zalando

Modern BTree techniques

DMITRY DOLGOV

31-01-2020

@aspis7

@aspis7

- D. Comet, ”Ubiquitous B-tree”, ACM Comp. Surv., vol. 11, no. 2, 1979.

- D. Comet, ”Ubiquitous B-tree”, ACM Comp. Surv., vol. 11, no. 2, 1979.
- Graefe, Goetz and Harumi A. Kuno. ”Modern B-tree techniques.” IEEE 27th International Conference on Data Engineering, 2011.

- ➔ RUM conjecture
- ➔ B-Tree
    - Key normalization & friends
    - SB-Tree
    - MDAM
- ➔ Hybrid, Bw-Tree, DPTree
- ➔ Trie
- ➔ Everything else

B-Tree

wB$^+$-Tree

HiKV

WORT

KISS-Tree

UB-Tree

B$^+$-Tree

NV-Tree

Masstree

CDDS-Tree

VAST-Tree

LHAM

B$_{link}$-Tree

FPTree

Skip List

Bw-Tree

FAST

MDAM

DPTree

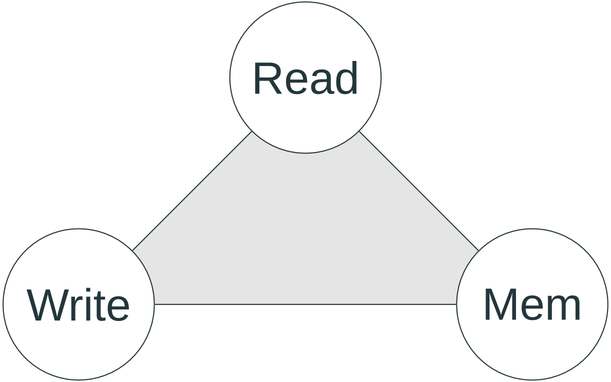
FASTFAIR

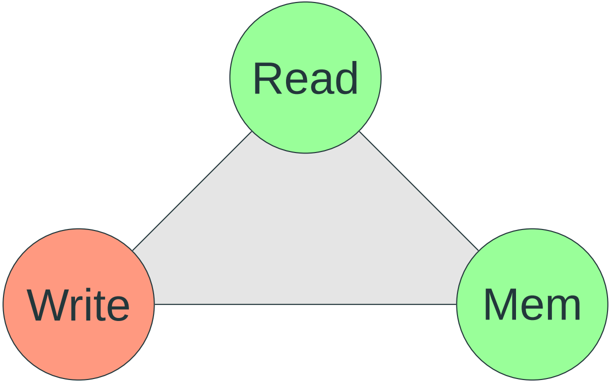
ART

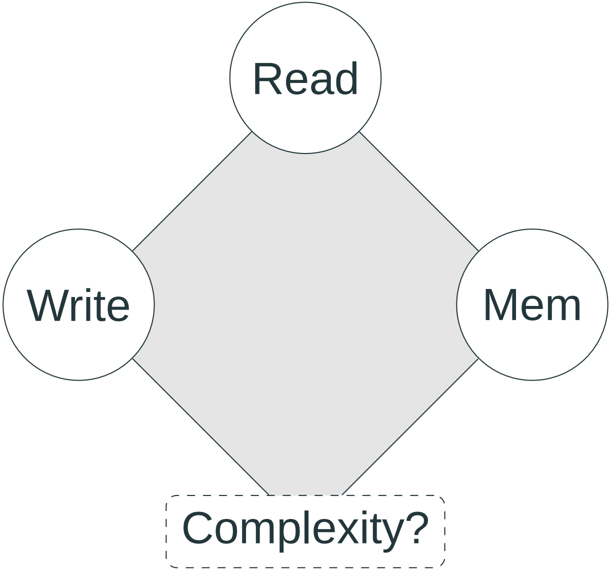
HOT

HV-Tree

Hybrid B$^+$-Tree


Read
Write
Mem


Read
Write
Mem



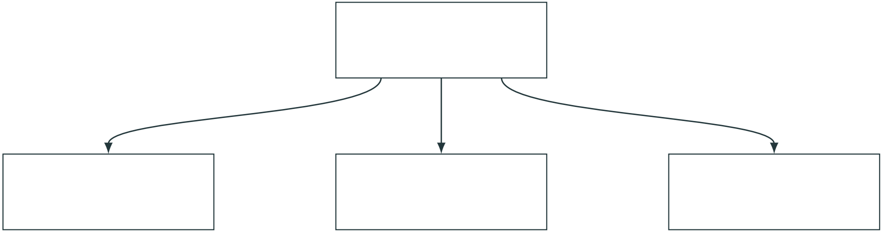
Read

Write

Mem

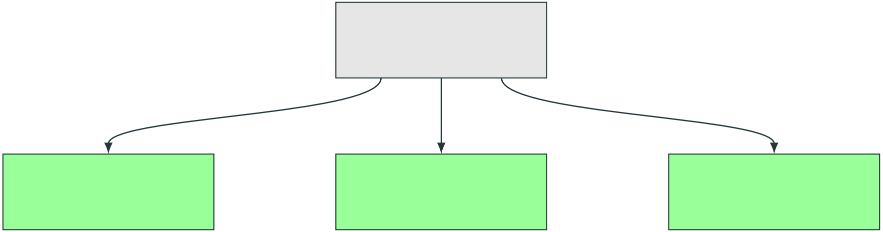
Complexity?

B^+Tree



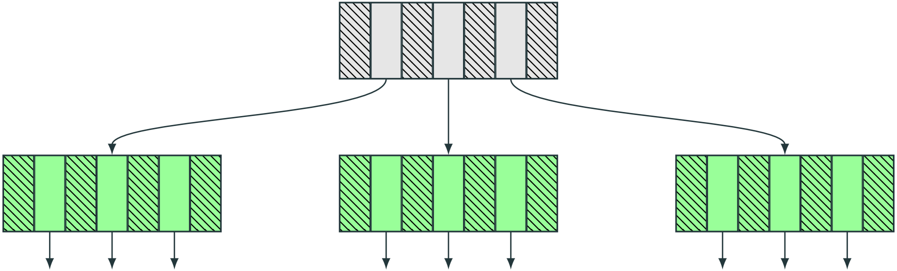
P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

