Friedrich-Alexander-Universität
Erlangen-Nürnberg

Write-Aware Timestamp Tracking:

Effective and Efficient Page Replacement for Modern Hardware

Demian Vöhringer[1] **Viktor Leis**[2]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg
[2]Technische Universität München

August 29, 2023 – VLDB

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

4 Goals of Page Replacement

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

4 Goals of Page Replacement

1. **Replacement Effectiveness**

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

4 Goals of Page Replacement

1. **Replacement Effectiveness**
2. **Write Awareness**

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

4 Goals of Page Replacement

1. **Replacement Effectiveness**
2. **Write Awareness**
3. **CPU Efficiency**

Revisit Page Replacement

Challenges

- **Storage:** HDD $\rightarrow$ SSD
- **CPU:** single-core $\rightarrow$ multi-core

4 Goals of Page Replacement

1. **Replacement Effectiveness**
2. **Write Awareness**
3. **CPU Efficiency**
4. **Multi-Core Scalability**

Revisit Page Replacement

Challenges

- **Storage:** HDD → SSD
- **CPU:** single-core → multi-core

4 Goals of Page Replacement

1. **Replacement Effectiveness**
2. **Write Awareness**
3. **CPU Efficiency**
4. **Multi-Core Scalability**

Used Strategies

- **LRU:** DB2, Oracle, InnoDB (MySQL / MariaDB), WiredTiger (MongoDB)
- **LRU2:** SQL Server
- **Clock-Sweep:** PostgreSQL

Tested Strategies

	Replacement Effectiveness	Write Awareness	CPU Efficiency	Multi-Core Scalability
random	$--$	no	$++$	$++$
CLOCK	$-$	no	$+$	$+$
LeanEvict	$-$	no	$+$	$+$
LRU2	$+$	no	$++$	$+$
LRU	$-$	no	$-$	$-$
CFLRU	$-$	yes	$-$	$-$
LRU_WSR	$-$	yes	$-$	$-$
Hyperbolic	$\sim$	no	$++$	$++$
ARC	$+$	no	$-$	$-$

Timestamp Tracking

more information

Timestamp Tracking

more information $\rightarrow$ more insights

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

Index[i]	1	2	3	4	...
Timestamp[t_i]	-	-	-	-	...

Tracking Array (per page)

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

i	1	2	3	4	...
Access at $t = 0$	0	-	-	-	...

Tracking Array (per page)

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

i	1	2	3	4	...
Access at $t = 8$	8	0	-	-	...

Tracking Array (per page)

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

i	1	2	3	4	...
Access at $t = 15$	15	8	0	-	...

Tracking Array (per page)

Timestamp Tracking

more information $\rightarrow$ more insights $\rightarrow$ better strategy

i	1	2	3	4	...
Access at $t = 42$	42	15	8	0	...

Tracking Array (per page)

How to use it to evict a page?

Page Value [PV]: high for good pages, low for bad ones

How to use it to evict a page?

Page Value [PV]: high for good pages, low for bad ones

approximate current access frequency

How to use it to evict a page?

Page Value [PV]: high for good pages, low for bad ones

approximate current access frequency

$$PV_{LRU}(t_{now}) = \frac{1}{age} = \frac{1}{t_{now} - t_1}$$

$$PV_{LRU2}(t_{now}) = \frac{1}{t_{now} - t_2}$$

$$PV_{LFU}(t_{now}) = len(\text{Tracking Array})$$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

$$SF_1(t_{now}) := \frac{1}{t_{now} - t_1} \qquad SF_2(t_{now}) := \frac{2}{t_{now} - t_2}$$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

$$SF_1(t_{now}) := \frac{1}{t_{now} - t_1} \qquad SF_2(t_{now}) := \frac{2}{t_{now} - t_2}$$

i	1	2	3	4
t_i	42	15	8	0

Evaluated at $t_{now} := 50$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

$$SF_1(t_{now}) := \frac{1}{t_{now} - t_1} \qquad SF_2(t_{now}) := \frac{2}{t_{now} - t_2}$$

i	1	2	3	4
t_i	42	15	8	0
$t_{now} - t_i$	8	35	42	50

Evaluated at $t_{now} := 50$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

$$SF_1(t_{now}) := \frac{1}{t_{now} - t_1} \qquad SF_2(t_{now}) := \frac{2}{t_{now} - t_2}$$

i	1	2	3	4
t_i	42	15	8	0
$t_{now} - t_i$	8	35	42	50
$SF_i(t_{now})$	1/8	2/35	3/42	4/50

Evaluated at $t_{now} := 50$

Subfrequencies

$$Subfrequency[SF]_i(t_{now}) := \frac{i}{t_{now} - t_i}$$

$$SF_1(t_{now}) := \frac{1}{t_{now} - t_1} \qquad SF_2(t_{now}) := \frac{2}{t_{now} - t_2}$$

i	1	2	3	4
t_i	42	15	8	0
$t_{now} - t_i$	8	35	42	50
$SF_i(t_{now})$	1/8	2/35	3/42	4/50
$SF_i(t_{now}) \approx$	0.13	0.06	0.07	0.08

