Practical Approaches to Optimizing Virtual Memory Buffer Management

第七組 鄧雅文、王偉力、吳吉加、周哲瑋

outline

1.Motivation

2.Introduction

3.Method

4.Result

5.Conclusion

Motivation

1. Rapid Growth of AI and Big Data

- Exponential increase in data volume, heightened demand for data access efficiency

2. Optimizing System from Middleware Perspective

- Not just model optimization, but also enhancing hardware-software interaction speed

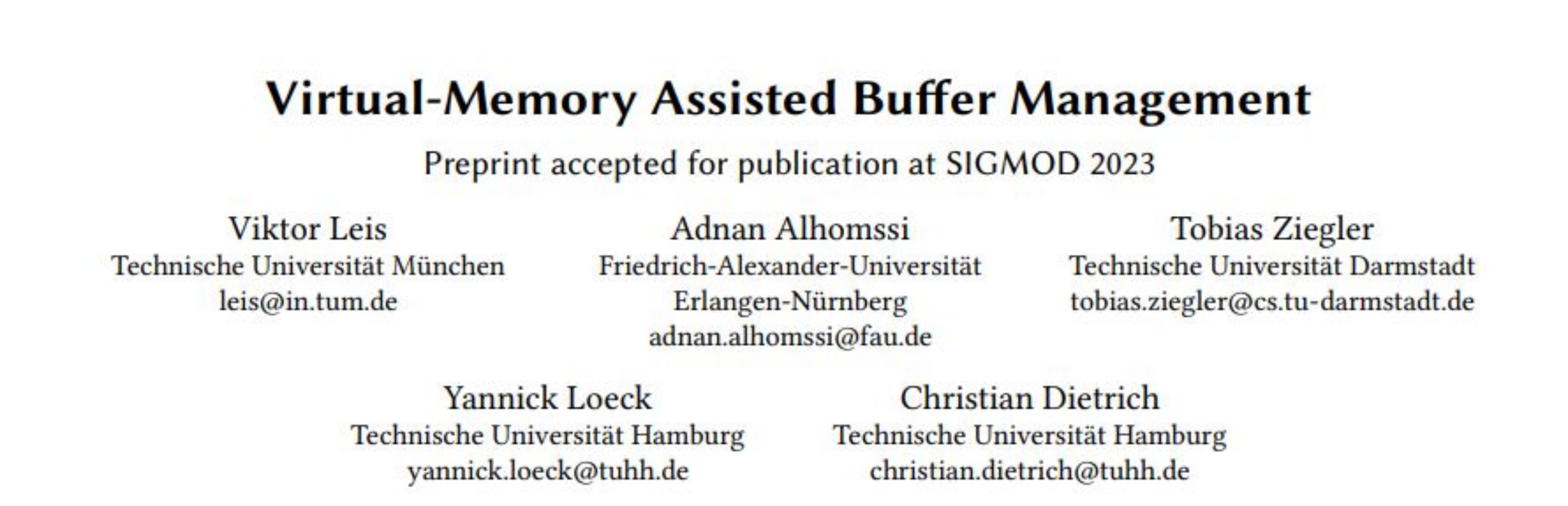
3. Improving Database Performance

- Based on VMCACHE code, exploring software-level improvements for better data access efficiency

4. Goals

- Provide efficient data management solutions for applications like AI or big data

Introduction



Virtual-Memory Assisted Buffer Management

Preprint accepted for publication at SIGMOD 2023

Viktor Leis
Technische Universität München
leis@in.tum.de

Adnan Alhomssi
Friedrich-Alexander-Universität
Erlangen-Nürnberg
adnan.alhomssi@fau.de

Tobias Ziegler
Technische Universität Darmstadt
tobias.ziegler@cs.tu-darmstadt.de

