CS 186 Discussion Section
Week 6

Peter Alvaro and Kuang Chen

March 2, 2009

I/O Cost Analysis

Fill in the I/O costs for the operations listed in the table. Assume that the relation R takes up B blocks of disk space and that it contains T tuples, the equality and range searches are performed on column R.A which contains U unique values, and that the tree indices, again on column R.A, have height H and L leaf blocks. Calculate the costs in terms of I/Os not time, as it is done in your books.

Let max and min represent the maximum and minimum values in column R.A, respectively.
Assume that the distribution of unique values (U) of attribute $R.A$ over the blocks of R is uniform.

File Type	Scan	$R.A = c$	$R.A > c$	Insert	Delete
Heap	B	$0.5B$	B	1	$0.5B + 1$
Sorted	B	$log_2 B + \frac{B}{U}$	$log_2 B + \frac{max-c}{max-min} B$	$log_2 B + 0.5B$	$log_2 B + 0.5B$
Clust. Tree	$H + B$	$H + \frac{B}{U}$	$H + \frac{max-c}{max-min} B$	$2H$	$2H$
Unclust. Tree	$H + L + T$	$H + \frac{L}{U} + \frac{T}{U}$	$H + \frac{max-c}{max-min}(L + T)$	$2H + 1$	$2H + 1$

As you can see by the assumptions, were taking a different twist while filling this table, which we will revisit when well talk about Cost Estimation for Query Optimization (Chapters 14 & 15). In order to understand some of the formulas, we need to introduce the notion of *reduction factor* of an operator as *the percentage (in terms of tuples or scanned blocks) returned from the execution of the operator, over the total number of tuples (or blocks) of the initial relation.* In other words, $rf = \frac{outputsize}{inputsize} \in [0, 1]$. We will also assume that the distribution of unique values (U) of attribute $R.A$ over the blocks of R is uniform. Finally, for the clustered tree case, we will regard that the index follows alternative one (p. 276 of your books), meaning that the data entries in the leaf nodes are the actual data records (in essence, the leaves are the blocks of the heap file), while for the unclustered tree case we follow alternative two (the data entry is a $< key, rid >$ pair, with the rid pointing to the corresponding record in the (separate) heap file).

Given the above, let us attempt to explain the equality search operation ($R.A = c$) for a sorted file. We need to access $log_2 B$ pages on average for our binary search, to locate the block with the first qualifying tuple. We then need to retrieve all the blocks containing tuples with the same value. The number of such blocks corresponds to the reduction factor of the equality selection operator, times the number of blocks of R, $\frac{1}{U}B$. For example, if there were 6 pages and 3 unique values in the search column, on average there would be 2 pages that would be retrieved. The same reasoning applies for the range selection predicate, only now, the percentage of blocks containing qualifying tuples will be $rf = \frac{max-c}{max-min}$, where max and min are the maximum and minimum values of the column $R.A$.

For the clustered tree case, besides the cost we have to pay to retrieve all blocks with qualifying tuples, we have to descend the tree. The cost for the latter is H, the height of the tree. The worst case for insertion/deletion is that we have to split/merge every node in the path from the root to the leaves: H to reach the leaves, and H going back up performing the splits.

Finally, for the unclustered tree case, each data entry contained in a leaf page can point to a different block in the file (remember that in an unclustered index, the ordering between the data entries and the tuples stored in the heap file of the relation does not match). Thus, for the equality search case, we have to descend the tree (H), visit $\frac{1}{U}L$, the leaf pages containing data entries with key value equal to c, and then follow the pointer of each qualifying data entry $\frac{1}{U}T$, that will take us to possibly different heap file blocks.

1 External Sorting

Activity

24 pages
2 tuples per page
4 available buffer slots

Algorithms

Two-way Merge Sort Algorithm:

```
// Given a file on disk; sorts it using three buffer pages
// Produce runs that are one page long: Pass 0
Read each page into memory, sort it, write it out
// Merge pairs of runs to produce longer runs until only
// one run (containing all records of input file) is left
While the number of runs at end of previous pass is > 1
    // Pass i= 1, 2, ...
    While there are runs to be merged from previous pass:
            Choose next two runs (from previous pass).
            Read each into an input buffer; page at a time.
            Merge the runs and write to the output buffer;
            force output buffer to disk one page at a time.
```

General Merge Sort Algorithm:

```
// Given a file on disk, sorts it using B buffer pages
// Produce runs that are B pages long: Pass 0
Read B pages into memory, sort them, write out a run
// Merge B-1 runs at a time to produce longer runs until only
// one run (containing all records of input file) is left
While the number of runs at end of previous pass is > 1:
    // Pass i = 1, 2...
    While there are runs to be merged from previous pass
    Choose next B-1 runs (from previous pass).
            Read each into an input buffer; page at a time.
            Merge the runs and write to the output buffer;
            force output buffer to disk one page at a time.
```