B^+Tree



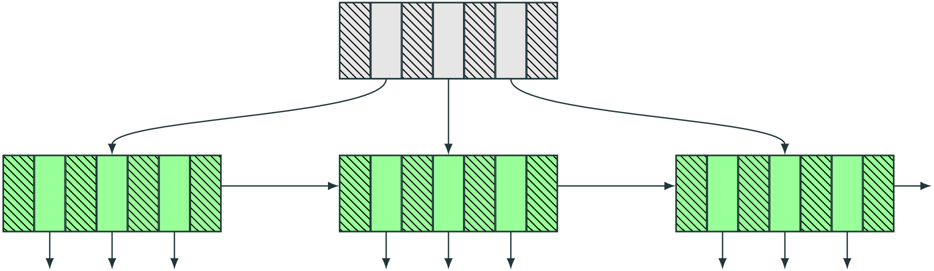
P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

B^{+}Tree



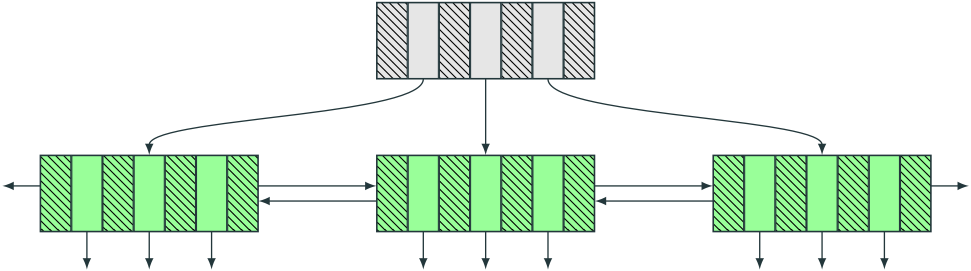
P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

B^+Tree



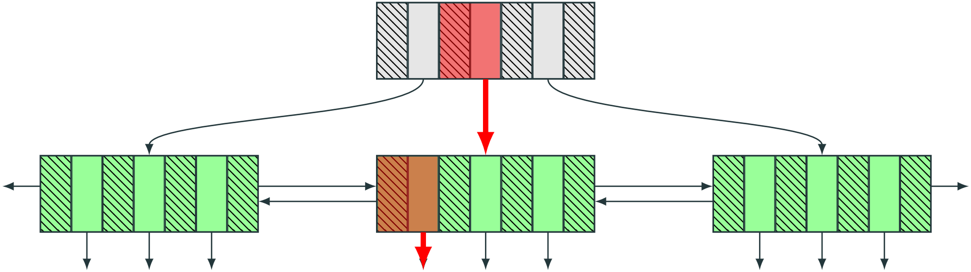
P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

B^+Tree



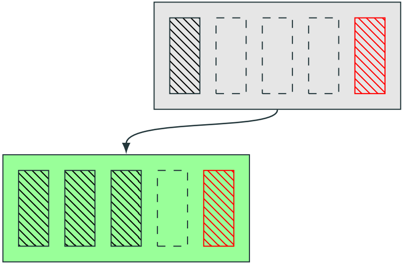
P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

B^+Tree

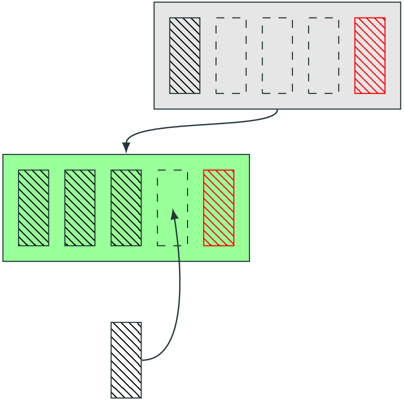


P. Lehman and S. Yao, Efficient Locking for Concurrent Operations on B-Trees, ACM Transactions on Database Systems, Vol 6, No. 4, December 1981, pp 650-670

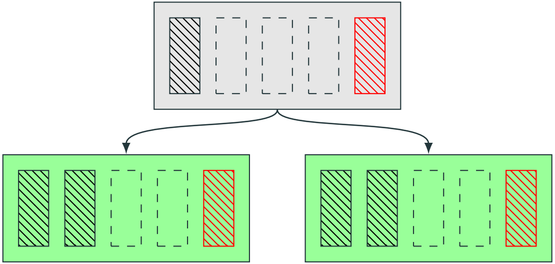
Page split



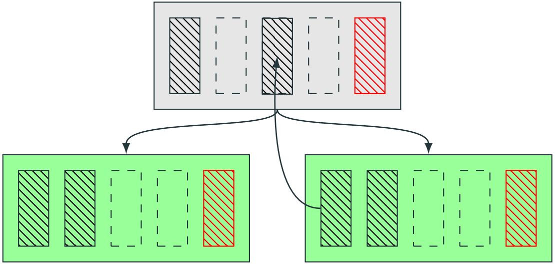
Page split



Page split



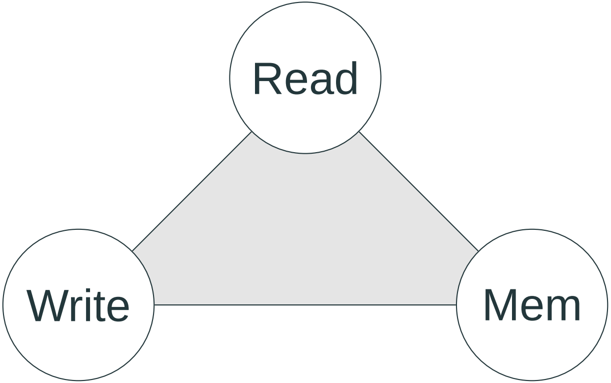
Page split

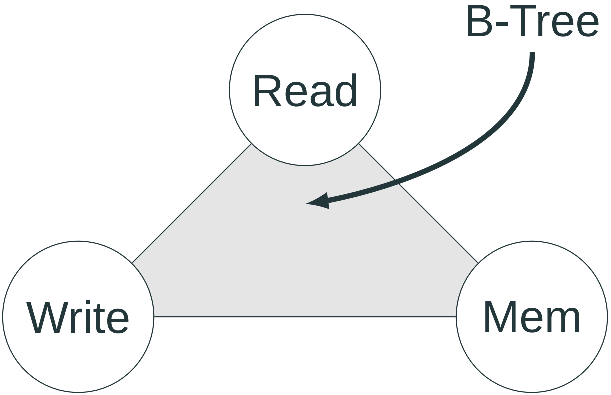


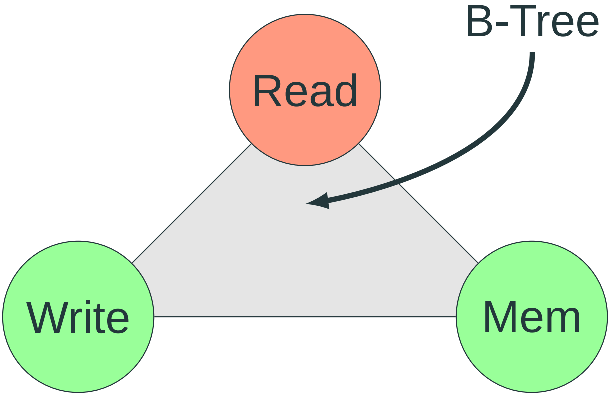
Page merge

”By adding periodic rebuilding of the tree, we obtain a data structure that is theoreticaly superior to standard B-trees in many ways. Our results suggest that rebalancing on deletion no only unnecessary but may be harmful.”

