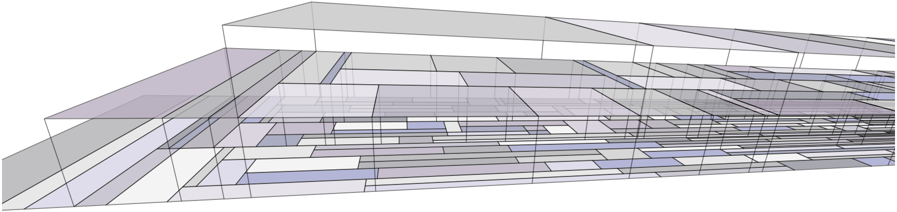
Aalto-yliopisto

Lectio praecursoria, October 22, 2010

Accessing Multiversion Data in Database Transactions



Tuukka Haapasalo
tuukka.haapasalo@tkk.fi

Background on databases

Databases

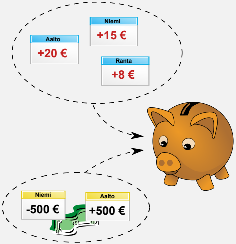
- Data storage for programs
- Examples:
  - Addresses in an address book
  - Bank account information
  - Calendar events
  - ...
  - Images, videos, music

Information retrieval

- Quick access to information
- Analogies: document archives, libraries
  - Books ordered by the author (in a library)
  - Easy to locate a certain book
  - Easy to locate all books written by a given author

Transactions

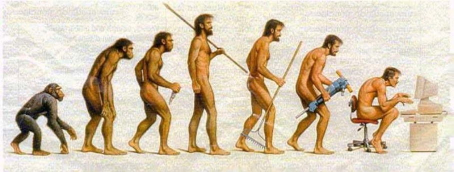
- Data are modified and accessed in transactions
- Atomicity
- Multiple concurrent updates
  - State remains consistent
  - Index structure remains intact



Multiversion databases

Multiversion databases

Difference to traditional databases?



Multiversion databases

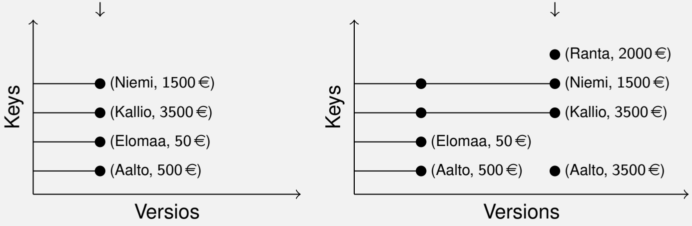
- Store the evolution of data
- What information was stored before?
- Examples:
  - What documents the archive consisted of when it was created?
  - Who were the users of the system on the June 6th, 2010?
  - What was the balance of Mr. X's bank account a month ago?

Modeling versioned data

Multiversion databases

Dissertation p. 12

Any change creates a new version (state) from the records of the index:



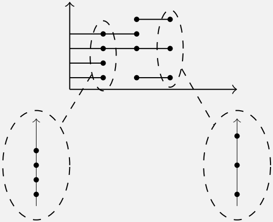
(a) Before modifications (b) After modifications

Queries can target previous versions in addition to the current version.

Optimality

Dissertation p. 35

A multiversion database index is optimal, if querying a version v is as efficient as in a single-version database index that only indexes the data items of the given version v.



- **Time-split B^+-tree (TSB-tree); Lomet and Salzberg [4]**
  - The first efficient multiversion index (1989)
  - Not optimal
  - Dissertation p. 55
- **Multiversion B^+-tree (MVBT); Becker et al. [1, 2]**
  - Second efficient multiversion index (1993–1996)
  - Optimal
  - Each update creates a new version
  - Dissertation p. 61
- **Multiversion access structure (MVAS); Varman and Verma [5]**
  - Third efficient multiversion index (1997)
  - Optimal according to a different (not so strict) definition
  - Each update creates a new version
  - Dissertation p. 69

