Homework 4 (by Yujing Zhang) - Solutions
Due: hard copy, at 3:00pm, Oct. 14, 2015

VERY IMPORTANT: Deposit hard copy of your answers, in class. Please
1. Separate your answers, on different page(s) for each question (staple additional pages, if needed).
2. Type the full info on each page: your name, Andrew ID, course#, Homework#, Question# on each question.

Reminders
• Plagiarism: Homework is to be done individually.
• Typeset all of your answers, please.
• Late homeworks: Standard policy: email (a) to all TAs, (b) with the subject line exactly 15-415 Homework Submission (HW 4), and (c) the count of slip-days you are using.

For your information:
• Graded out of 100 points. 6 questions total. Expected effort: ≈ 3-6h.

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffers</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>B Tree</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>B+ Tree</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Extendible Hashing</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Linear Hashing</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Sorting</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Revision: 2015/10/27 19:54
Question 1: Buffers

On separate page, with ‘[course-id] [hw#] [question#] [andrew-id] [your-name]’

Graded by: Anna Etzel

Consider a buffer pool with 4 frames (F1-F4); and a file with pages numbered from P1 to P7 inclusive. Consider the 10 read-requests for the pages, as shown in the table below. Consider three different buffer replacement policies: least recently used (LRU), most recently used (MRU), and ‘clock’. Assume the following:

- The buffer pool is initially empty.
- The frames are unpinned, immediately after use.
- The pointer for the ‘clock’ policy, starts at frame F1 at the very beginning; then, it moves whenever we need to replace the contents of a frame, as the textbook says.

(a) [5 points] For each policy, which page is evicted at which time as requests come in? Fill in the table below. (Put a “-” (dash) if nothing is evicted at certain timestamp)

<table>
<thead>
<tr>
<th>timestamp</th>
<th>request</th>
<th>evicted page (LRU)</th>
<th>evicted page (MRU)</th>
<th>evicted page (Clock)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>P1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t2</td>
<td>P2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t3</td>
<td>P3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t4</td>
<td>P2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t5</td>
<td>P3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t6</td>
<td>P4</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t7</td>
<td>P5</td>
<td>P1</td>
<td>P4</td>
<td>P1</td>
</tr>
<tr>
<td>t8</td>
<td>P1</td>
<td>P2</td>
<td>-</td>
<td>P2</td>
</tr>
<tr>
<td>t9</td>
<td>P6</td>
<td>P3</td>
<td>P1</td>
<td>P3</td>
</tr>
<tr>
<td>t10</td>
<td>P7</td>
<td>P4</td>
<td>P6</td>
<td>P4</td>
</tr>
</tbody>
</table>

Solution:

<table>
<thead>
<tr>
<th>timestamp</th>
<th>request</th>
<th>evicted (LRU)</th>
<th>evicted (MRU)</th>
<th>evicted (Clock)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>P1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t2</td>
<td>P2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t3</td>
<td>P3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t4</td>
<td>P2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t5</td>
<td>P3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t6</td>
<td>P4</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>t7</td>
<td>P5</td>
<td>P1</td>
<td>P4</td>
<td>P1</td>
</tr>
<tr>
<td>t8</td>
<td>P1</td>
<td>P2</td>
<td>-</td>
<td>P2</td>
</tr>
<tr>
<td>t9</td>
<td>P6</td>
<td>P3</td>
<td>P1</td>
<td>P3</td>
</tr>
<tr>
<td>t10</td>
<td>P7</td>
<td>P4</td>
<td>P6</td>
<td>P4</td>
</tr>
</tbody>
</table>

Grading info: -1 for each wrong blank
(b) [5 points] Draw the final state of the buffer for each policy.

<table>
<thead>
<tr>
<th></th>
<th>LRU:</th>
<th>MRU:</th>
<th>Clock:</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRU:</td>
<td>5 1 6 7</td>
<td>7 2 3 5</td>
<td>5 1 6 7</td>
</tr>
<tr>
<td>MRU:</td>
<td>7 2 3 5</td>
<td>1 2 3 15</td>
<td>13 14 15 12</td>
</tr>
<tr>
<td>Clock:</td>
<td>5 1 6 7</td>
<td>5 1 6 7</td>
<td>13 14 15 12</td>
</tr>
</tbody>
</table>

Solution:

Grading info: -2 for each wrong blank

(c) [5 points] For each of the three policies, draw the final states of the buffer for a sequential scan of pages numbered from P1 to P15 inclusive. (That is, scanning 15 pages from page P1 to page P15, in order)