S. Sen and R. E. Tarjan, ”Deletion without rebalancing in multiway search trees.” ISAAC, pp. 832-841, 2009


Read
Write
Mem


B-Tree
Read
Write
Mem


B-Tree
Read
Write
Mem

Key normalization

1	”Dirk”	”Bart”	1	0..01	1	11..00	∅	1	10..10	∅
2	”Todd”	Null	1	0..10	1	01..00	∅	0		
Null	””	Null	0	1	∅	0				

R. C. Singleton, ”An efficient algorithm for sorting with minimal storage”, Communications of the ACM, vol. 12, no.3 pp. 185-186, 1969

Prefix truncation

Smith Jack	01-02-2019
Smith Jane	02-03-2019
Smith James	03-04-2019

Prefix truncation

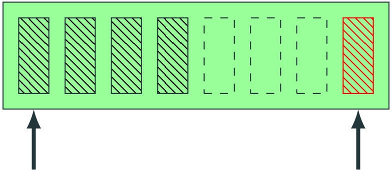
Smith Jack	01-02-2019
Smith Jane	02-03-2019
Smith James	03-04-2019

Prefix truncation

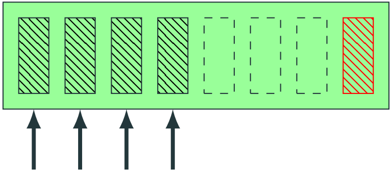
Smith Ja

ck	01-02-2019
ne	02-03-2019
mes	03-04-2019

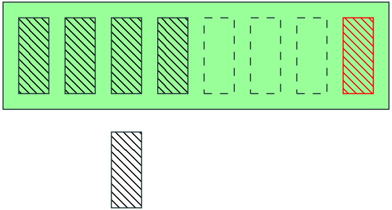
Prefix truncation



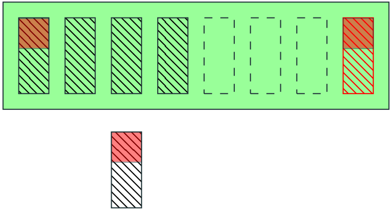
Prefix truncation



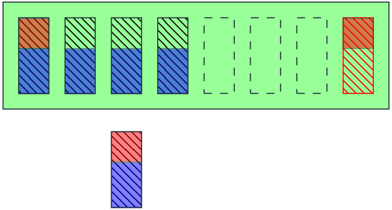
Dynamic prefix truncation



Dynamic prefix truncation



Dynamic prefix truncation



Suffix truncation

Johnson David	04-02-2019
Miller Kevin	01-02-2019
Miller Ned	02-03-2019
Miller Steven	03-04-2019
Smith Jack	01-02-2019

R. Bayer, K. Unterauer, ”Prefix B-trees”, ACM Transactions on Database Systems, vol. 2, no. 1, pp. 11-26, 1977

Suffix truncation

	Johnson David	04-02-2019
	Miller Kevin	01-02-2019
⇨	Miller Ned	02-03-2019
	Miller Steven	03-04-2019
	Smith Jack	01-02-2019

R. Bayer, K. Unterauer, ”Prefix B-trees”, ACM Transactions on Database Systems, vol. 2, no. 1, pp. 11-26, 1977

Suffix truncation

	Johnson David	04-02-2019
	Miller Kevin	01-02-2019
⇨	Miller Ned	02-03-2019
	Miller Steven	03-04-2019
	Smith Jack	01-02-2019

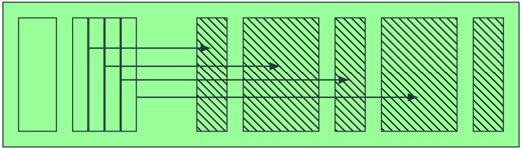
R. Bayer, K. Unterauer, ”Prefix B-trees”, ACM Transactions on Database Systems, vol. 2, no. 1, pp. 11-26, 1977

Suffix truncation

Johnson David	04-02-2019
Miller Kevin	01-02-2019
Miller Ned	02-03-2019
Miller Steven	03-04-2019
Smith Jack	01-02-2019

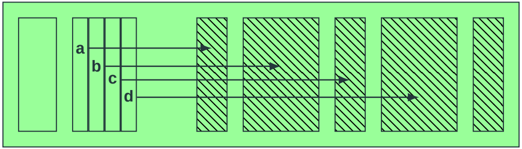
R. Bayer, K. Unterauer, ”Prefix B-trees”, ACM Transactions on Database Systems, vol. 2, no. 1, pp. 11-26, 1977

Indirection vector



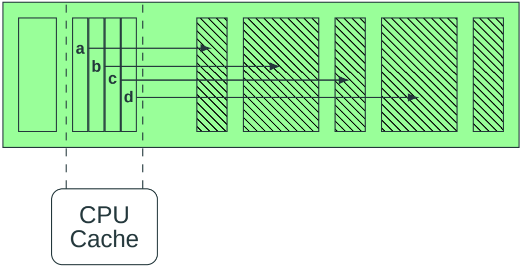
G. Graefe, Per-Ake Larson, ”B-tree indexes and CPU caches”, International Conference on Data Engineering, pp. 349-358, 2001

Indirection vector



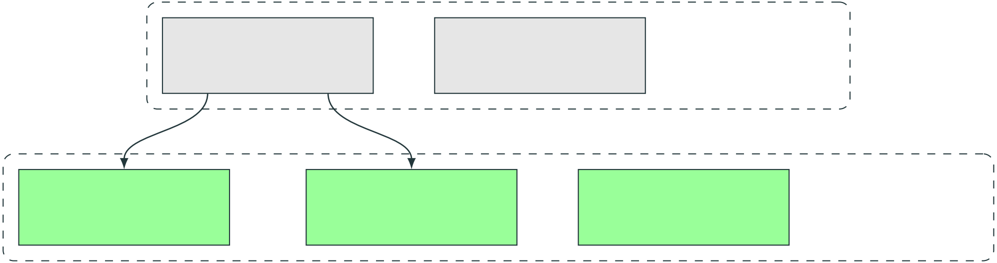
G. Graefe, Per-Ake Larson, ”B-tree indexes and CPU caches”, International Conference on Data Engineering, pp. 349-358, 2001

