Bottlenecks Uncovered: A Component-Wise Breakdown of the Runtime of an OLTP System

Max Gilbert

Technische Universität Kaiserslautern

m_gilbert13@cs.uni-kl.de

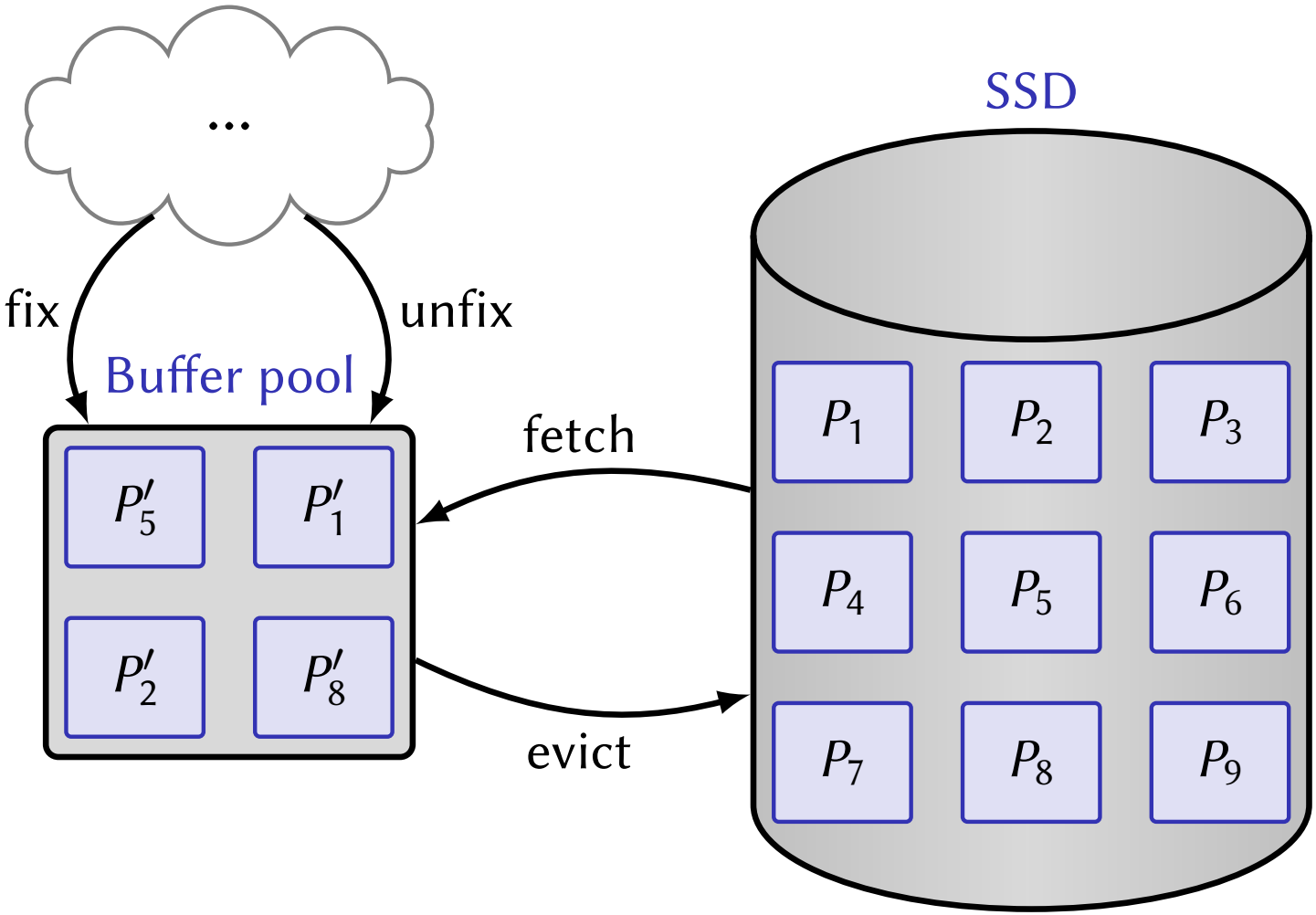
September 30, 2020

Table of Contents

Section 1

Page Eviction by the Buffer Pool Management

The Buffer Pool



Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference		Since first reference
References	No consideration				
	Most recent reference				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration				
	Most recent reference				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration				
	Most recent reference				
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			
	Most recent reference				
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO
	Most recent reference				
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference				
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference		LRU		
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		**Age**			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference		LRU MRU		
	Some recent references				
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference		LRU MRU		
	Some recent references			LRU-K	
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference		LRU MRU		
	Some recent references		SLRU	LRU-K	
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference		LRU MRU CLOCK		
	Some recent references		SLRU	LRU-K	
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK		
	Some recent references		SLRU	LRU-K	
	All references				

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK		
	Some recent references		SLRU	LRU-K	
	All references		GCLOCK-V1		

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references		GCLOCK-V1		

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references		GCLOCK-V1 DGCLOCK-V1		

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references		GCLOCK-V1 DGCLOCK-V1		

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		LFUDA

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2		
	Some recent references		SLRU	LRU-K	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		LRD-V1 LFUDA

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2		
	Some recent references		SLRU	LRU-K LRD-V2	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		LRD-V1 LFUDA

Extended Page Replacement Algorithm Classification by Effelsberg and Härder From [EH]

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2 LeanStore		
	Some recent references		SLRU	LRU-K LRD-V2	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		LRD-V1 LFUDA

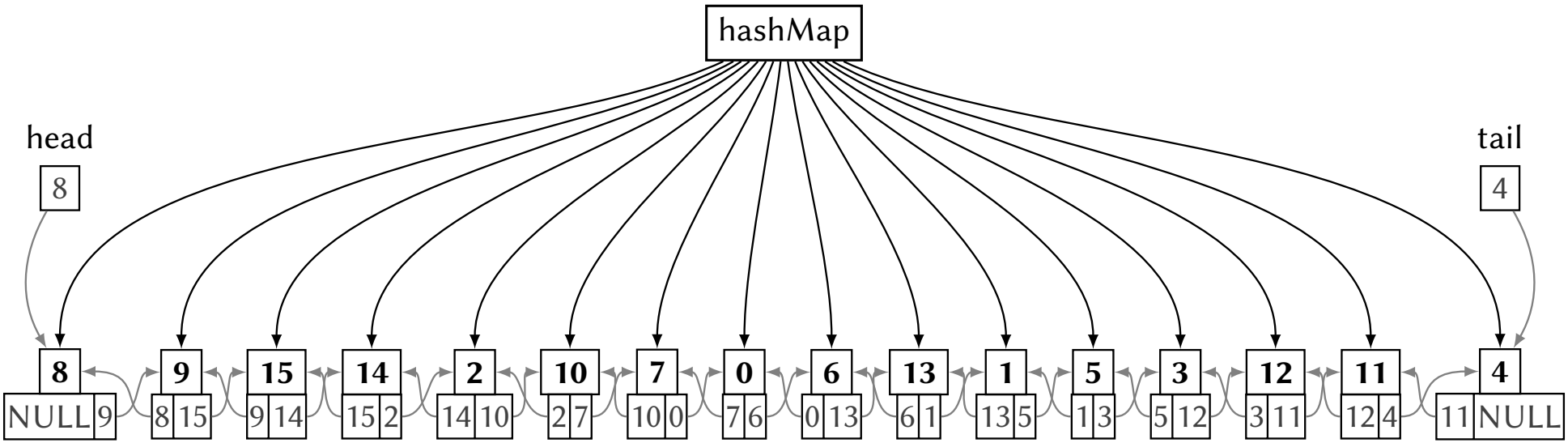
Subsection 2

Least Recently Used

Subsubsection 1

Hash-Map-Doubly-Linked-List Implementation

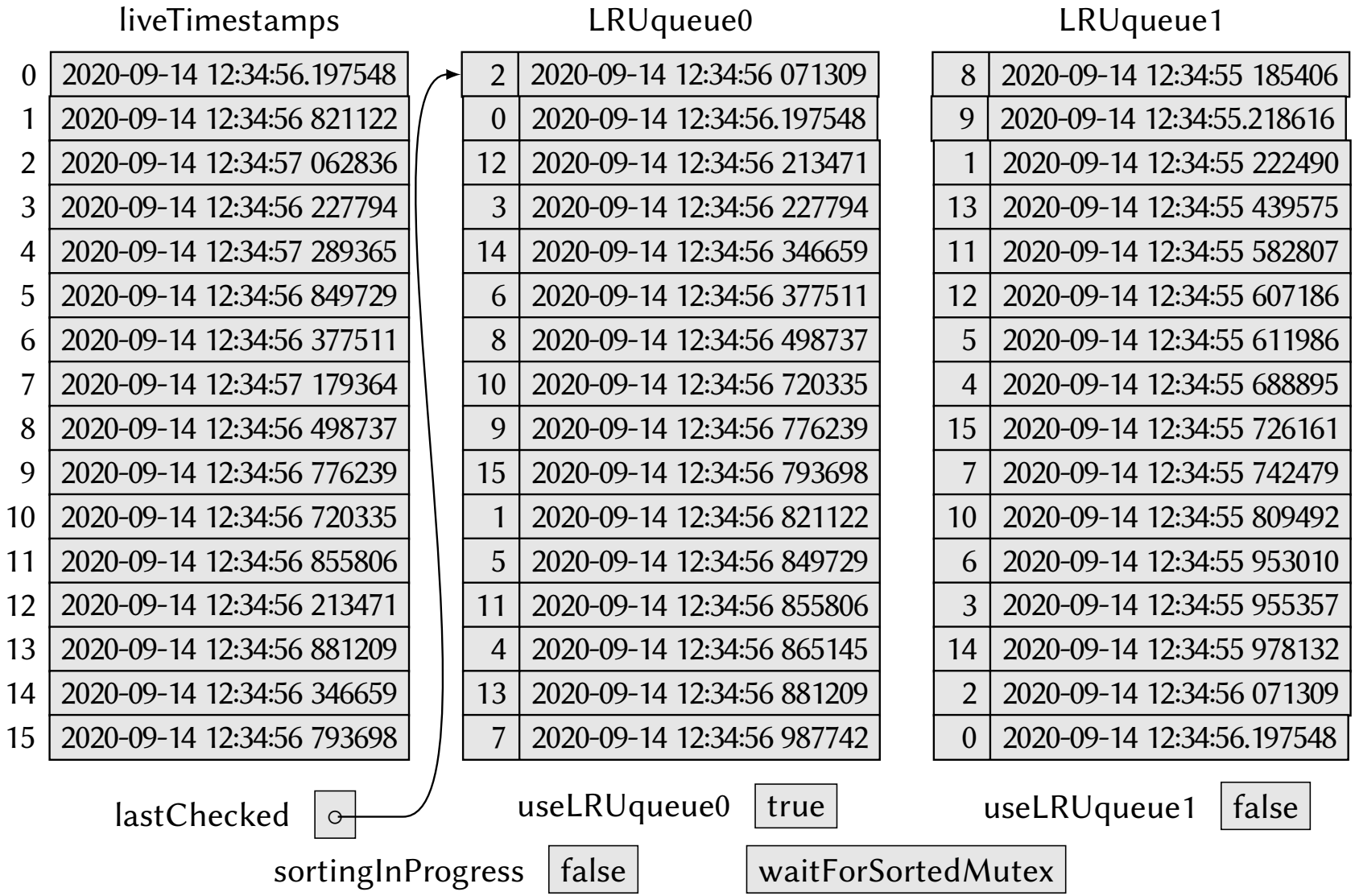
Hash-Map-Doubly-Linked-List



Subsubsection 2

Timestamp-Sorting Implementation

Page Reference Statistics

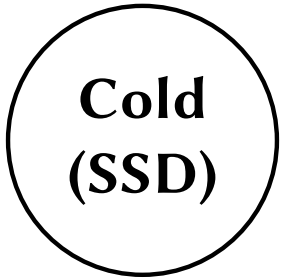


Subsection 3

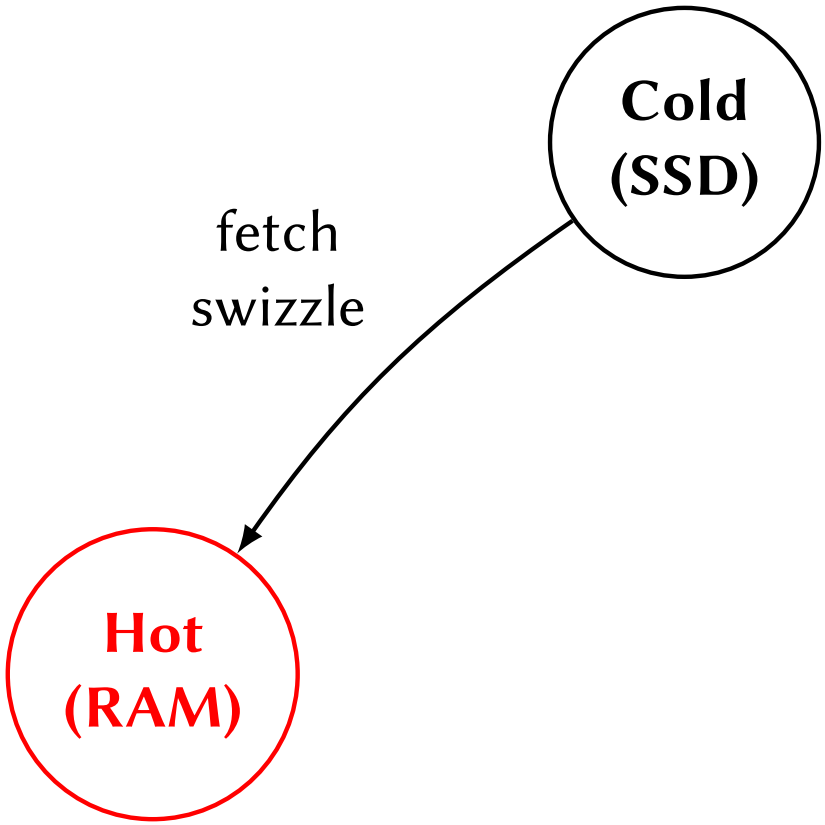
LeanStore Replacement [Lei+]

Basic Concept (Pointer Swizzling)

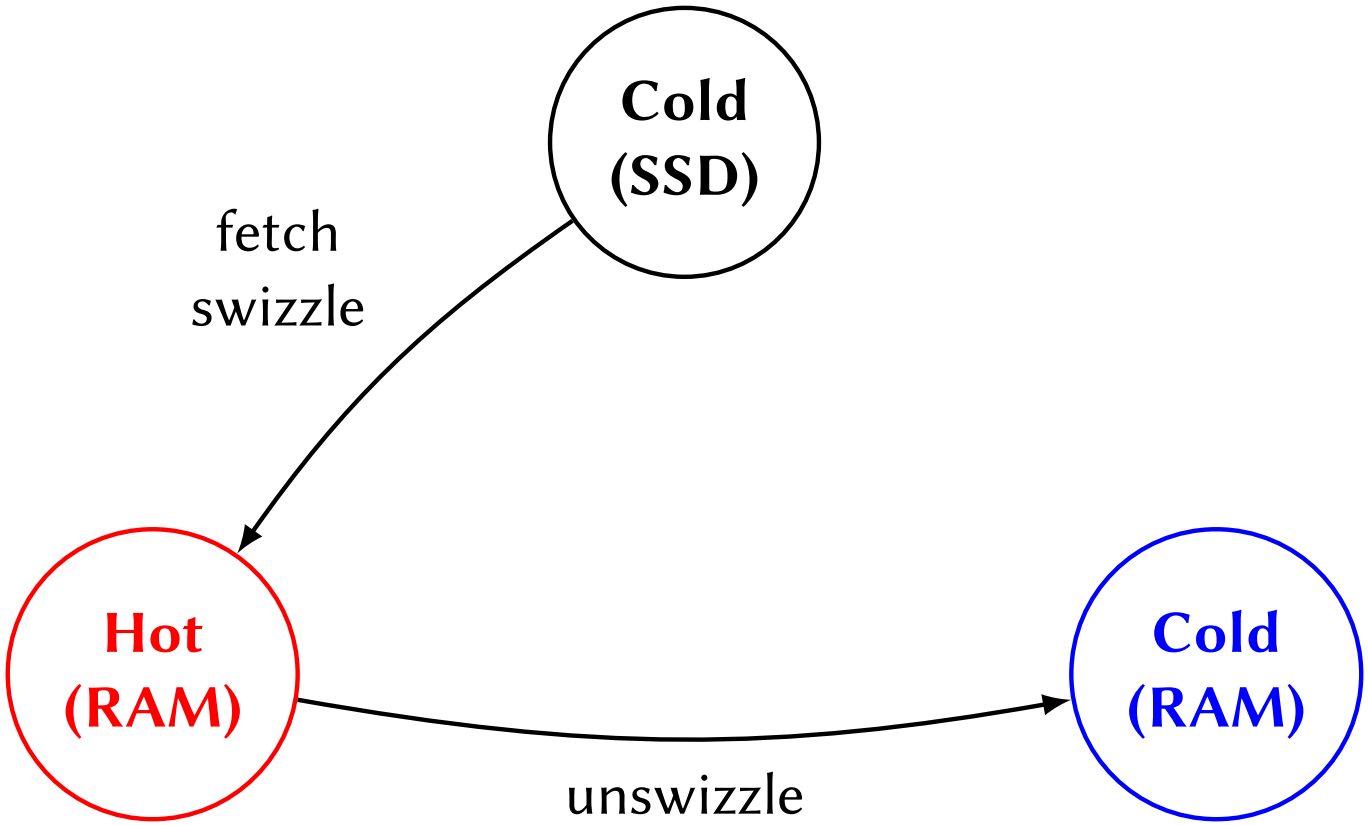
Basic Concept (Pointer Swizzling)



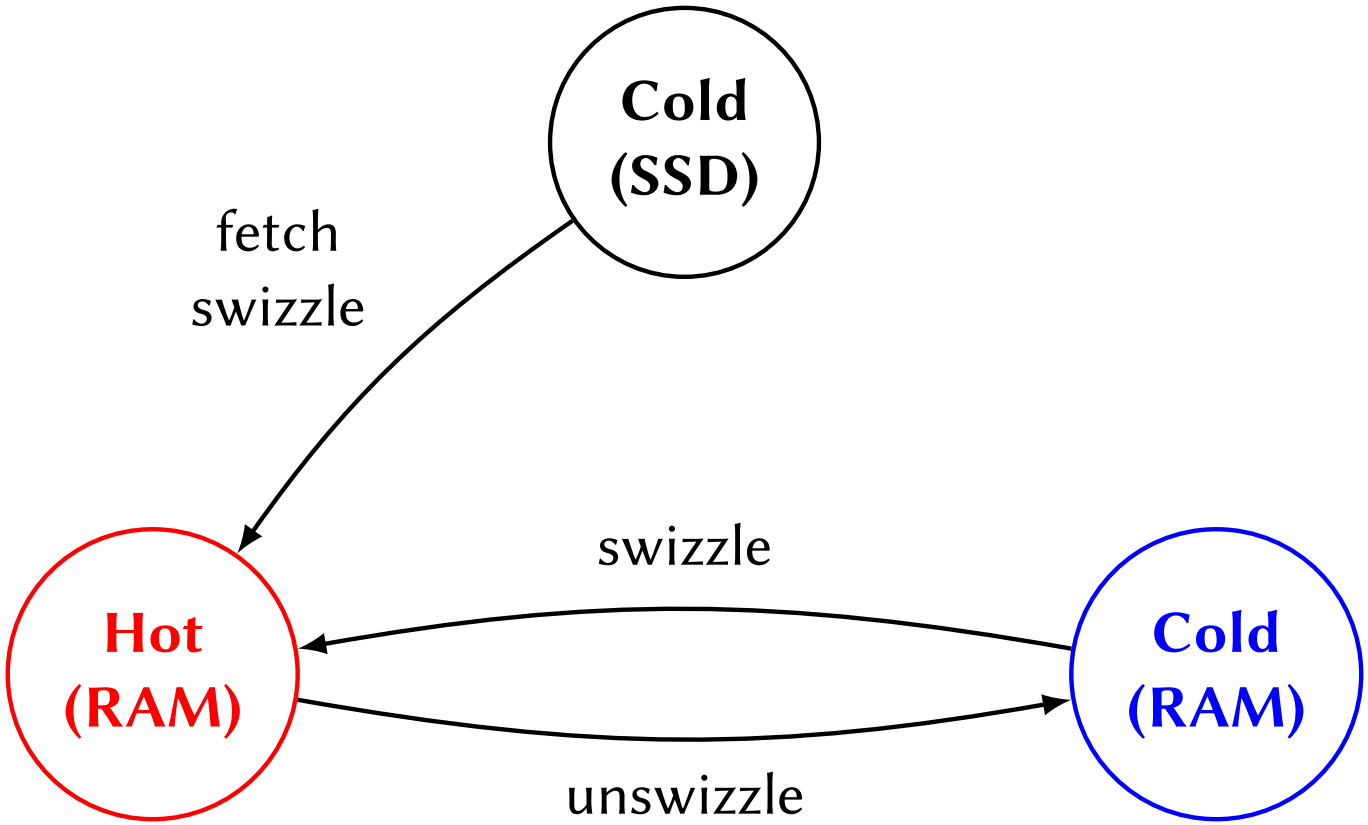
Basic Concept (Pointer Swizzling)



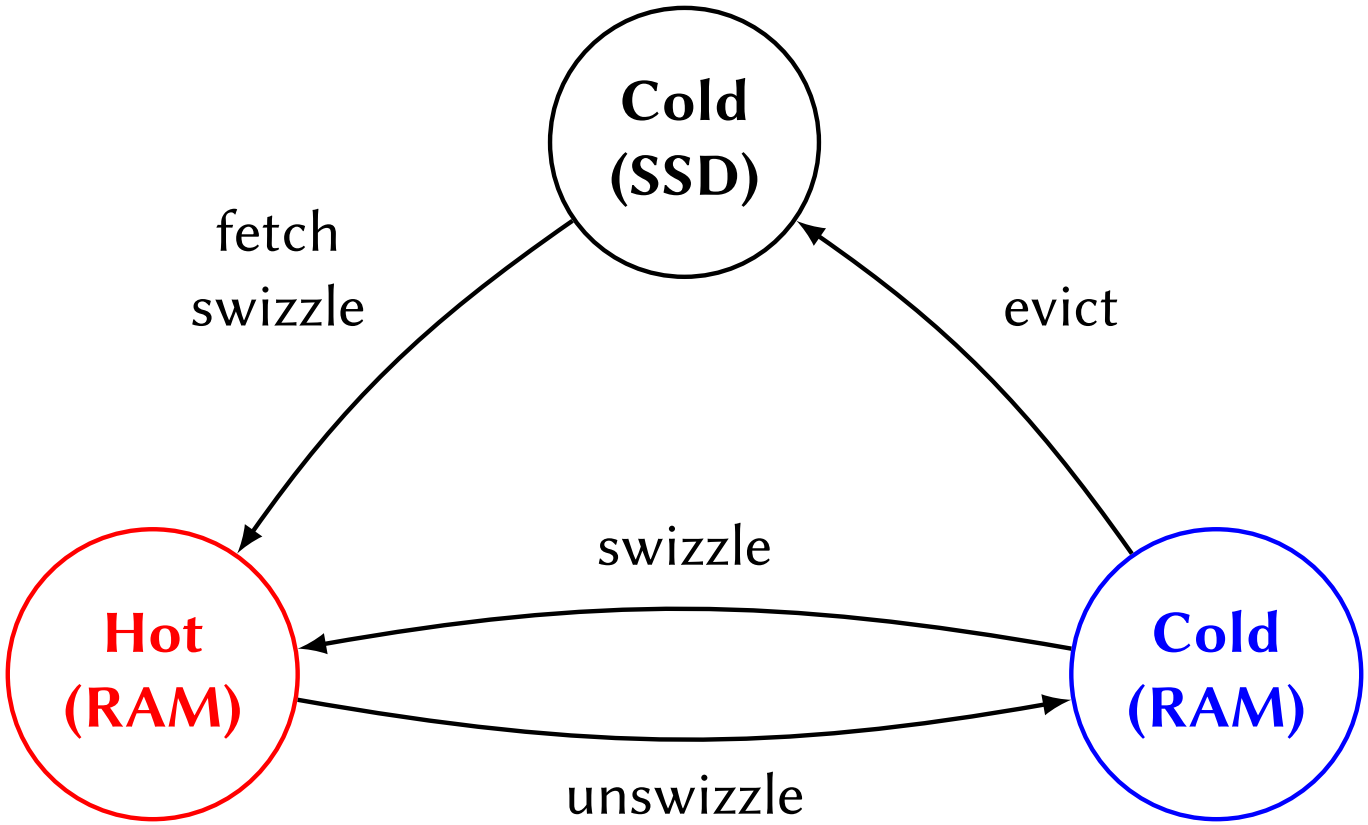
Basic Concept (Pointer Swizzling)



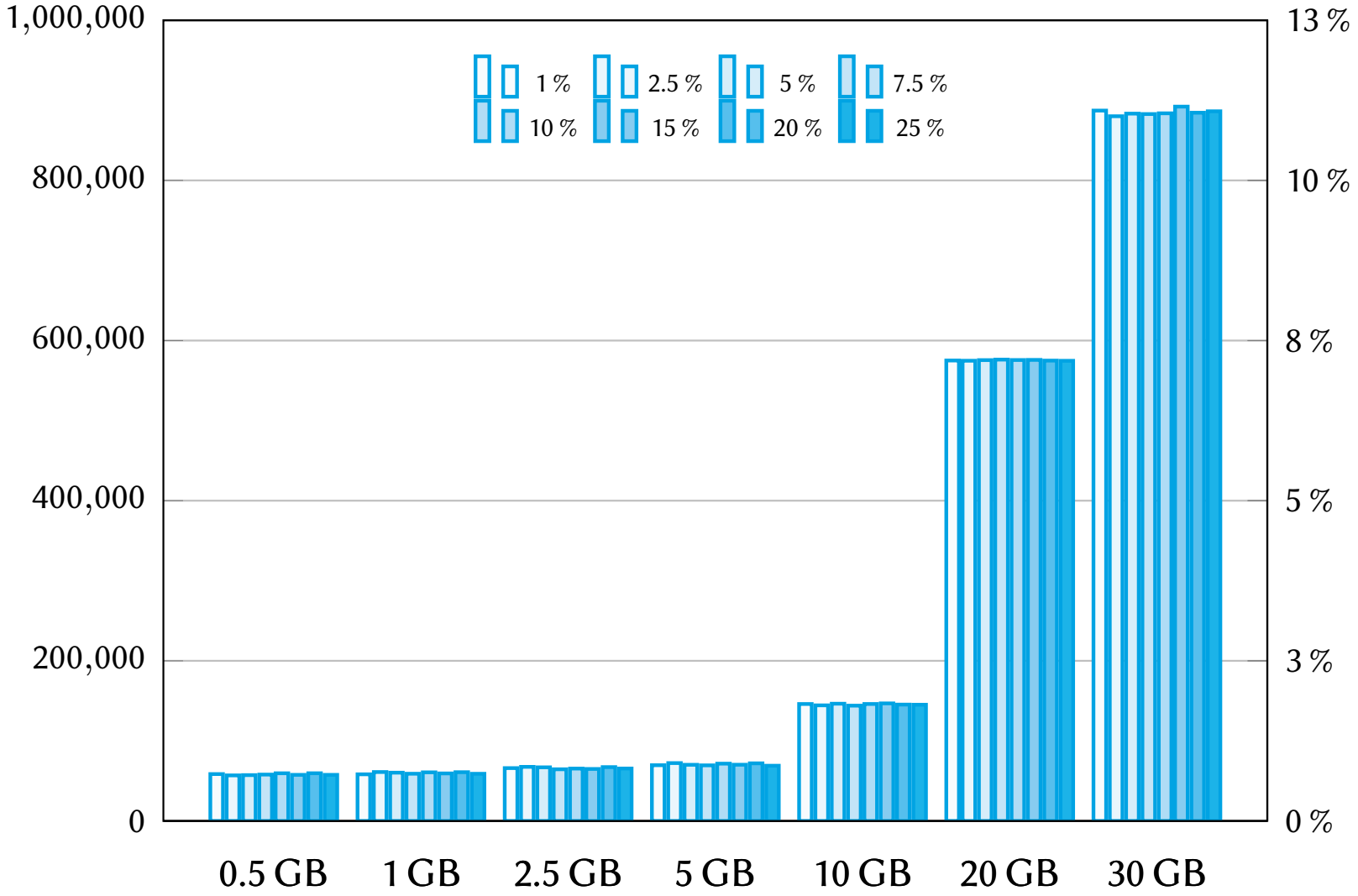
Basic Concept (Pointer Swizzling)



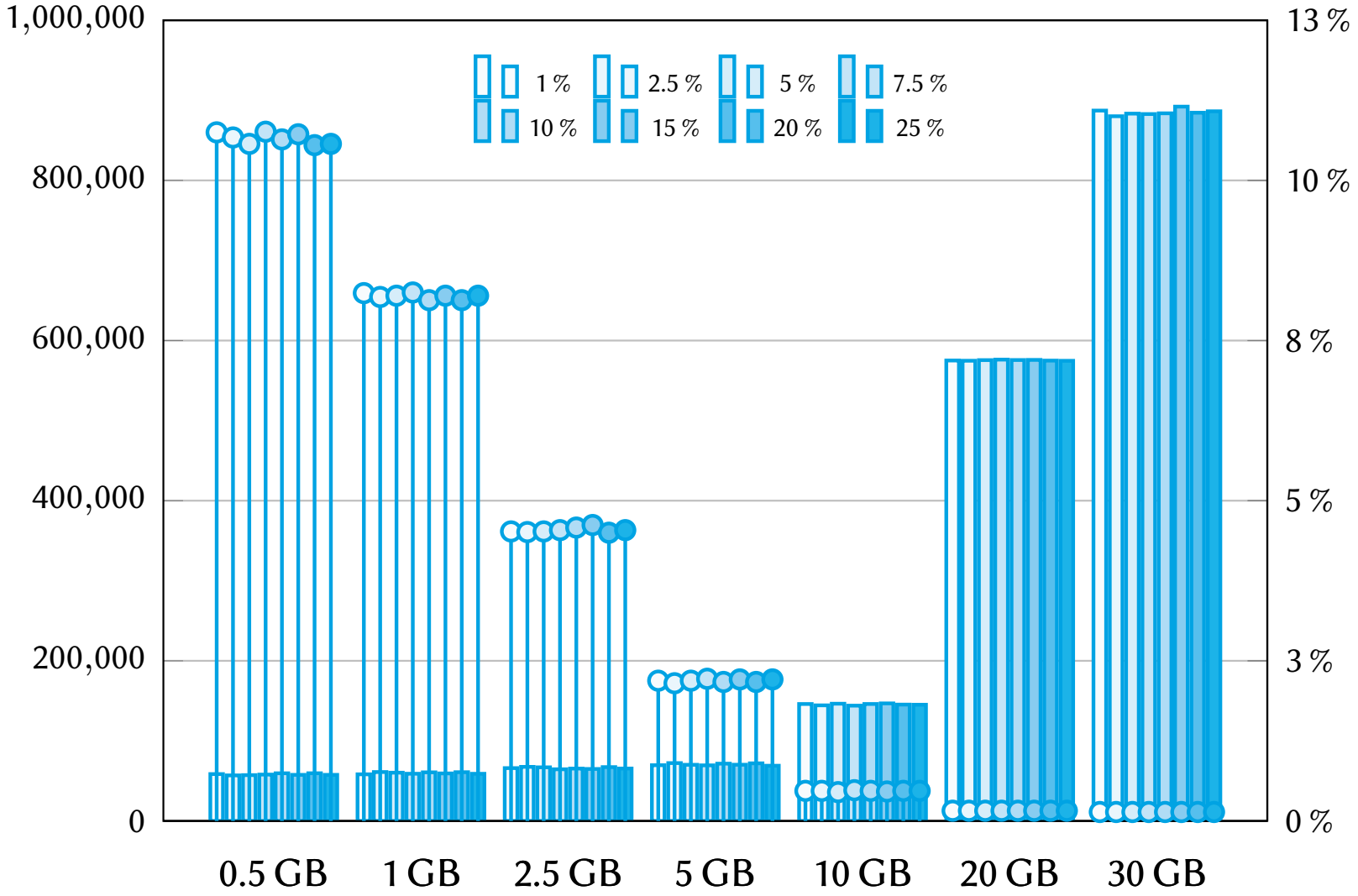
Basic Concept (Pointer Swizzling)