Evaluated at $t_{now} := 50$

Page Value

$$Page\ Value[PV](t_{now}) := \max_i SF_i(t_{now})$$

Page Value

$$Page\ Value[PV](t_{now}) := \max_i SF_i(t_{now})$$

i	1	2	3	4
$SF_i(50) \approx$	0.13	0.06	0.07	0.08

$$PV(50) = SF_1(50) \approx 0.13$$

Compare again against LRU and LFU

$$PV_{LRU}(t_{now}) = \frac{1}{age} = \frac{1}{t_{now} - t_1}$$

$$PV_{LRU2}(t_{now}) = \frac{1}{t_{now} - t_2}$$

$$PV_{LFU}(t_{now}) = len(\text{Tracking Array})$$

$$PV_{WATT}(t_{now}) = \max_i \frac{i}{t_{now} - t_i}$$

Compare again against LRU and LFU

$$PV_{LRU}(t_{now}) = \frac{1}{age} = \frac{1}{t_{now} - t_1} = SF_1(t_{now})$$

$$PV_{LRU2}(t_{now}) = \frac{1}{t_{now} - t_2}$$

$$PV_{LFU}(t_{now}) = len(\text{Tracking Array})$$

$$PV_{WATT}(t_{now}) = \max_i \frac{i}{t_{now} - t_i}$$

Compare again against LRU and LFU

$$PV_{LRU}(t_{now}) = \frac{1}{age} = \frac{1}{t_{now} - t_1} = SF_1(t_{now})$$

$$PV_{LRU2}(t_{now}) = \frac{1}{t_{now} - t_2} \equiv SF_2(t_{now})$$

$$PV_{LFU}(t_{now}) = len(\text{Tracking Array})$$

$$PV_{WATT}(t_{now}) = \max_i \frac{i}{t_{now} - t_i}$$

Compare again against LRU and LFU

$$PV_{LRU}(t_{now}) = \frac{1}{age} = \frac{1}{t_{now} - t_1} = SF_1(t_{now})$$

$$PV_{LRU2}(t_{now}) = \frac{1}{t_{now} - t_2} \equiv SF_2(t_{now})$$

$$PV_{LFU}(t_{now}) = len(\text{Tracking Array}) = \max_i \frac{i}{const} =: SF^*_{max}(t_{now})$$

$$PV_{WATT}(t_{now}) = \max_i \frac{i}{t_{now} - t_i}$$

Refinements

↯Write Awareness

↯Memory Consumption

↯Bursts

↯Scans and Onetime Accesses

↯Finding Eviction Candidates

Refinements

ϟWrite Awareness

$$PV(t_{now}) := PV^*_{access}(t_{now}) + write\_weight \cdot PV^*_{write}(t_{now})$$

ϟMemory Consumption

ϟBursts

ϟScans and Onetime Accesses

ϟFinding Eviction Candidates

Refinements

ϟWrite Awareness

$$PV(t_{now}) := PV^*_{access}(t_{now}) + write\_weight \cdot PV^*_{write}(t_{now})$$

ϟMemory Consumption

limit Tracking Array size

ϟBursts

ϟScans and Onetime Accesses

ϟFinding Eviction Candidates

Refinements

ϟWrite Awareness

$$PV(t_{now}) := PV^*_{access}(t_{now}) + write\_weight \cdot PV^*_{write}(t_{now})$$

ϟMemory Consumption

limit Tracking Array size

ϟBursts

group accesses to epochs

ϟScans and Onetime Accesses

ϟFinding Eviction Candidates

Refinements

ϟWrite Awareness

$$PV(t_{now}) := PV^*_{access}(t_{now}) + write\_weight \cdot PV^*_{write}(t_{now})$$

ϟMemory Consumption

limit Tracking Array size

ϟBursts

group accesses to epochs

ϟScans and Onetime Accesses

dampening of $SF_1(t_{now}) = SF^*_1(t_{now})/10$

ϟFinding Eviction Candidates

Refinements

↯Write Awareness

$$PV(t_{now}) := PV^*_{access}(t_{now}) + write\_weight \cdot PV^*_{write}(t_{now})$$