Indirection vector



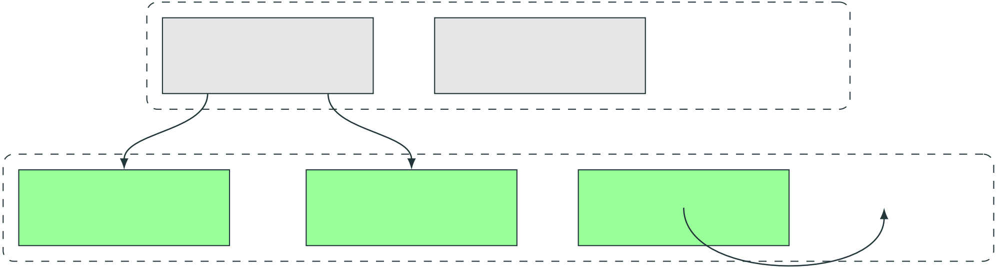
G. Graefe, Per-Ake Larson, ”B-tree indexes and CPU caches”, International Conference on Data Engineering, pp. 349-358, 2001

SB-tree



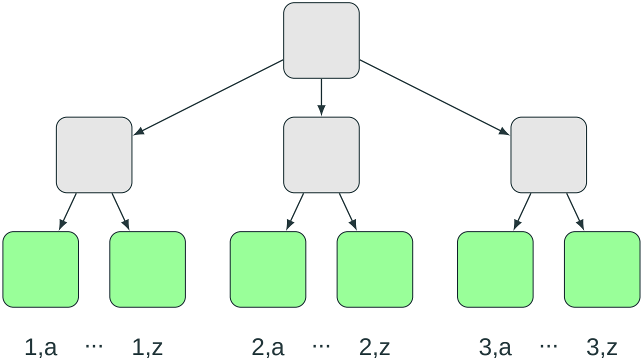
P. E. O’Neil, ”The SB-tree: An index-sequential structure for high performance sequential access”, Acta Informatica, vol. 29, no. 3. pp. 241-265, 1992

SB-tree



P. E. O'Neil, "The SB-tree: An index-sequential structure for high performance sequential access", Acta Informatica, vol. 29, no. 3. pp. 241-265, 1992

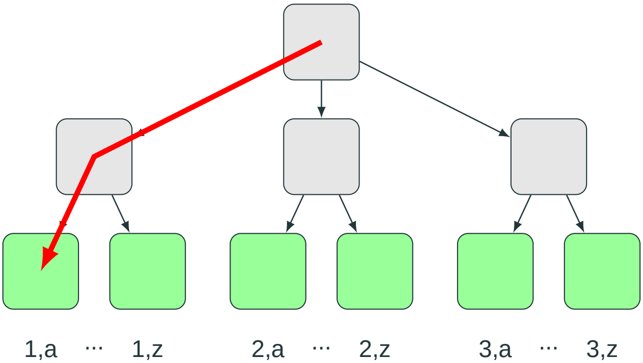
Multidimentional Access Method



1,a ··· 1,z   2,a ··· 2,z   3,a ··· 3,z

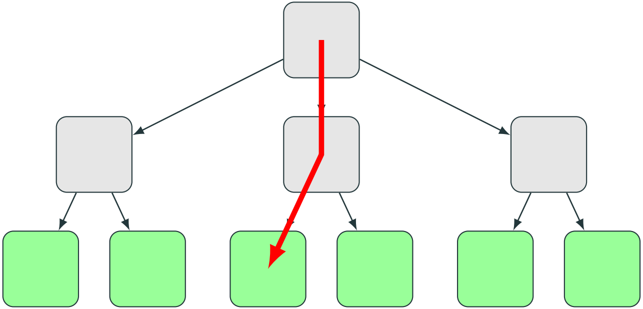
H. Leslie, R. Jain, D. Birdsall, H. Yaghmai, ”Efficient search of multidimentionel B-trees”,
ACM Transactions on Database Systems, vol. 6, no. 4, pp. 650-670, 1995

Multidimentional Access Method



H. Leslie, R. Jain, D. Birdsall, H. Yaghmai, ”Efficient search of multidimentionel B-trees”, ACM Transactions on Database Systems, vol. 6, no. 4, pp. 650-670, 1995

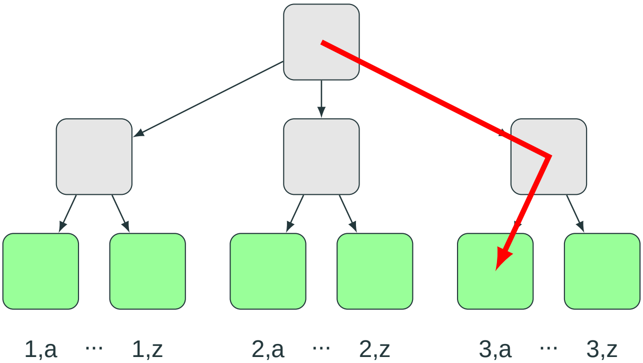
Multidimentional Access Method



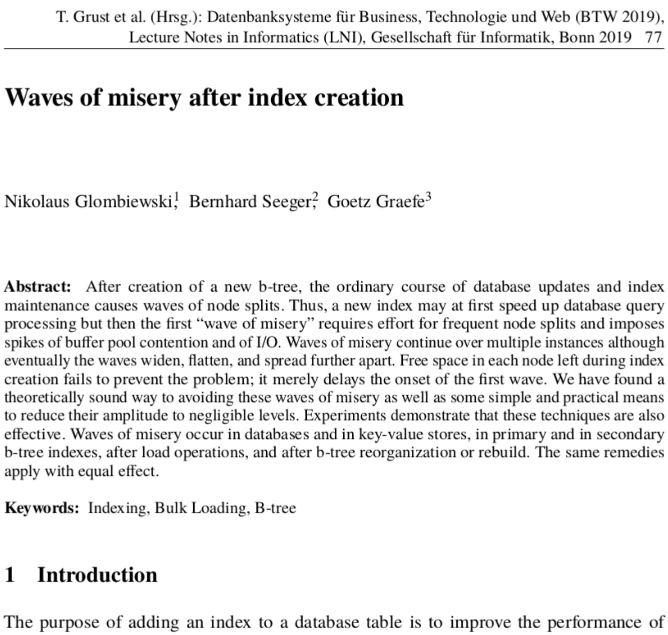
1,a ··· 1,z 2,a ··· 2,z 3,a ··· 3,z

H. Leslie, R. Jain, D. Birdsall, H. Yaghmai, ”Efficient search of multidimentionel B-trees”,
ACM Transactions on Database Systems, vol. 6, no. 4, pp. 650-670, 1995