Yannick Loeck
Technische Universität Hamburg
yannick.loeck@tuhh.de

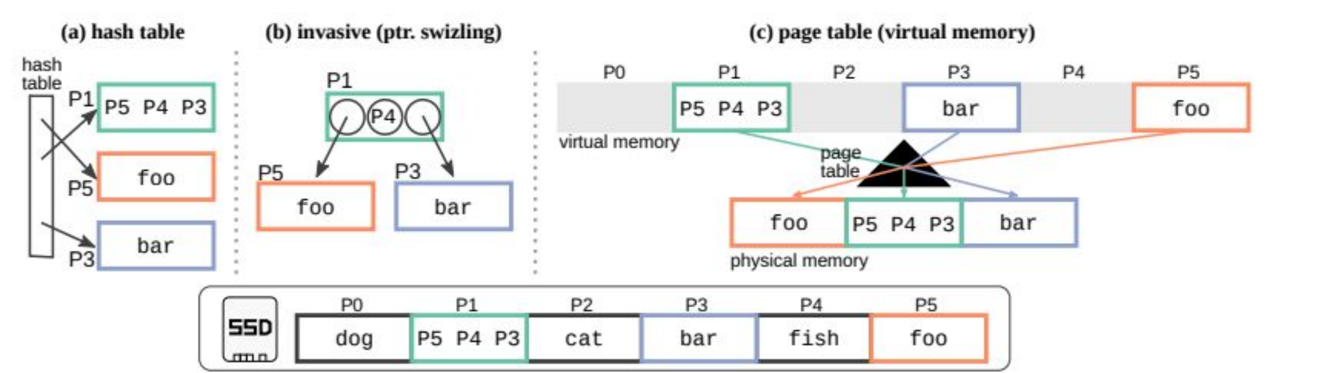
Christian Dietrich
Technische Universität Hamburg
christian.dietrich@tuhh.de