↯Memory Consumption

limit Tracking Array size

↯Bursts

group accesses to epochs

↯Scans and Onetime Accesses

dampening of $SF_1(t_{now}) = SF^*_1(t_{now})/10$

↯Finding Eviction Candidates

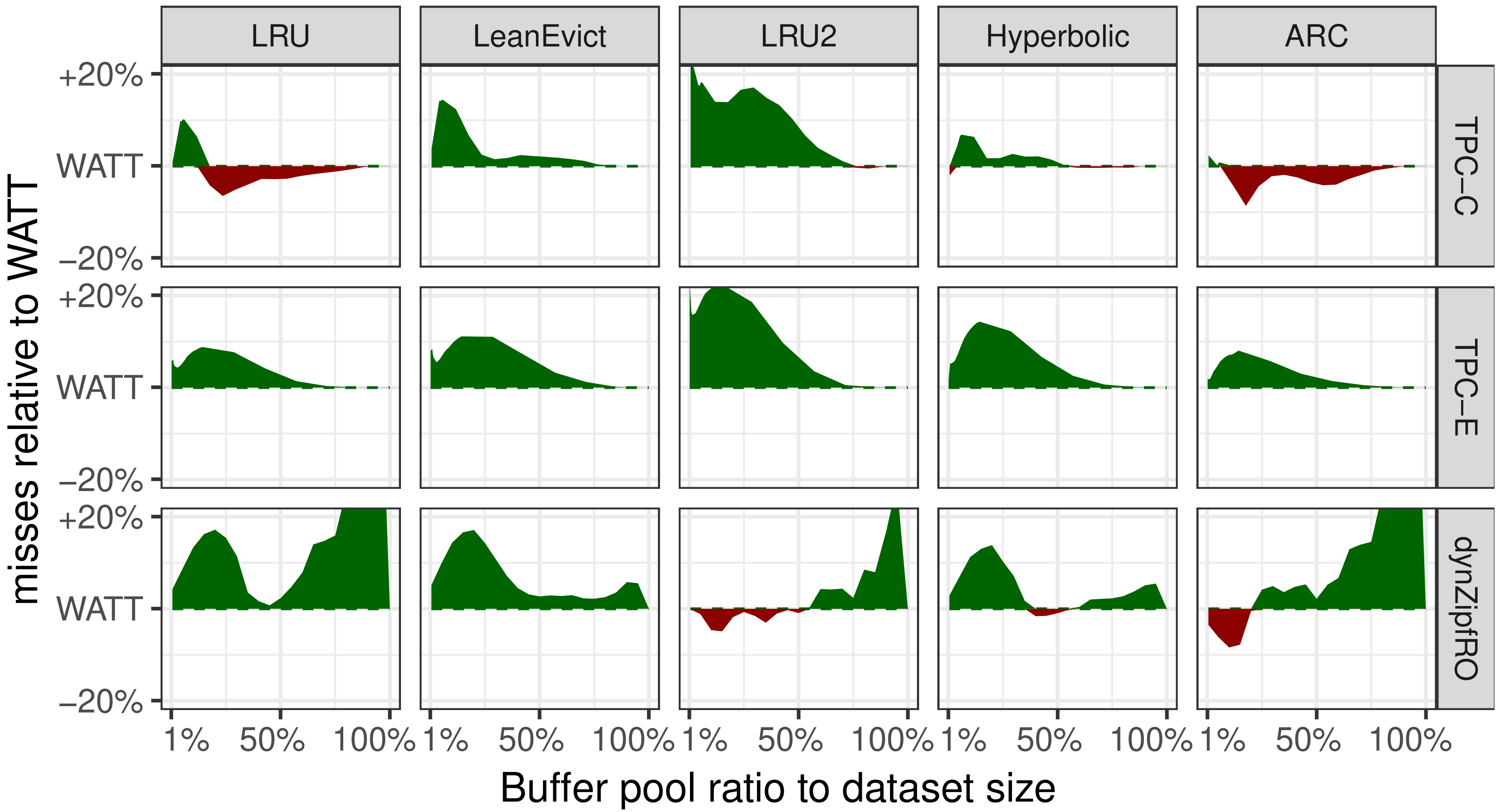
Random Sampling

Simulation-based Evaluation

Replacement Effectiveness (Goal 1)

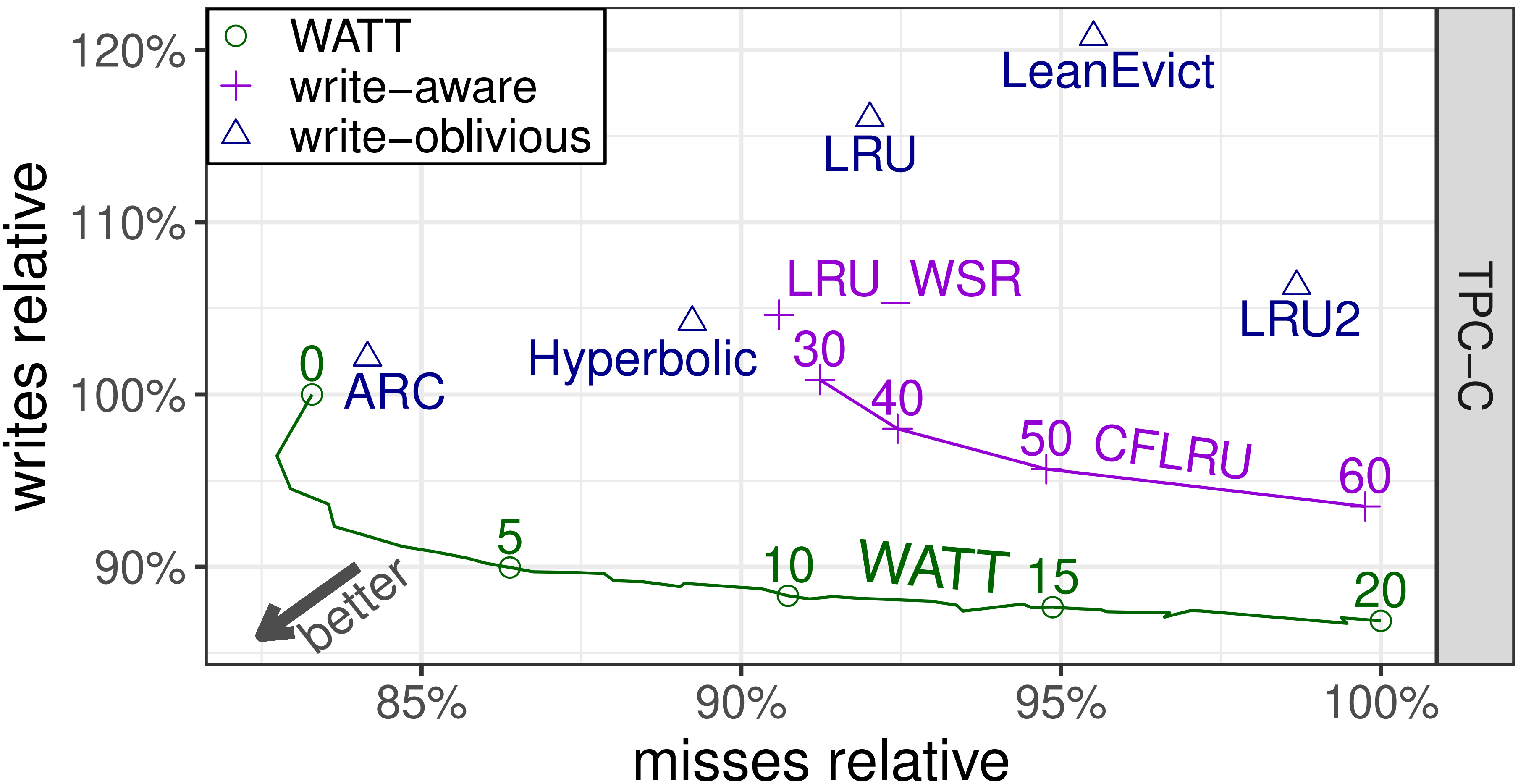
Simulation-based Evaluation

Replacement Effectiveness (Goal 1)



Simulation-based Evaluation

Write Awareness (Goal 2)