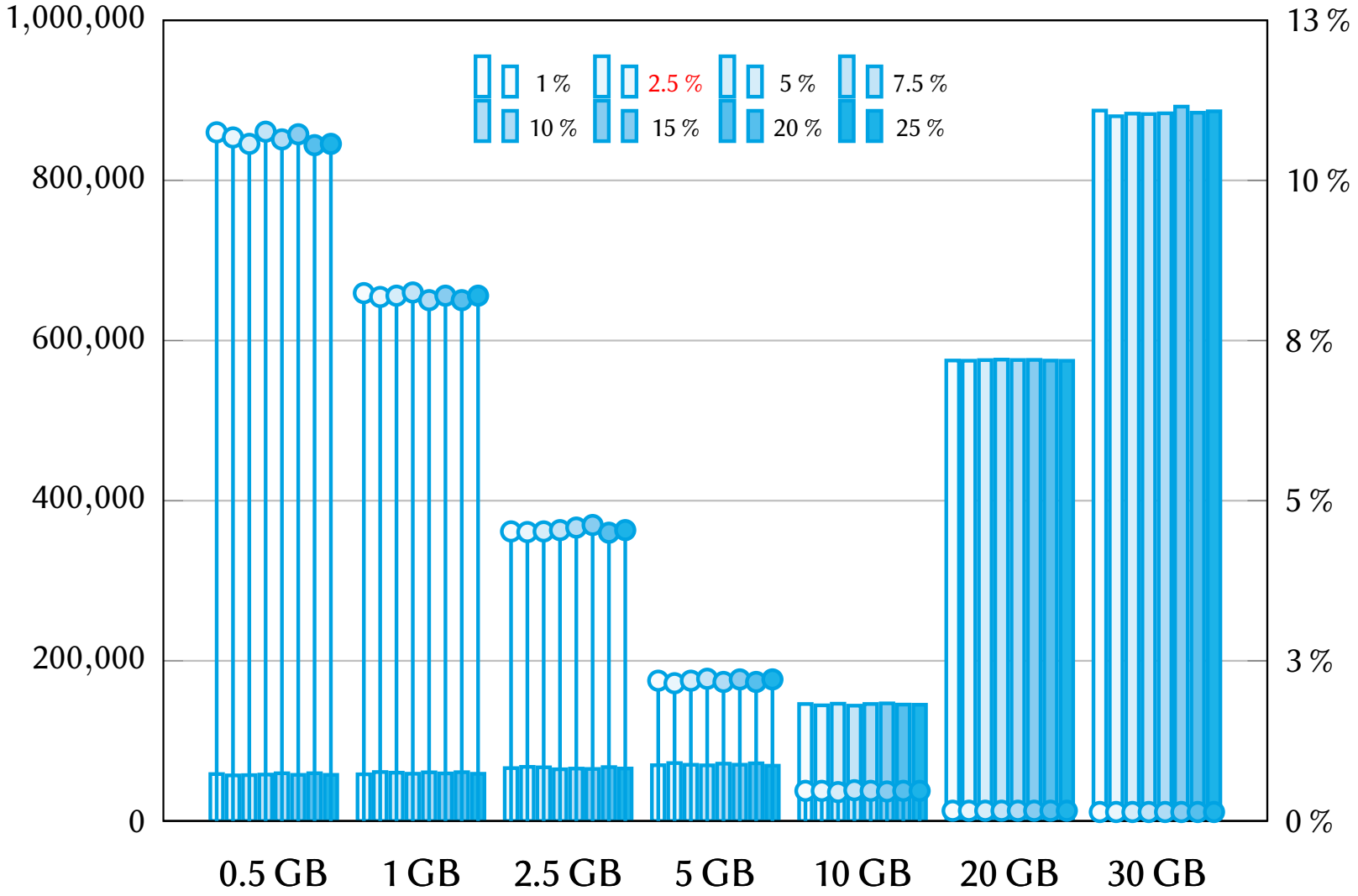
Performance Evaluation



Performance Evaluation



Performance Evaluation



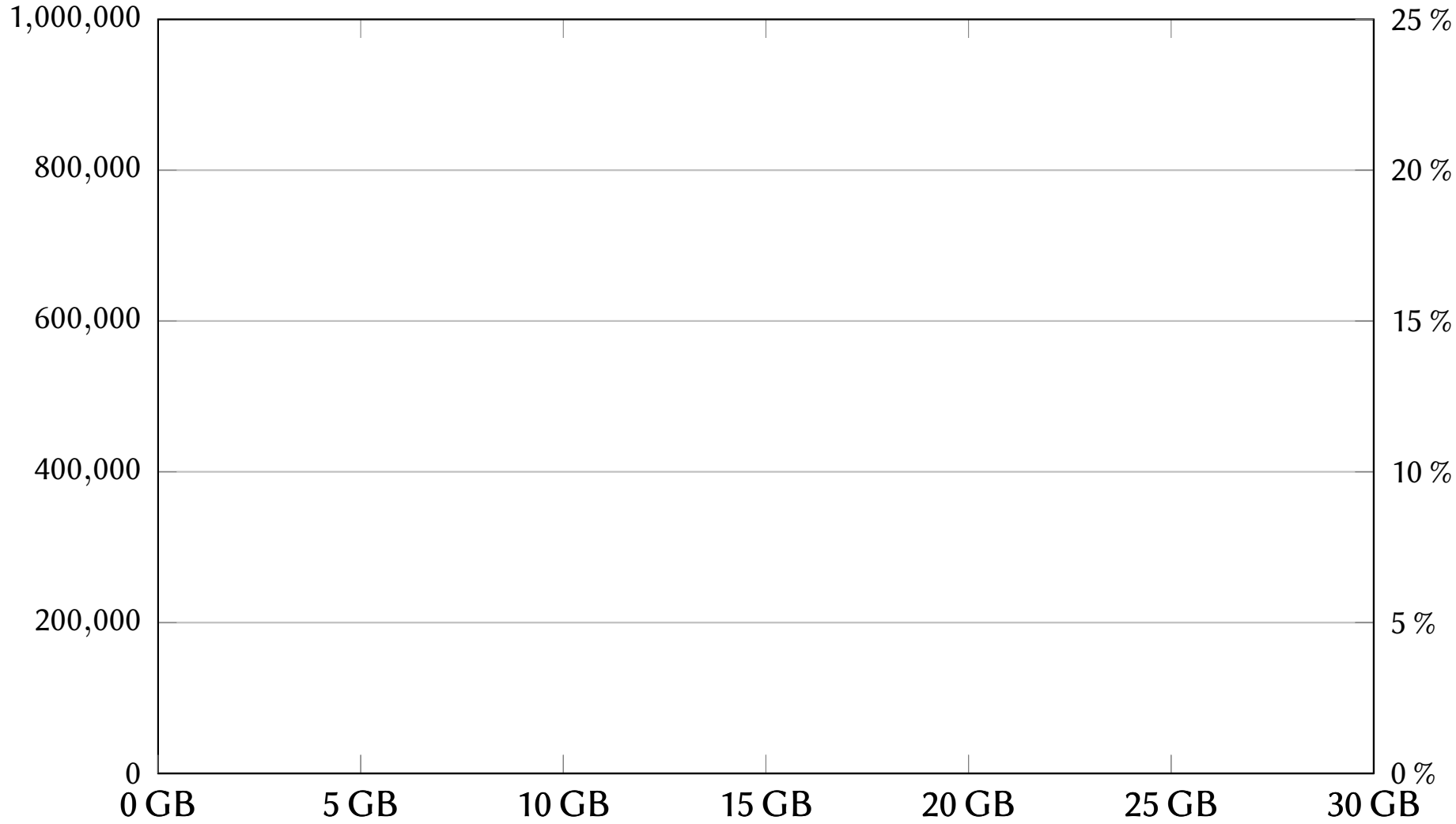
Subsection 4

All Page Replacement Algorithms

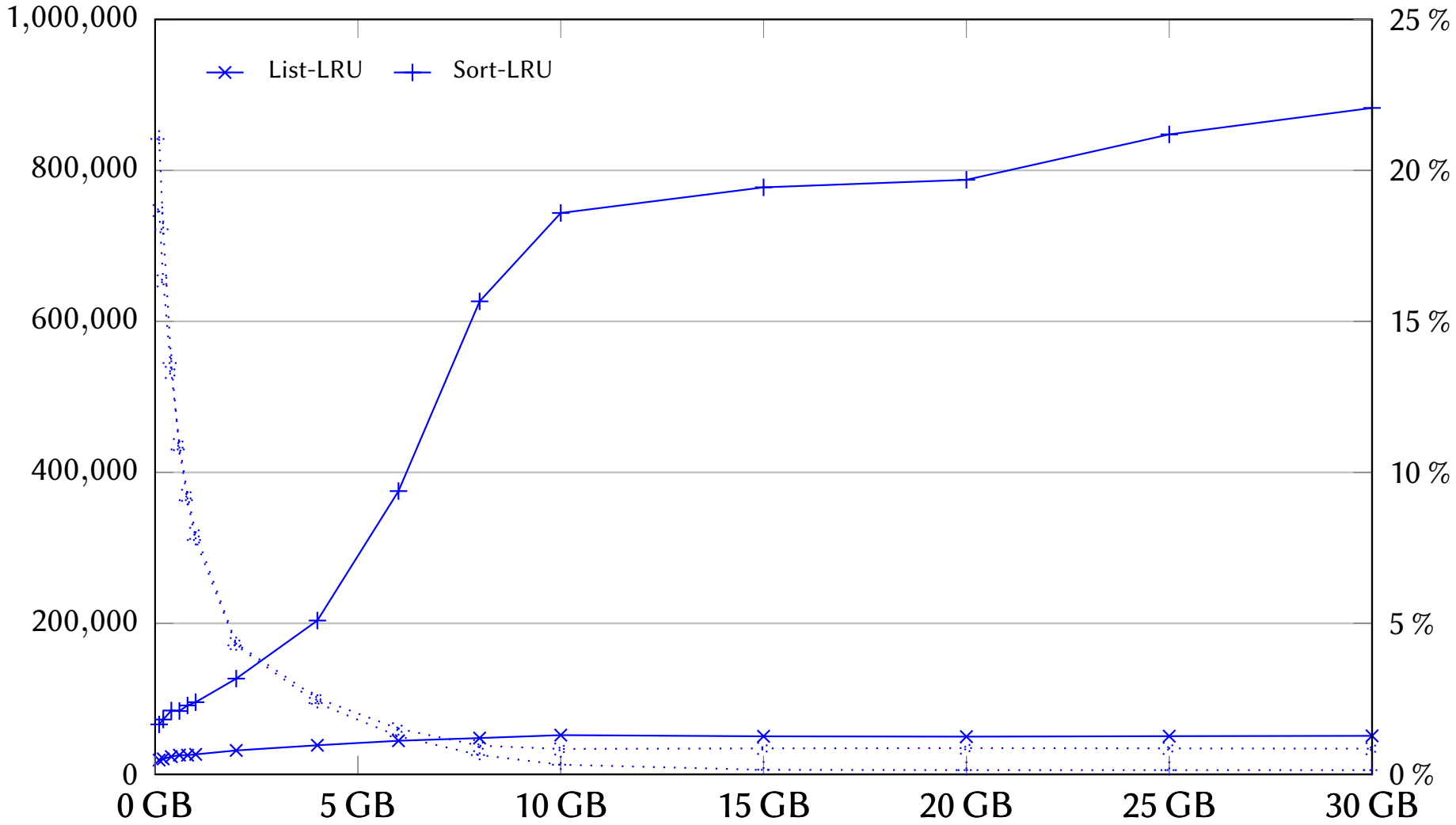
Overview

Consideration during selection decision		Age			
		No consideration	Since most recent reference	Since some recent reference	Since first reference
References	No consideration	RANDOM			FIFO FILO
	Most recent reference	ZCLOCK	LRU MRU CLOCK GCLOCK-V2 DGCLOCK-V2 LeanStore		
	Some recent references		SLRU	LRU-K LRD-V2	
	All references	LFU	GCLOCK-V1 DGCLOCK-V1		LRD-V1 LFUDA