Introduction

Larger-than-memory Database

VMCACHE : VM + PT + TLB, DBMS-driven , easy to implement

page management, page states, page table length



Method

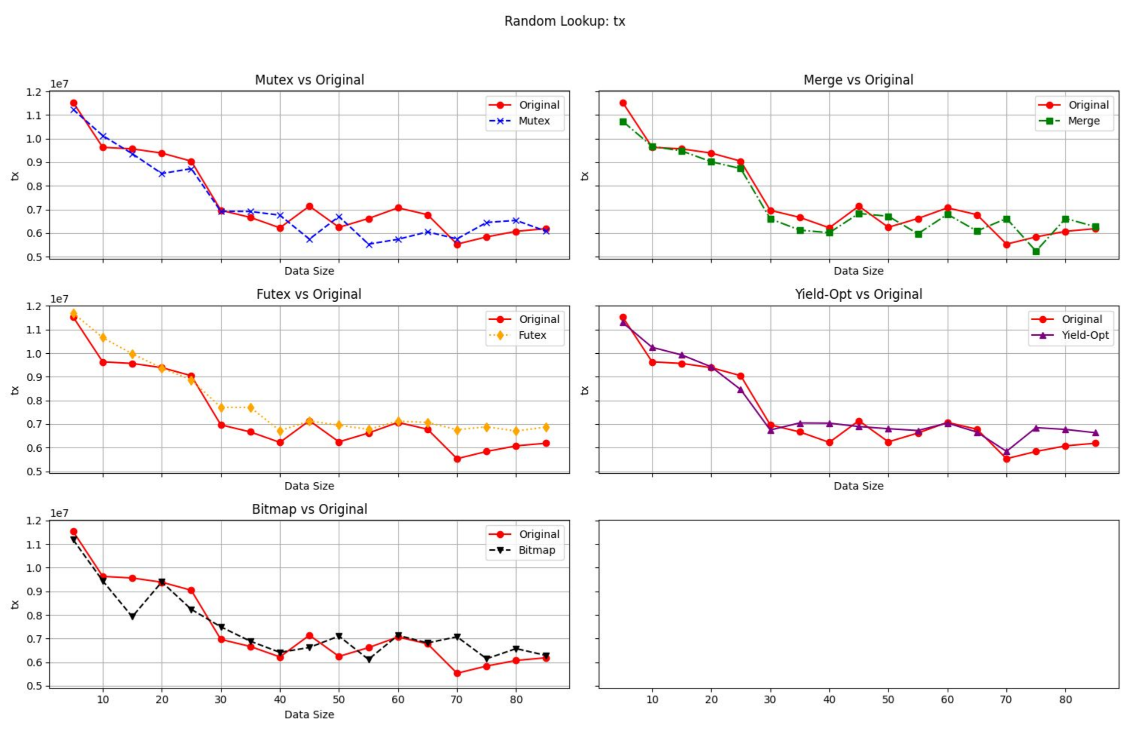
Improvements Based on Original Code

Five methods tested with random access and TPC-C benchmark

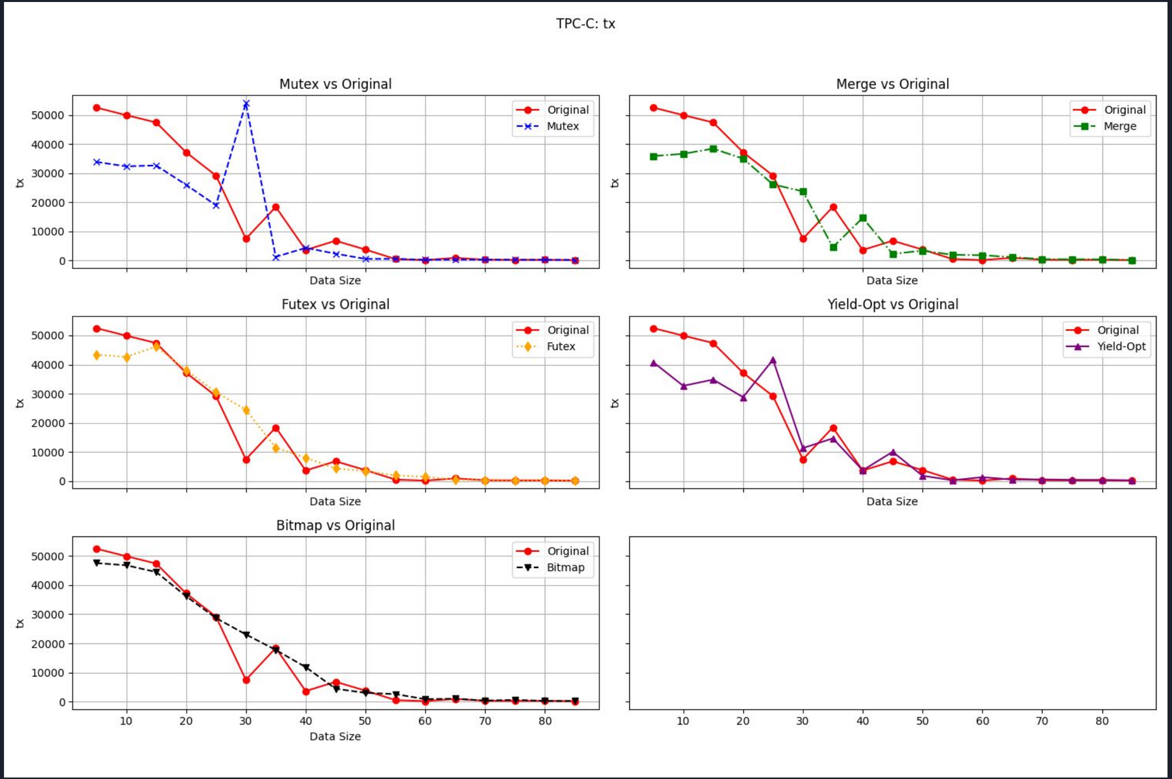
1. **Mutex**: Addresses synchronization issues
2. **Futex**: Handles synchronization with a fast userspace mutex, lighter and faster than mutex
3. **Merge**: B-tree inner node merge
4. **Yield-Opt**: Dynamically adjusts waiting strategies
5. **Bitmap**:   Manages free space