System Evaluation

Implementation

- implemented in leanstore

System Evaluation

Implementation

- implemented in leanstore
- lock free access to Tracking Array

System Evaluation

Implementation

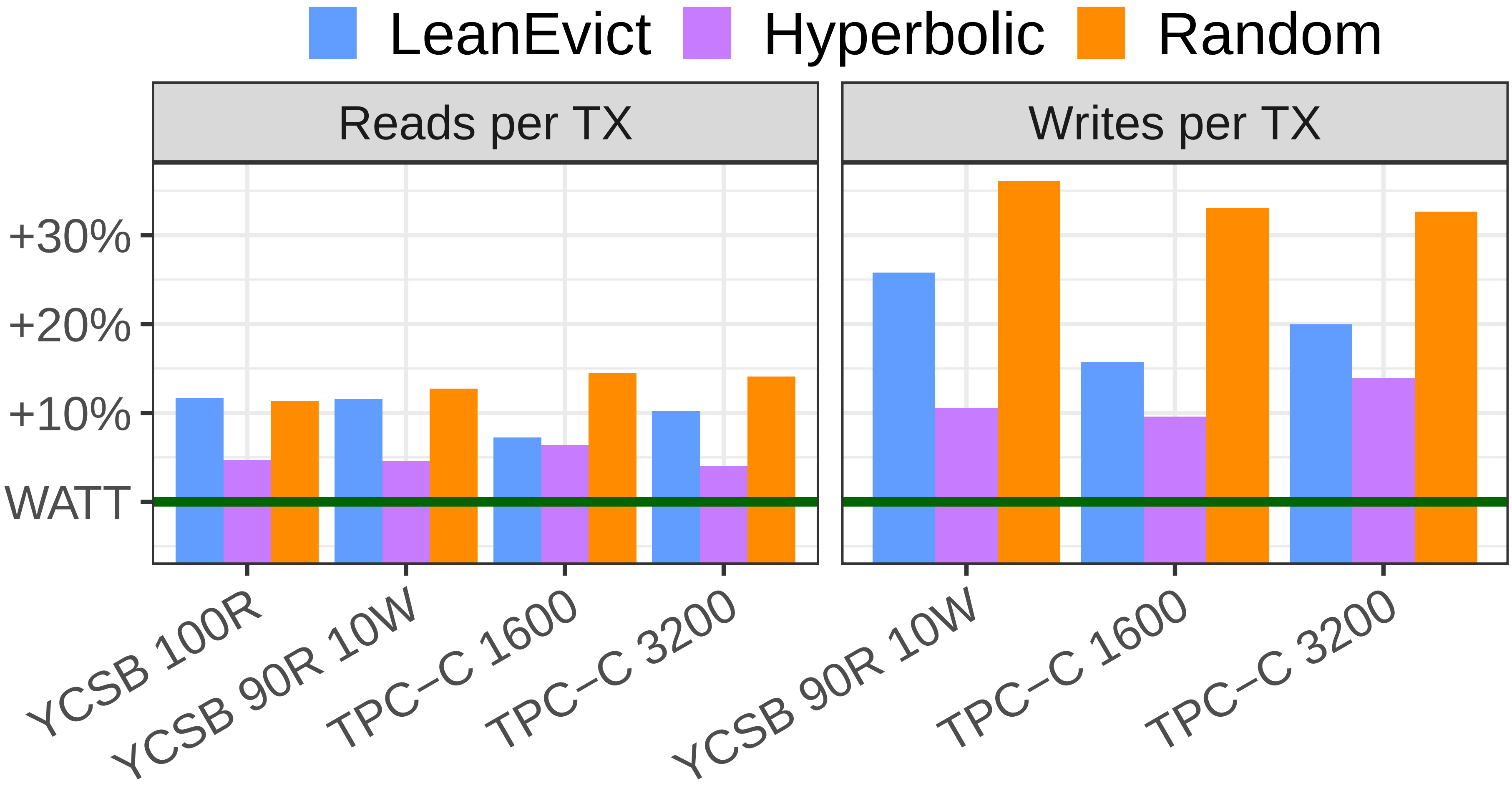
- implemented in leanstore
- lock free access to Tracking Array
- SIMD vectorization to calculate Page Value

System Evaluation

Implementation

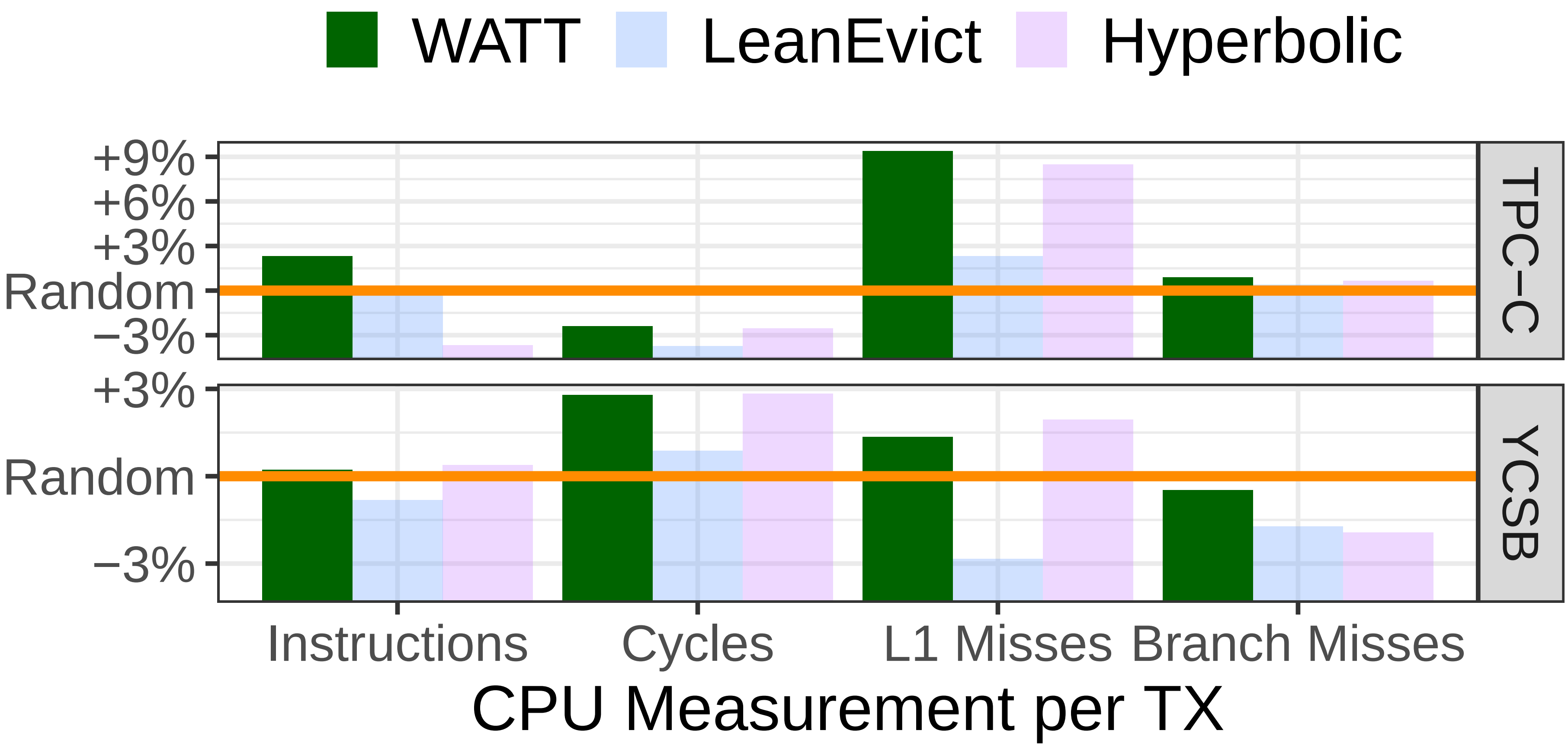
- implemented in leanstore
- lock free access to Tracking Array
- SIMD vectorization to calculate Page Value
- Prefetching of Tracking Arrays for Page Value calculation