<table>
<thead>
<tr>
<th></th>
<th>LRU:</th>
<th>MRU:</th>
<th>Clock:</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRU:</td>
<td>13 14 15 12</td>
<td>1 2 3 15</td>
<td>13 14 15 12</td>
</tr>
<tr>
<td>MRU:</td>
<td>1 2 3 15</td>
<td>1 2 3 15</td>
<td>13 14 15 12</td>
</tr>
<tr>
<td>Clock:</td>
<td>13 14 15 12</td>
<td>13 14 15 12</td>
<td>13 14 15 12</td>
</tr>
</tbody>
</table>

Solution:

Grading info: -2 for each wrong blank
Question 2: B Tree ................................... [20 points]

On separate page, with ‘[course-id] [hw#] [question#] [andrew-id] [your-name]’

Graded by: Anna Etzel

(a) Produce the most dense possible B-tree of order \( d = 2 \) (at most 4 keys per page), containing as keys the integers 1 through 24 inclusive. For example, the most dense B-tree with keys from 1 to 9 has height \( h=2 \), and is shown in Figure 1:

![Figure 1: dense B-tree of order \( d=2 \), with height \( h=2 \)](http://www.cs.cmu.edu/~christos/courses/dbms.F15/hws/HW4/tree-template.docx)

For your drawing convenience, here is the MSWord template for tree nodes:

i. [2 points] How many nodes does the structure have?

Solution: 6

ii. [3 points] Draw the final structure.

Solution:

(b) Produce the most sparse B-tree of order \( d = 1 \), containing the keys 1 through 15 inclusive.

i. [2 points] How many nodes does the structure have?

Solution: 15

ii. [2 points] What is the height of the structure?

Solution: 4

iii. [1 point] How many keys are in the root?

Solution: 1

iv. [2 points] Which key(s) is/are in the root?

Solution: 8

(c) Consider an empty B-tree of order \( d = 3 \). Using the standard B-tree algorithm given in the foils (2-to-1 split, no deferred splits), insert keys from 1 to 32 (inclusive), in order.

Question 2 continues...
i. [2 points] What is the height of the final structure?
   Solution: 3

ii. [1 point] How many keys are in the root?
   Solution: 1

iii. [2 points] Which key(s) is/are in the root?
   Solution: 16

iv. [1 point] How many keys are in the last (= right-most) leaf node?
   Solution: 4

v. [2 points] Which key(s) is/are in the last leaf node?
   Solution: 29,30,31,32

Homework 4 continues...
Question 3: B+ Tree ............................. [30 points]
On separate page, with ‘[course-id] [hw#] [question#] [andrew-id] [your-name]’

Graded by: Dana Van Aken

Consider the following B+ tree of order $d=2$ and height $h=3$ levels, shown in Figure 2.

Please make the following assumptions:

- (A) For each part of the problem, disregard previous parts and apply the instruction on the tree structure in Figure 2.
- (B) With respect to “$\geq$”, follow the convention used in the textbook, and in Figure 2 that is, the left pointer is for $<$, the right one for $\geq$.
- (C) In case of underflow, if you can borrow from both siblings, choose the one on the left.
- (D) In case of overflow, promote the key in the middle.
- (E) If a large part of the tree remained unchanged, you may type ‘(unchanged)’, instead of drawing it, to make your solution cleaner.

For each question except for the last one (‘e’) below, draw the tree after the specified operation, always starting from the tree of Figure 2.

Grading info: -1 for including any keys added in (a) and (b) in subsequent problems (per occurrence).

(a) [5 points] Insert 10$^\ast$.

Solution:

(b) [5 points] Starting from the B+tree of Figure 2 insert 20$^\ast$.

Question 3 continues...
Solution:

Here is the left subtree of root. The right subtree is the same.

(c) [5 points] Starting from the B+tree of Figure 2, delete 7∗. (according to assumption C)

Solution:

Grading info:
- 2 for the below answer (see assumption D).
- Full penalty for forgetting to delete the key from either the leaf or non-leaf node (or both).

(d) [5 points] Starting from the B+tree of Figure 2, delete 17∗, 18∗, and 26∗.

Solution:

Grading info: Full penalty for forgetting to delete one or more keys from either the leaf or non-leaf node (or both).

Question 3 continues...
(e) Consider the 12 records with keys 1*, 2*, . . . , 12*. Suppose they are bulk-loaded in an *initially empty* B+ tree of order \( d=1 \), using the algorithm from the textbook.

i. [3 points] How many levels are in the resulting tree? (The B+ tree of Figure 2 has height \( h=3 \) levels; The fully-packed leaves are at level '1').

