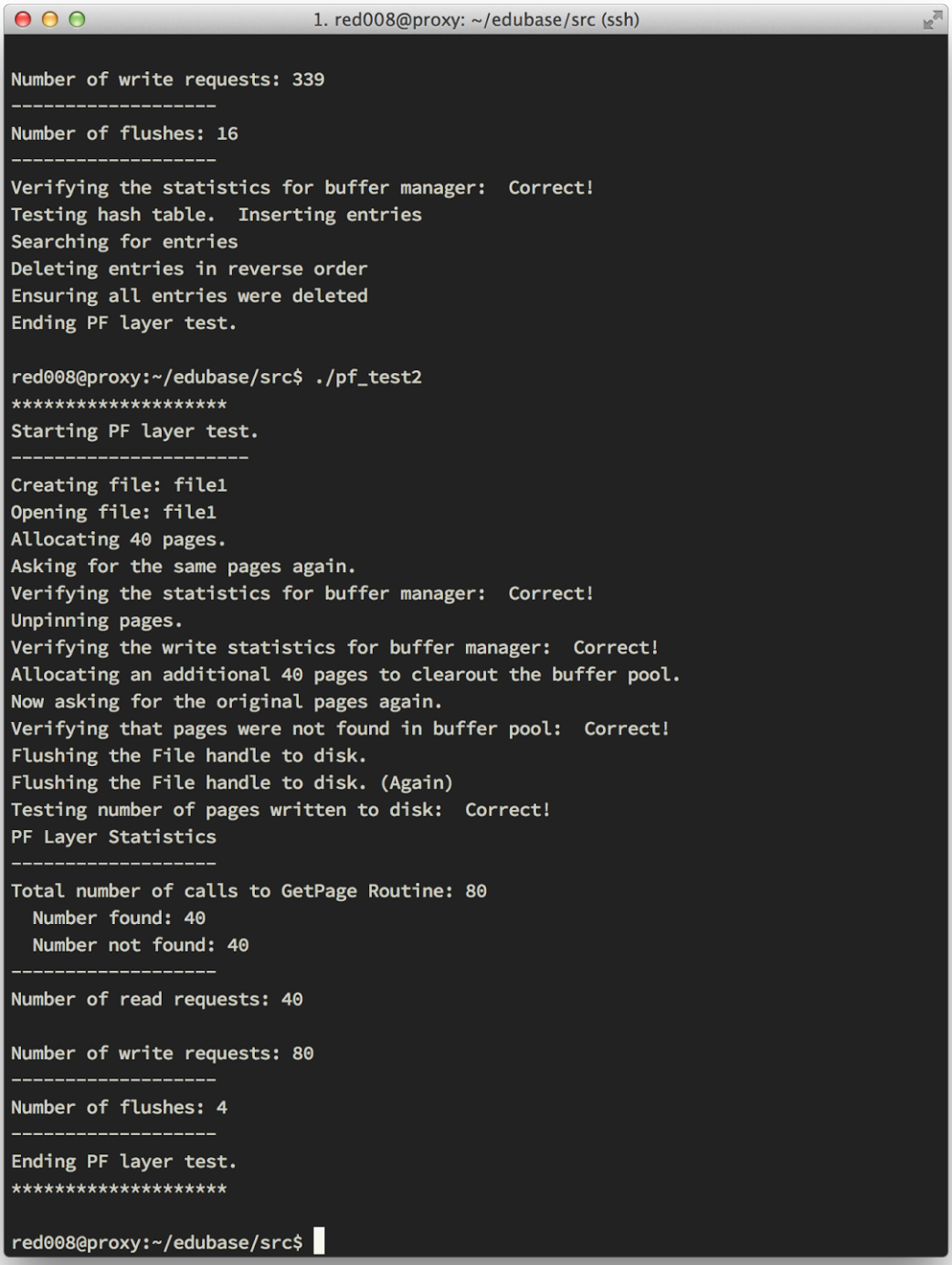
```
@@ -236,6 +236,10 @@ RC PF_BufferMgr::GetPage(int fd, PageNum pageNum, char **ppBuffer,
236 236
237 237         // Page is alredy in memory, just increment pin count
238 238         bufTable[slot].pinCount++;
    239 +       bufTable[slot].hists.push_back(time(NULL));
    240 +       if (bufTable[slot].hists.size() > K_FOR_LRU_K)
    241 +         // Keep only K(in LRU-K) histories
    242 +         bufTable[slot].hists.pop_front();
239 243  #ifdef PF_LOG
240 244         sprintf (psMessage, "Page found in buffer.  %d pin count.\n",
241 245                  bufTable[slot].pinCount);
@@ -385,6 +389,7 @@ RC PF_BufferMgr::UnpinPage(int fd, PageNum pageNum)
385 389  #endif
386 390
387 391     // If unpinning the last pin, make it the most recently used page
    392 +   bufTable[slot].hists.pop_front();
388 393     if (--(bufTable[slot].pinCount) == 0) {
389 394        if ((rc = Unlink(slot)) ||
390 395                 (rc = LinkHead (slot)))
@@ -784,10 +789,19 @@ RC PF_BufferMgr::InternalAlloc(int &slot)
784 789     }
785 790     else {
786 791
787     -       // Choose the least-recently used page that is unpinned
788     -       for (slot = last; slot != INVALID_SLOT; slot = bufTable[slot].prev) {
789     -          if (bufTable[slot].pinCount == 0)
790     -             break;
    792 +      // Choose the least-recently used page that is unpinned
    793 +      int &candidate = slot, accessedTime = -1;
    794 +       for (candidate = last;
    795 +            candidate != INVALID_SLOT;
    796 +            candidate = bufTable[candidate].prev) {
    797 +          if (bufTable[slot].pinCount == 0) {
    798 +             if (accessedTime == -1 ||
    799 +                 bufTable[slot].hists.front() < accessedTime) {
    800 +                slot = candidate;
    801 +                accessedTime = bufTable[candidate].hists.back();
    802 +                break;
    803 +             }
    804 +          }
791 805        }
792 806
793 807        // Return error if all buffers were pinned
@@ -910,6 +924,8 @@ RC PF_BufferMgr::InitPageDesc(int fd, PageNum pageNum, int slot)
910 924     bufTable[slot].pageNum  = pageNum;
911 925     bufTable[slot].bDirty   = FALSE;
912 926     bufTable[slot].pinCount = 1;
    927 +   bufTable[slot].hists = deque<time_t>();
    928 +   bufTable[slot].hists.push_back(time(NULL));
913 929
914 930     // Return ok
915 931     return (0);
```