Replacement Effectiveness (Goal 1) Write Awareness (Goal 2)



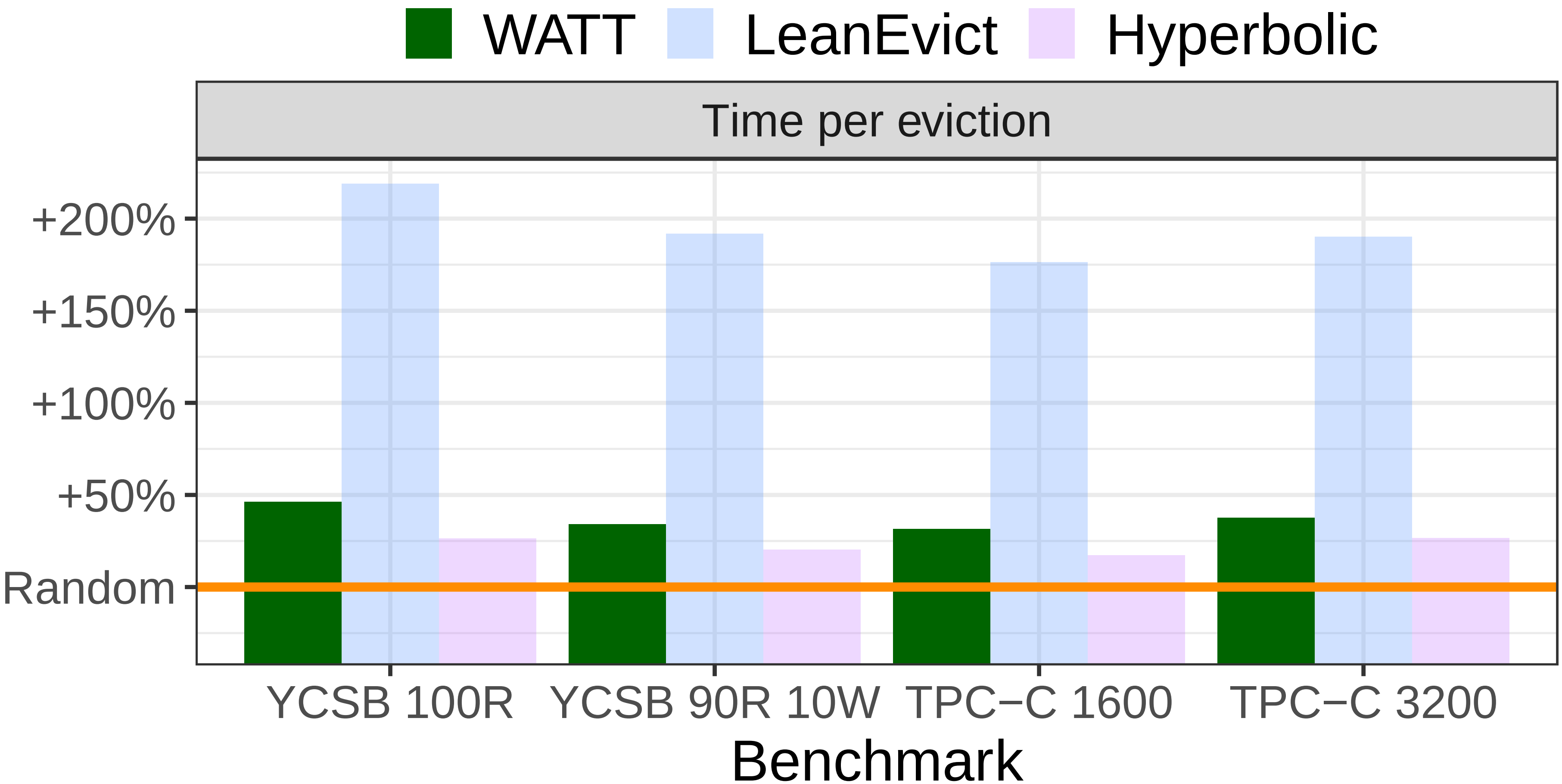
CPU Efficiency (Goal 3)