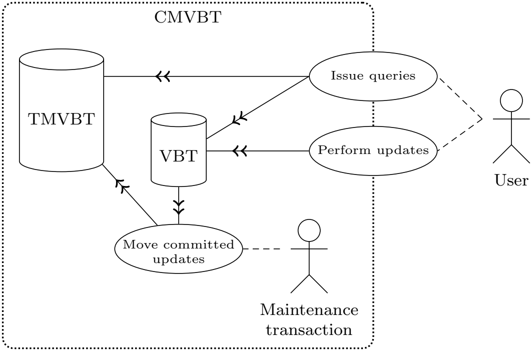
Our research

Contributions

1. Transactions to the MVBT index: the *transactional MVBT* (TMVBT)
   - Only a single updating transaction at a time
   - As efficient as the MVBT
   - Dissertation p. 75
2. The *concurrent MVBT* (CMVBT) for concurrent updating transactions
   - CMVBT = TMVBT + VBT
   - VBT = a versioned B^+-tree
   - Dissertation p. 111
3. Experimental evaluation
   - CMVBT is as efficient as the TSB-tree in the general situation
   - CMVBT is more efficient for key-range queries
   - Dissertation p. 137

Concurrent multiversion B$^+$-tree [3]

Dissertation p. 113



Experimental evaluation

Experimental evaluation

Tested index structures

Indexes we have evaluated

- **CMVBT** index
- **TSB-tree**

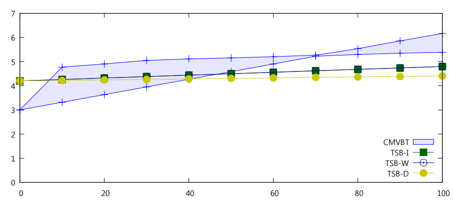
Also implemented

- **TMVBT** index (one transaction at a time)
- **VBT** index (alone)

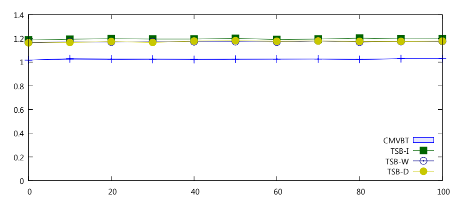
Queries and updates, short transactions

Dissertation p. 147

Page fixes



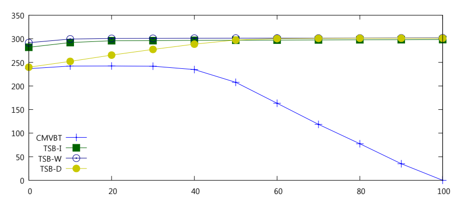
Page reads from disk



Key-range queries

Dissertation p. 151

Page reads from disk (almost identical to the number of page fixes in this test):



Summary

Summary

Index structures

- **TMVBT** = transactional, optimal MVBT
- **CMVBT** = **TMVBT** + **VBT**
- Multiple updating transactions can operate on the **CMVBT** concurrently

Kokeelliset tulokset

- **CMVBT** is as efficient as the **TSB-tree** in the general situation
- **CMVBT** is more efficient than the **TSB-tree** in key-range queries
- **CMVBT** takes 10–60 % more space than the **TSB-tree**

References I

[1] B. Becker, S. Gschwind, T. Ohler, B. Seeger, and P. Widmayer. On optimal multiversion access structures. In *Proceedings of the 3rd International Symposium on Advances in Spatial Databases*, pages 123–141, 1993.

[2] B. Becker, S. Gschwind, T. Ohler, B. Seeger, and P. Widmayer. An asymptotically optimal multiversion B-tree. *The VLDB Journal*, 5(4):264–275, 1996.

[3] T. Haapasalo, I. Jaluta, S. Sippu, and E. Soisalon-Soininen. Concurrent updating transactions on versioned data. In *Proceedings of the 2009 International Database Engineering and Applications Symposium*, pages 77–87, September 2009.

[4] D. Lomet and B. Salzberg. Access methods for multiversion data. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pages 315–324, 1989.

[5] P. J. Varman and R. M. Verma. An efficient multiversion access structure. *IEEE Transactions on Knowledge and Data Engineering*, 9(3):391–409, 1997.