Multidimentional Access Method



H. Leslie, R. Jain, D. Birdsall, H. Yaghmai, ”Efficient search of multidimentionel B-trees”, ACM Transactions on Database Systems, vol. 6, no. 4, pp. 650-670, 1995



T. Grust et al. (Hrsg.): Datenbanksysteme für Business, Technologie und Web (BTW 2019),
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019 77

Waves of misery after index creation

Nikolaus Glombiewski[1], Bernhard Seeger[2], Goetz Graefe[3]

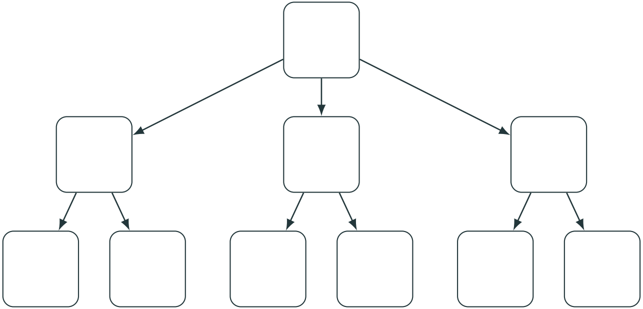
Abstract: After creation of a new b-tree, the ordinary course of database updates and index maintenance causes waves of node splits. Thus, a new index may at first speed up database query processing but then the first "wave of misery" requires effort for frequent node splits and imposes spikes of buffer pool contention and of I/O. Waves of misery continue over multiple instances although eventually the waves widen, flatten, and spread further apart. Free space in each node left during index creation fails to prevent the problem; it merely delays the onset of the first wave. We have found a theoretically sound way to avoiding these waves of misery as well as some simple and practical means to reduce their amplitude to negligible levels. Experiments demonstrate that these techniques are also effective. Waves of misery occur in databases and in key-value stores, in primary and in secondary b-tree indexes, after load operations, and after b-tree reorganization or rebuild. The same remedies apply with equal effect.

Keywords: Indexing, Bulk Loading, B-tree

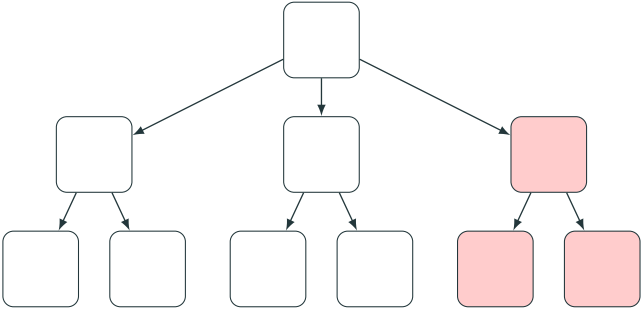
1 Introduction

The purpose of adding an index to a database table is to improve the performance of

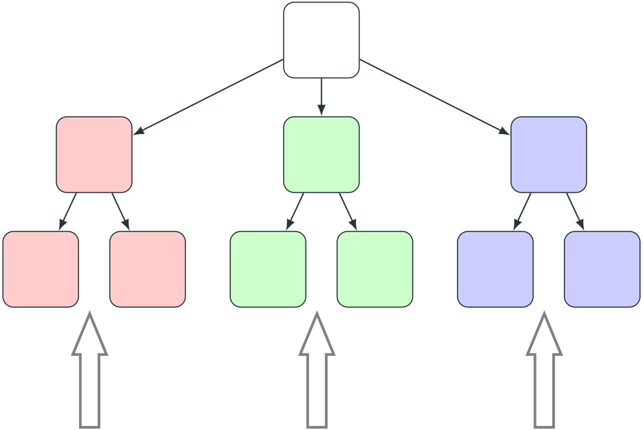
Partitioned B-tree



Partitioned B-tree

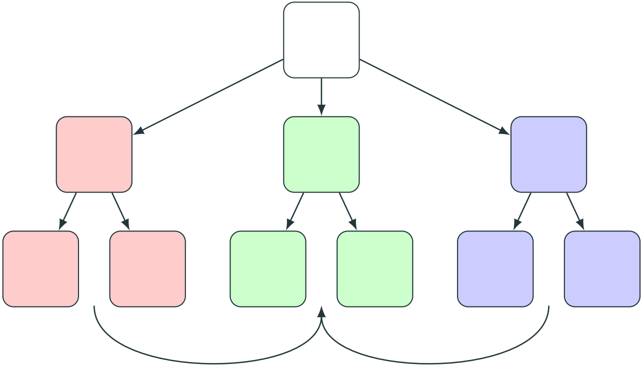


Partitioned B-tree



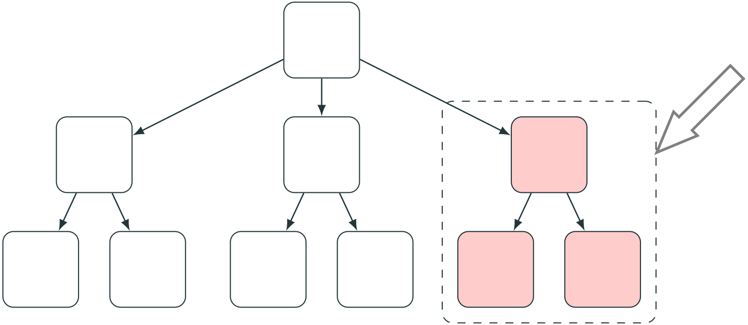
G. Graefe, ”Sorting and indexing with partitioned B-Trees”, Classless Inter Domain Routing, 2003

Partitioned B-tree



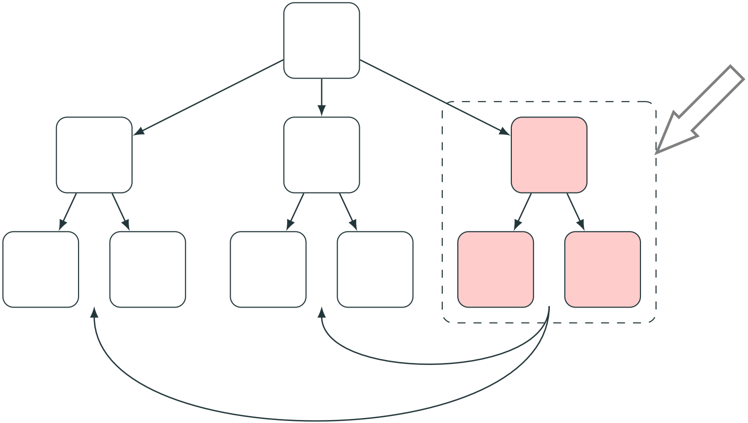
G. Graefe, "Sorting and indexing with partitioned B-Trees", Classless Inter Domain Routing, 2003

Partitioned B-tree



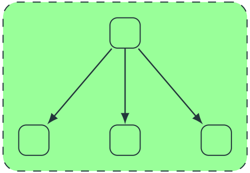
Riegger Christian, Vincon Tobias, Petrov Ilia. (2017). Write-optimized indexing with partitioned b-trees. 296-300.