테스트 결과:

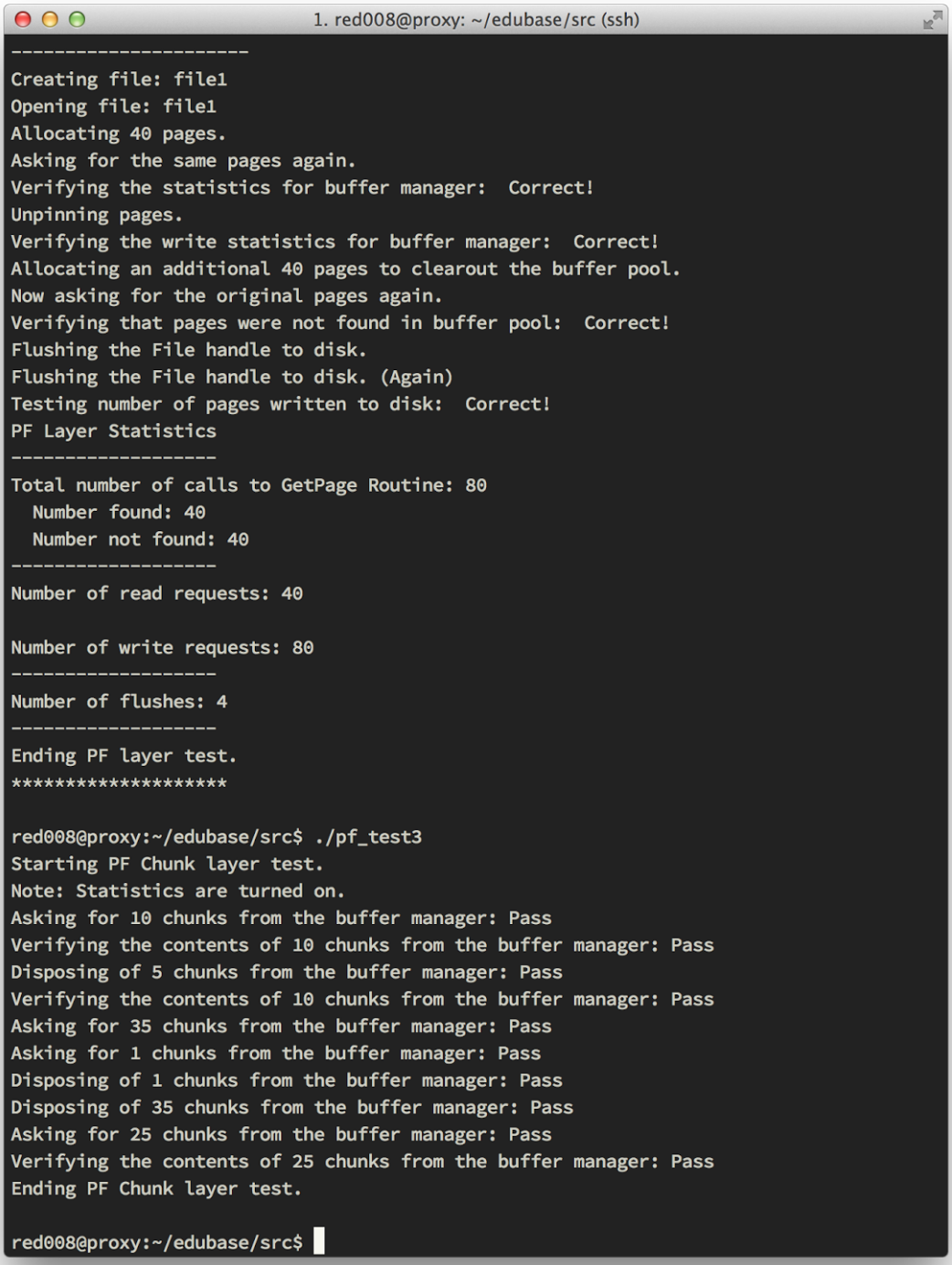
pf_test1



pf_test2


1. red008@proxy: ~/edubase/src (ssh)
Number of write requests: 339
Number of flushes: 16
Verifying the statistics for buffer manager: Correct!
Testing hash table. Inserting entries
Searching for entries
Deleting entries in reverse order
Ensuring all entries were deleted
Ending PF layer test.
red008@proxy:~/edubase/src$ ./pf_test2
Starting PF layer test.
Creating file: file1
Opening file: file1
Allocating 40 pages.
Asking for the same pages again.
Verifying the statistics for buffer manager: Correct!
Unpinning pages.
Verifying the write statistics for buffer manager: Correct!
Allocating an additional 40 pages to clearout the buffer pool.
Now asking for the original pages again.
Verifying that pages were not found in buffer pool: Correct!
Flushing the File handle to disk.
Flushing the File handle to disk. (Again)
Testing number of pages written to disk: Correct!
PF Layer Statistics
Total number of calls to GetPage Routine: 80
  Number found: 40
  Number not found: 40
Number of read requests: 40
Number of write requests: 80
Number of flushes: 4
Ending PF layer test.
red008@proxy:~/edubase/src$

pf_test3


1. red008@proxy: ~/edubase/src (ssh)

Creating file: file1
Opening file: file1
Allocating 40 pages.
Asking for the same pages again.
