UNIVERSITY OF WISCONSIN–MADISON
COMPUTER SCIENCES DEPARTMENT

CS 564: DATABASE MANAGEMENT SYSTEMS

Assignment 3: Introduction to B+ Trees
Due:March 27th, 5 p.m.

Instructor: Raghu Ramakrishnan

# 1 Introduction

In this assignment, you will answer several questions about B+ trees, in particular, their current implementation in Minibase. You will create data files which are then input to our B+ tree trace generator to produce "trace" files. You will then use the "trace" files as input to our B+ tree visualization tool.

*You will carry out this assignment in teams of two with the same partner as for the first assignment.* This assignment involves no programming beyond creating data files for inserting/deleting entries in a B+ tree. Try and use the visualization tool to help you think through the problems!

# 2 Available Documentation

You should begin by reading the chapter on Tree Structured Indexes, in particular the sections on B+ tree insertion and deletion. The visualization tool provides on-line help.

# 3 The Exercises

1. Start with the provided data file "e1.dat" which contains commands to initialize a B+ tree of order 2. When you run our B+ tree trace generator program with this data file (genTrace < e1.dat), it will print a trace to `stdout`. To save the trace to a file, redirect output to the file. Use this file as the input to the B+ tree visualization tool. Work with the tool to see what the tree's structure is. Draw a picture of the tree similar to Figure 5.10 in the text.

2. How does the structure of this tree differ from the one depicted in Figure 5.10? Actually, both B+ trees are indexing the same set of data records. Why do we get two different B+ trees?

3. Modify the data file from Exercise 1 to insert the key 23 as the last operation on the tree. What happens? How many page writes and reads are there in this insertion? What can you deduce about the insertion algorithm implemented in Minibase, compared to the one in the text?

4. Modify your data file from Exercise 1 to delete 29, 33, and 34 at the very end. The command to delete a key X is "delete X". How many page writes and reads are required in this series of deletion? What can you deduce about Minibase's deletion algorithm by comparing it to the one in the text?

5. Given what you have deduced about Minibase's B+ tree insertion and deletion algorithms, modify your data file in Exercise 1 so the internal structure of the tree exactly matches Figure 5.14. What did you have to do?

6. Using the data file "e2.dat", find examples of each of the following:

(a) An insertion that causes a leaf split according to the basic algorithm, but that could be handled by re-distribution with the leaf's right sibling.

(b) An insertion that causes a leaf split even if we consider the possibility of re-distribution with the leaf's right sibling.

(c) A series of one or more insertions that also causes a non-leaf split, but does not change the height of the tree.

(d) A series of one or more insertions that changes the height of the tree.

(e) A query that retrieves data from exactly 3 leaf pages. How many pages are fetched from disk to answer this query?

7. Using insertions of unique keys only, construct a tree of order 2 and height 4 that has as **few** records as possible.

8. Using insertions only, construct a tree of order 2 and height 3 that has as **many** records as possible.

# 4 Compiling Your Code and Running the Tests

The B+ tree trace generator program (genTrace) and the data files(e1.dat and e2.dat) are in `~cs564-1/spring96/assign3/`.
You have to copy these files into your directories.
The visualization tool can be run by typing

```
/p/course/cs564-raghu/spring96/assign3/bin/view_btree
```

Note that you have to run the visualization tool from the directory in which your trace is stored.

# 5 What to Turn In, and When

You should turn in:

1. Your solutions in the form of modified data files.

2. Brief answers to the questions in a separate file. Use any word processing software that you want, but please make the file easily readable. (Plain text is preferable.)

The assignment is due at 5PM on March 27th. The solution will be made public then, and solutions turned in after that time will not receive any credit. So be sure to turn in whatever you have, even if it is not working fully, at that time.