START – Self-Tuning Adaptive Radix Tree

Philipp Fent, Michael Jungmair, Andreas Kipf, Thomas Neumann

Technische Universität München

April 20, 2020
Learned indexes

- Small and efficient
- Adapt to data distribution
- Fast for read-only workloads
Learned indexes
- Small and efficient
- Adapt to data distribution
- Fast for read-only workloads

ART – Adaptive Radix Tree
- Used in practice
- Well understood
- Fast for various workloads
- But: limited adaption to data distribution
- and slower than a well-trained machine learning model
Learned indexes
- Small and efficient
- Adapt to data distribution
- Fast for read-only workloads

START Self Tuning ART
- Bridges the gap
- Adapts to data: 85% faster!
- Keep robustness

ART – Adaptive Radix Tree
- Used in practice
- Well understood
- Fast for various workloads
- But: limited adaption to data distribution
- and slower than a well-trained machine learning model
Self-Tuning ART

Byte 1

Byte 2

Byte 3

Byte 4

ART

START

Philipp Fent et al. START – Self-Tuning Adaptive Radix Tree
Self-Tuning ART

Byte 1

Byte 2

Byte 3

Byte 4

ART

cost model

optimizer

introduce multilevel nodes

START
Multilevel Nodes

Rewired memory nodes

- Dense regions with many keys
- Subtrees of full nodes
- Keep data continuous
Multilevel Nodes

Rewired memory nodes
- Dense regions with many keys
- Subtrees of full nodes
- Keep data continuous

Sparse regions
- Already use path compression
- Still: improve common prefixes
- Reduce chains of small nodes
Rewired Memory Nodes

Node256

256 children

...
Rewired Memory Nodes

Node256
256 children

Node16M
64K Node256
Rewired Memory Nodes

| Memory File | Virtual Pages | Physical Pages |
|-------------|---------------|----------------|
| 'a'         | 'b'           | 'b'            |
|             |               | (φ)            |
| (φ)         | 'b'           |                 |
| (φ)         |                 | (φ)            |

Note: The diagram illustrates the rewiring of memory nodes between virtual and physical pages.
Rewired Memory Nodes

virtual pages

physical pages

page 1

page 2

page 3

page 4

∅
Rewired Memory Nodes

virtual pages

physical pages

page 1

page 2

page 3

page 4

∅
Rewired Memory Nodes

virtual pages

physical pages

page 1

page 2

page 3

page 4

∅
Rewired Memory Nodes

virtual pages

physical pages

page 1

access(19)

page 2

page 3

page 4

∅
Rewired Memory Nodes

virtual pages

physical pages

page 1

page 2

page 3

page 4

access(11)
Multilevel Node4

- Helps reduce sparse affix nodes
- Use all 64 Bytes of a cacheline
Cost Model for Node Lookup

- Minimize the average child lookup cost
- Consider individual node cost

| [ns/lookup]     | Levels |
|-----------------|--------|
| Node4           | 1      |
| Node16          | 1      |
| Node48          | 1      |
| Node256         | 1      |
| Rewired64K      | 2      |
| Rewired16M      | 3      |
| MultiNode4      | 2-4    |
Cost Model for Node Lookup

- Minimize the average child lookup cost
- Consider individual node cost

| Node       | Levels | Cached | Header | Cached | Uncached |
|------------|--------|--------|--------|--------|----------|
| Node4      | 1      | 7      |        | 7      | 68       |
| Node16     | 1      | 5      |        | 77     | 162      |
| Node48     | 1      | 2      |        | 165    | 168      |
| Node256    | 1      | 2      |        | 88     | 92       |
| Rewired64K | 2      | 6      |        | 87     | 162      |
| Rewired16M | 3      | 6      |        | 88     | 165      |
| MultiNode4 | 2-4    | 6      |        | 6      | 68       |
Bottom-up Optimization
Bottom-up Optimization

25
[130, 80, ...]

20
[60, ...]

5
[20, ...]

10
[30, ...]

5
[25, ...]
Bottom-up Optimization

![Diagram of a binary tree with leaf nodes labeled with intervals and internal nodes labeled with 2 and 15. The intervals are: 25 [130, 80, ...], 20 [60, ...], 5 [20, ...], 10 [30, ...], 5 [25, ...].]
Bottom-up Optimization

1-level: \[25 + 15 \]

2-level: \[\begin{align*}
20 & \quad \{60, \ldots\} \\
25 & \quad \{130, 80, \ldots\} \\
40 \cdot 2 + 215 & \\
130 + 85 & \\
80 + 55 & \\
215 & \quad \{255, 130, 85, 55, \ldots\}
\end{align*}\]
Performance

https://learned.systems/sosd

Datasets

Latency [ns / op]

Indexes

- ART
- START
- RS
- RMI

Philipp Fent et al.

START – Self-Tuning Adaptive Radix Tree
Conclusion

- START – Best of both worlds?
- Adapt to real-world data distribution
- Still with robust underpinning of ART
- Inserts still possible and efficient
- But: might degrade multilevel nodes

[github.com/jungmair/START](https://github.com/jungmair/START)

[fent@in.tum.de](mailto:fent@in.tum.de)