Access

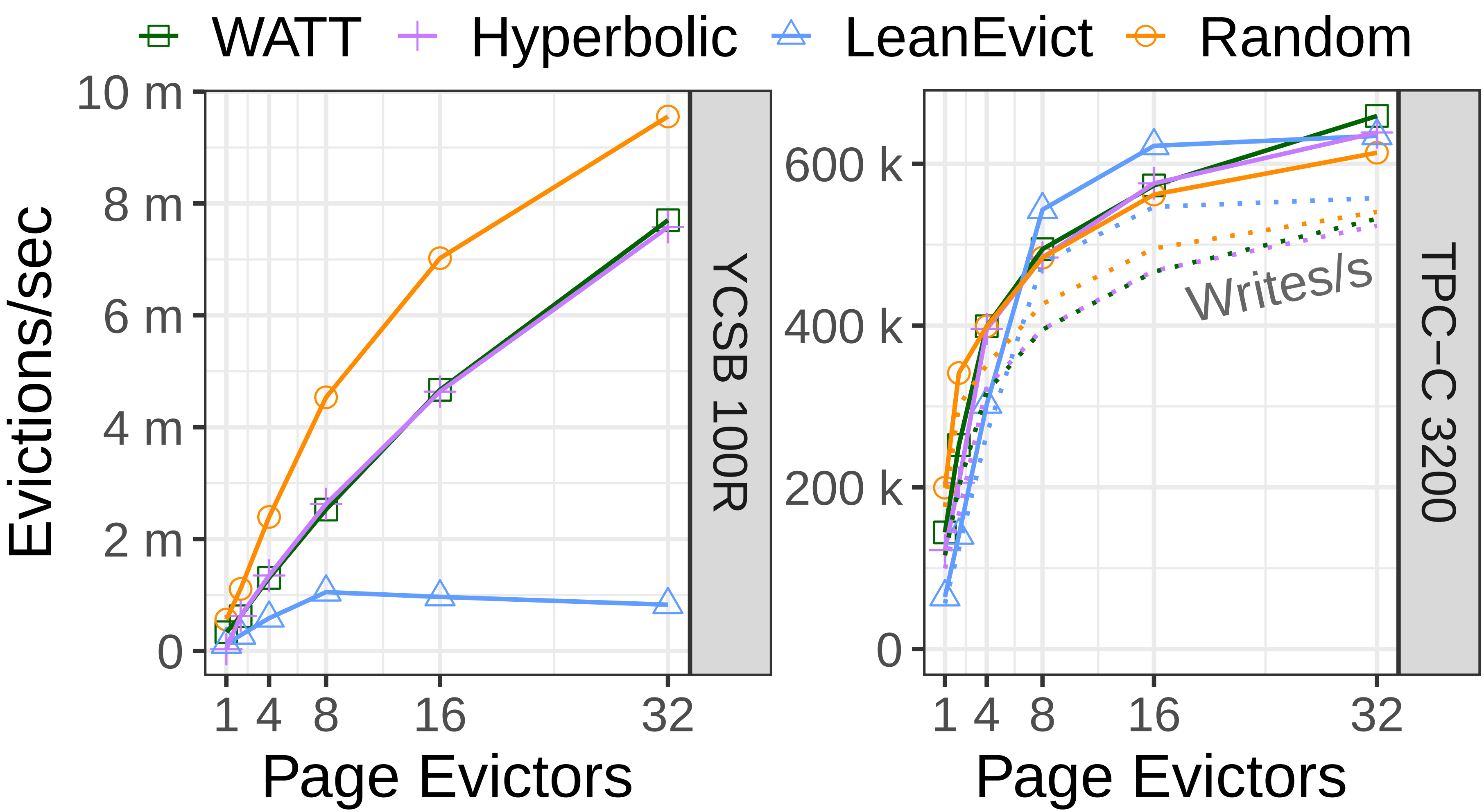


CPU Efficiency (Goal 3)

Eviction



Multi-Core Scalability (Goal 4)



4 Goals

	Replacement Effectiveness	Write Awareness	CPU Efficiency	Multi-Core Scalability
random	−−	no	++	++
CLOCK	−	no	+	+
LeanEvict	−	no	+	+
LRU2	+	no	++	+
LRU	−	no	−	−
CFLRU	−	yes	−	−
LRU_WSR	−	yes	−	−
Hyperbolic	~	no	++	++
ARC	+	no	−	−
WATT (our)	++	yes	++	++

Conclusion

- Write-Aware Timestamp Tracking:
  Effective and Efficient Page Replacement for Modern Hardware
- 4 Goals of Page Replacement
- Paper and Artifacts: leanstore.io

Leanstore on VLDB 2023 :
D2, Tue 3:30 PM - 5:00 PM
What Modern NVMe Storage Can Do, And How To Exploit It:
High-Performance I/O for High-Performance Storage Engines
Gabriel Haas (TUM); Viktor Leis (TUM)

C8, Thu 3:30 PM - 5:00 PM
Scalable and Robust Snapshot Isolation for High-Performance Storage Engines
Adnan Alhomssi (FAU); Viktor Leis (TUM)

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Write-Aware Timestamp Tracking:

Effective and Efficient Page Replacement for Modern Hardware

Demian Vöhringer[1] **Viktor Leis**[2]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg
[2]Technische Universität München

August 29, 2023 – VLDB

More Leanstore:

D2: What Modern NVMe Storage Can Do, And How To Exploit It: High-Performance I/O for High-Performance Storage Engines