Partitioned B-tree



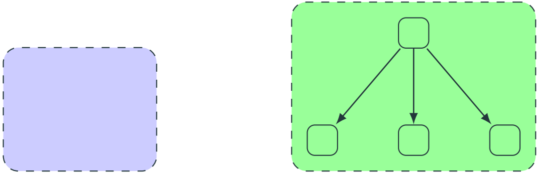
Riegger Christian, Vincon Tobias, Petrov Ilia. (2017). Write-optimized indexing with partitioned b-trees. 296-300.

Hybrid indexes



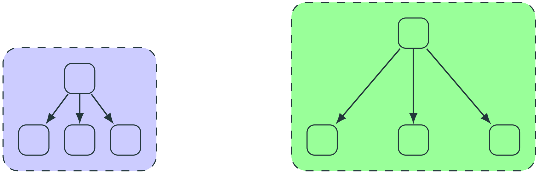
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Hybrid indexes



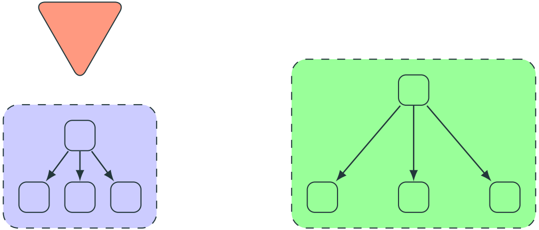
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Hybrid indexes



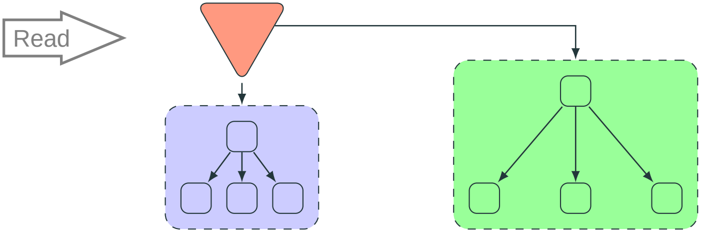
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Hybrid indexes



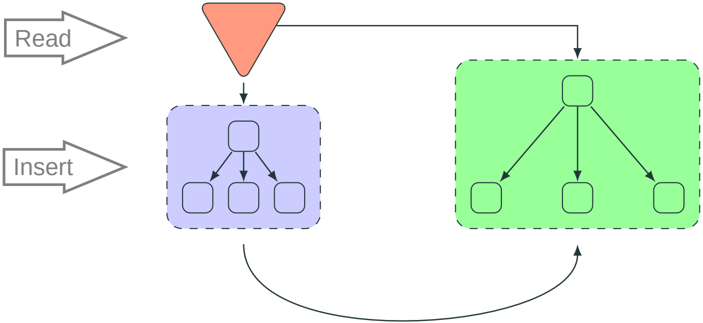
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD ’16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Hybrid indexes



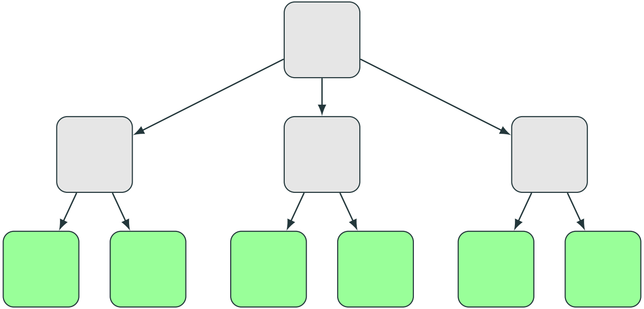
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Hybrid indexes



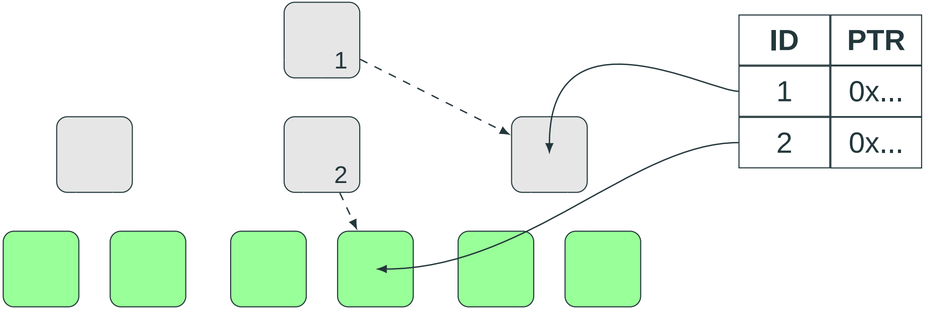
Huanchen Zhang, David G. Andersen, Andrew Pavlo, Michael Kaminsky, Lin Ma, and Rui Shen. 2016. Reducing the Storage Overhead of Main-Memory OLTP Databases with Hybrid Indexes. In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16). Association for Computing Machinery, New York, NY, USA, 1567–1581.

Bw-Tree



Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

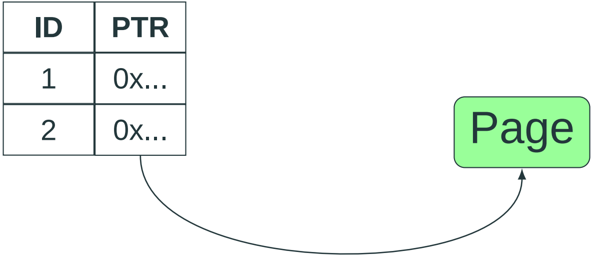
Bw-Tree



ID	PTR
1	0x...
2	0x...

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

Bw-Tree

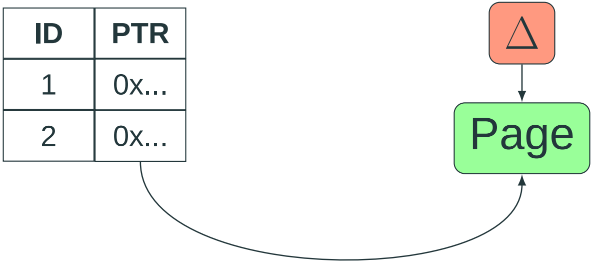


ID	PTR
1	0x...
2	0x...