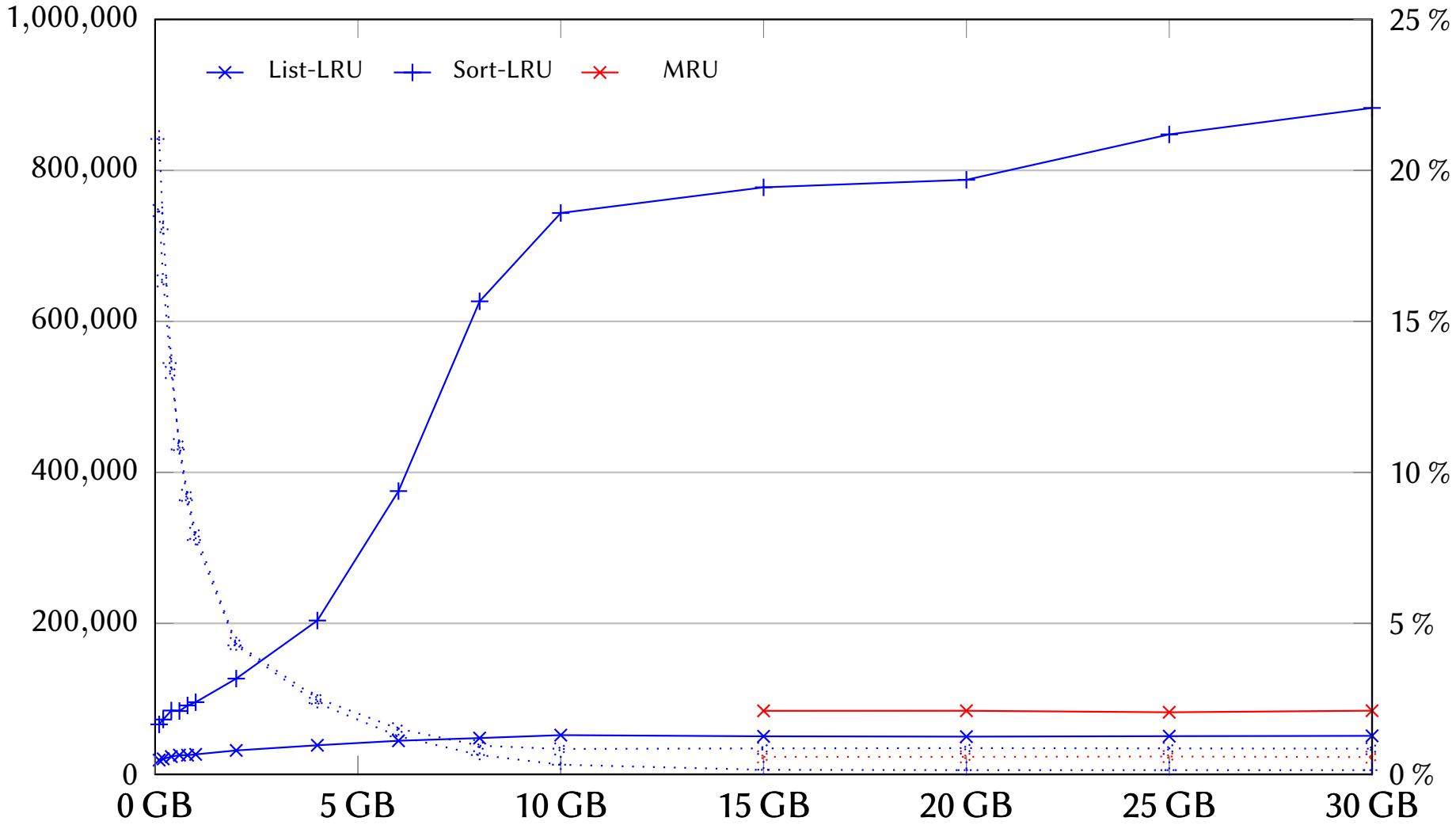
Performance Evaluation



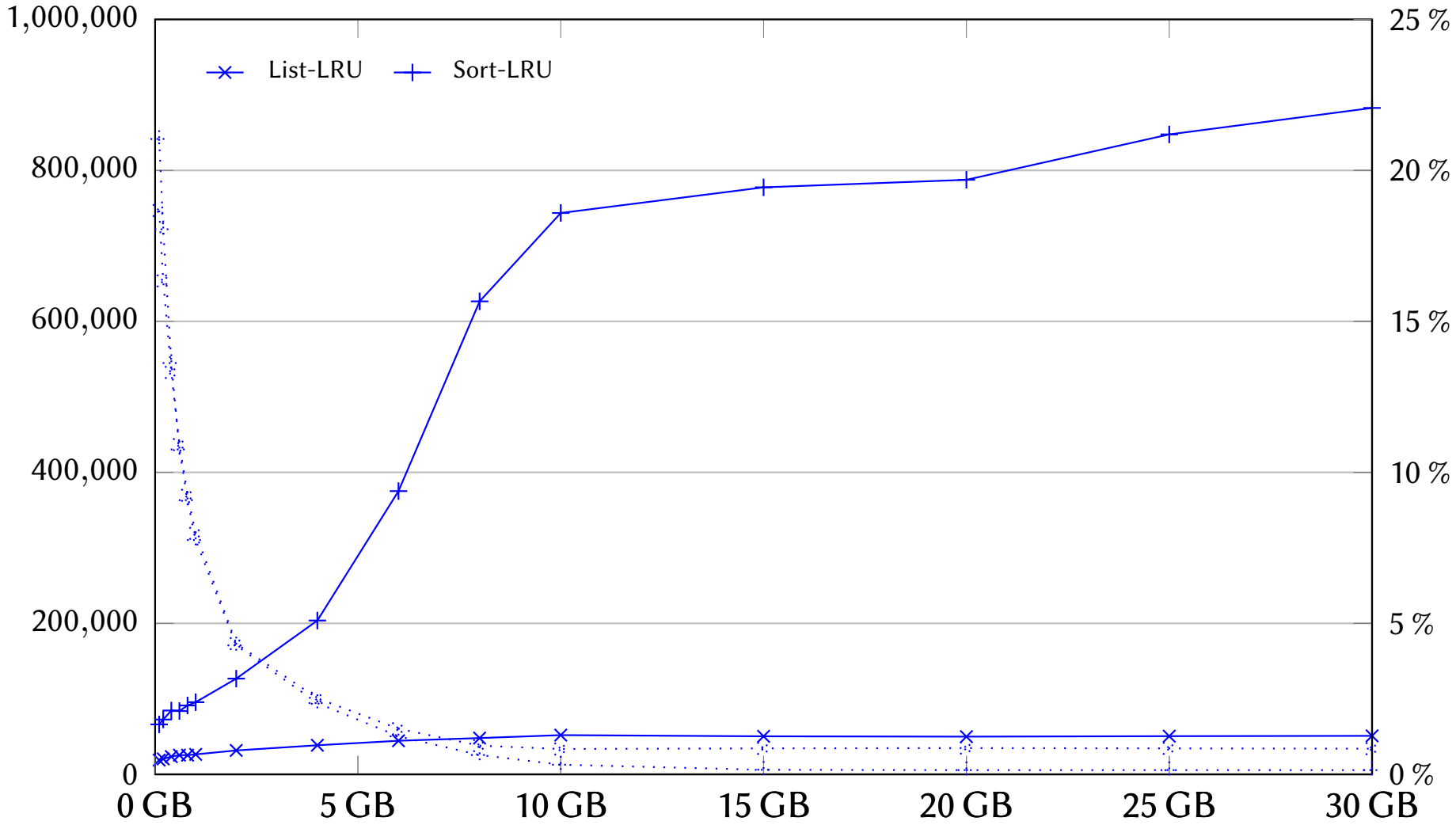
Performance Evaluation



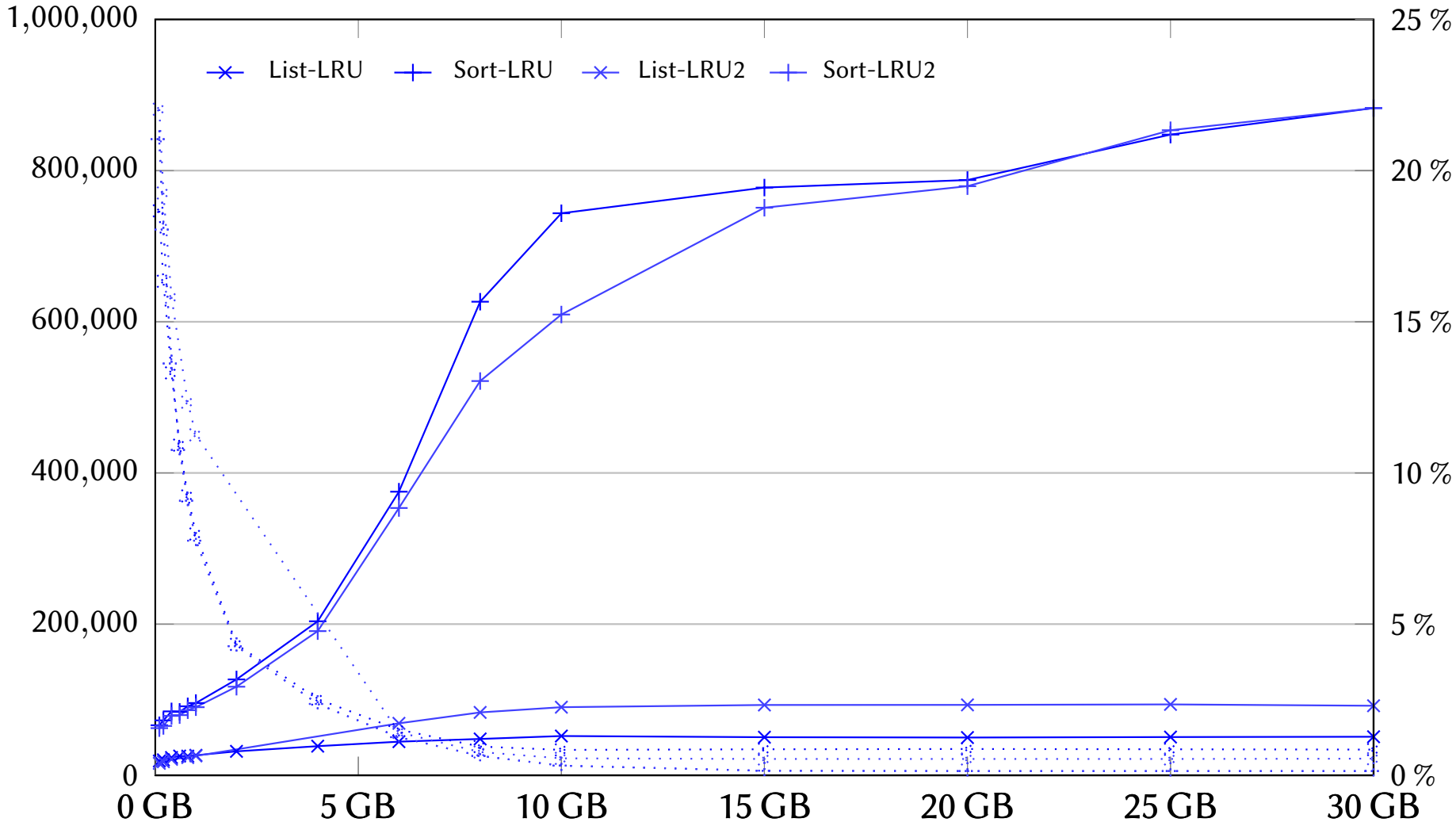
Performance Evaluation



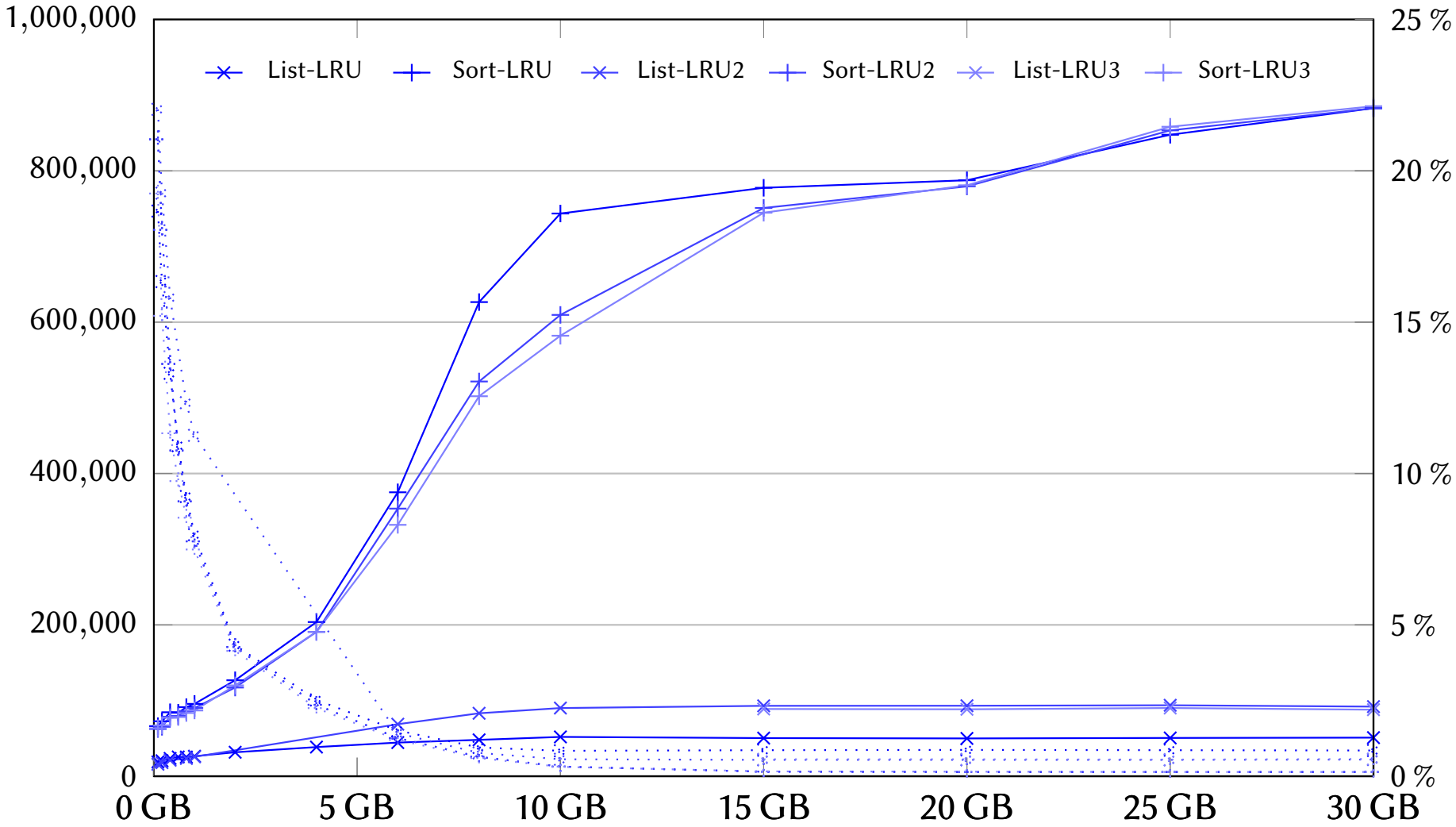
Performance Evaluation



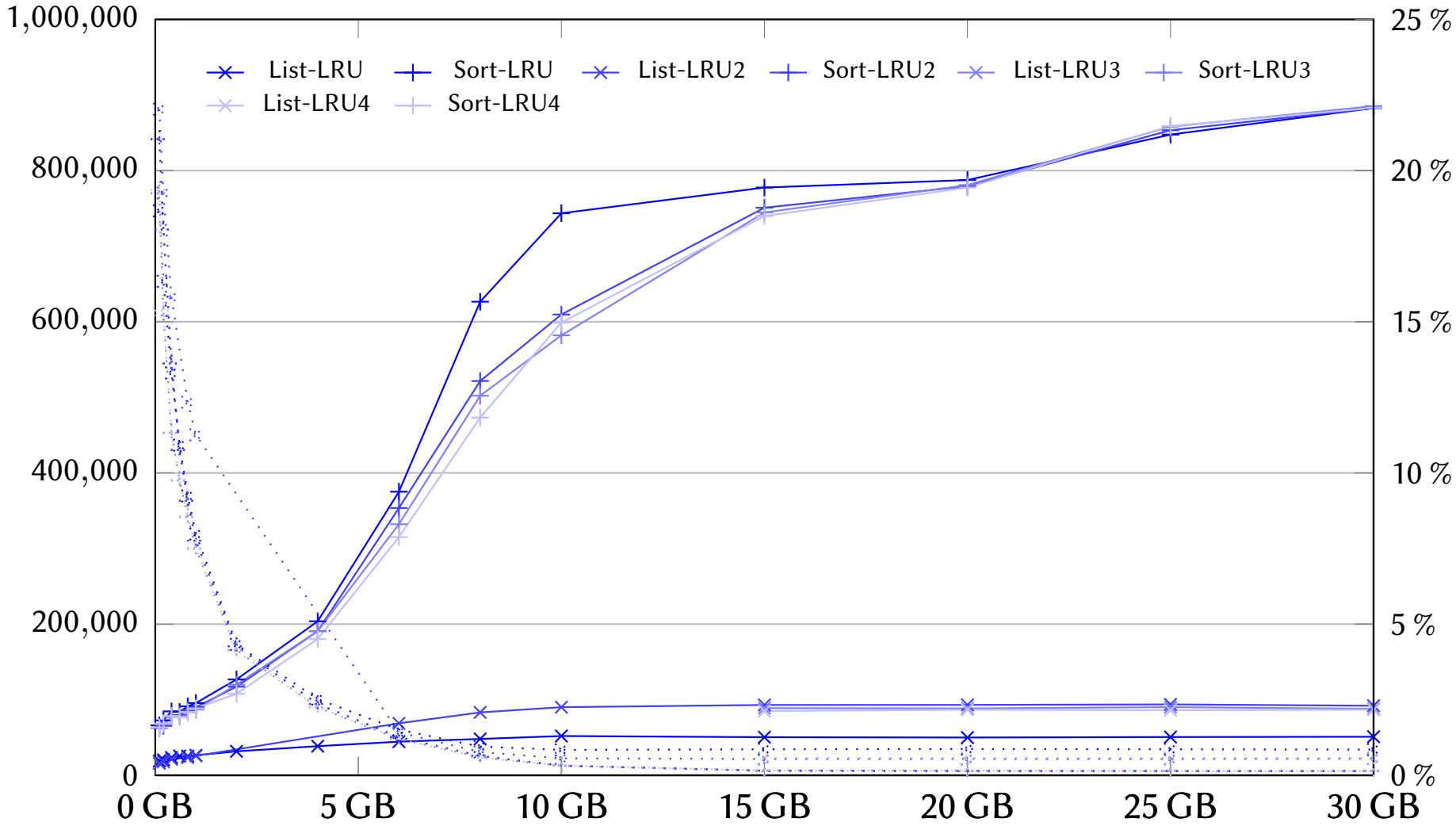
Performance Evaluation



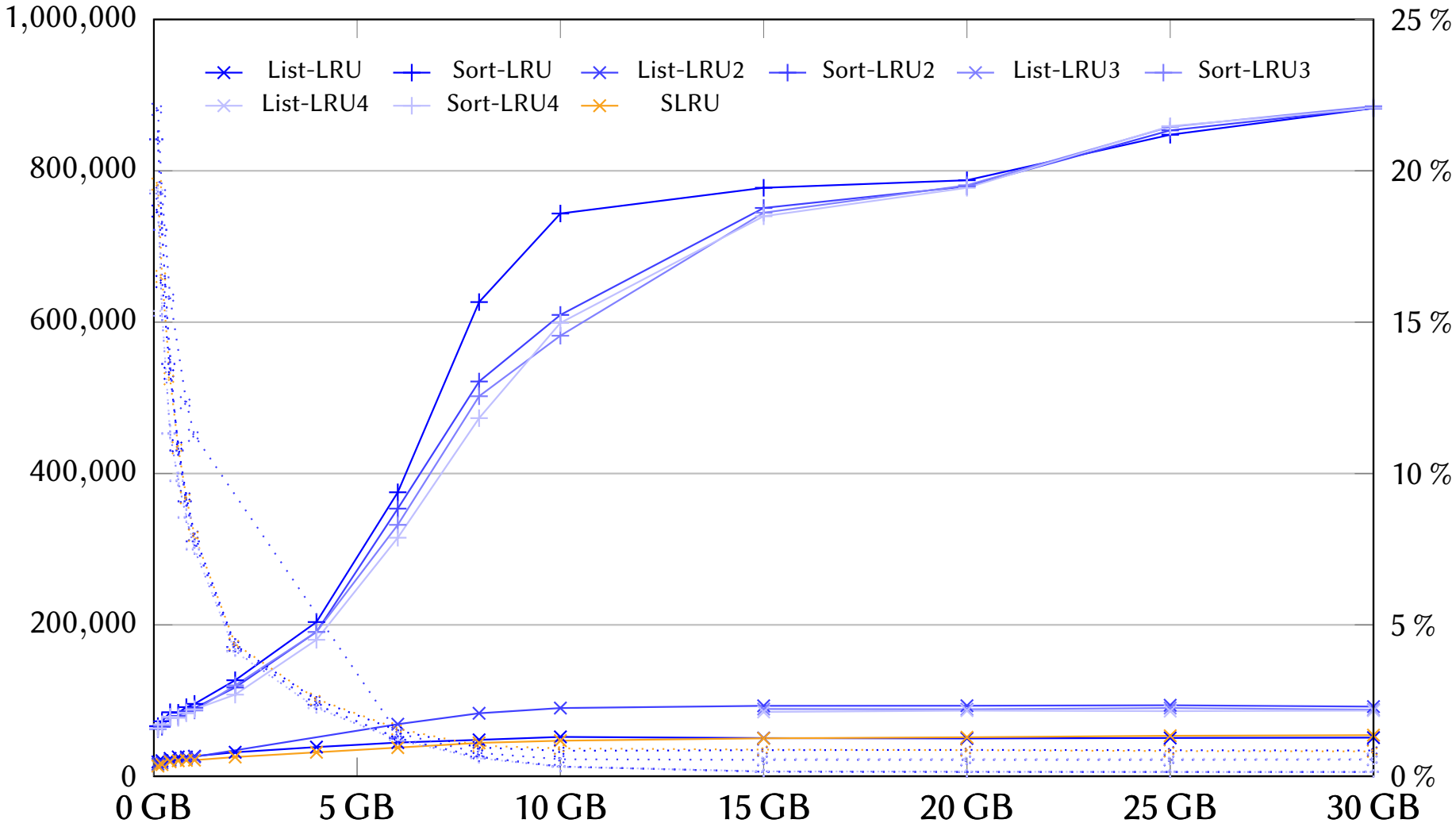
Performance Evaluation



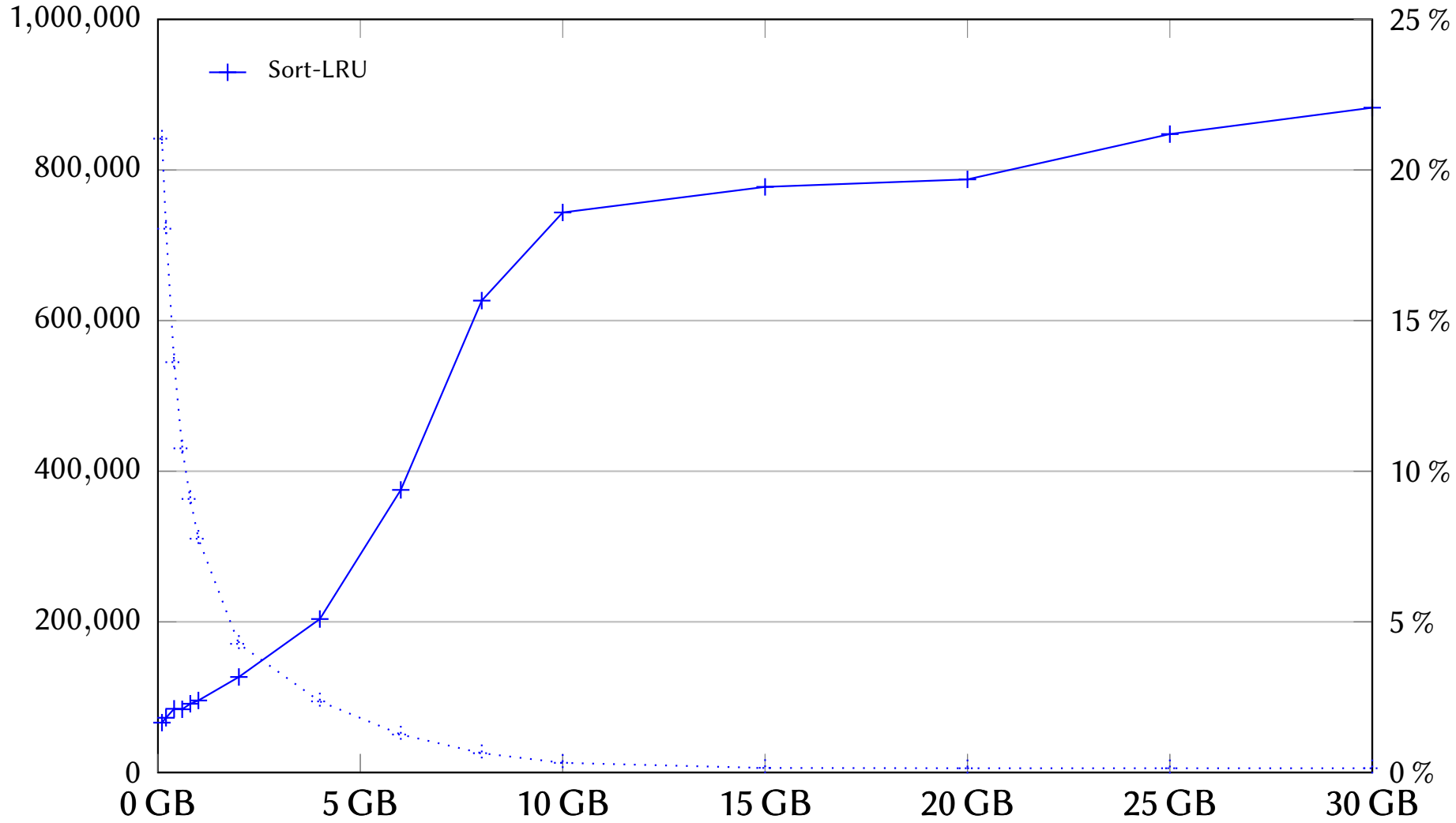
Performance Evaluation



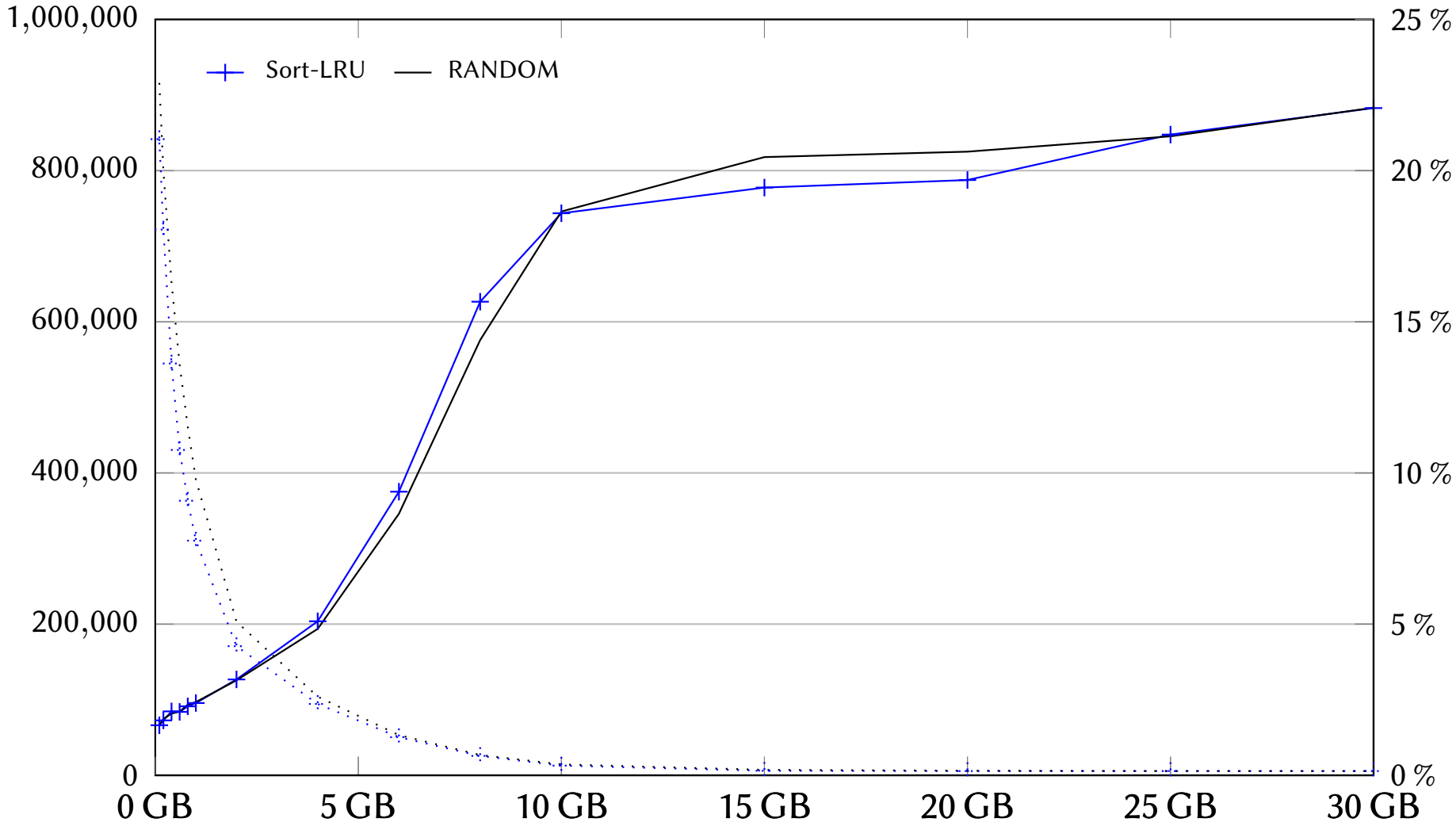
Performance Evaluation



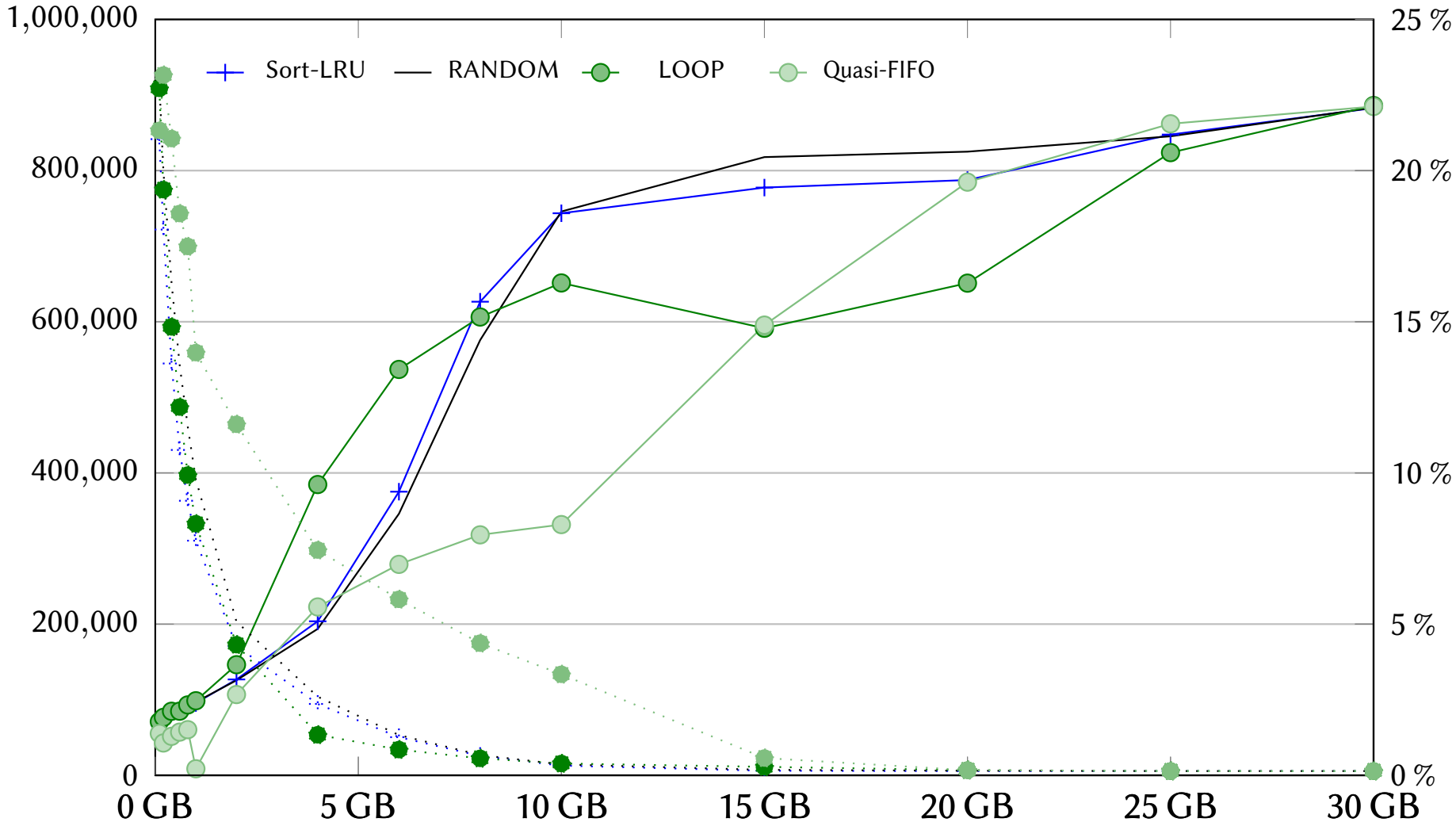
Performance Evaluation



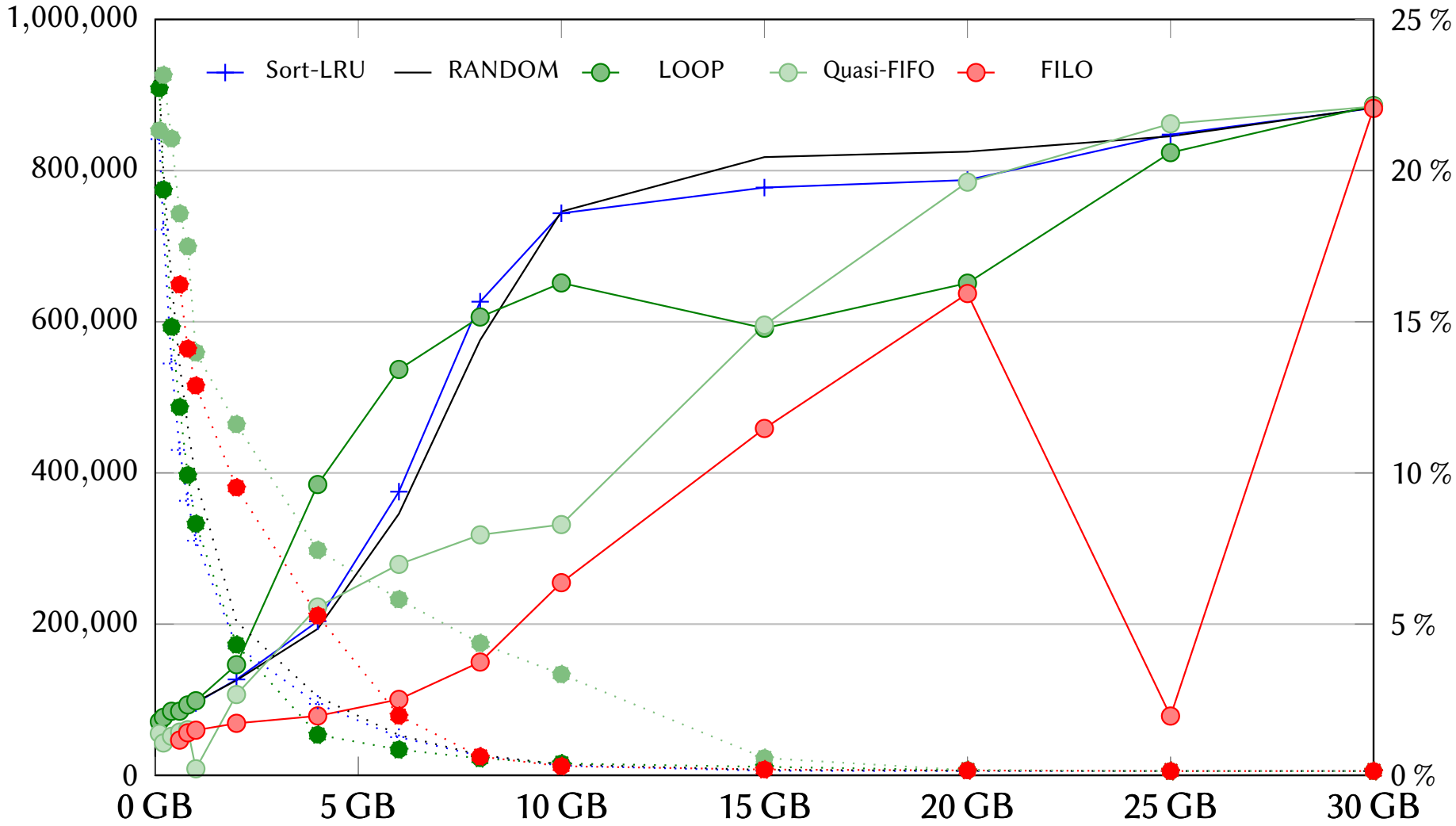
Performance Evaluation



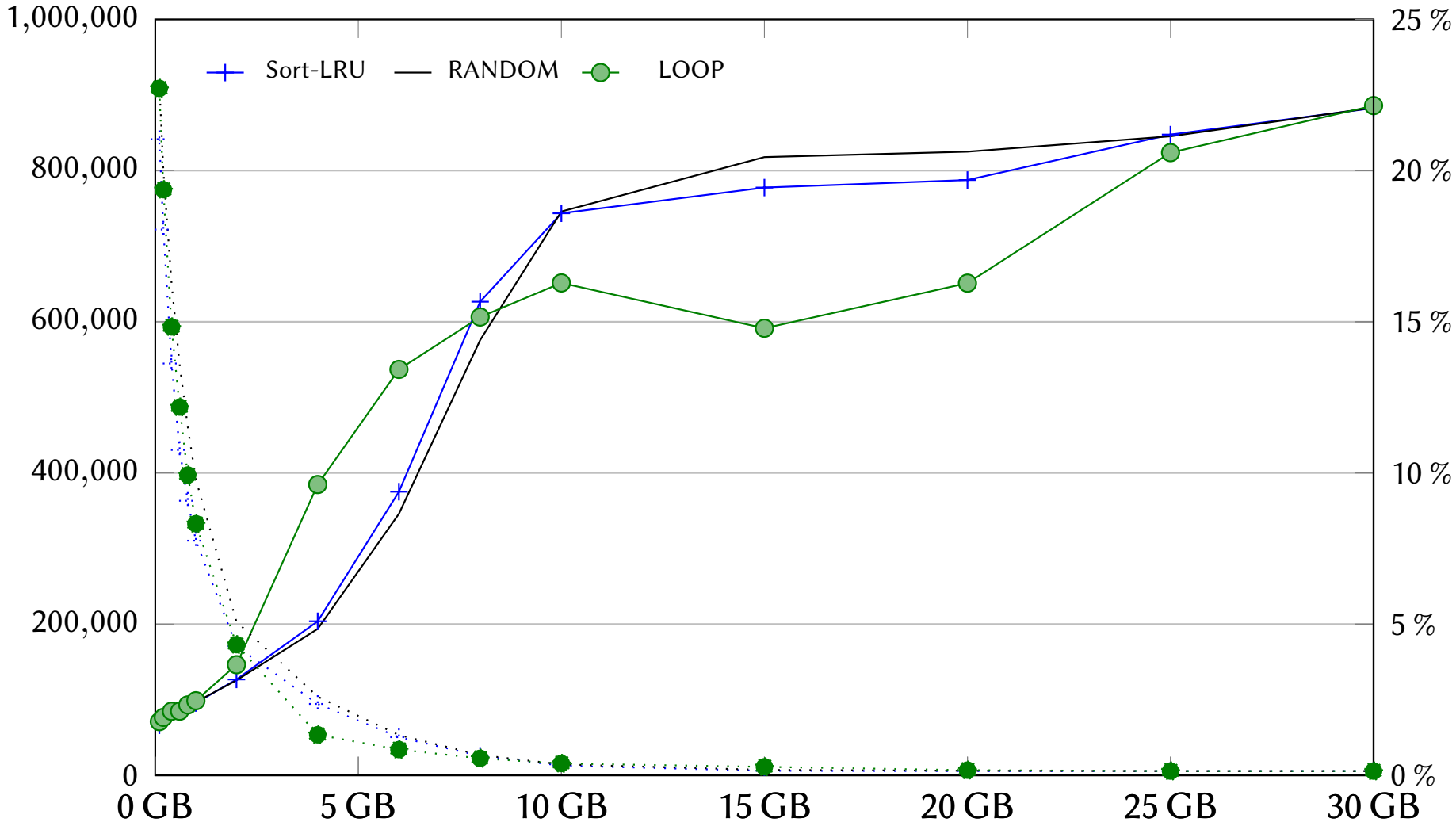
Performance Evaluation



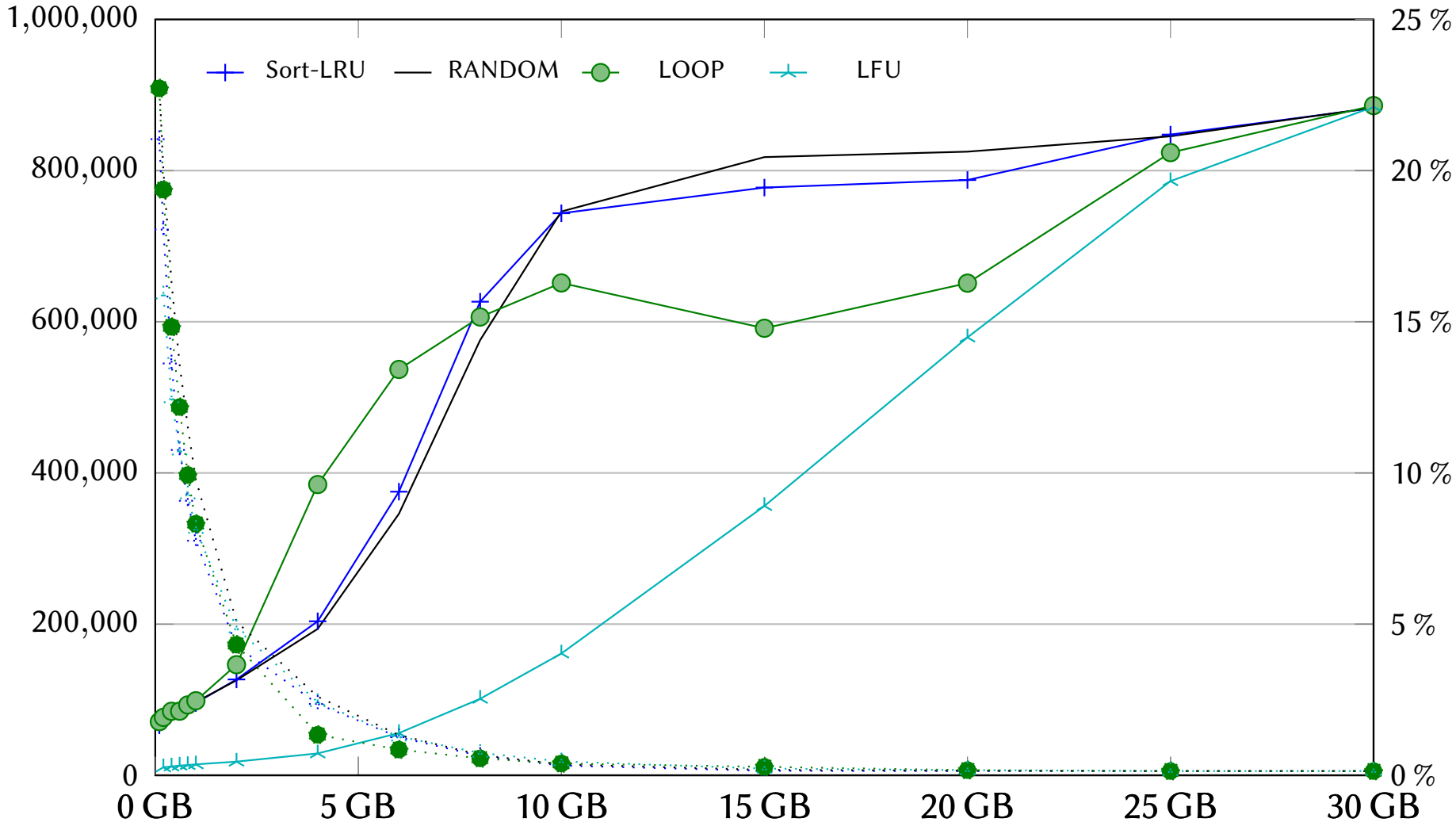
Performance Evaluation



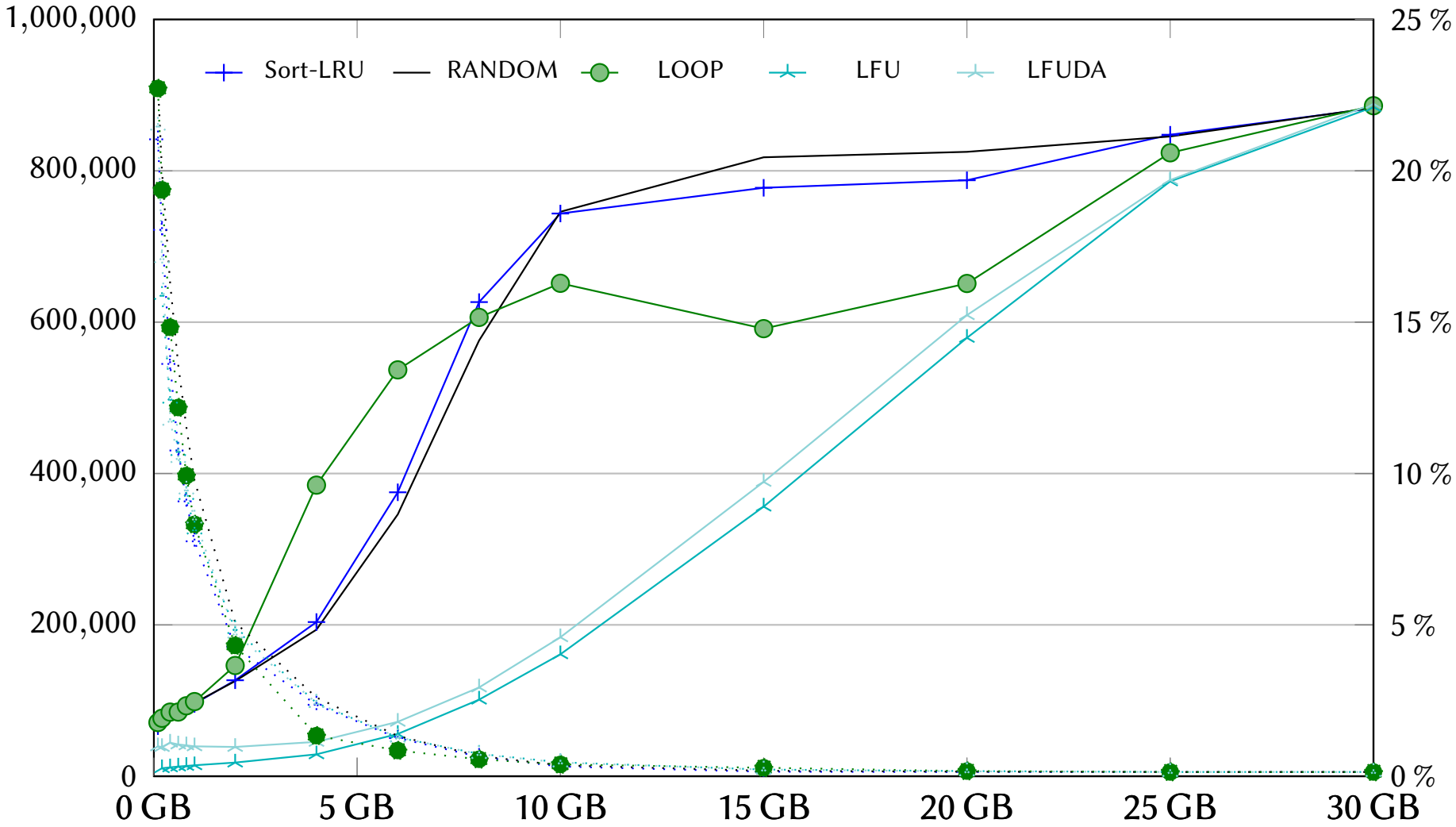
Performance Evaluation



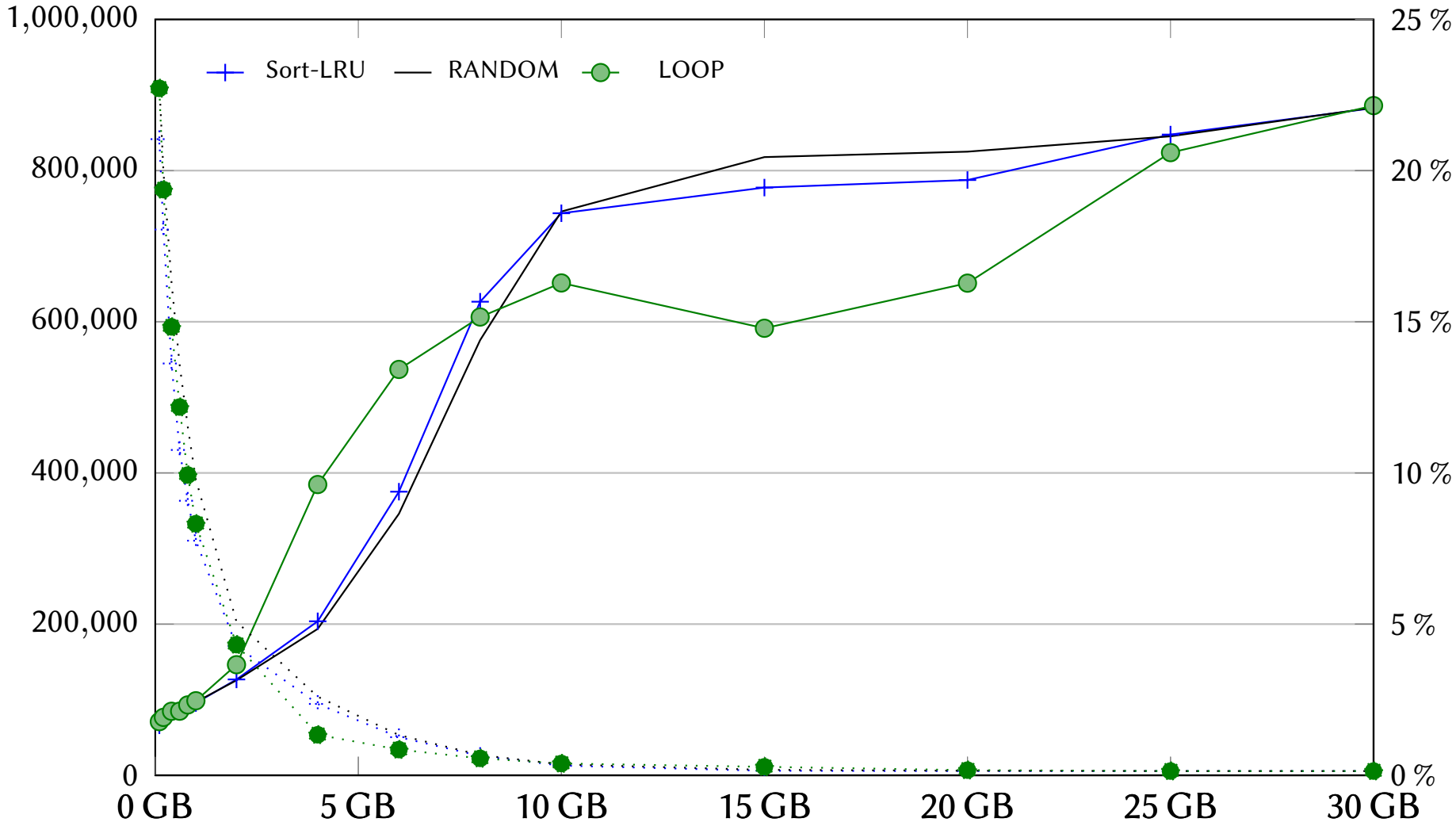
Performance Evaluation



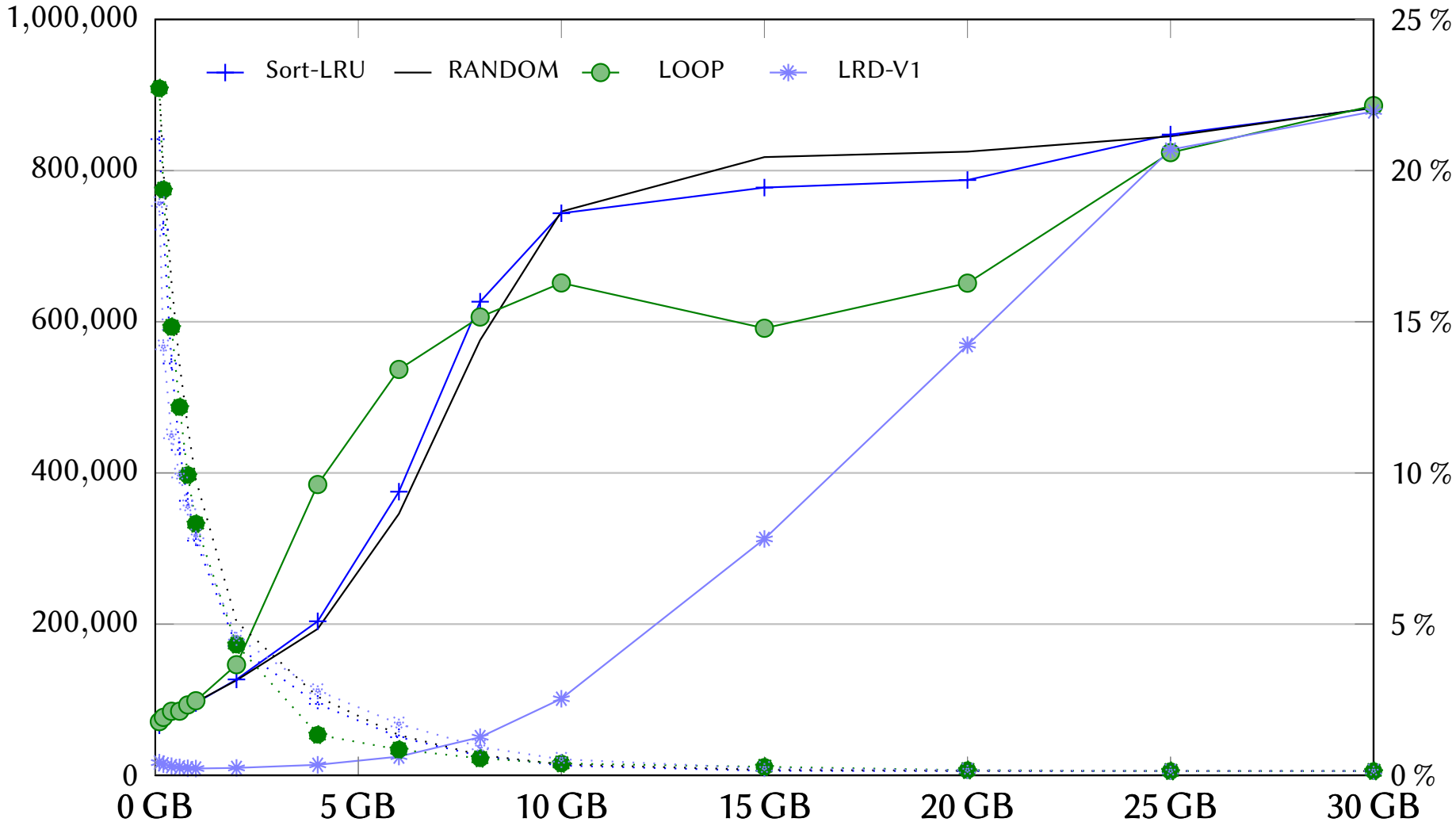
Performance Evaluation



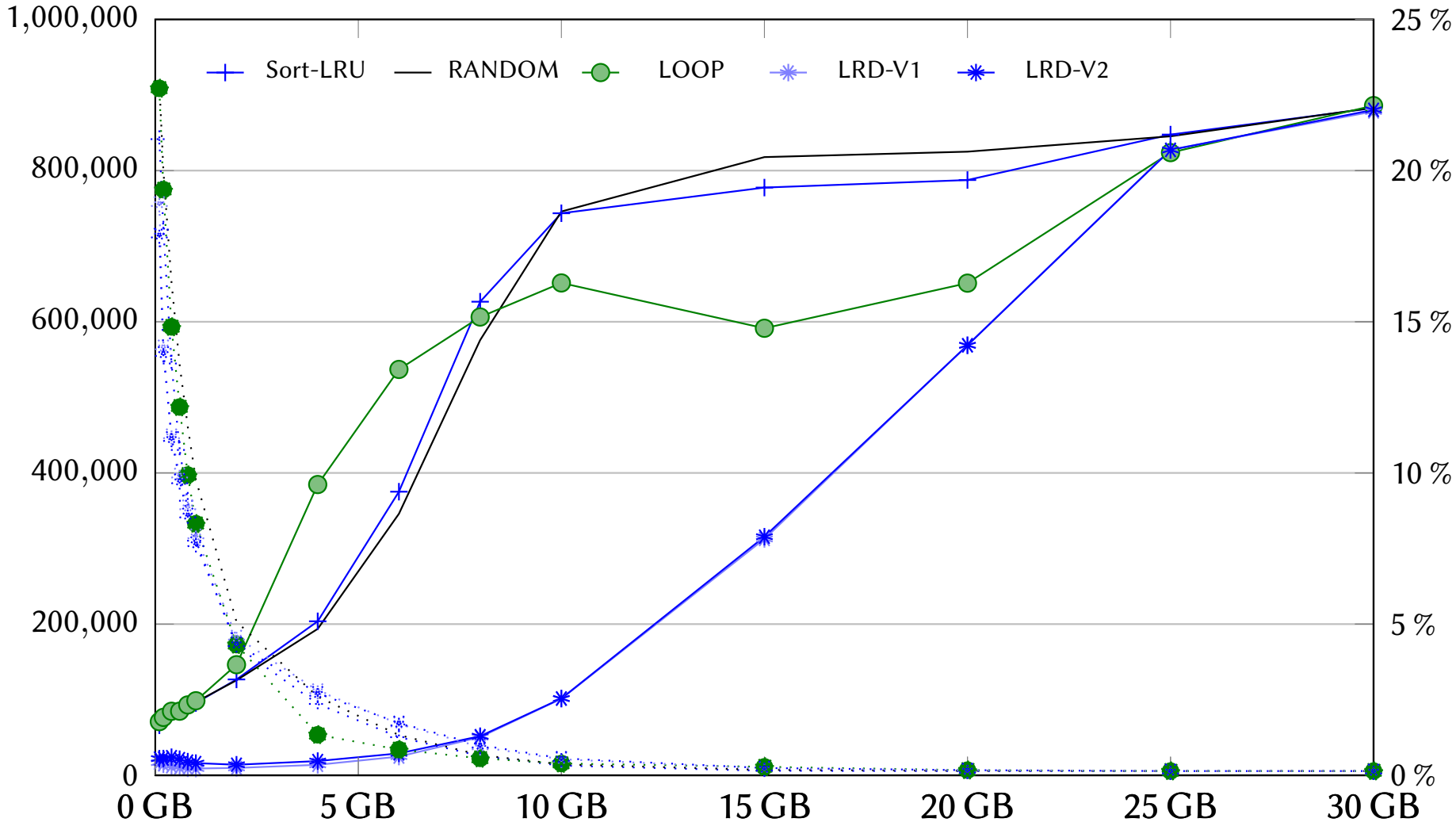
Performance Evaluation



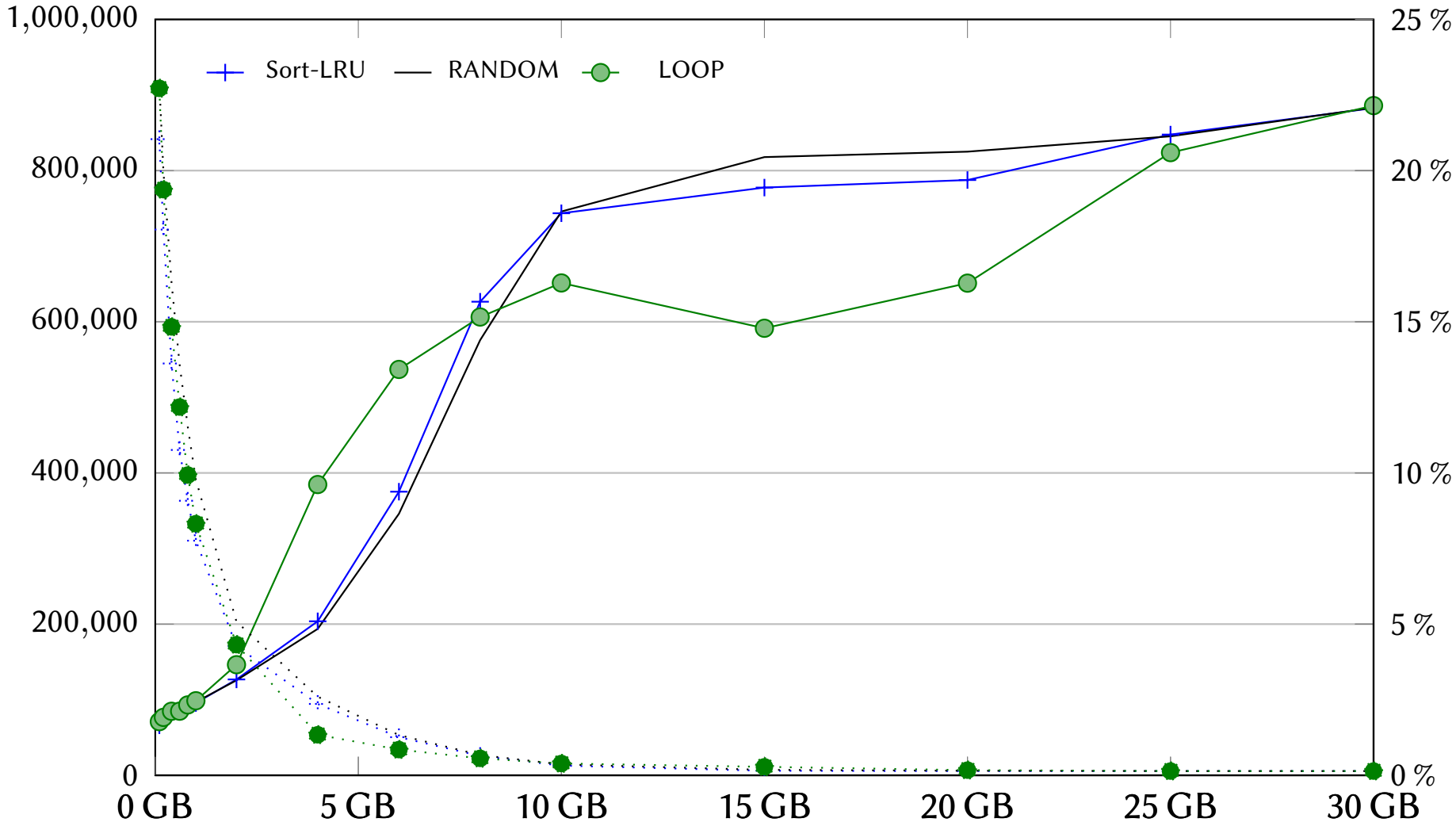
Performance Evaluation



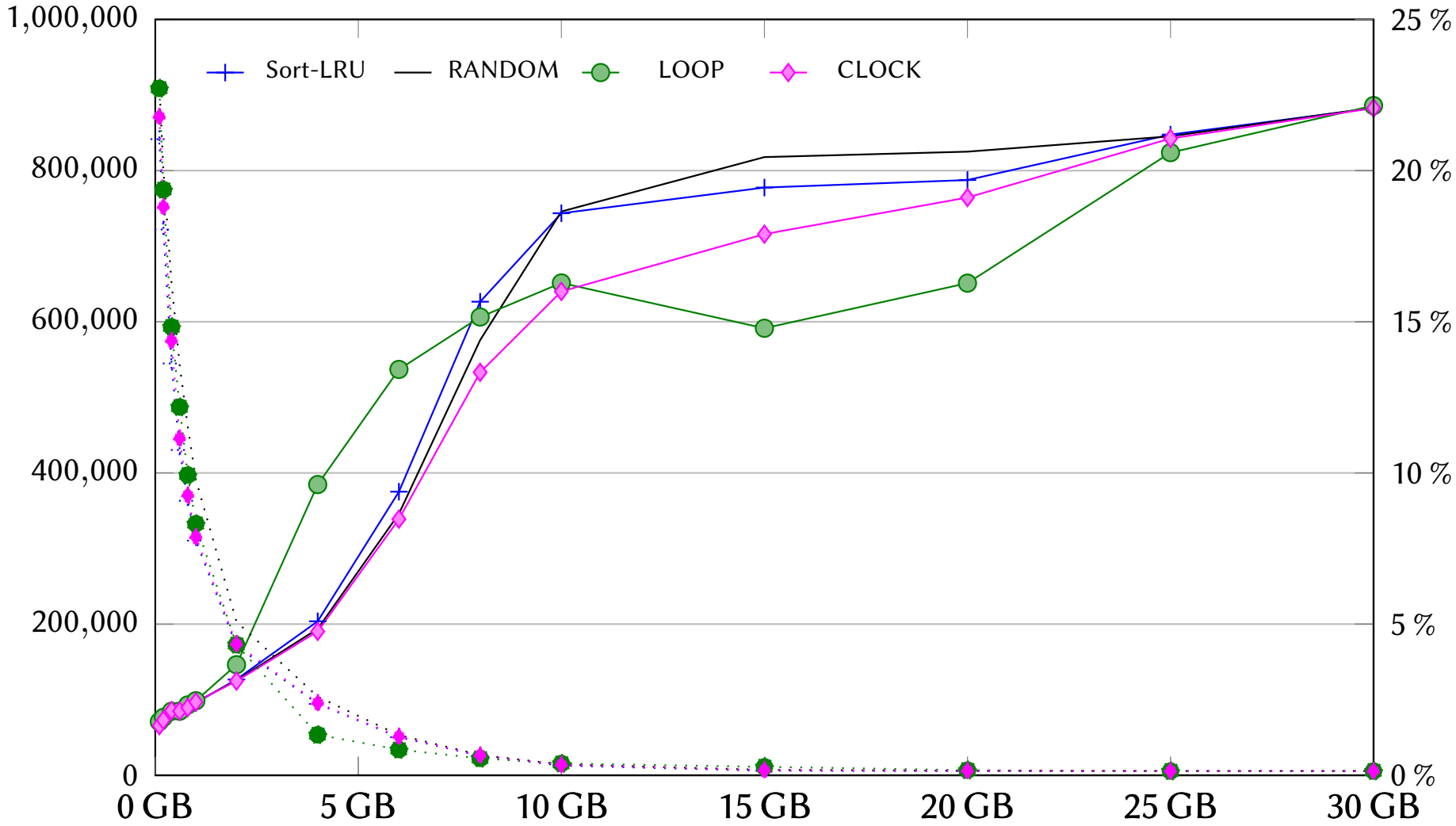
Performance Evaluation



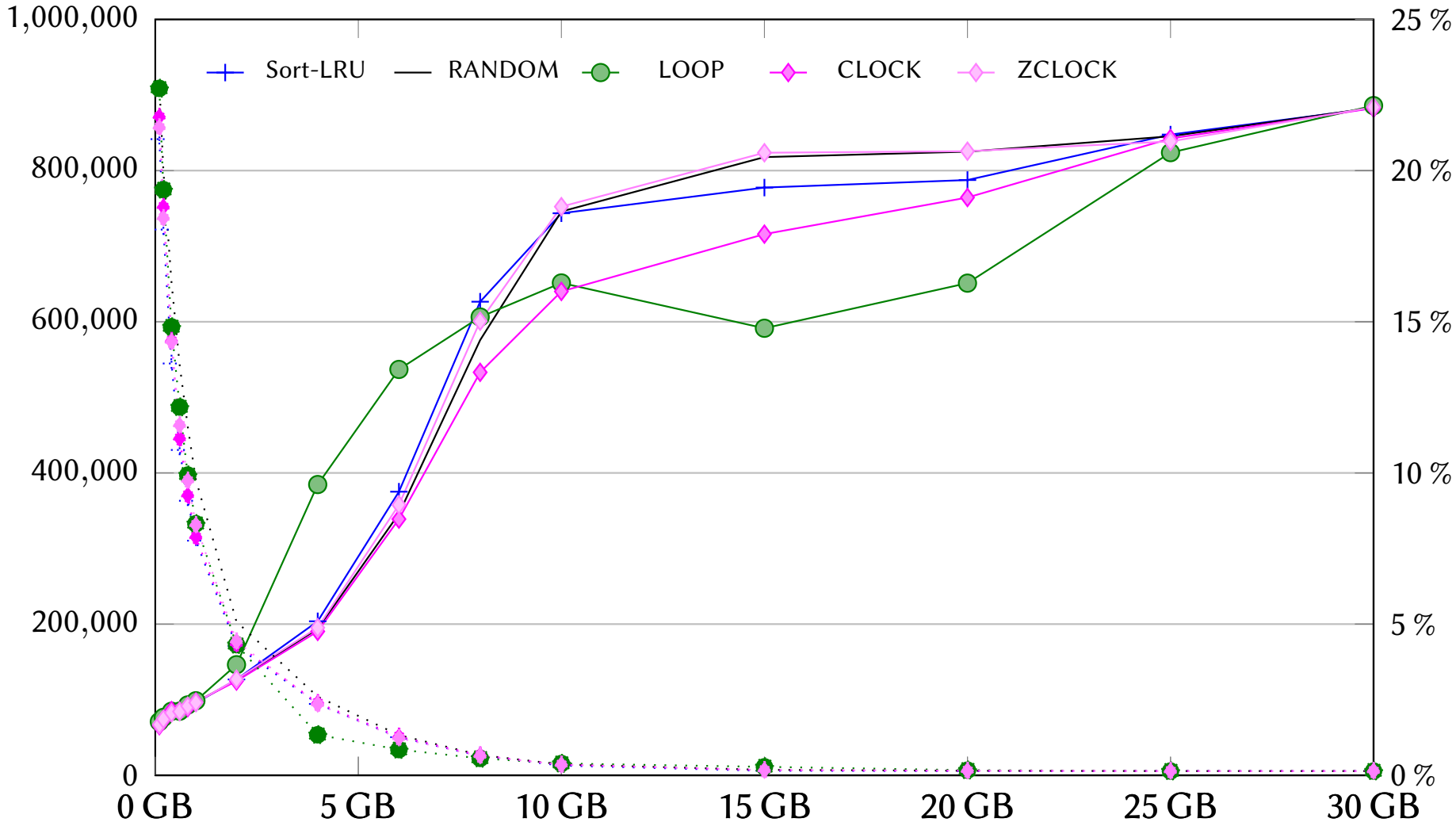
Performance Evaluation



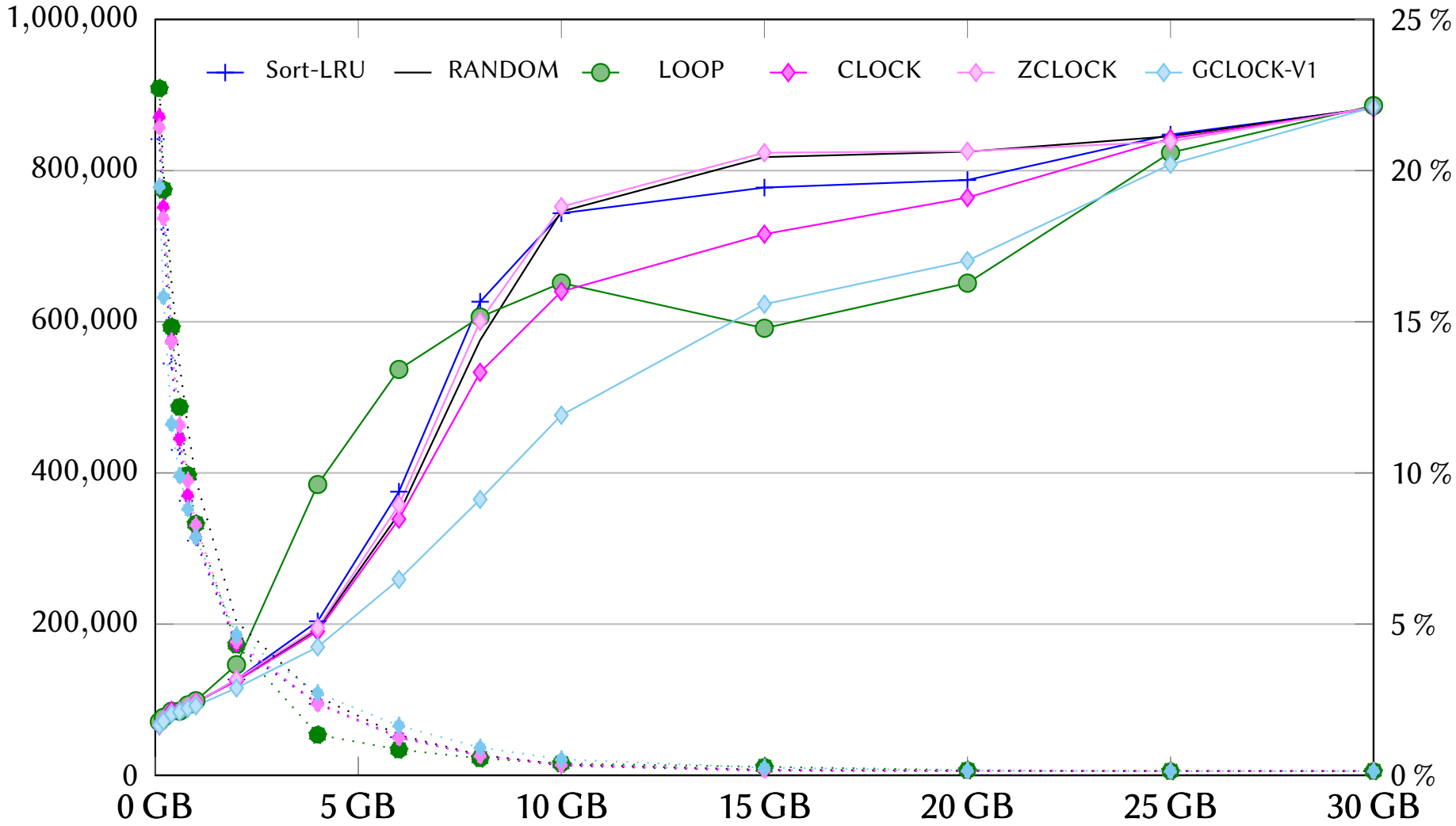
Performance Evaluation



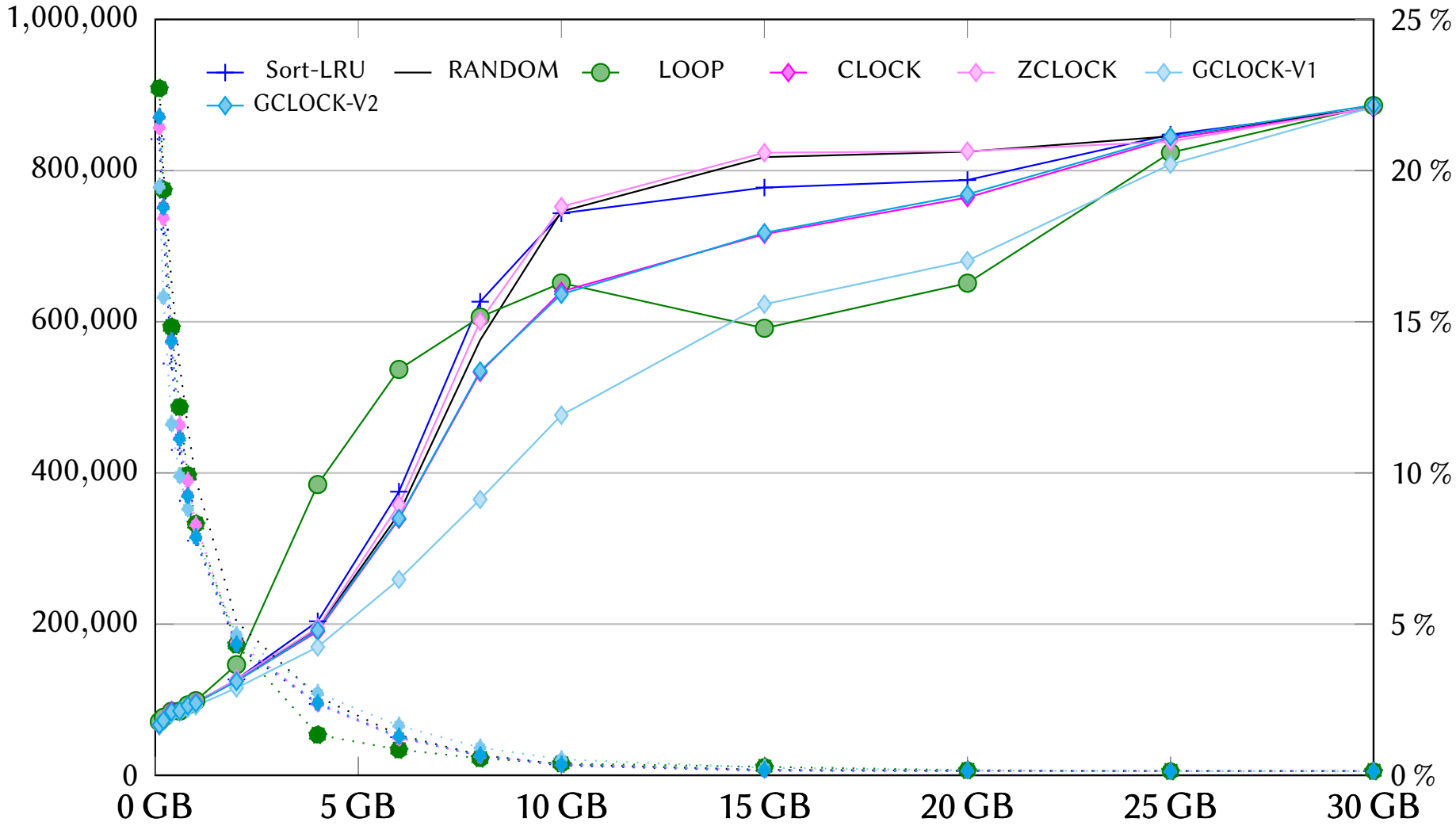