Verifying the statistics for buffer manager: Correct!
Unpinning pages.
Verifying the write statistics for buffer manager: Correct!
Allocating an additional 40 pages to clearout the buffer pool.
Now asking for the original pages again.
Verifying that pages were not found in buffer pool: Correct!
Flushing the File handle to disk.
Flushing the File handle to disk. (Again)
Testing number of pages written to disk: Correct!
PF Layer Statistics

Total number of calls to GetPage Routine: 80
  Number found: 40
  Number not found: 40

Number of read requests: 40
Number of write requests: 80

Number of flushes: 4

Ending PF layer test.
****
red008@proxy:~/edubase/src$ ./pf_test3
Starting PF Chunk layer test.
Note: Statistics are turned on.
Asking for 10 chunks from the buffer manager: Pass
Verifying the contents of 10 chunks from the buffer manager: Pass
Disposing of 5 chunks from the buffer manager: Pass
Verifying the contents of 10 chunks from the buffer manager: Pass
Asking for 35 chunks from the buffer manager: Pass
Asking for 1 chunks from the buffer manager: Pass
Disposing of 1 chunks from the buffer manager: Pass
Disposing of 35 chunks from the buffer manager: Pass
Asking for 25 chunks from the buffer manager: Pass
Verifying the contents of 25 chunks from the buffer manager: Pass
Ending PF Chunk layer test.
red008@proxy:~/edubase/src$