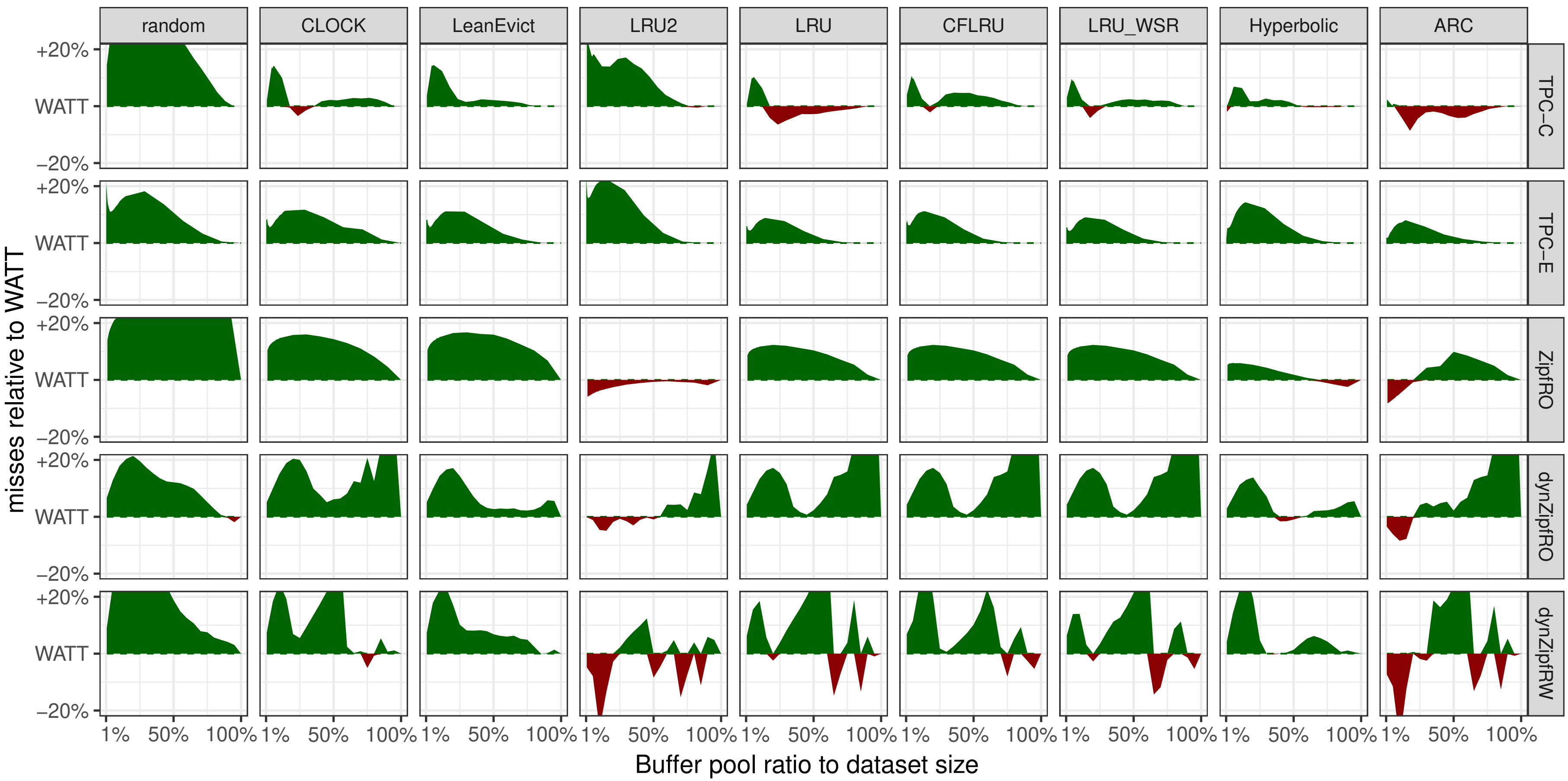
C8: Scalable and Robust Snapshot Isolation for High-Performance Storage Engines

Space Used

Memory Overhead	random	CLOCK	LeanEvict	LRU2	LRU	CFLRU	LRU_WSR	Hyperbolic	ARC	WATT
Bytes*	0	$\frac{1}{8}$	8	8	24	24	24+$\frac{1}{8}$	8	48	50
Overhead† [≈%]	0	0	0.2	0.2	0.6	0.6	0.6	0.2	1.2	1.2

*Using: timestamp = 4 Byte, pointer or PageID = 8 Byte, doubly-linked list = 2 pointers + 1 PageID = 24 Byte
†4KB pagesize