Performance Evaluation



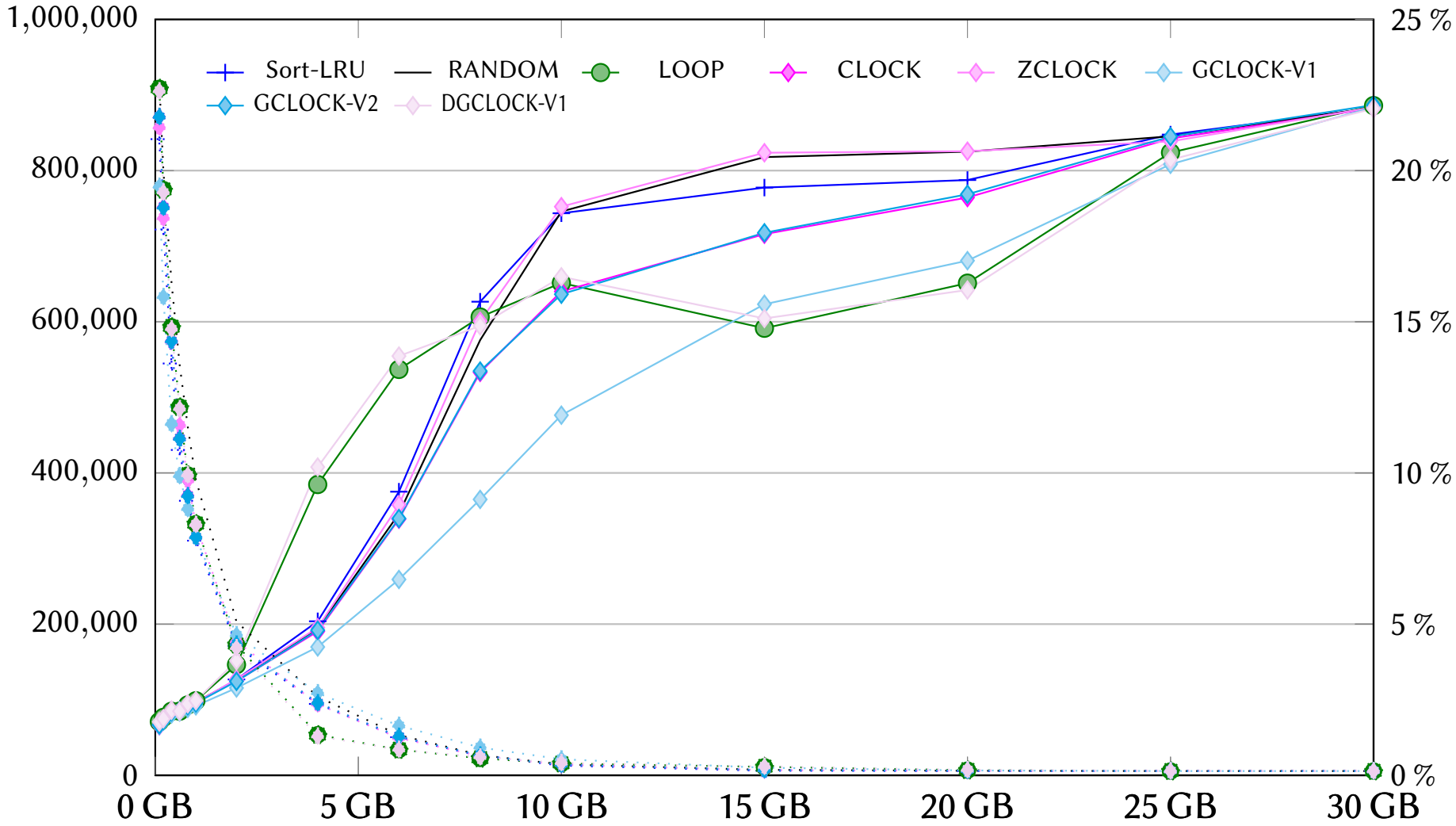
Performance Evaluation



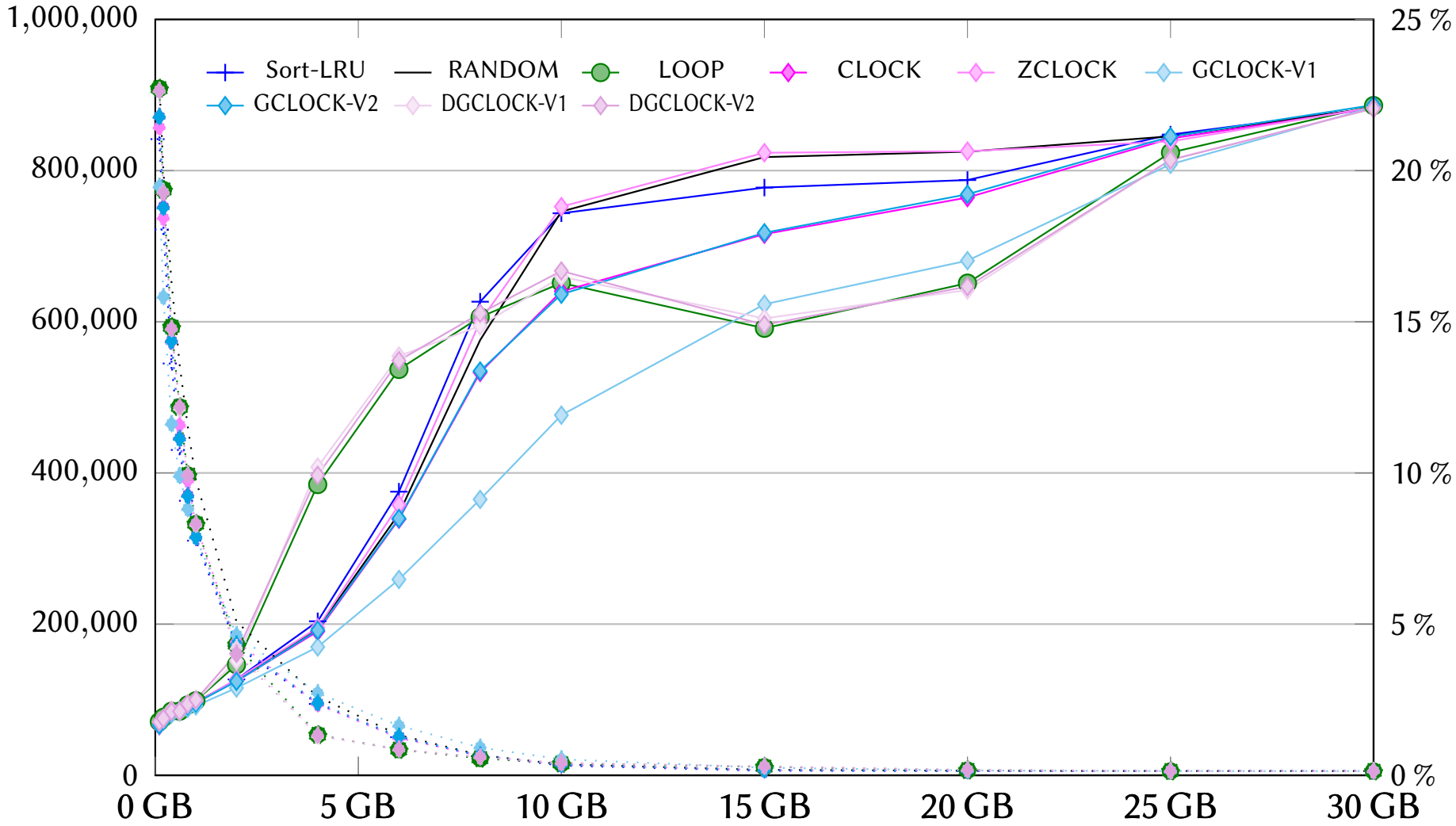
Performance Evaluation



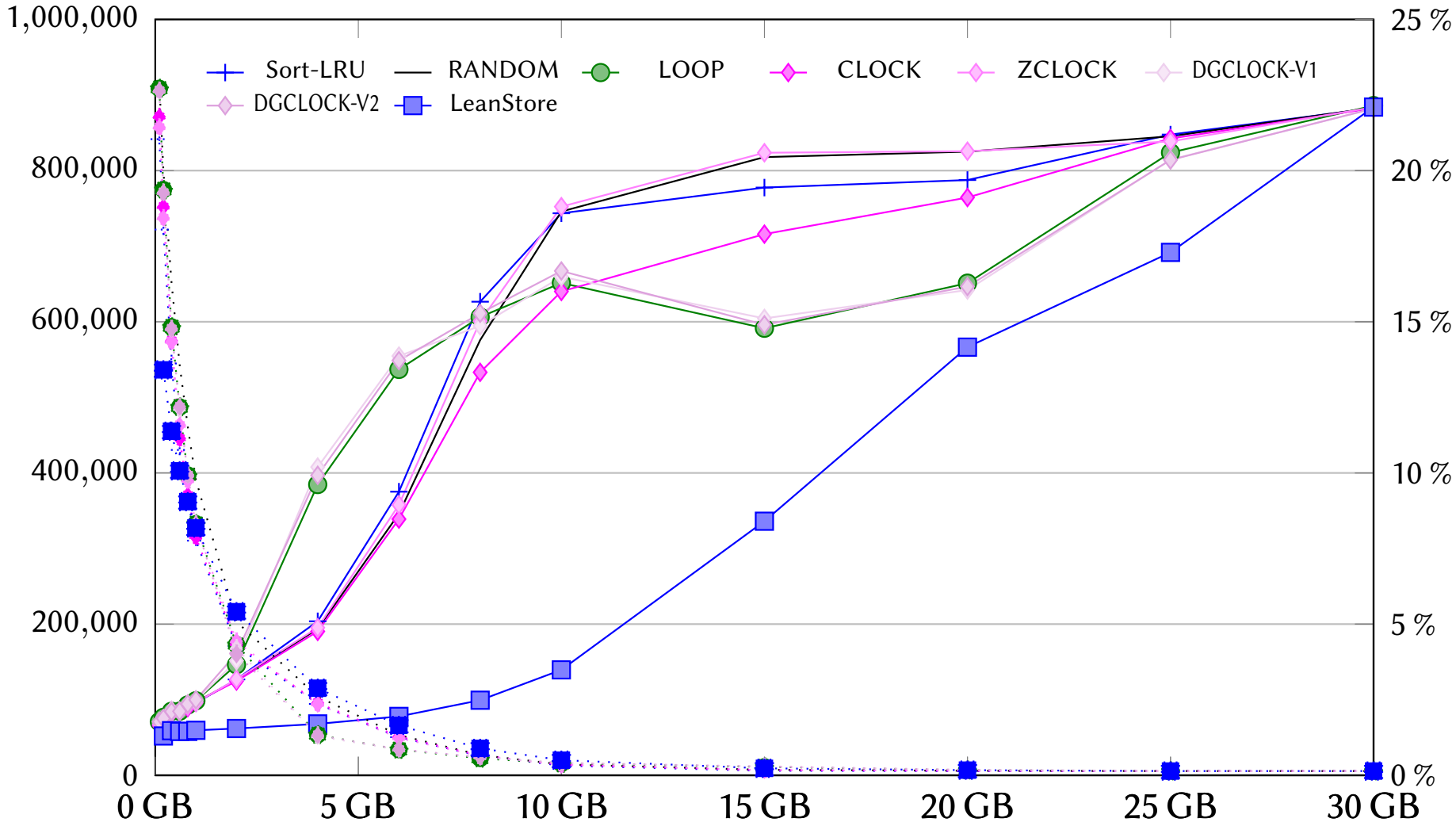
Performance Evaluation



Performance Evaluation



Performance Evaluation



Conclusion

Simplicity is the Key

RANDOM and LOOP outperform the more complex competitors

Conclusion

Simplicity is the Key

RANDOM and LOOP outperform the more complex competitors (at least in combination with *Zero*, *TPC-C* and the specific benchmark configuration)

Section 2

Component-Wise Performance Evaluation of OLTP Systems

Subsection 1

OLTP through the Looking Glass, and What We Found There [Har+]

OLTP Through the Looking Glass, and What We Found There

Stavros Harizopoulos
HP Labs
Palo Alto, CA
stavros@hp.com

Daniel J. Abadi
Yale University
New Haven, CT
dna@cs.yale.edu

Samuel Madden, Michael Stonebraker
Massachusetts Institute of Technology
Cambridge, MA
{madden, stonebraker}@csail.mit.edu

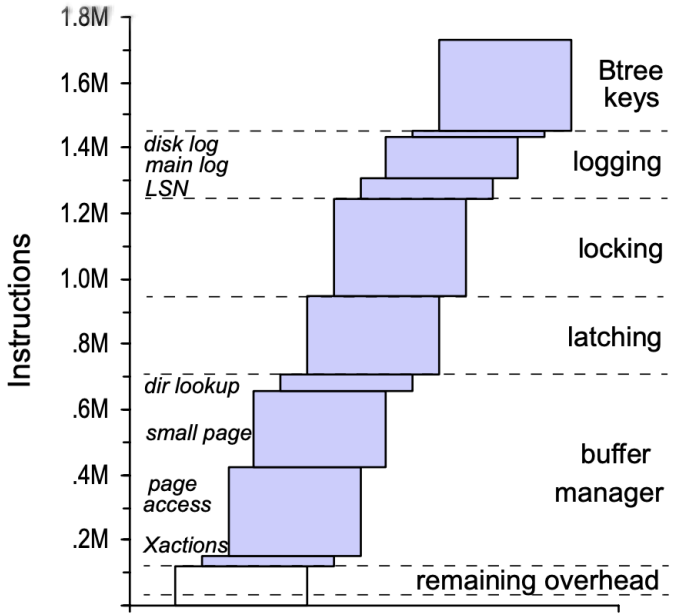


Figure 7. Expanding breakdown for New Order (see Section 3.2 for the labels on the left column).

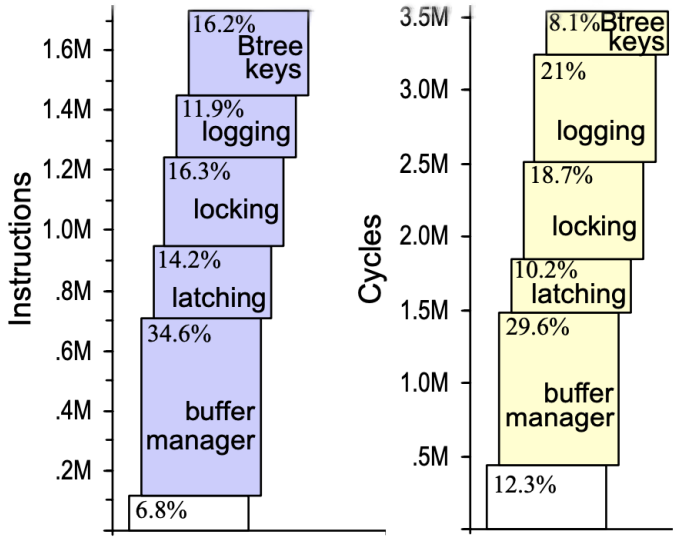


Figure 8. Instructions (left) vs. Cycles (right) for New Order.

The Author's Assumptions

The Author's Assumptions

- No more larger-than-memory DBs

The Author's Assumptions

- No more larger-than-memory DBs
