TECHNICAL UNIVERISTY OF CRETE
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DATA AND FILE STRUCTURES

3rd Assignment

Deadline: 28 May 2021
The assignment is solitary.

B+ Trees

Implement a B+ tree on the disk. The pages of the disk (as well as the nodes of the tree) have size N = 256 bytes. Each key is associated with a data field of size m = 32 bytes. The numbers N, m are implementation parameters.

Nodes are B+-tree, they store pointers and keys without their data (they do not contain a data field). They store information such as the number of keys (how many keys are stored in the node), pointers to the pages in the layer below, a pointer to the layer above. Internal nodes are an index structure for the data. Each node is linked by an index to its adjacent node at the same level (nodes at each level, as in the sheets, create a list).

In particular, nodes in the leaves contain pointers to an external data file. That is, each key (like all nodes) has a key that, instead of pointing to a node in the layer below, points to another external file (the location where the data field is stored in the file).

For a given N=256 bytes, compute the degree n of B+tree. The degree of the tree is related to the number of pointers and keys in nodes, does not depend on m, and does not affect the number of data stored in the external file.

Follow and implement the steps below:

- Insert a random key
- Deleting a random key value
- Random key value search
- Key range queries in the K value range. That is, for the range [K1, K2], K = K2 - K1
- Performance and documentation of results

Enter n $=10^5$ unique keys with values from 1 to $10^{6.}$ Fill in the values in the table below:

1. Average number of disk accesses per import. After you have imported all keys, make an additional 20 imports and count the average number of disk accesses.

2. Average number of disk accesses per search and delete. That is, for 20 searches and 20 deletions of data, count the average number of disk accesses.

3. For 20 value range searches (i.e. for different K1, K2 where K=K2- K1, for K=10 and 1000) measure the average number of disk accesses.

Average number of disk accesses per import	**Average number of disk accesses per random search**	**Average number of disk accesses per deletion**	**Average number of disk accesses for value range K (K=10)**	**Average number of disk accesses for value range K (K=1000)**

Comment (separate text file) on the performance of all methods and try to justify the performance of each method. **Pay particular attention to the interpretation of the results and do not just show the values.**

Recommendations:

- See http://people.cs.vt.edu/~shaffer/Book/, other book or WWW.
- Most of the codes you will find involves implementation in main memory, whereas the exercise asks for implementation on disk.
- Variables and pointer constants are of integer (4 bytes) type.
- Initially do the implementation in main memory and without a data field. Then you can convert it for disk.
- Do not proceed with the implementation until you have a good understanding of the B+-tree functionality and the differences from B-tree.
- **The implementation must be B+tree and not B-tree. Exercises with B-tree implementation are not graded**.
- The content of the data field (it can be anything) is not important

Deliverables: A compressed zip file containing the following:

- A report 2 pages maximum with the results requested i.e. **Pay particular attention to the interpretation of the results** and do not just show the values.
- A report describing in 1-2 pages how the code was built (i.e. for each question what is the general idea of the solution in 3-4 sentences), there are clear compiler and execution instructions, what errors it has (if any, cases where the program does not work, or cases where it does more than the exercise asks for, what you used from off-the-shelf programs or information sources.

Even point to WWW resources such as Wikipedia or Stackoverflow (full address of relevant pages).

- The code contains brief comments explaining the implementation. Add comments in javadoc format at the beginning of each class and method. Also javadoc comments before each member variable of the classes. And wherever are required within your code.
- In addition to the above, the exercises are graded with a 10 if:
    - The zip is complete.
    - The codes pass through a compiler and run normally and correctly in a windows or Linux environment (Note: You must use relative directory paths and not absolute ones, which only apply to your computers).
- Copies (even of part of the implementation) are zeroed out.