Solution: 3

ii. [3 points] How many keys are in the root?

Solution: 2

iii. [4 points] Which key(s) is/are in the root?

Solution: 5 and 9

Homework 4 continues…
Question 4: Extendible Hashing ....................... [10 points]

Graded by: Jinliang Wei

This is a modification of Exercise 11.1, p. 386-387 of the textbook. Assume we have the following records where we indicate the hashed key in parenthesis (in binary). Consider an Extendible Hashing structure.

- Each bucket can hold up to 3 records.
- Initially the structure is empty (only one empty bucket).

Consider the result after the records above have been inserted in the order shown, using the lowest-bits for the hash function, as the textbook does. That is, records in a bucket of local depth \( d \), agree on their rightmost \( d \) bits.

- ‘a’ [00001]
- ‘b’ [01000]
- ‘c’ [00000]
- ‘d’ [10100]
- ‘e’ [10101]
- ‘f’ [00010]
- ‘g’ [11000]
- ‘h’ [10001]

(a) [2 points] What is the global depth of the resulting directory?

Solution: 3

(b) [3 points] Which record causes the first split? Which record causes the second split?

Solution: (1)d (2)f

(c) After inserting records ‘a’-‘h’, suppose that we insert record ‘i’ [10000].

i. [2 points] How many buckets will we have now?

Solution: 5 buckets

ii. [2 points] What is the local depth of the bucket containing record ‘i’?

Solution: depth: 4

iii. [1 point] Which other keys are in the same bucket with record ‘i’? (Type ‘alone’, if ‘i’ is by itself.)

Solution: c [00000]

Homework 4 continues...
Question 5: Linear Hashing .......................... [10 points]

On separate page, with 'course-id [hw#] [question#] [andrew-id] [your-name]'

Graded by: Jiaxi Xiong

For the same records as in Question 4 (‘a’ - ‘h’), consider a linear hashing structure
where pages can hold up to 3 records. A bucket consists of 1 main page, and zero or
more overflow pages. Initially the structure is empty (only one empty bucket/page).
Consider the result after the records above have been inserted in the order shown.

- Use the algorithm in the textbook (p. 380 and on, section 11.3)
- Assume that a split is 'triggered' whenever the insertion of the new record caused
  the creation of an overflow page.

(a) [3 points] How many buckets will we have? (In the textbook, there are 5 buckets
    in Figure 11.9, that is, overflow pages don’t count as ‘buckets’.)

Solution: 4

(b) [4 points] List all the records in the bucket that contains record ‘b’ (including
    records in the overflow pages for that bucket).

Solution: b, c, d, g; g is in the overflow page.

(c) In the hash table that contains records ‘a’-‘h’, insert records ‘j’ [11100], ‘k’ [00011]
    in this order.

i. [1 point] How many buckets will we have?

Solution: 4 buckets.

ii. [2 points] List all the elements in the bucket which contains the record ‘k’
    [00011] (including records in any overflow page(s) for that bucket). Type ‘alone’,
    if ‘k’ is by itself.

Solution: alone

Homework 4 continues...
Question 6: Sorting ........................................... [15 points]
On separate page, with ‘[course-id] [hw#] [question#] [andrew-id] [your-name]’

Graded by: Yujing Zhang

- *Hint:* This is a modified version of Exercise 13.1 in the textbook.
- Feel free to study the answers to the odd-numbered exercises, that are on the web.

Suppose you have a file with $N = 8 \times 10^7$ pages, and $B$ buffers (each able to hold one page).

(a) [4 points] What is the total I/O cost of sorting the file using the two way merge sort (with $B=3$)?

**Solution:** $4.48 \times 10^9$ or $4.16 \times 10^9$

*Grading info:* -2: no final result

(b) Suppose we have $B=100$ buffers, and we are using the external sorting algorithm discussed in chapter 13.3 (p.427, Figure 13.6) of the textbook.

i. [3 points] How many runs will you produce in the first pass?

**Solution:** $8 \times 10^5$

*Grading info:* -2: no final result

ii. [4 points] Now assume that we have a disk with $16$ms I/O time. What is the total running time to sort the file?

**Solution:** $1.024 \times 10^{10}$ms

*Grading info:* -1 for missing time unit

iii. [4 points] With $B=100$ buffer pages, what is the maximum size $N_{max}$ of the file (in number of pages) that we can sort with 4 passes?

**Solution:** $97029900$

*Grading info:* -2: no final result

End of Homework 4