- Thus, multithreaded and interleaved transaction execution no more needed

The Author's Assumptions

- No more larger-than-memory DBs
- Thus, multithreaded and interleaved transaction execution no more needed
- Many applications do not need ACID transactions

The Author's Assumptions

- No more larger-than-memory DBs
- Thus, multithreaded and interleaved transaction execution no more needed
- Many applications do not need ACID transactions
- Replication instead of logging for recovery

The Author's Methodology

The Author's Methodology

Gradually ...

The Author's Methodology

Gradually ...

- ... make the DBS in-memory—removing buffer pool—, ...

The Author's Methodology

Gradually ...

- ... make the DBS in-memory—removing buffer pool—, ...
- ... remove latching

The Author's Methodology

Gradually ...

- ... make the DBS in-memory—removing buffer pool—, ...
- ... remove latching and locking, ...

The Author's Methodology

Gradually ...

- ... make the DBS in-memory—removing buffer pool—, ...
- ... remove latching and locking, ...
- ... remove transaction logging and LSN maintenance, ...

The Author's Methodology

Gradually ...

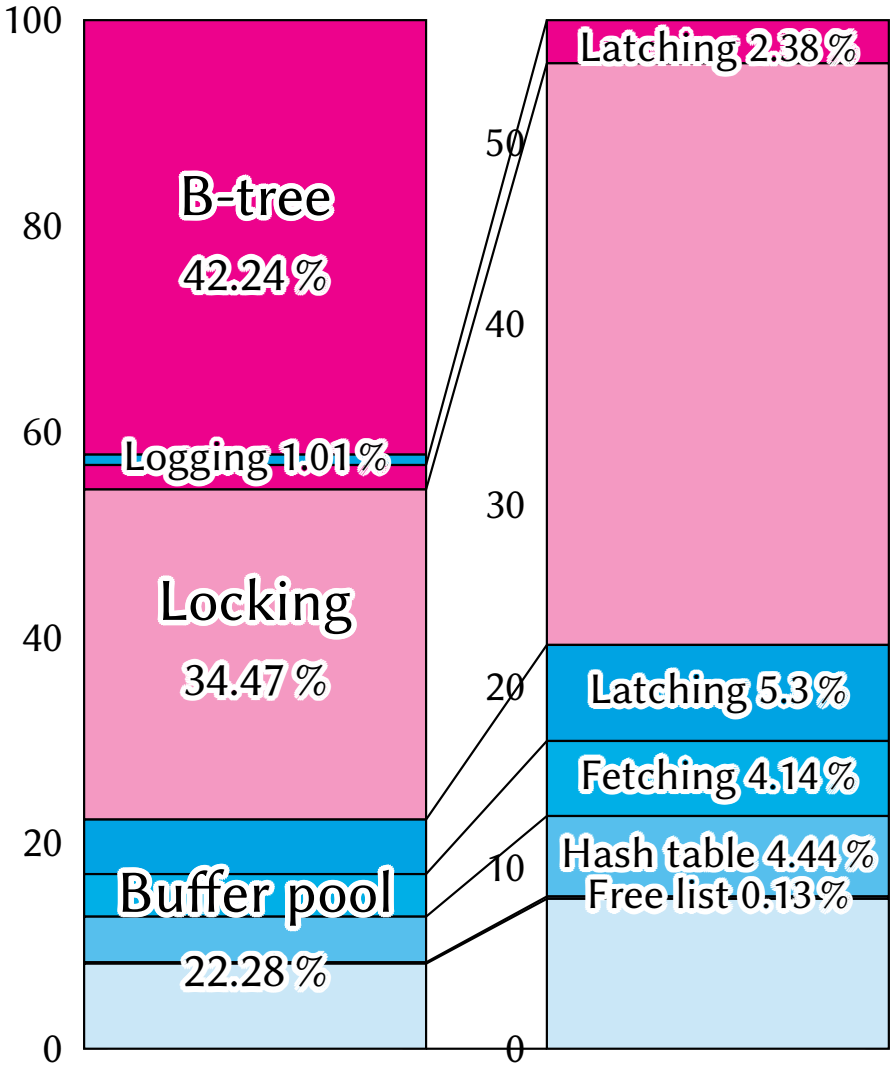
- ... make the DBS in-memory—removing buffer pool—, ...
- ... remove latching and locking, ...
- ... remove transaction logging and LSN maintenance, ...
- ... and hand-optimize B-tree code for in-memory DBS.