Page

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

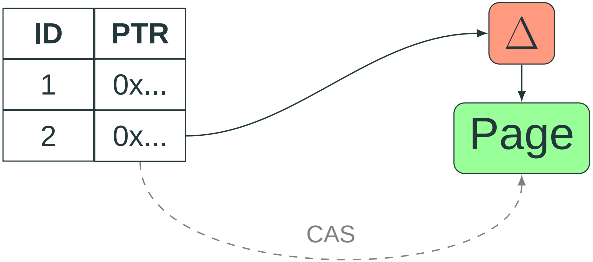
Bw-Tree



ID	PTR
1	0x...
2	0x...

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

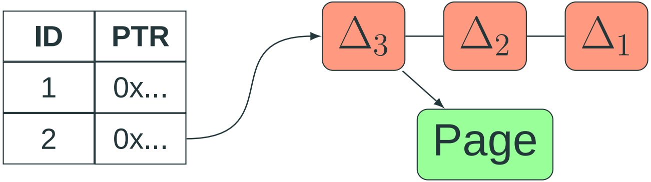
Bw-Tree



ID	PTR
1	0x...
2	0x...

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

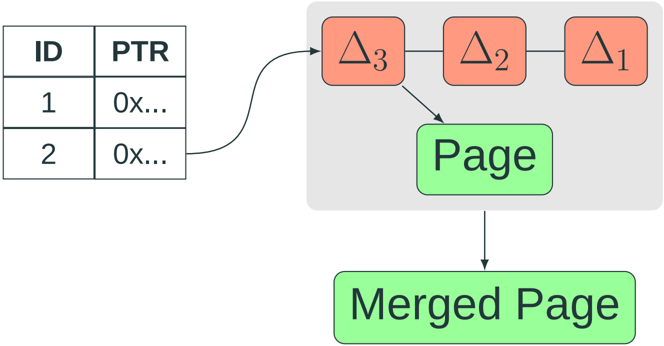
Bw-Tree



ID	PTR
1	0x...
2	0x...

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

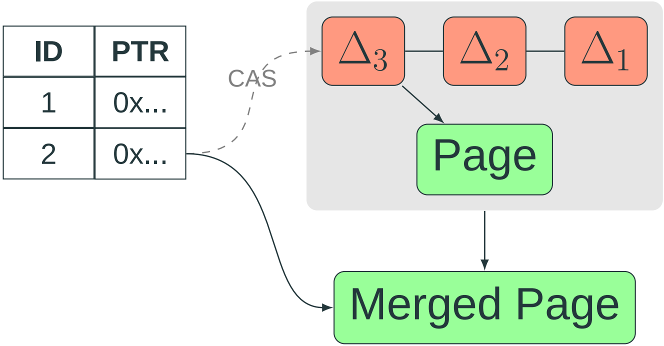
Bw-Tree



ID	PTR
1	0x...
2	0x...

Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

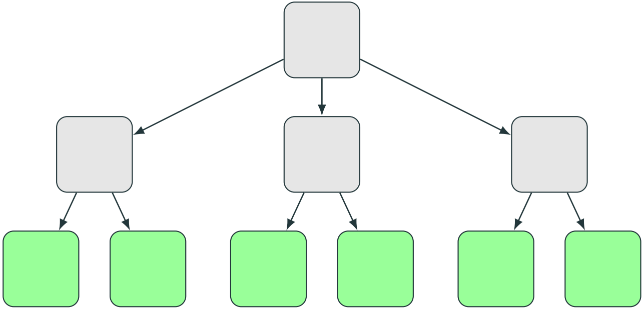
Bw-Tree



ID	PTR
1	0x...
2	0x...

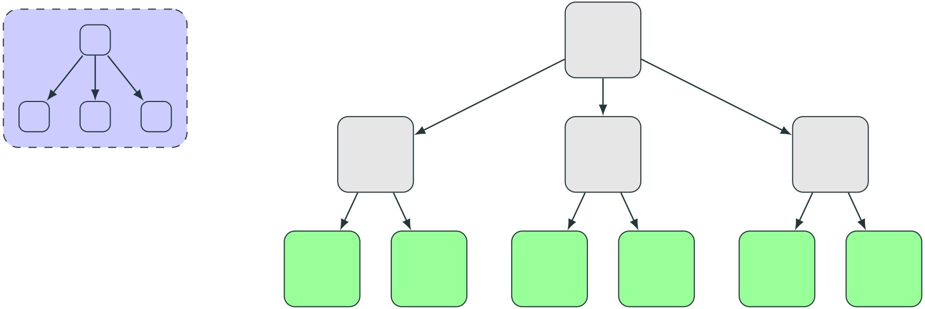
Levandoski Justin, Lomet David, Sengupta Sudipta. (2012). The Bw-Tree: A B-tree for New Hardware Platforms. Proceedings - International Conference on Data Engineering.

DPTree



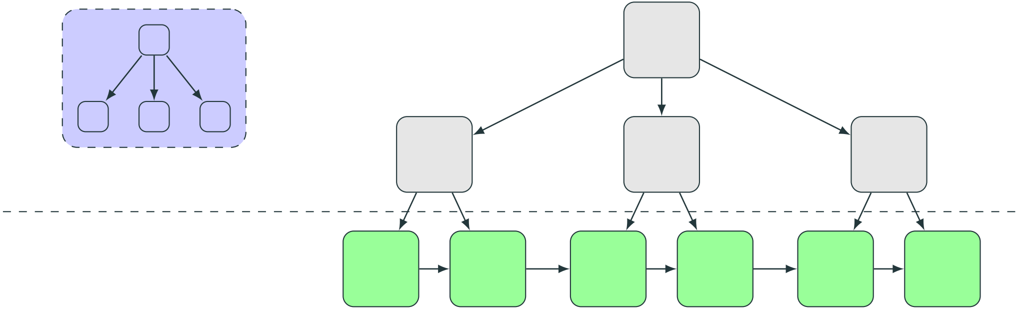
Zhou Xinjing, Shou Lidan, Chen Ke, Hu Wei, Chen Gang. (2019). DPTree: differential indexing for persistent memory. Proceedings of the VLDB Endowment.

