UiT

NORGES
ARKTISKE
UNIVERSITET

INF-2202 (Fall 2016)

ASSIGNMENT #1

CONCURRENT B+TREES

Tim A. Teige & Lars Ailo Bongo

23.08.2016

Overview

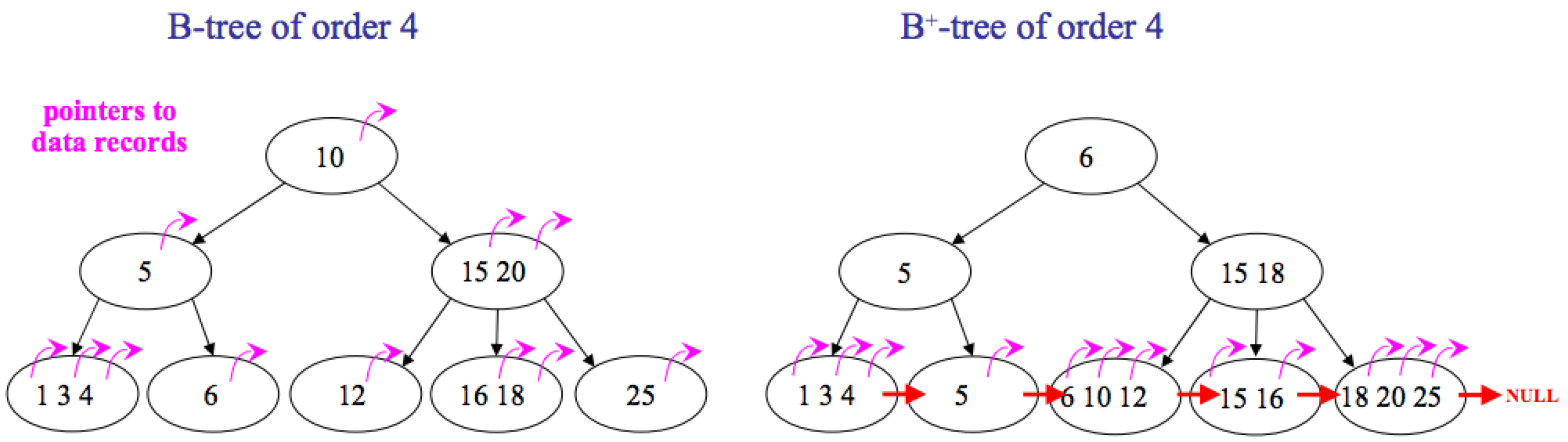
- Your task is to transform the given B+tree code in «precode/bpt.c», into a concurrent B+tree.

- There are provided papers on B+trees inside the «btrees-notes» directory. There are also provided papers on concurrent B+trees. These are inside «concurrent-btrees-papers».

- **Deadline: 12.09 end of the day.**

B-tree vs B+tree

- A B^+-tree can be viewed as a B-tree in which each node contains only keys (not pairs), and to which an additional level is added at the bottom with linked leaves



About the precode

- The bpt.c file contains a single-threaded and complete B+tree implementation in c.
- Multi-threaded benchmark and correctness test have been provided in the source code.
- If your concurrency control implementation is correct, the benchmark and correctness test should run without any errors when using multiple threads (eg., no segmentation faults, all inserted keys are searchable).
- Note that the correctness test will pass when using a single thread.

Compiling and running

- You can use any modern compiler (C99 or above) to compile the code
- Run the program with the –h argument to see a list of accepted run parameters and to display help.

```
$ gcc –g bpt.c –o bpt –lpthread –lm
$ ./bpt –h
```

- Use the ‘-t 1’ switch to run the sequential and multi-threaded correctness test.
- To use more than 1 thread, add the –n «#threads» parameter.

Requirements and limitations

- Modify the B+tree Search, Insert and Delete operations to support concurrency
- You may use any known techniques or your own
- Use the POSIX Thread API, please contact the T.A. if you will use other methods.
- You can not change the benchmark or the correctness test.
- The program must be able to run using any number of threads.
- A report describing the implementation and design of the concurrency control, and a performance analysis (speedup, efficiency)
- Analysis should be done using the benchmark provided with different settings for the update ratio (-u «x») and the number of threads (-n «y»).

Github workflow

- Clone the class repository: https://github.com/uit-inf-2202-f16/uit-inf-2202-f16.github.io
- The assignment is located under assignments/assignment-1
- We will use Github classroom for delivery, further instructions on how to deliver will be given on the next group session.
- We are currently waiting for private repositories from Github.

Grading

- Based on your submitted code and report, a PASS or FAIL grade will be given.
- Be sure to follow the requirements and limitations.

Disclaimer

- Please do not publicize or share your solution or code anywhere without our permission.
- This is an individual assignment, please do not share code or copy other’s code.
- Group discussion and sharing ideas is highly recomended.