Subsection 2

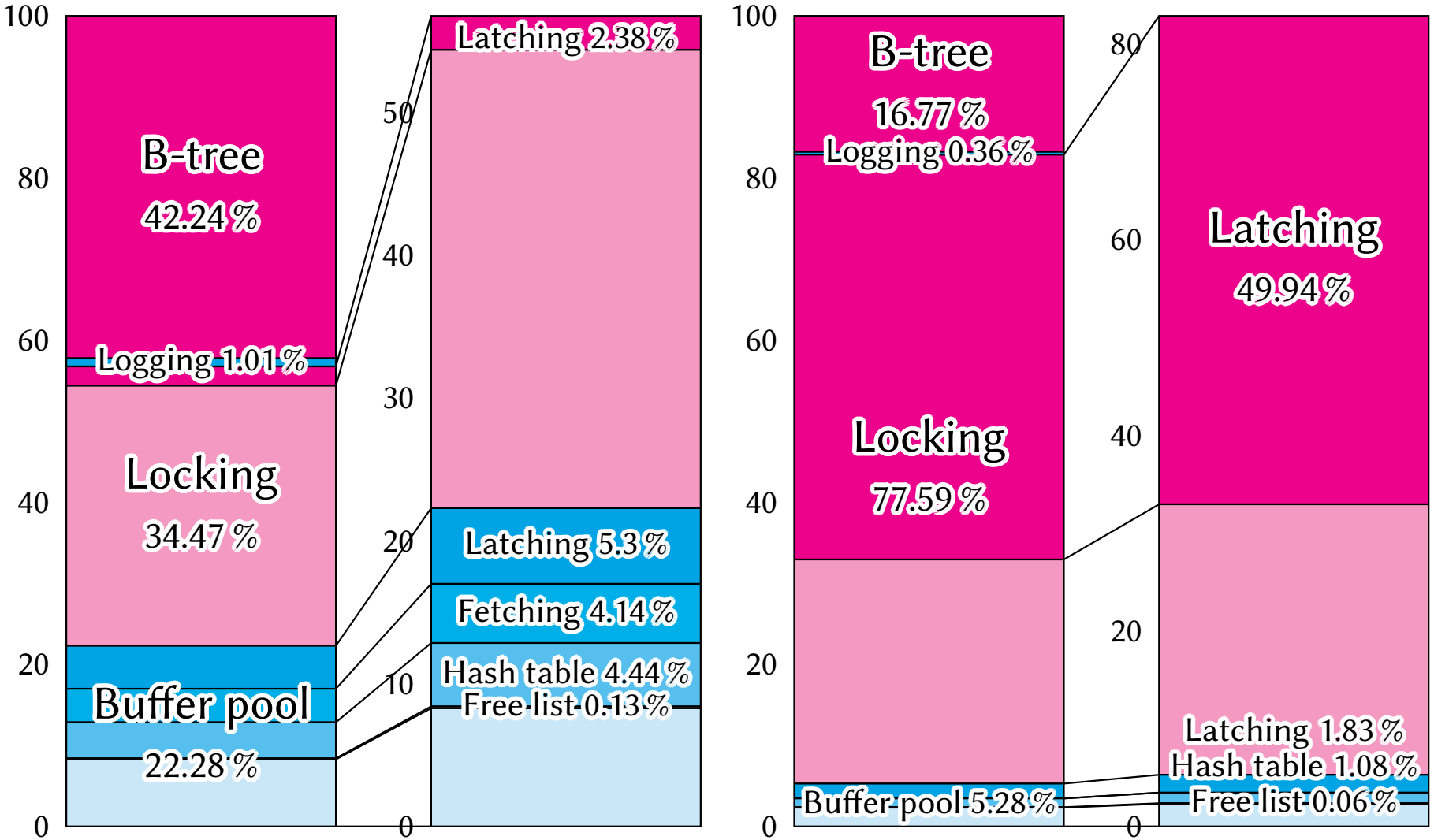
Results

Read-Only YCSB

Read-Only YCSB

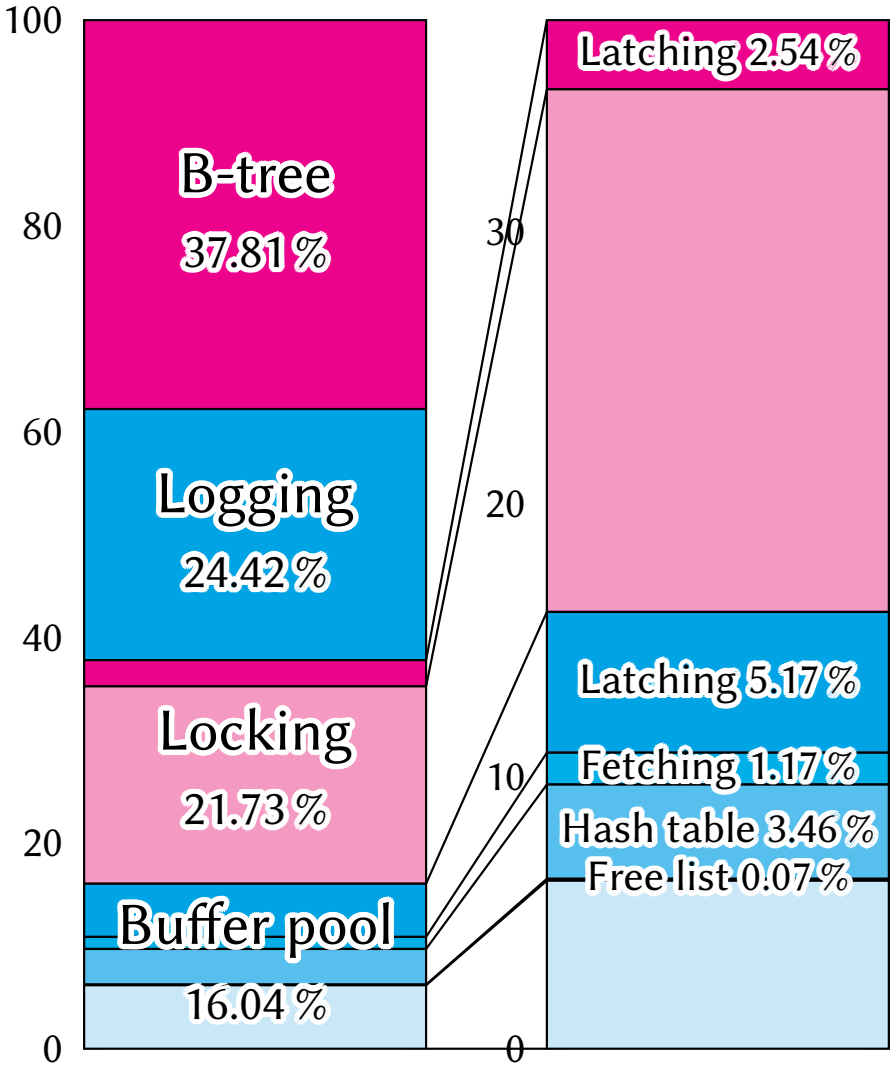


Read-Only YCSB

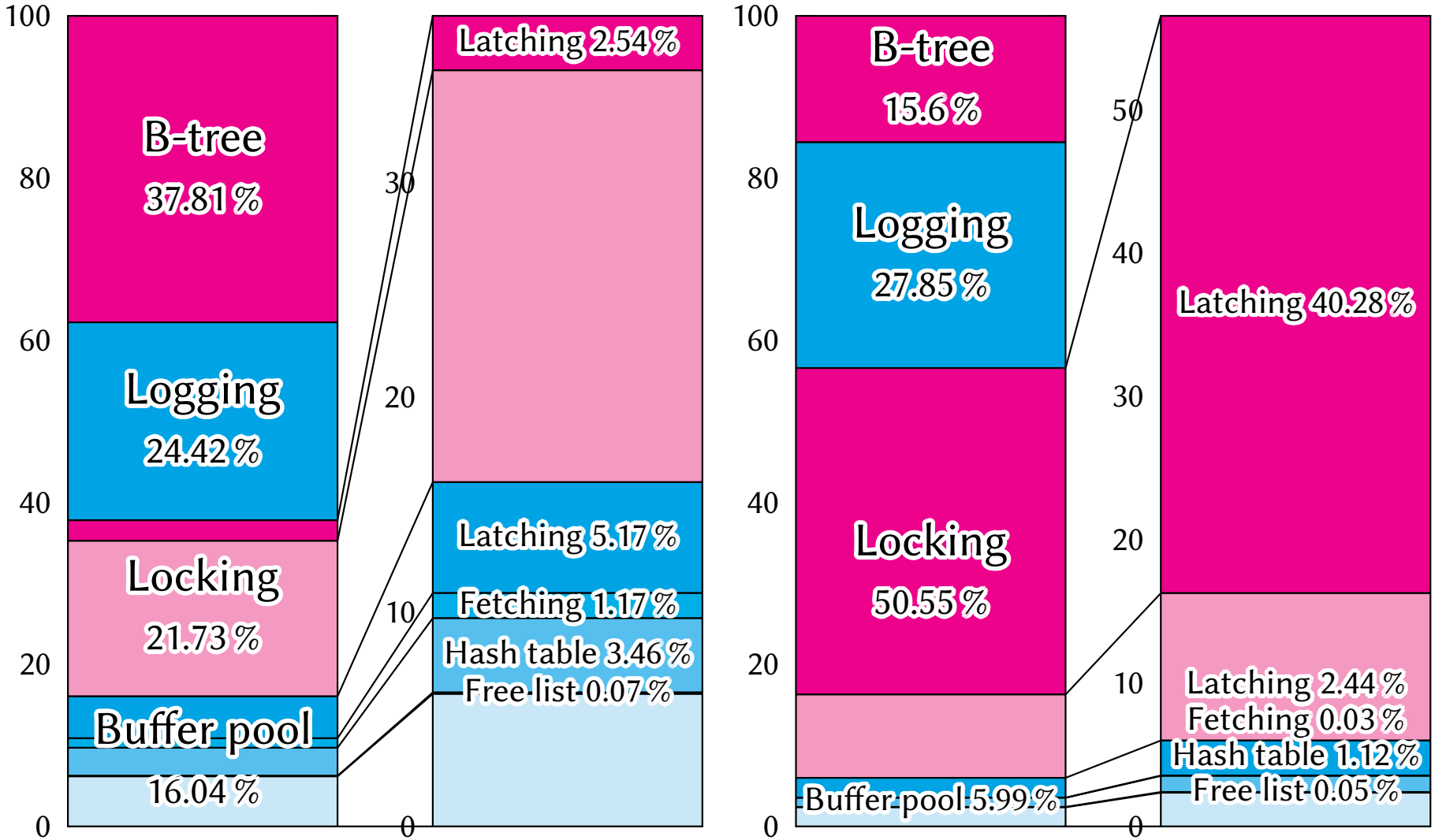


Update-Only YCSB

Update-Only YCSB

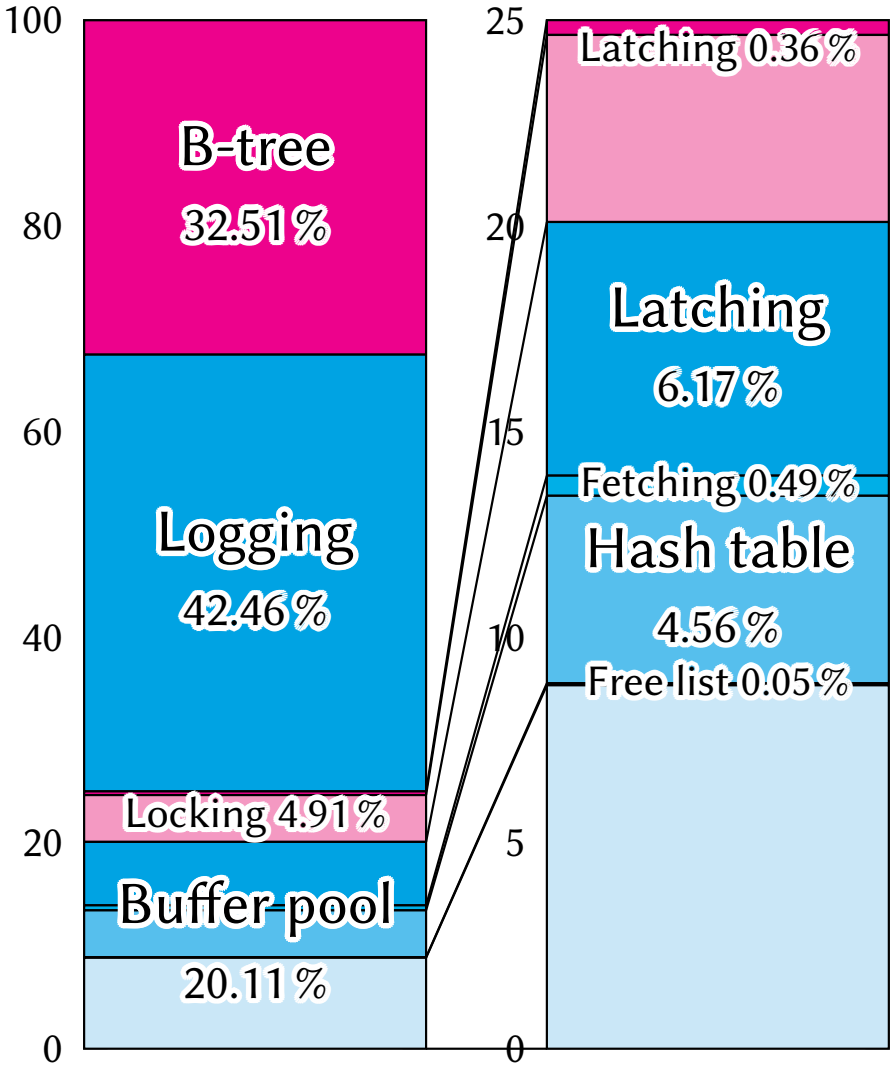


Update-Only YCSB

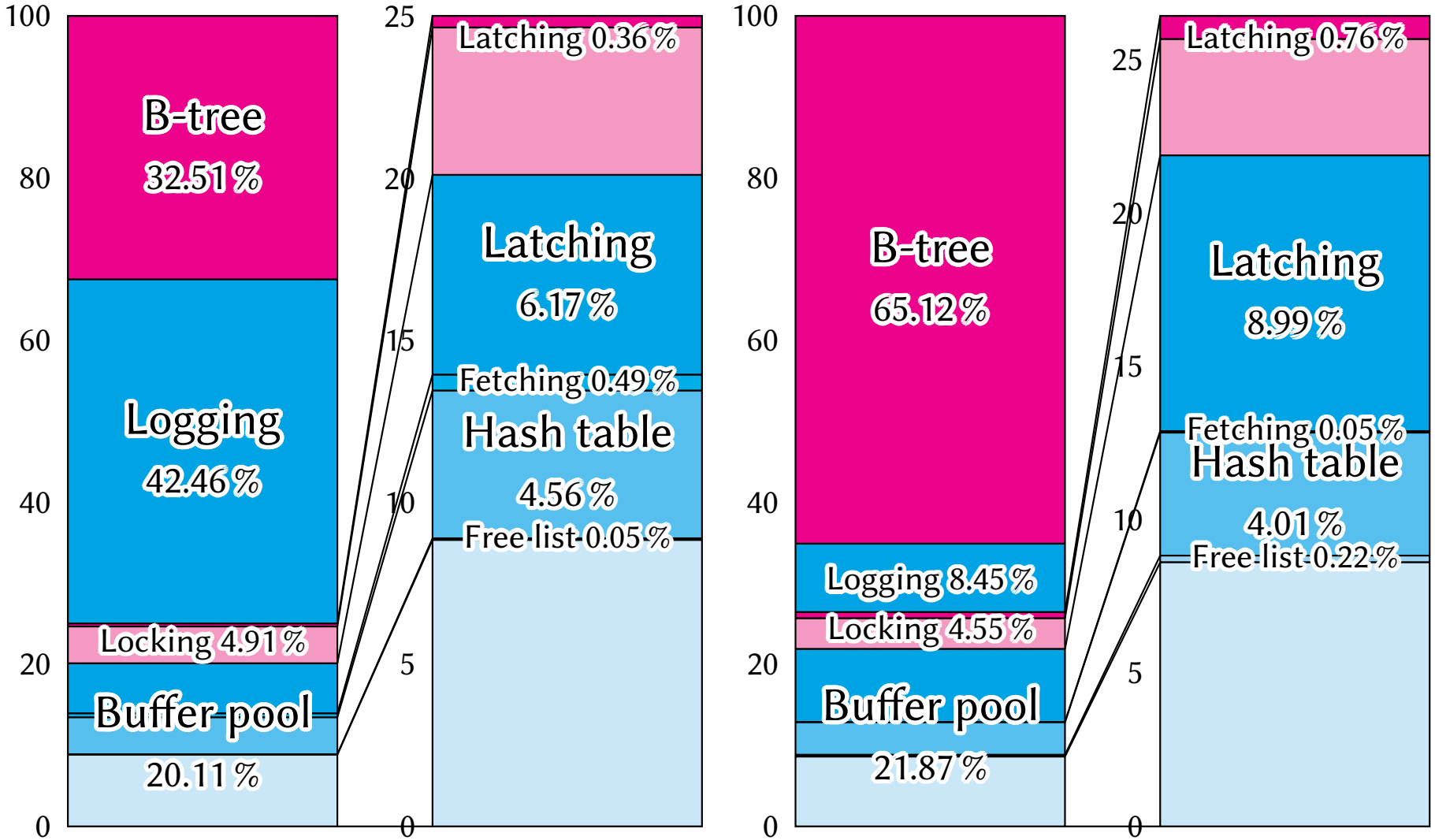


TPC-C

TPC-C


100
80
60
40
20
0
B-tree
32.51 %
Logging
42.46 %
Locking 4.91 %
Buffer pool
20.11 %
25
20
15
10
5
0
Latching 0.36 %
Latching
6.17 %
Fetching 0.49 %
Hash table
4.56 %
Free list 0.05 %

TPC-C


100
80
60
40
20
0
B-tree
32.51 %
Logging
42.46 %
Locking 4.91 %
Buffer pool
20.11 %
25
20
15
10
5
0
Latching 0.36 %
Latching
6.17 %
Fetching 0.49 %
Hash table
4.56 %
Free list 0.05 %
100
80
60
40
20
0
B-tree
65.12 %
Logging 8.45 %
Locking 4.55 %
Buffer pool
21.87 %
25
20
15
10
5
0
Latching 0.76 %
Latching
8.99 %
Fetching 0.05 %
Hash table
4.01 %
Free list 0.22 %

Conclusion

Conclusion

Complex OLTP (and OLAP)

The index structures (here Foster B-tree) are the most performance-critical components.

Conclusion

Complex OLTP (and OLAP)

The index structures (here Foster B-tree) are the most performance-critical components.

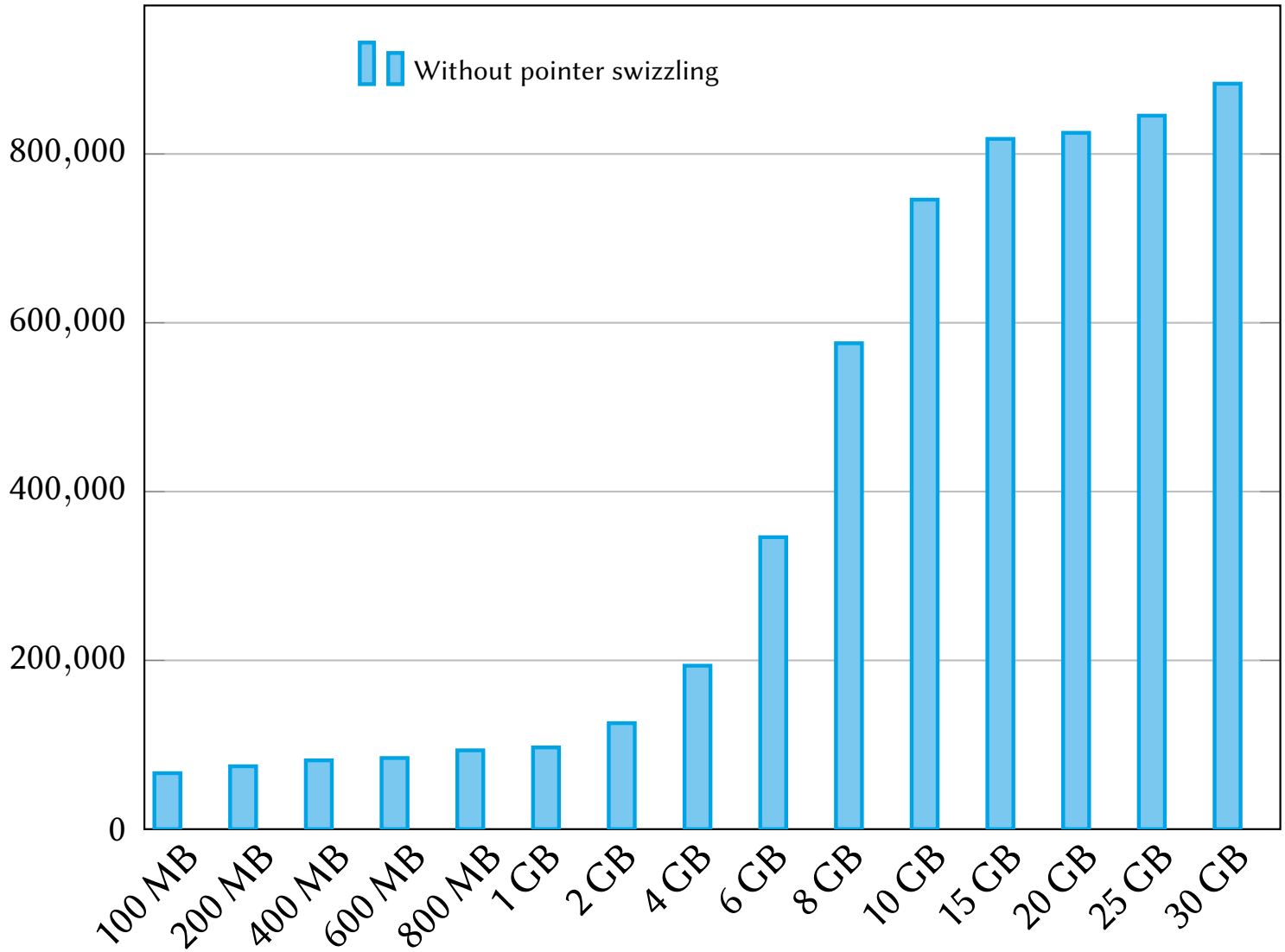
Simple Workloads

A global latch in the transaction manager limits the performance.

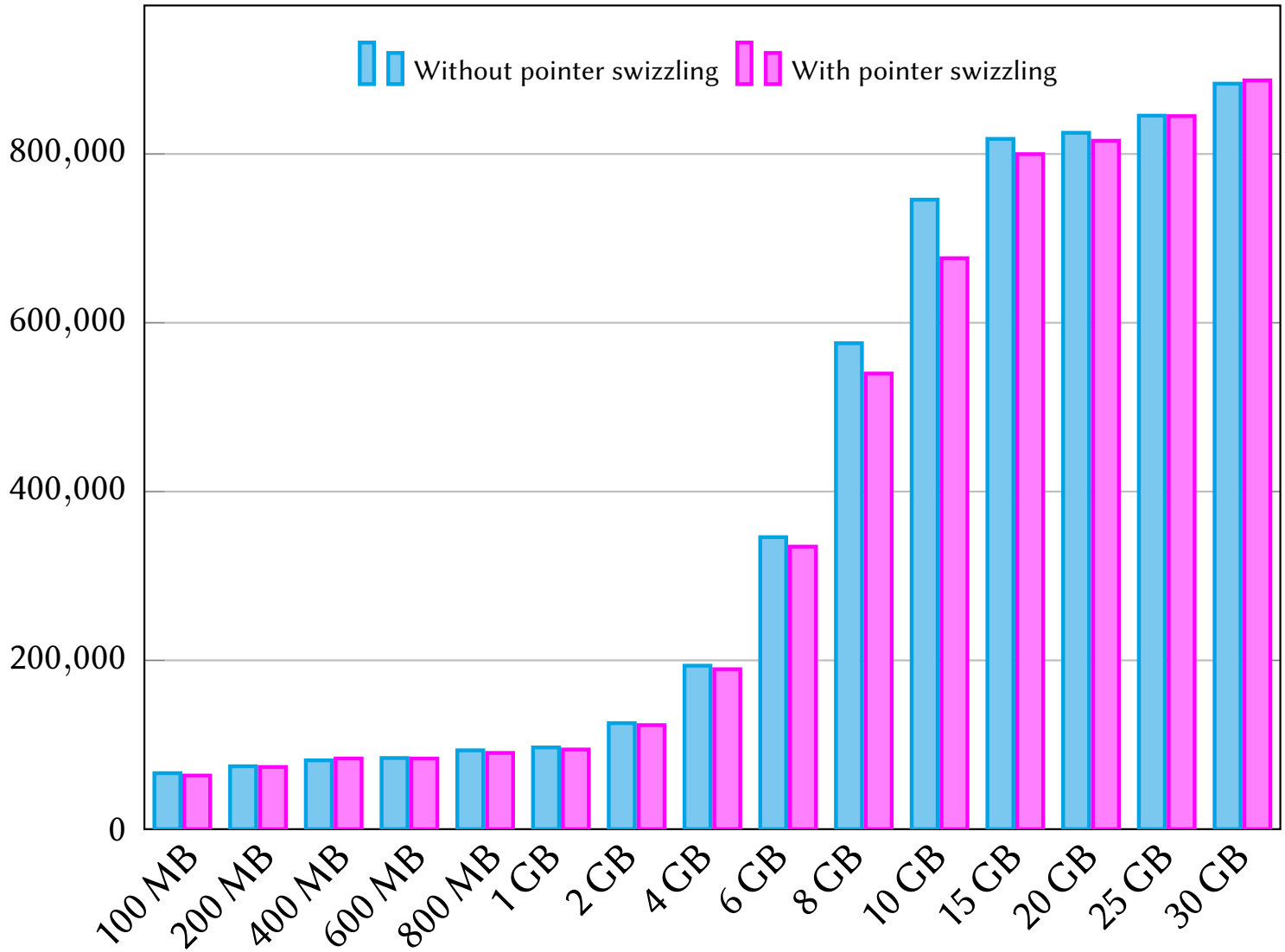
Subsection 4

Optimizations of OLTP Systems

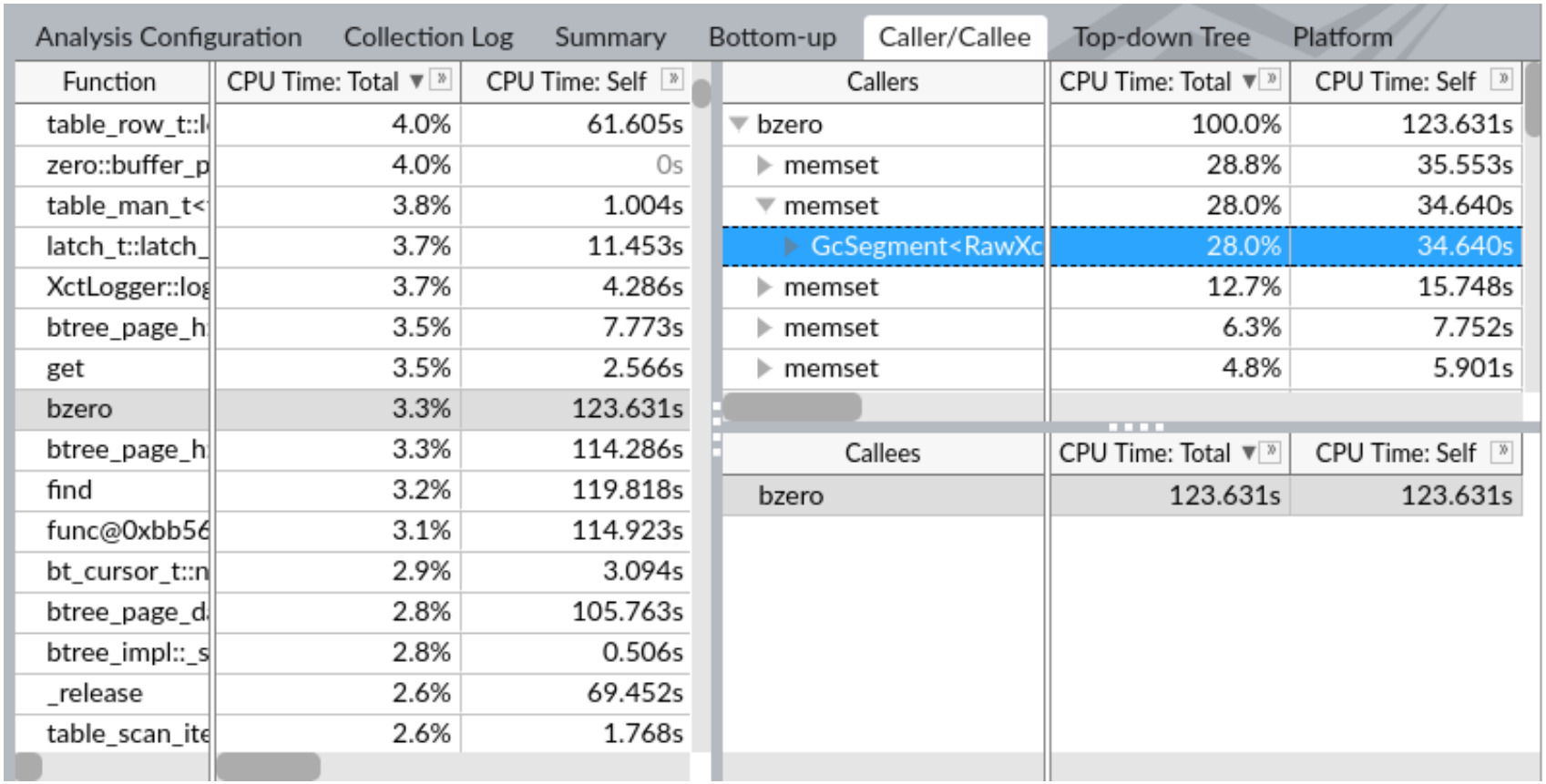
Pointer Swizzling in the Buffer Pool [Gra+]



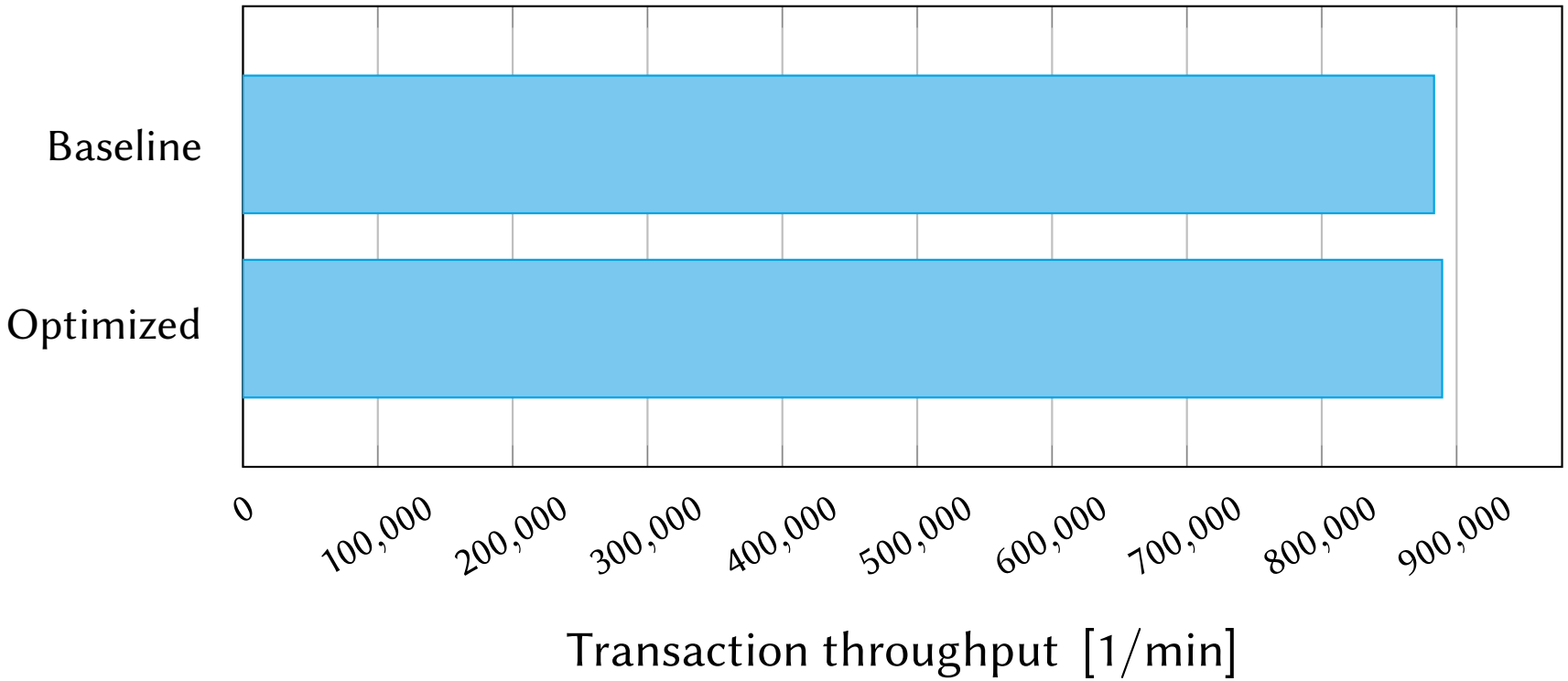
Pointer Swizzling in the Buffer Pool [Gra+]



System Call: bzero



System Call: bzero



References I

- László A. Bélády. “A Study of Replacement Algorithms for Virtual-Storage Computer”.
- F. J. Corbató. “A Paging Experiment with the Multics System”. In.
- Wolfgang Effelsberg and Theo Härder. “Principles of Database Buffer Management”.
- Goetz Graefe et al. “In-Memory Performance for Big Data”.
- Stavros Harizopoulos et al. “OLTP through the Looking Glass, and What We Found There”.
- Viktor Leis et al. “LeanStore: In-Memory Data Management beyond Main Memory”.

References II

Elizabeth J. O'Neil, Patrick E. O'Neil, and Gerhard Weikum. "The LRU-K Page Replacement Algorithm for Database Disk Buffering".

Your Turn to Ask ...

Page Replacement Algorithm Classification by Bélády From [Bél]

Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 No statistics

Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 RANDOM, FIFO and FILO

Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 RANDOM, FIFO and FILO

Class 2 Statistics about the latest references of *pages in the buffer pool*

Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 RANDOM, FIFO and FILO

Class 2 LRU, MRU, LRU-K, SLRU, CLOCK, ZCLOCK, GCLOCK, DGCLOCK, LRD, LFU, LFU-Aging, LFUDA, 2Q, MQ and LeanStore

Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 RANDOM, FIFO and FILO

Class 2 LRU, MRU, LRU-K, SLRU, CLOCK, ZCLOCK, GCLOCK, DGCLOCK, LRD, LFU, LFU-Aging, LFUDA, 2Q, MQ and LeanStore

Class 3 Statistics about each time *any page* was fetched from the database and each time it was evicted from the buffer pool

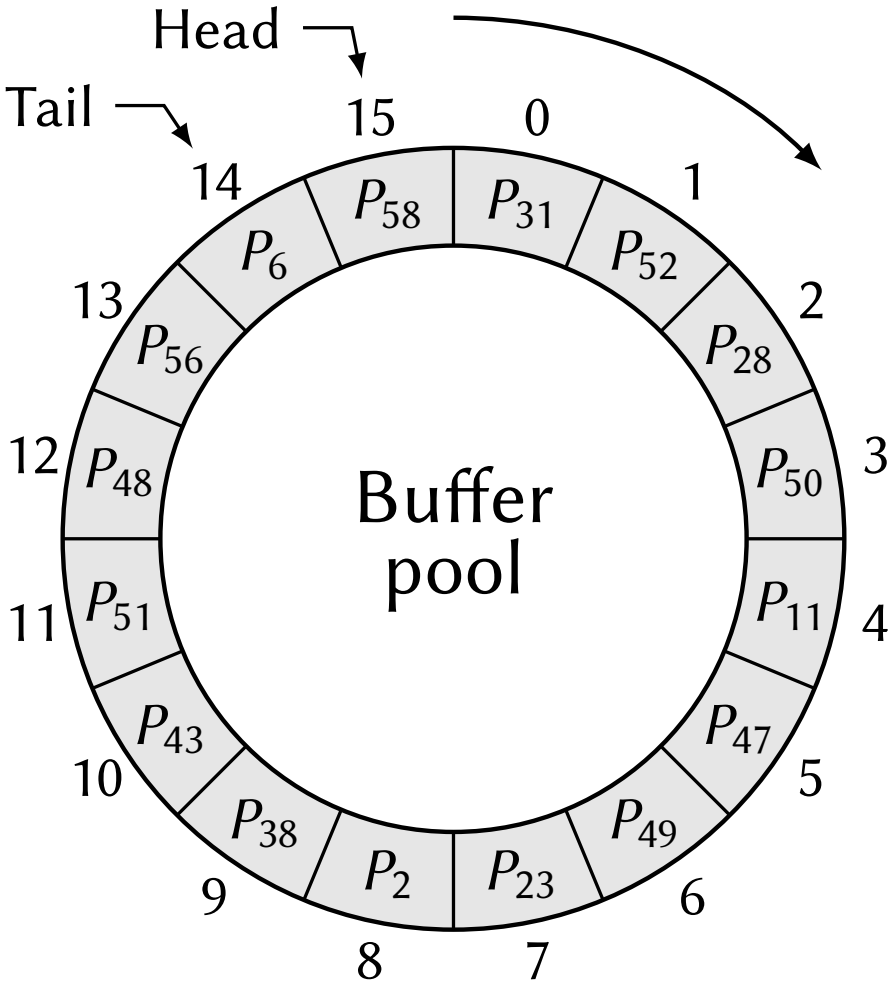
Page Replacement Algorithm Classification by Bélády From [Bél]

Class 1 RANDOM, FIFO and FILO

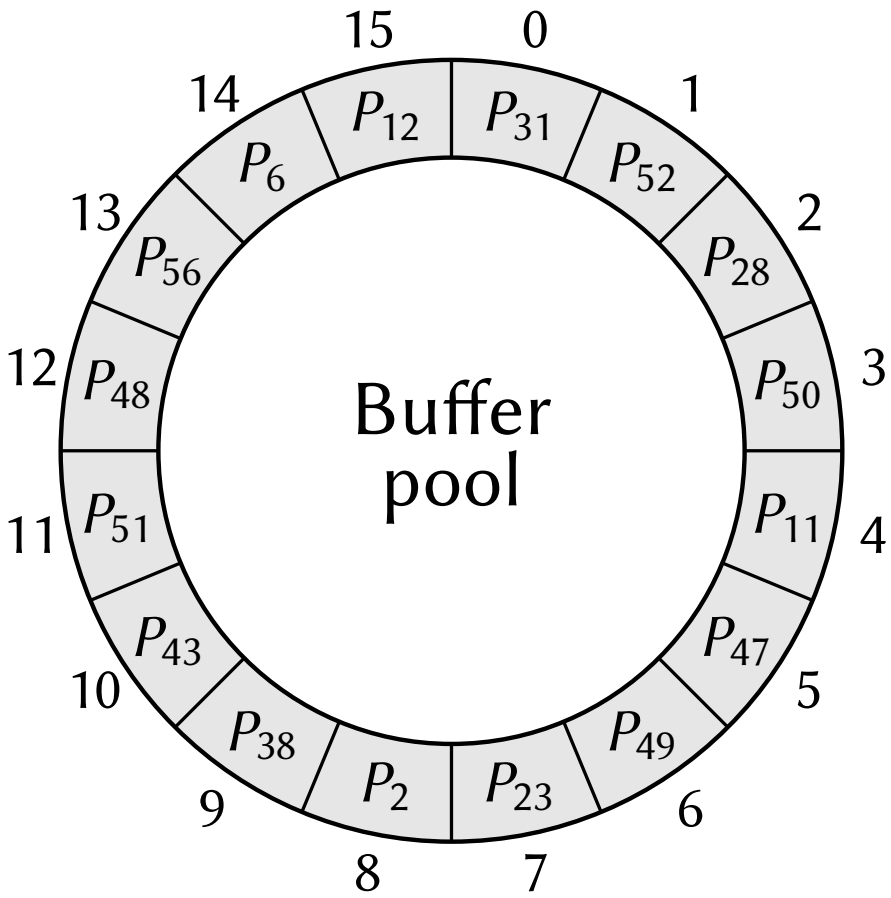
Class 2 LRU, MRU, LRU-K, SLRU, CLOCK, ZCLOCK, GCLOCK, DGCLOCK, LRD, LFU, LFU-Aging, LFUDA, 2Q, MQ and LeanStore