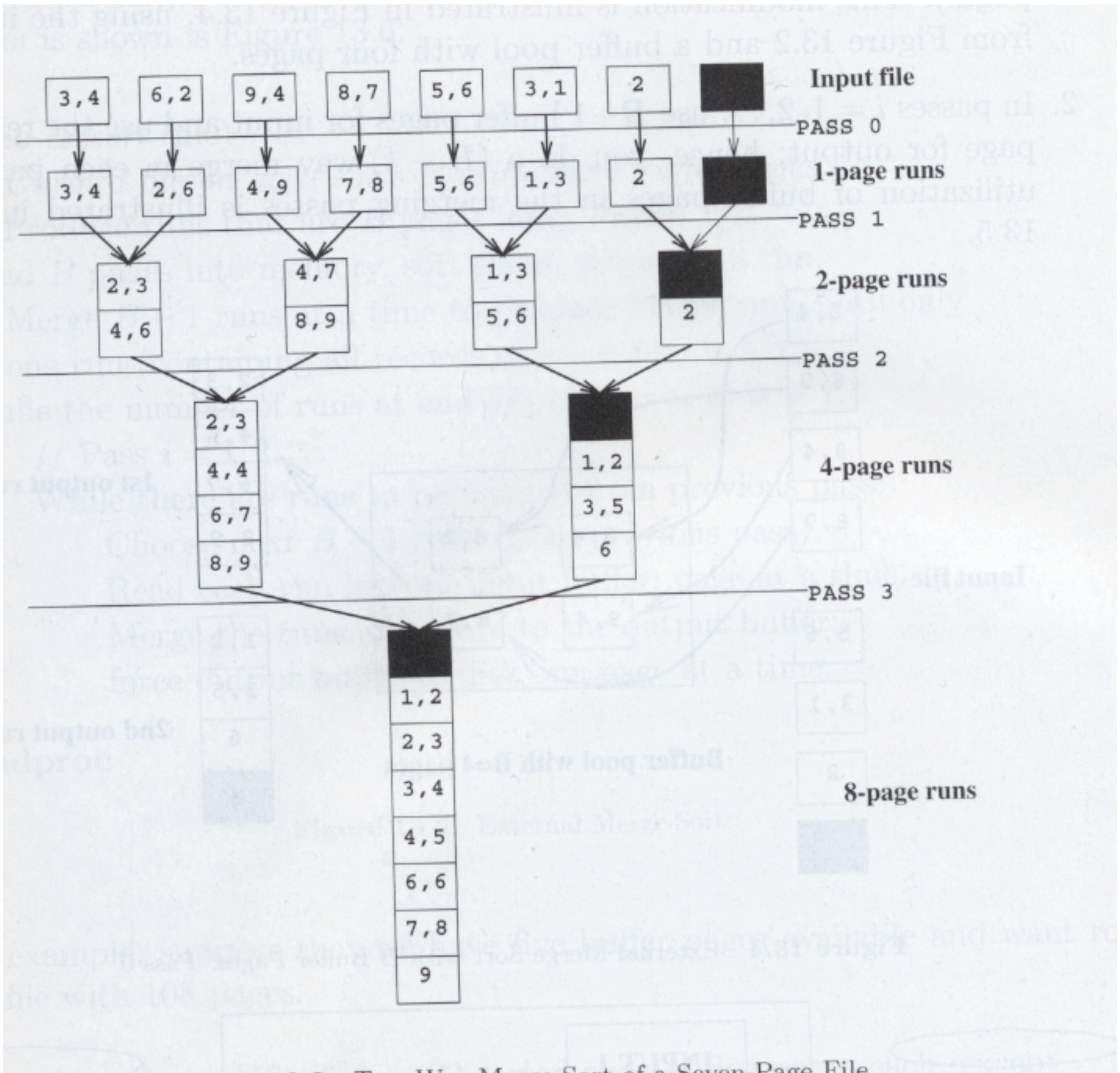

Figure 13.2 Two-Way Merge Sort of a Seven-Page File

1.1 Exercises

Three scenarios:

1. A file with 10,000 pages and three available buffer pages
2. A file with 20,000 pages and five available buffer pages
3. A file with 2,000,000 pages and seventeen available buffer pages

Answer the following questions for each of the above scenarios, assuming that our most general external sorting algorithm is used:

1. How many runs will you produce in the first pass? *(1)* $\lceil 10,000/3 \rceil$, *(2)* $\lceil 20,000/5 \rceil$, *(3)* $\lceil 2,000,000/17 \rceil$
2. How many passes will it take to sort the file completely?

3. What is the total I/O cost of sorting the file?
4. How many buffer pages do you need to sort the file completely in just two passes?