DPTree



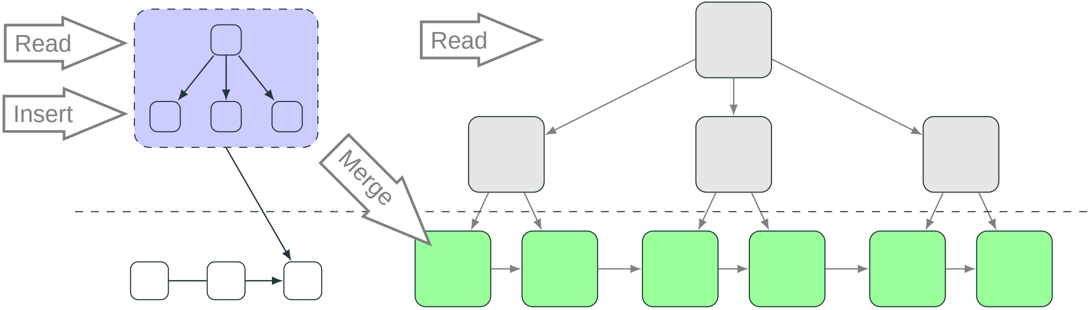
Zhou Xinjing, Shou Lidan, Chen Ke, Hu Wei, Chen Gang. (2019). DPTree: differential indexing for persistent memory. Proceedings of the VLDB Endowment.

DPTree



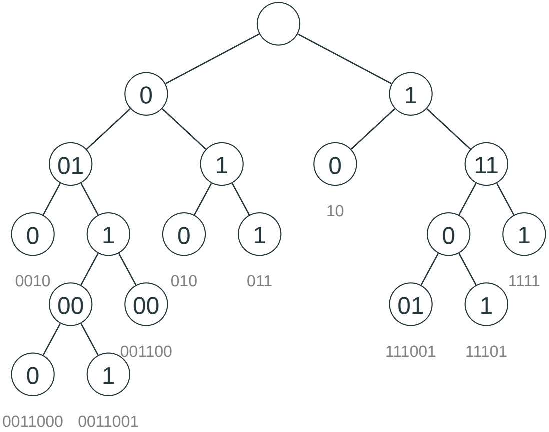
Zhou Xinjing, Shou Lidan, Chen Ke, Hu Wei, Chen Gang. (2019). DPTree: differential indexing for persistent memory. Proceedings of the VLDB Endowment.

DPTree

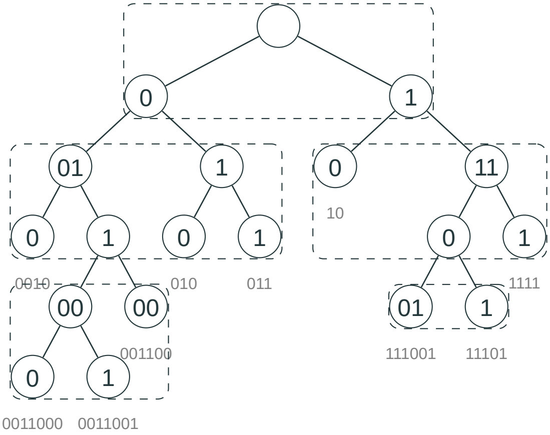


Zhou Xinjing, Shou Lidan, Chen Ke, Hu Wei, Chen Gang. (2019). DPTree: differential indexing for persistent memory. Proceedings of the VLDB Endowment.

Trie

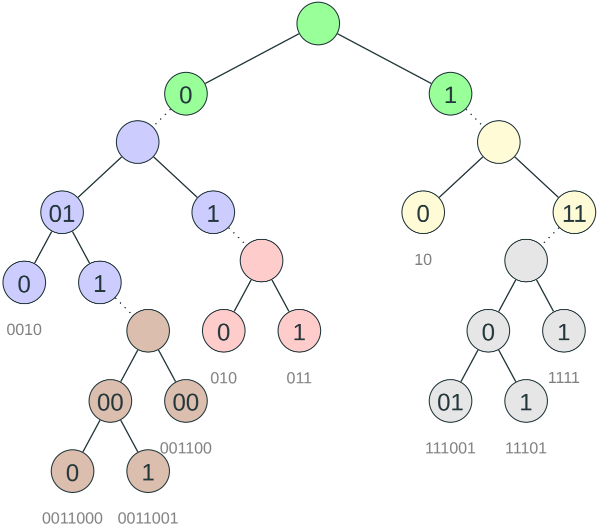

0
1
01
1
0
11
10
0
1
0
1
0
1
0010
010
011
1111
00
00
01
1
001100
111001
11101
0
1
0011000
0011001

Trie



Leis V., Kemper Alfons, Neumann Thomas. (2013). The adaptive radix tree: ARTful indexing for main-memory databases. Proceedings - International Conference on Data Engineering. 38-49.

Trie



Binna Robert, Zangerle Eva, Pichl Martin, Specht Guenther, Leis Viktor. (2018). HOT: A Height Optimized Trie Index for Main-Memory Database Systems. 521-534.

Is that all?

@Unzucht

➔ Buffered (Lazy) B-Tree

- ➔ Buffered (Lazy) B-Tree
- ➔ Cache conscious B-Tree

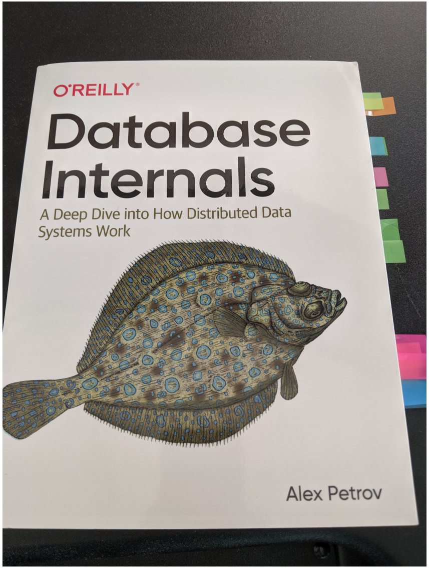
- ➔ Buffered (Lazy) B-Tree
- ➔ Cache conscious B-Tree
- ➔ Learned indexes

- ➔ Buffered (Lazy) B-Tree
- ➔ Cache conscious B-Tree
- ➔ Learned indexes
- ➔ UB-Tree

- ➔ Buffered (Lazy) B-Tree
- ➔ Cache conscious B-Tree
- ➔ Learned indexes
- ➔ UB-Tree
- ➔ Skip List

- ➔ Buffered (Lazy) B-Tree
- ➔ Cache conscious B-Tree
- ➔ Learned indexes
- ➔ UB-Tree
- ➔ Skip List
- ➔ LSM Tree

- Buffered (Lazy) B-Tree
- Cache conscious B-Tree
- Learned indexes
- UB-Tree
- Skip List
- LSM Tree
- ...



O'REILLY®

Database Internals

A Deep Dive into How Distributed Data Systems Work

Alex Petrov

Questions?

- 🐦 @erthalion
- ✉ dmitrii.dolgov at zalando dot de
- ✉ 9erthalion6 at gmail dot com