Class 3 ARC, CAR, CART, LIRS, CLOCK-Pro, DLIRS and CLOCK-Pro+

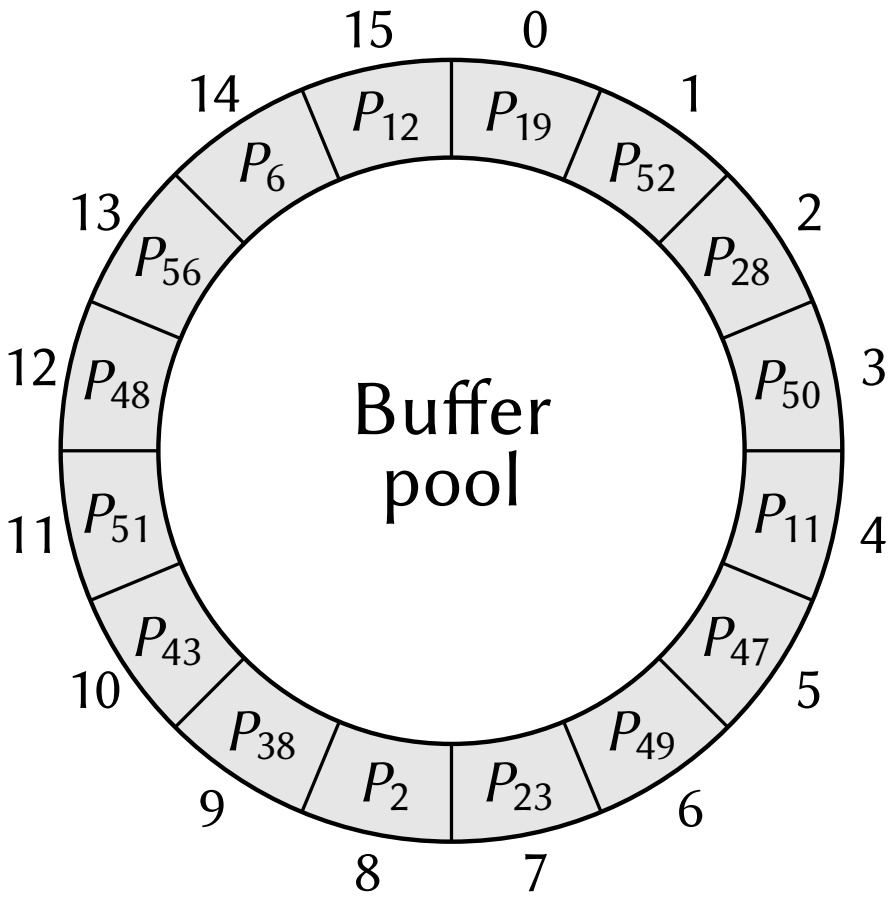
LOOP Algorithm



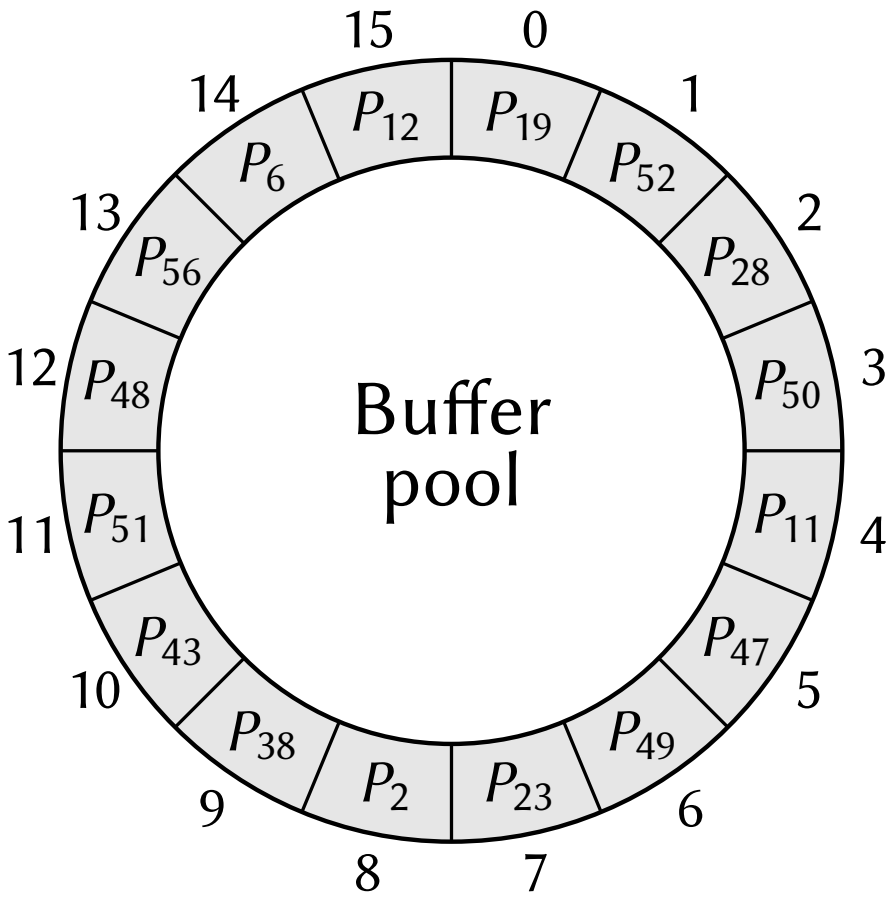
LOOP Algorithm



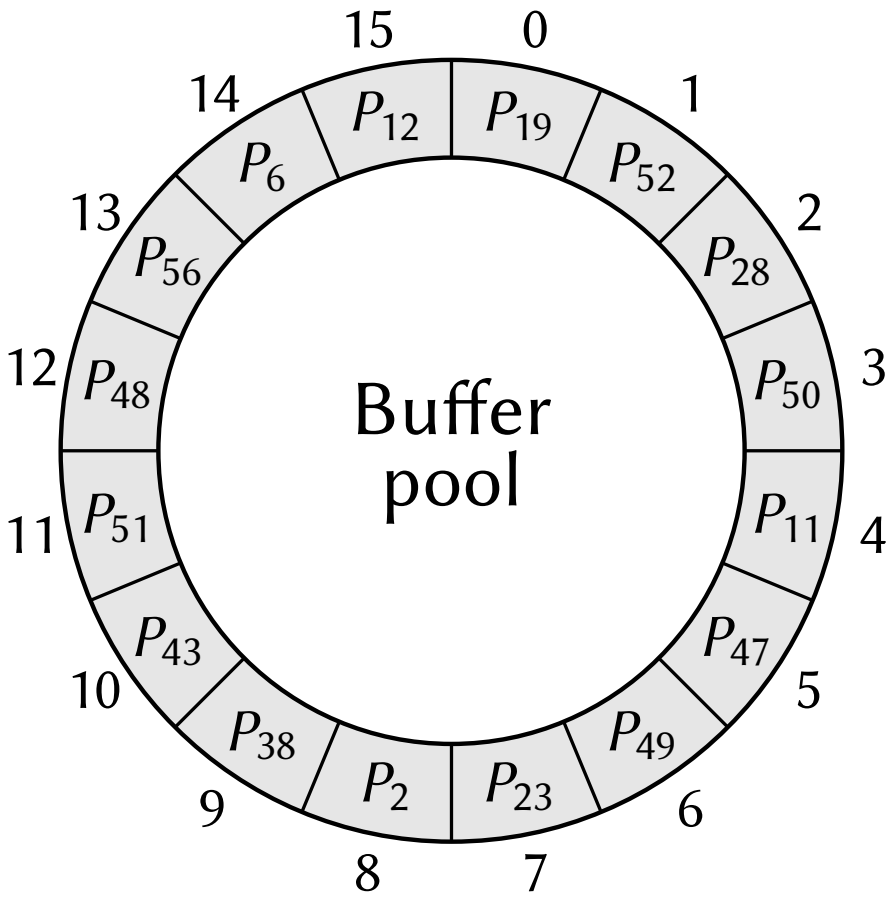
LOOP Algorithm



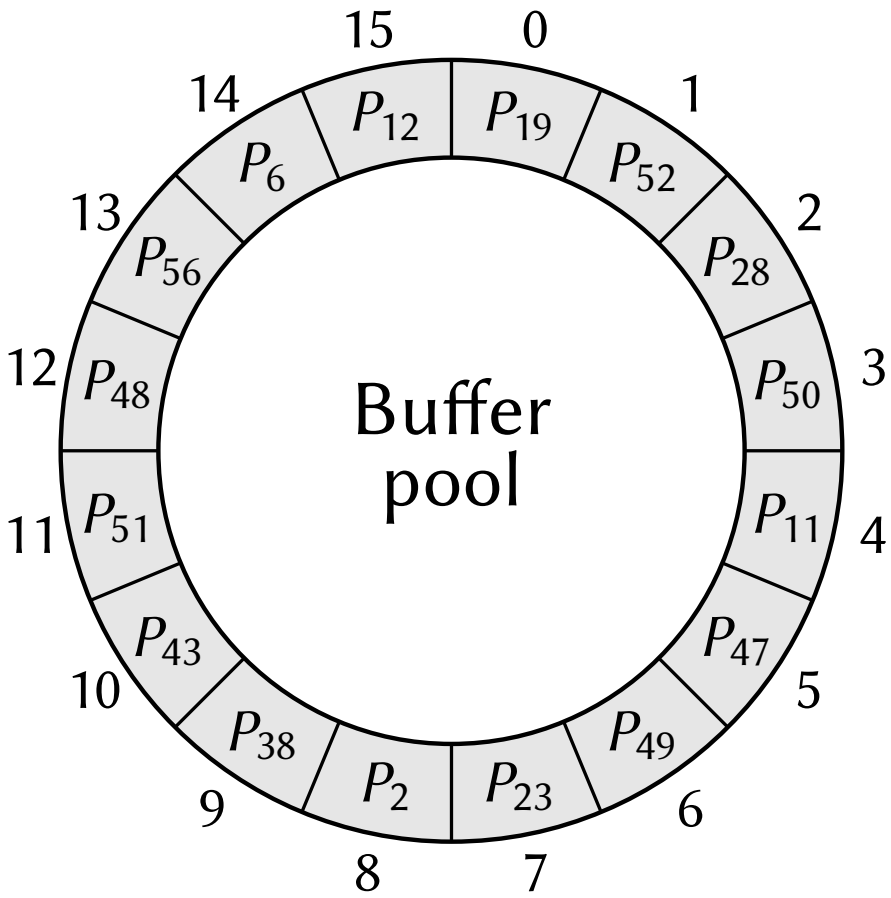
LOOP Algorithm



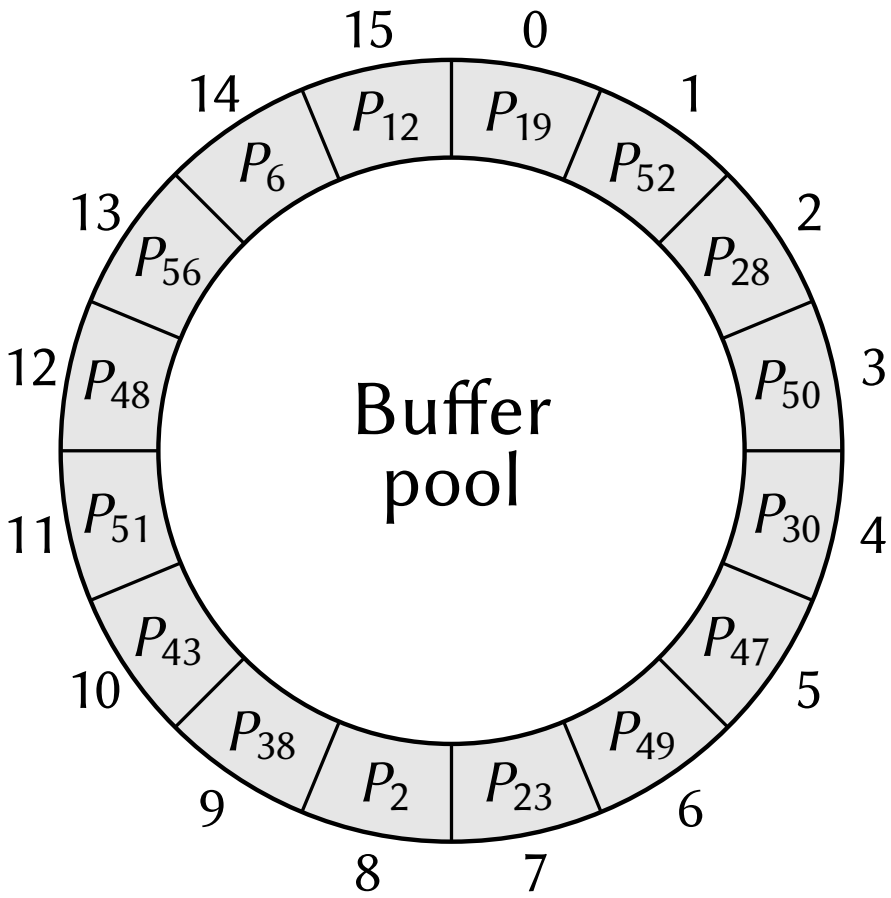
LOOP Algorithm



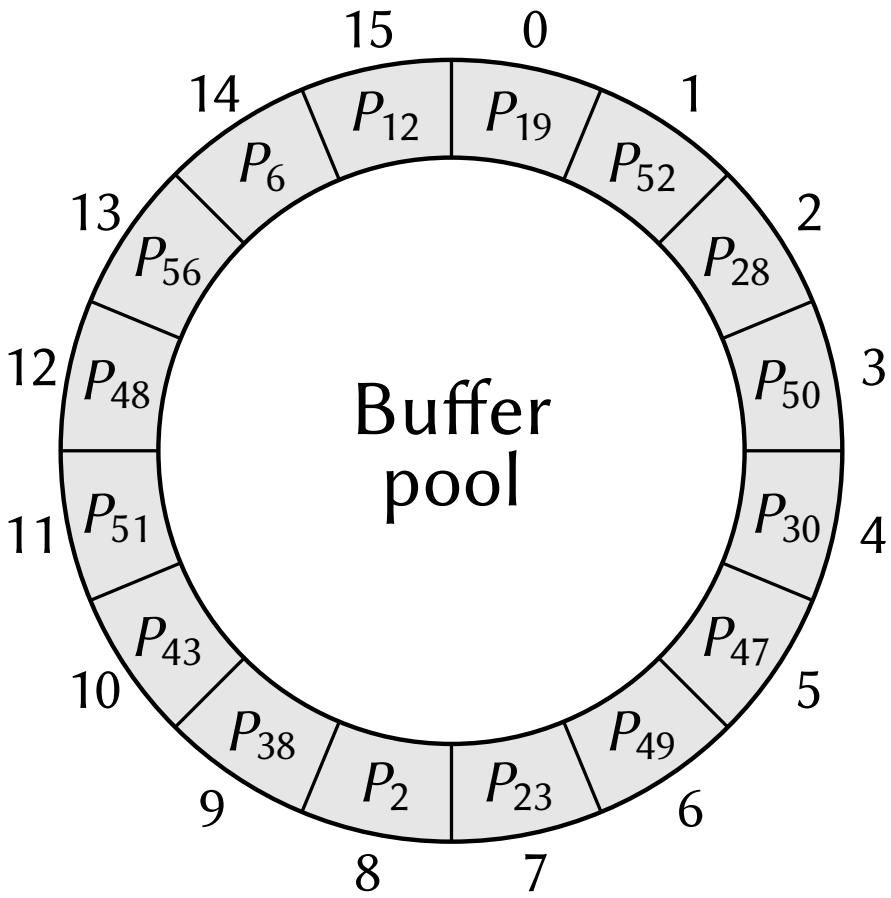
LOOP Algorithm



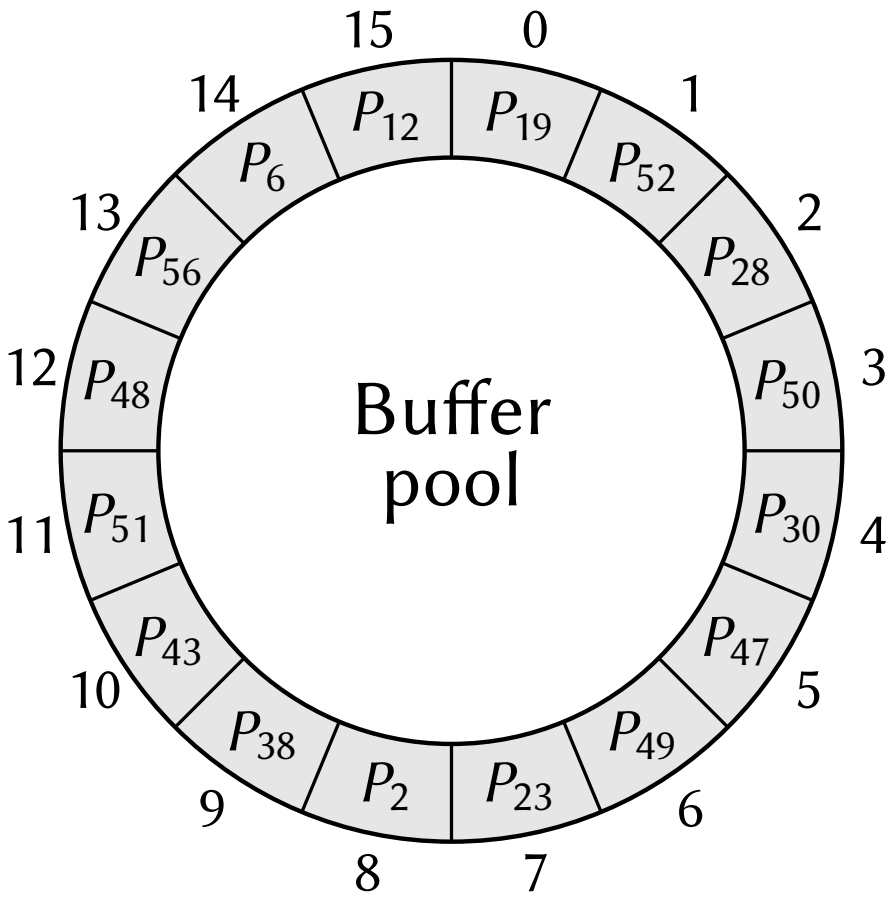
LOOP Algorithm



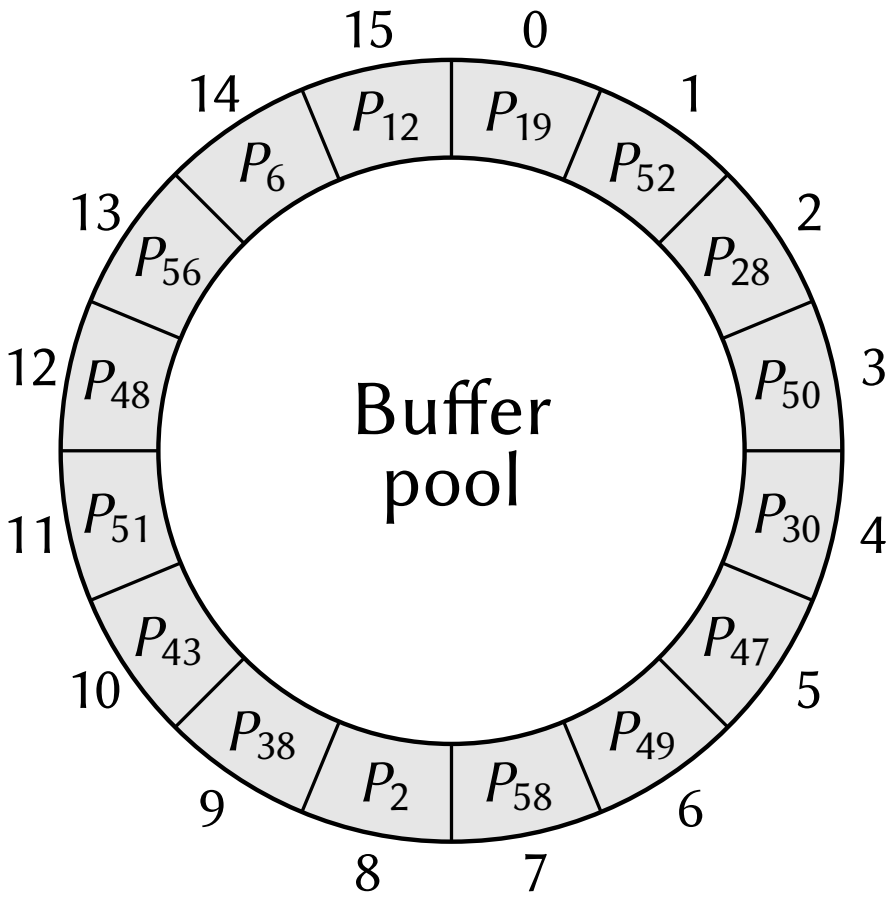
LOOP Algorithm



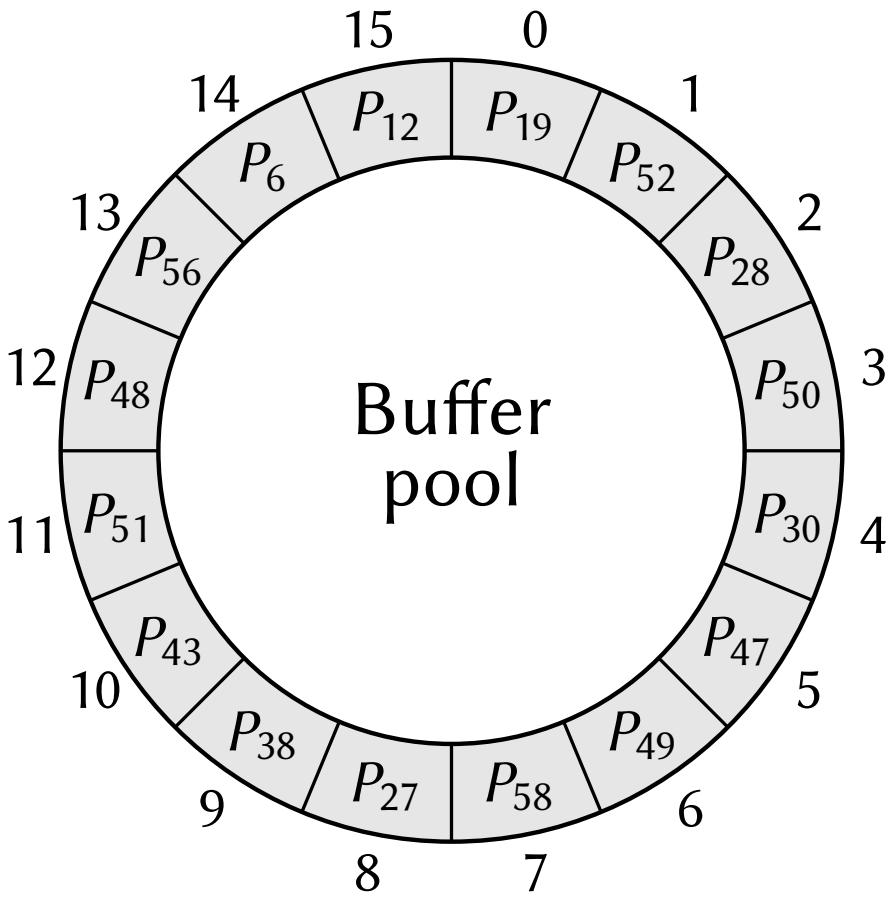
LOOP Algorithm



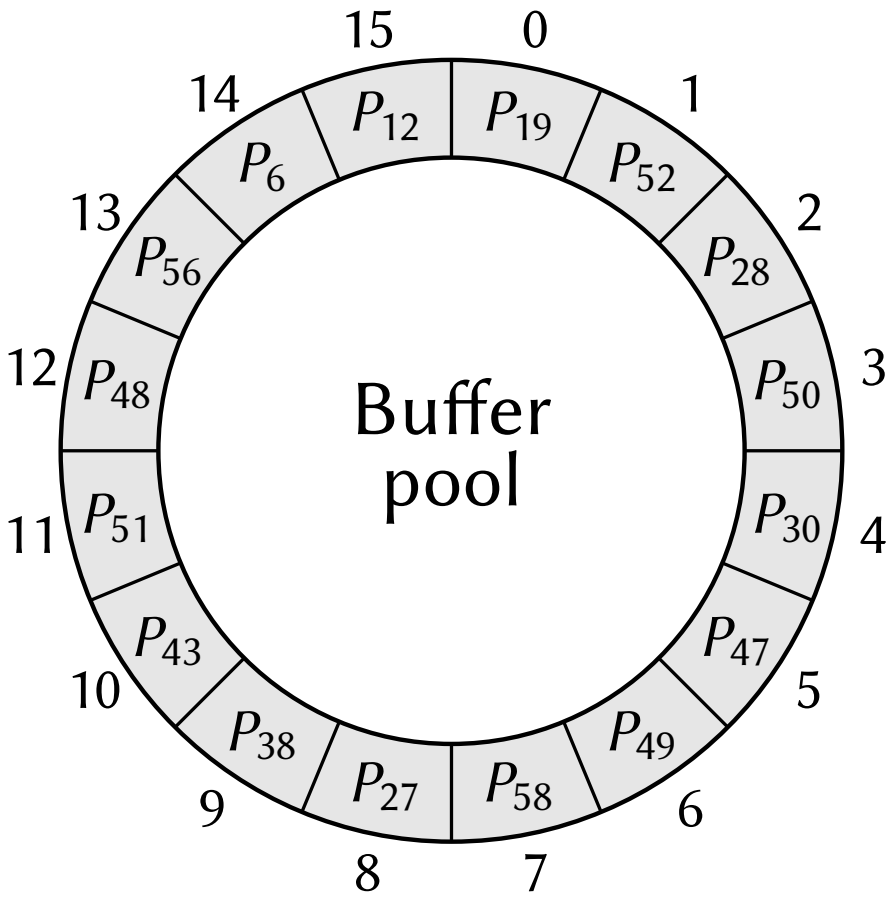
LOOP Algorithm



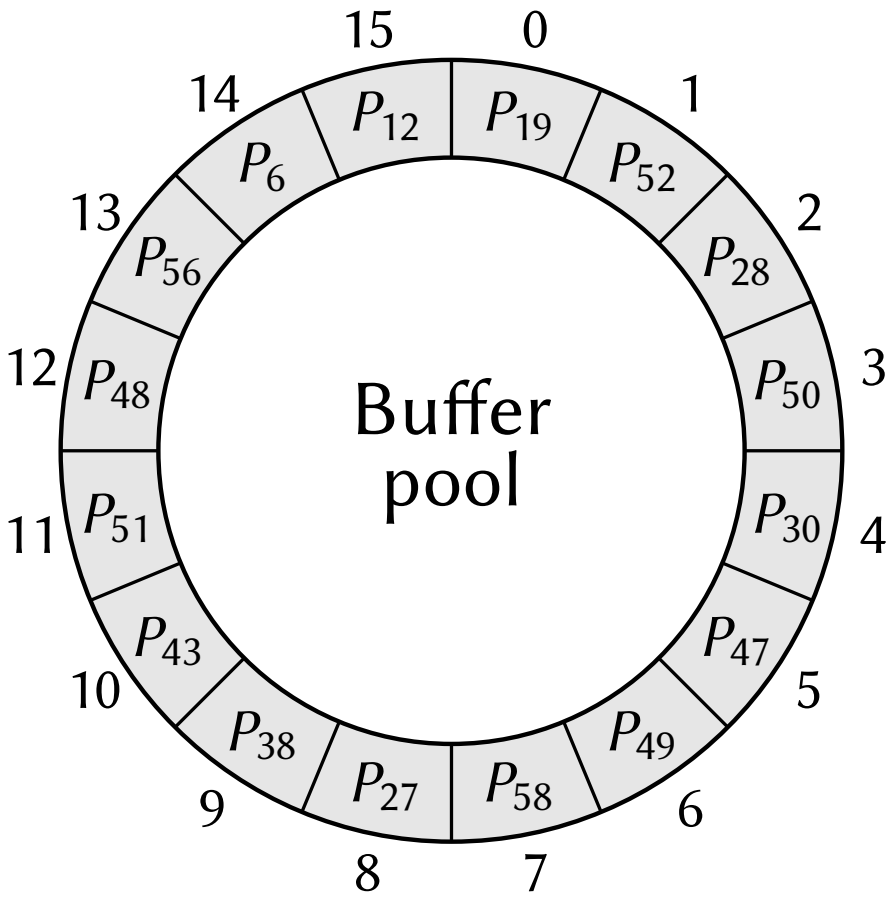
LOOP Algorithm



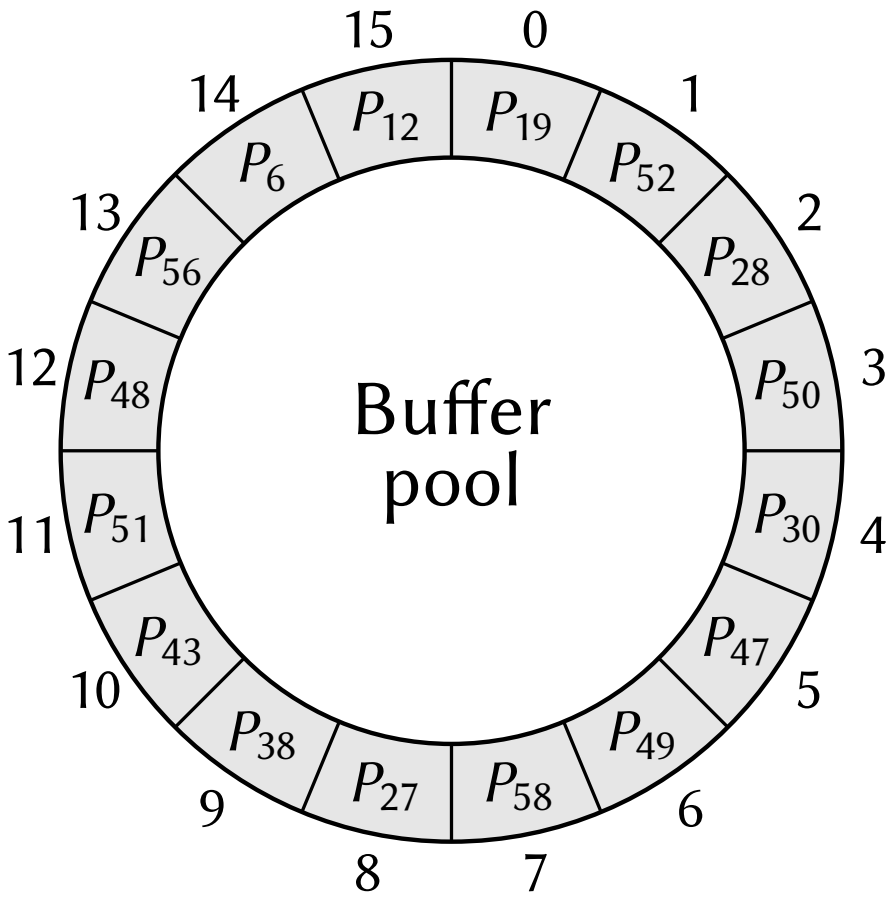
LOOP Algorithm



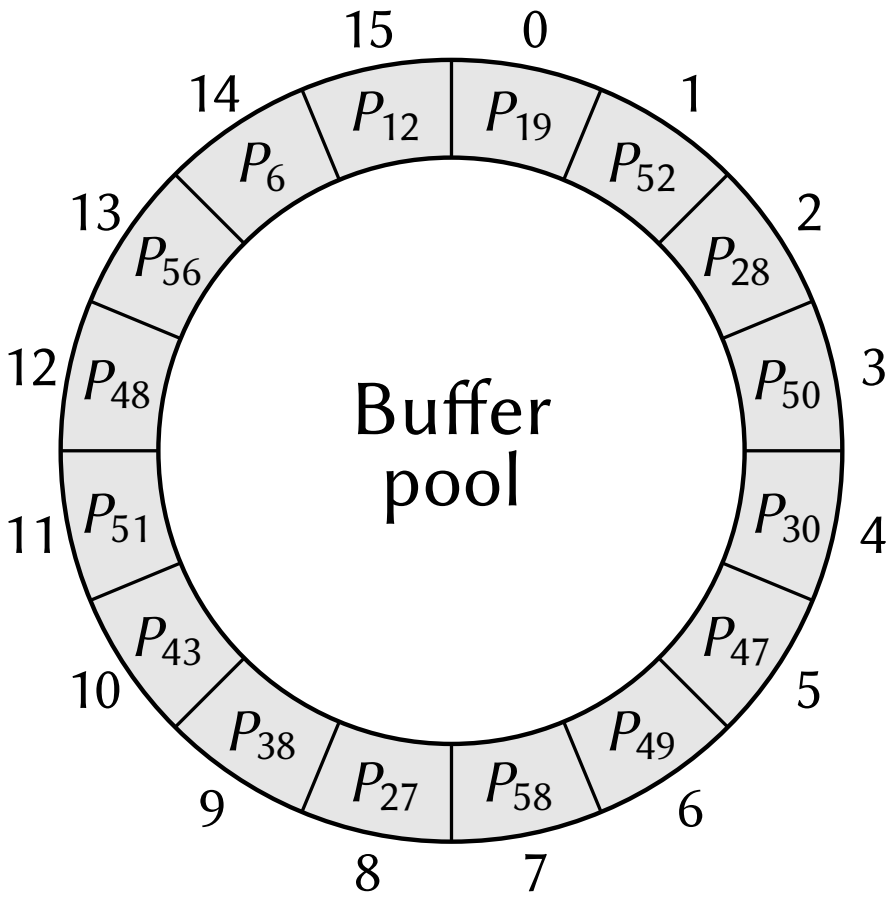
LOOP Algorithm



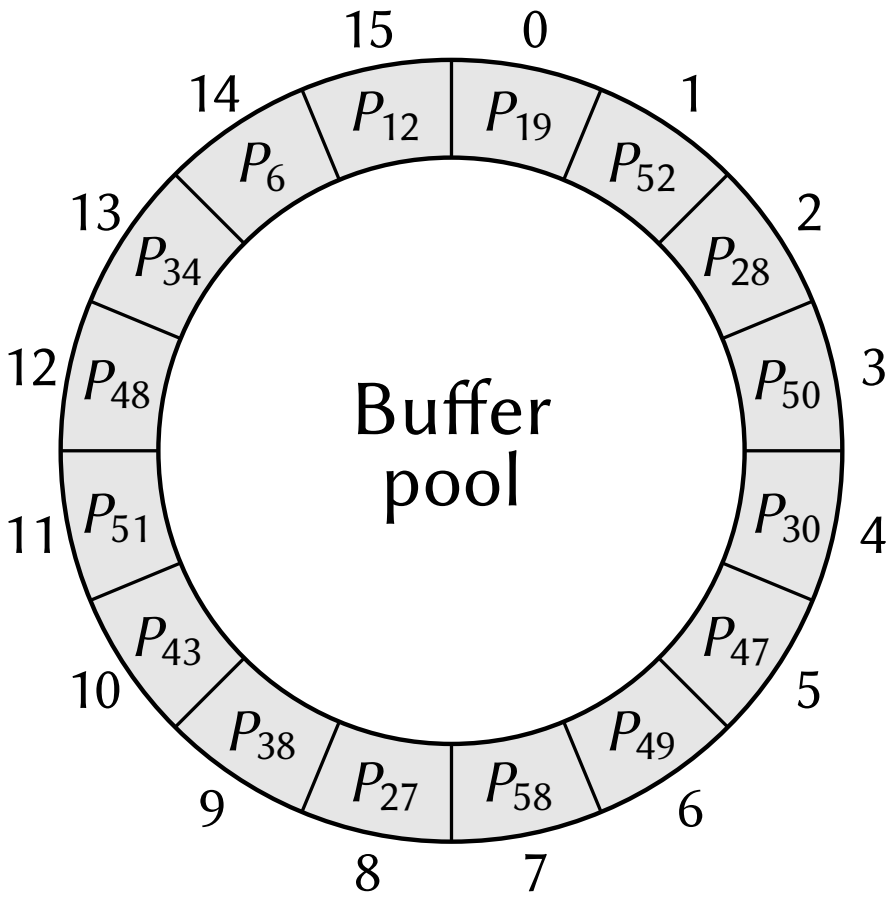
LOOP Algorithm



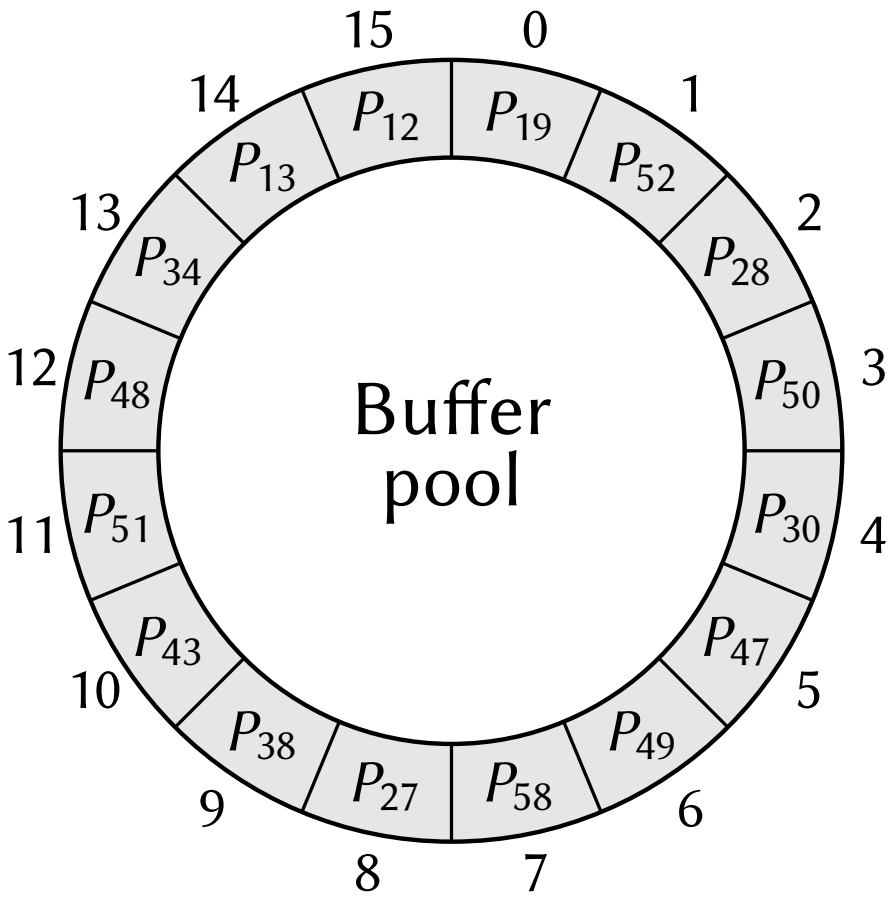
LOOP Algorithm



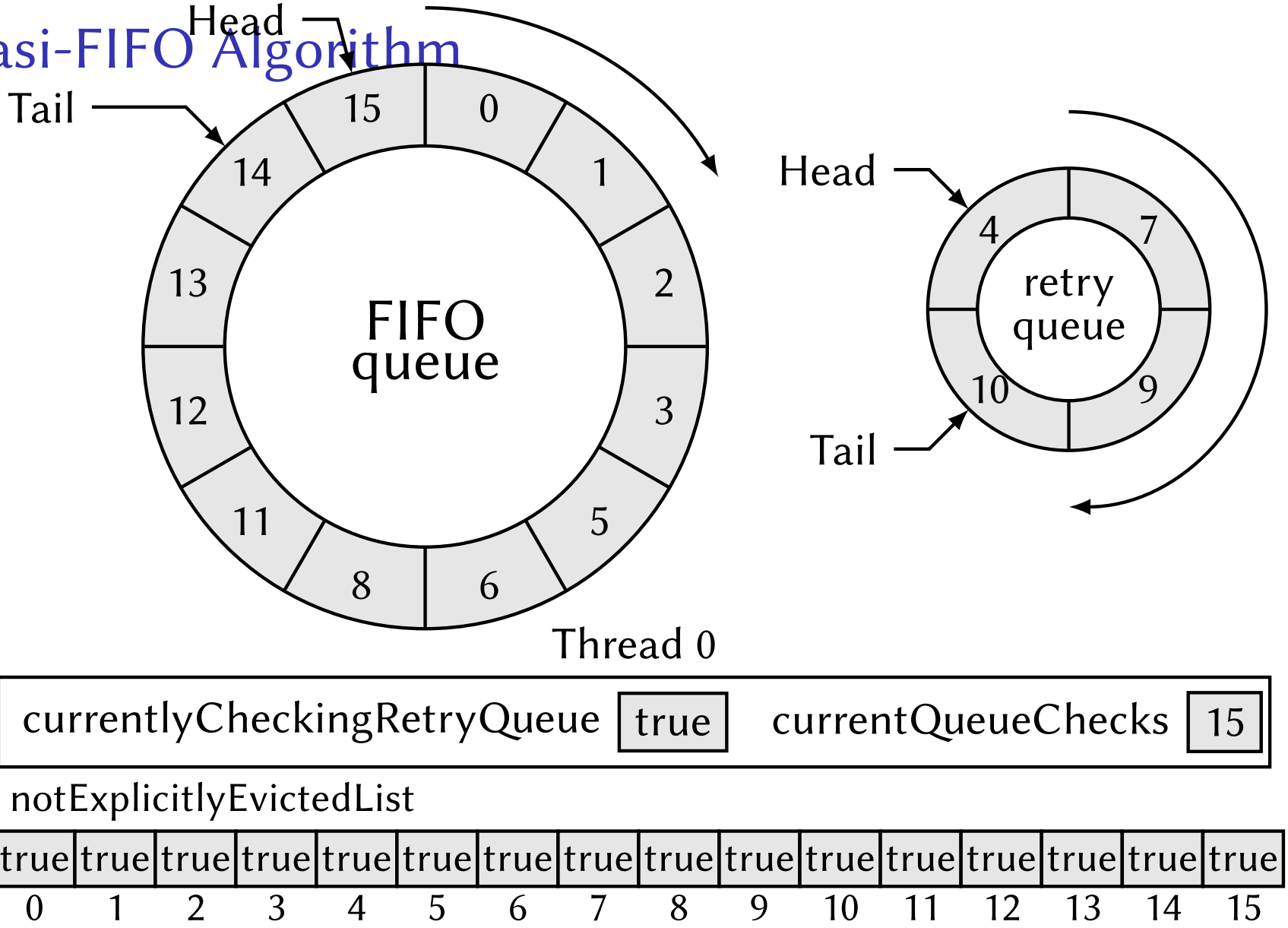
LOOP Algorithm



LOOP Algorithm



Quasi-FIFO Algorithm



LRU-K Implementations

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes
  - Ghost references are in a segment at the head of the queue

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes
  - Ghost references[1] are in a segment at the head of the queue

[1] Ghost references are used for pages with $< k$ references

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes
  - Ghost references[1] are in a segment at the head of the queue
- *Timestamp-Sorting*

[1] Ghost references are used for pages with $< k$ references