Implementation Division: method 1~4 周哲瑋，5 鄧雅文

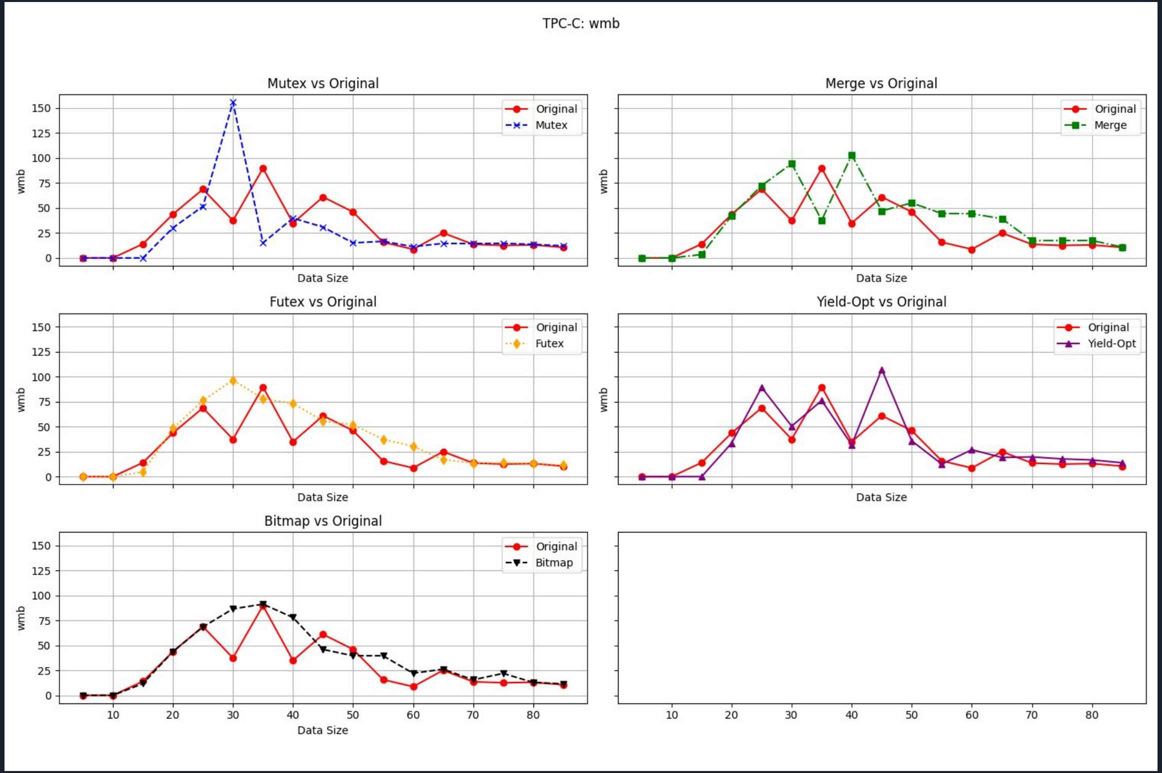
Result


Random Lookup: tx
Mutex vs Original
Merge vs Original
Futex vs Original
Yield-Opt vs Original
Bitmap vs Original
Original
Mutex
Merge
Futex
Yield-Opt
Bitmap
1e7
tx
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
Data Size
10
20
30
40
50
60
70
80

Result


TPC-C: tx
Mutex vs Original
Original
Mutex
Merge vs Original
Original
Merge
Futex vs Original
Original
Futex
Yield-Opt vs Original
Original
Yield-Opt
Bitmap vs Original
Original
Bitmap
tx
Data Size
0
10000
20000
30000
40000
50000
10
20
30
40
50
60
70
80

Result


TPC-C: wmb
Mutex vs Original
Original
Mutex
Merge vs Original
Original
Merge
Futex vs Original
Original
Futex
Yield-Opt vs Original
Original
Yield-Opt
Bitmap vs Original
Original
Bitmap
wmb
Data Size
0
25
50
75
100
125
150
10
20
30
40
50
60
70
80

Conclusion

Implementation of 5 Improvement Methods:

Overall Performance:

- Futex, Yield-Opt, and Bitmap show slight advantages over the original version

Testing Environment:

- Used Google Colab, limited to testing with up to 4 threads
- Unable to test with varying thread counts
- Long testing times

More Details : https://github.com/5000user5000/vmcache/tree/master

Thanks for listening