Full Comparison



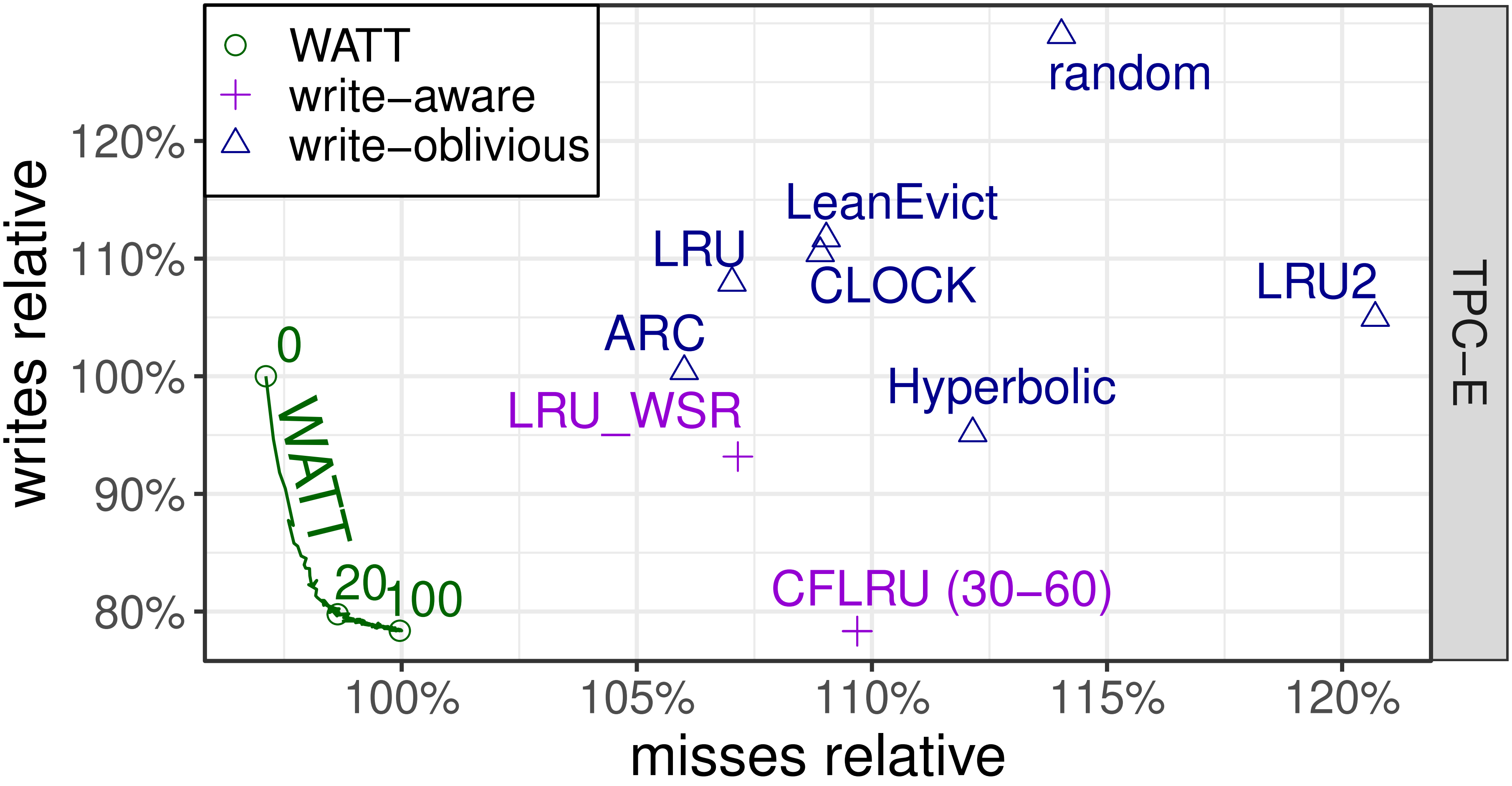
PageTracker::track()

```
atomic<uint32_t> globalTrackerTime // time (epoch)

class PageTracker // sizeof(PageTracker) < cache_line
    atomic<uint32_t> accessLog[8], writeLog[4]
    atomic<uint8_t> accessHead, writeHead

void PageTracker::track()
    // Compare last tracked and current epoch
    uint8_t oldPos = accessHead.load()
    uint32_t now = globalTrackerTime.load()
    if (now != accessLog[oldPos])
        // Store current epoch if they differ
        uint8_t pos = (oldPos+1) % 8
        accessLog[pos].store(now, memory_order_release)
        accessHead.store(pos, memory_order_release)
```

PageTracker::PVaccess()

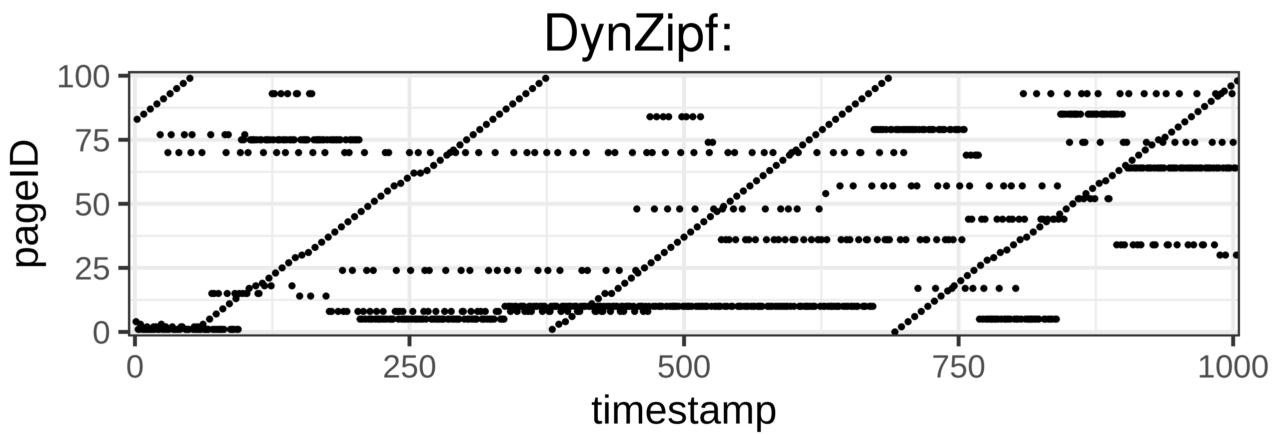
```
// Table of precalculated quotients i
// with dampening of 0.1
float iQuotient[] = {{0.1, 8, 7, 6, 5, 4, 3, 2},
                     {2, 0.1, 8, 7, 6, 5, 4, 3},···}

float PageTracker::PVaccess()
   uint8_t     head = accessHead.load()
   uint32_t     now = globalTrackerTime.load()
   __m256i      ts8 = _mm256_loadu_si256(accessLog)
   __m256i     now8 = _mm256_set1_epi32(now)
   __m256i ageInt8 = _mm256_sub_epi32(now8, ts8)
   __m256     age8  = _mm256_cvtepi32_ps(ageInt8)
   __m256        i8 = _mm256_loadu_ps(iQuotient[head])
   __m256 subfreq8 = _mm256_div_ps(i8, age8)
   return             _mm256_reduce_max_ps(subfreq8)
```

Write Aware (TPC-E)



Simulation Traces*

	Writes	Accesses	Pages	Hot pages†	Top 10 pages
TPC-C	15.6%	1 M	16,128	8	36.0%
TPC-E	5.7%	1.5 M	65,656	24	30.4%
dynZipfRO	0%	2 M	20,000	1	4.5%



*Traces published: github.com/itodnerd/WATT-traces/tree/main/WATT_competition_traces
†more than 2% of all accesses

LeanStore Traces

Benchmark	YCSB		TPC-C	
Configuration	100R	90R 10 W	1600	3200
Dataset Size	400 GB	400 GB	264 GB	588 GB