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes
  - Ghost references[1] are in a segment at the head of the queue
- *Timestamp-Sorting*
  - k timestamps per buffer frame

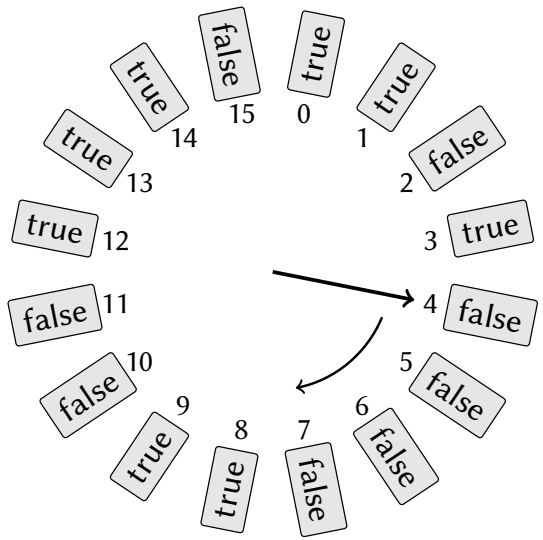
[1] Ghost references are used for pages with $< k$ references

LRU-K Implementations

- *Hash-Map-Doubly-Linked-List*
  - Entries in the queue are reference IDs instead of frame indexes
  - Ghost references[1] are in a segment at the head of the queue
- *Timestamp-Sorting*
  - k timestamps per buffer frame
  - Ghost references are represented by timestamps very early in the past

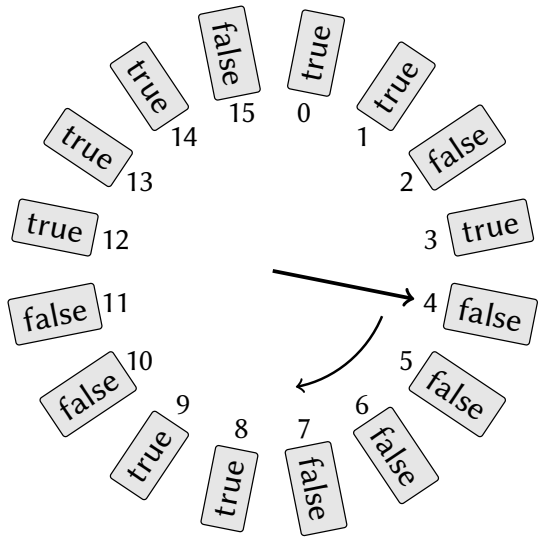
[1] Ghost references are used for pages with $< k$ references

CLOCK Variants



CLOCK Variants

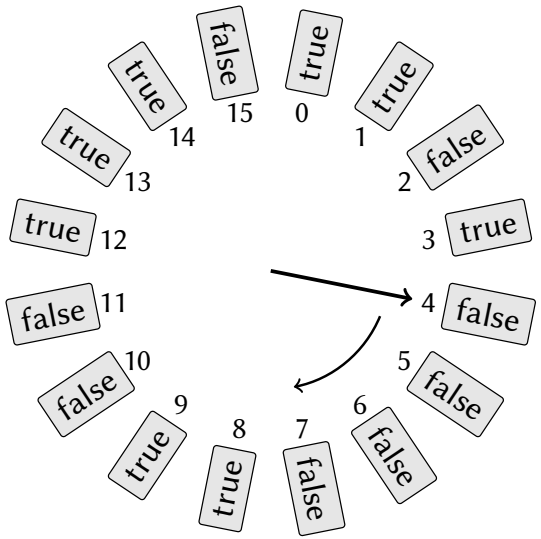
Fix Set usage-bit on *page fix*



CLOCK Variants

Fix Set usage-bit on *page fix*

Unfix Set usage-bit on *page unfix*

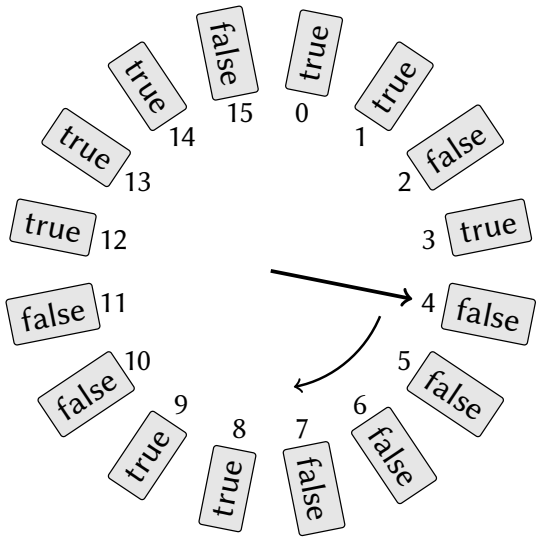


CLOCK Variants

Fix Set usage-bit on *page fix*

Unfix Set usage-bit on *page unfix*

FixUnfix Set usage-bit on *page fix* and on *page unfix*



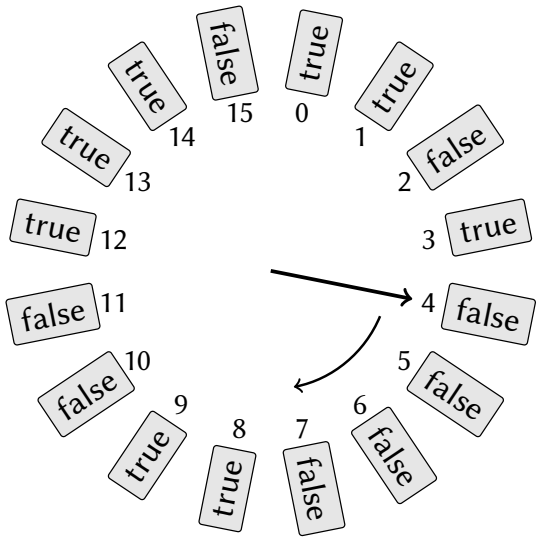
CLOCK Variants

Fix Set usage-bit on *page fix*

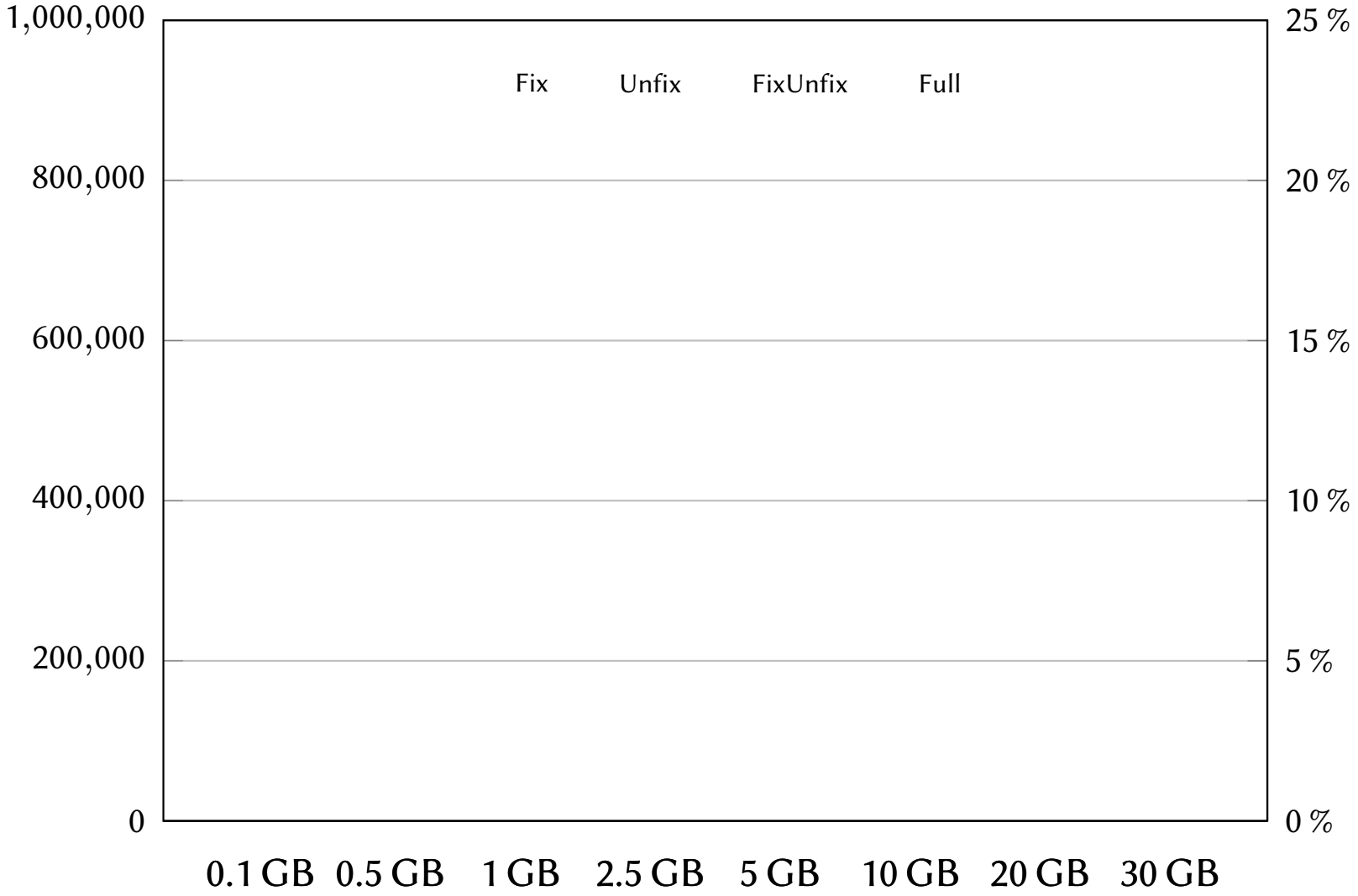
Unfix Set usage-bit on *page unfix*

FixUnfix Set usage-bit on *page fix* and on *page unfix*

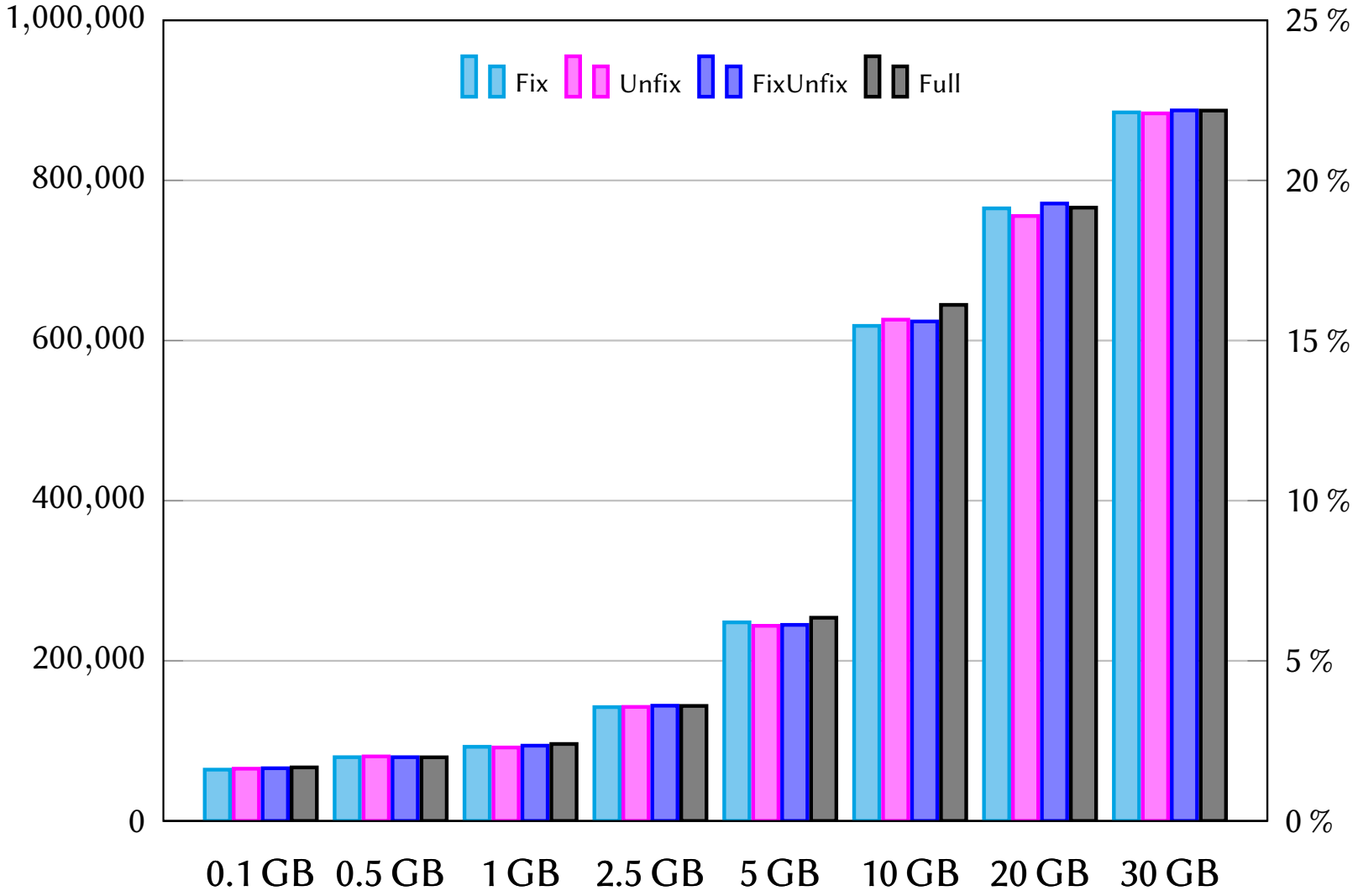
Full Like FixUnfix but also set usage-bit when page cannot be evicted



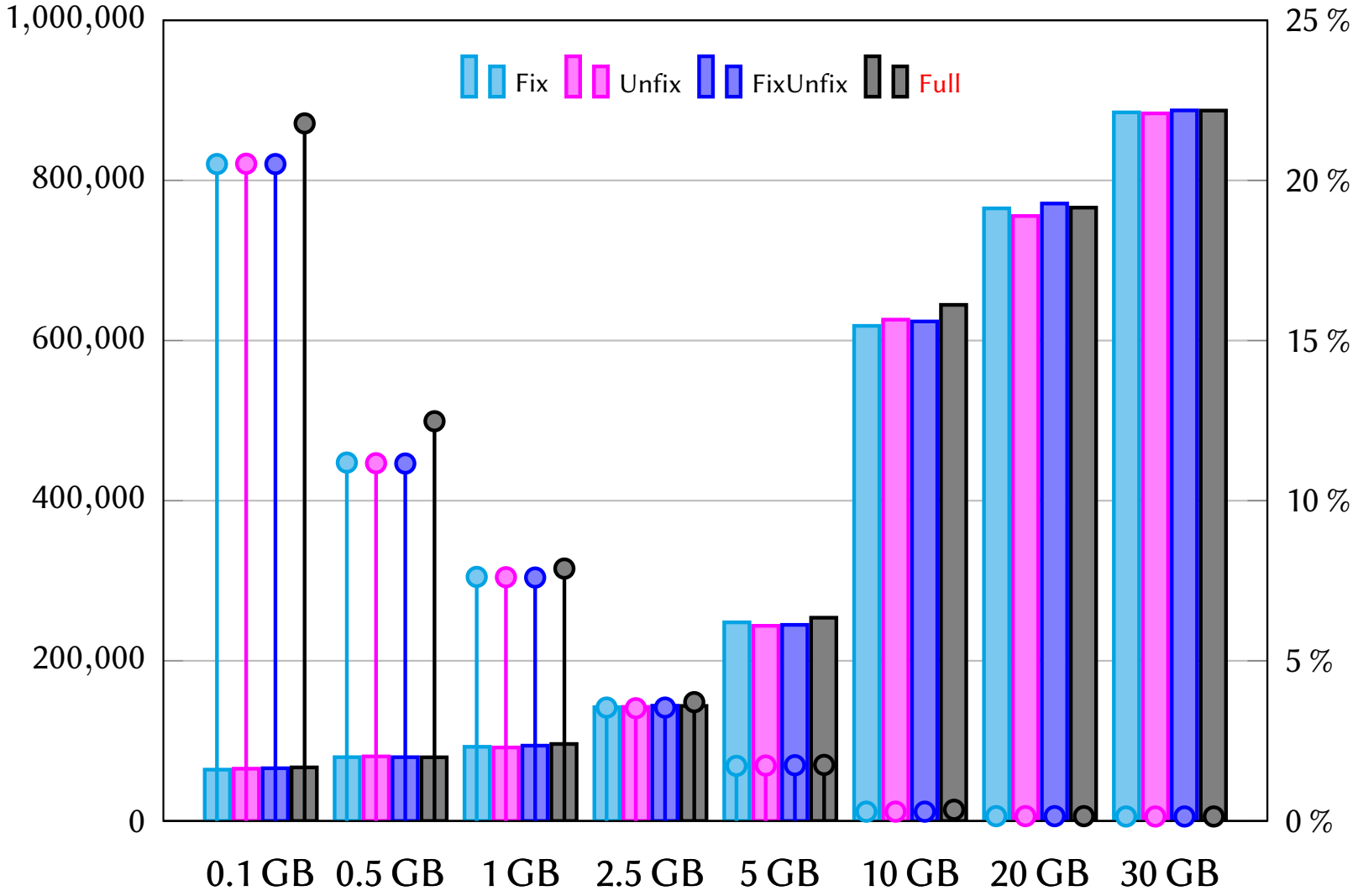
Performance Evaluation



Performance Evaluation



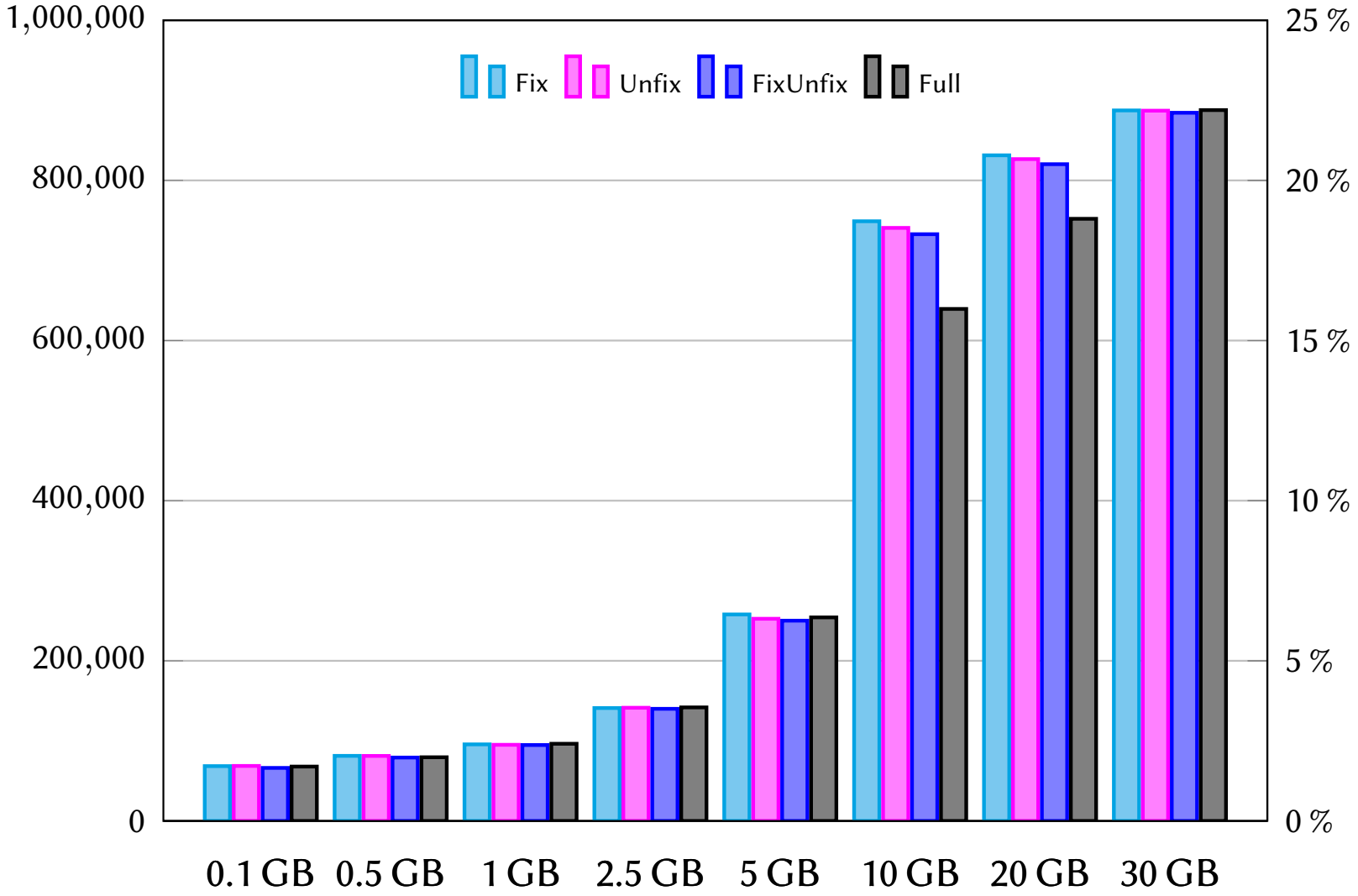
Performance Evaluation



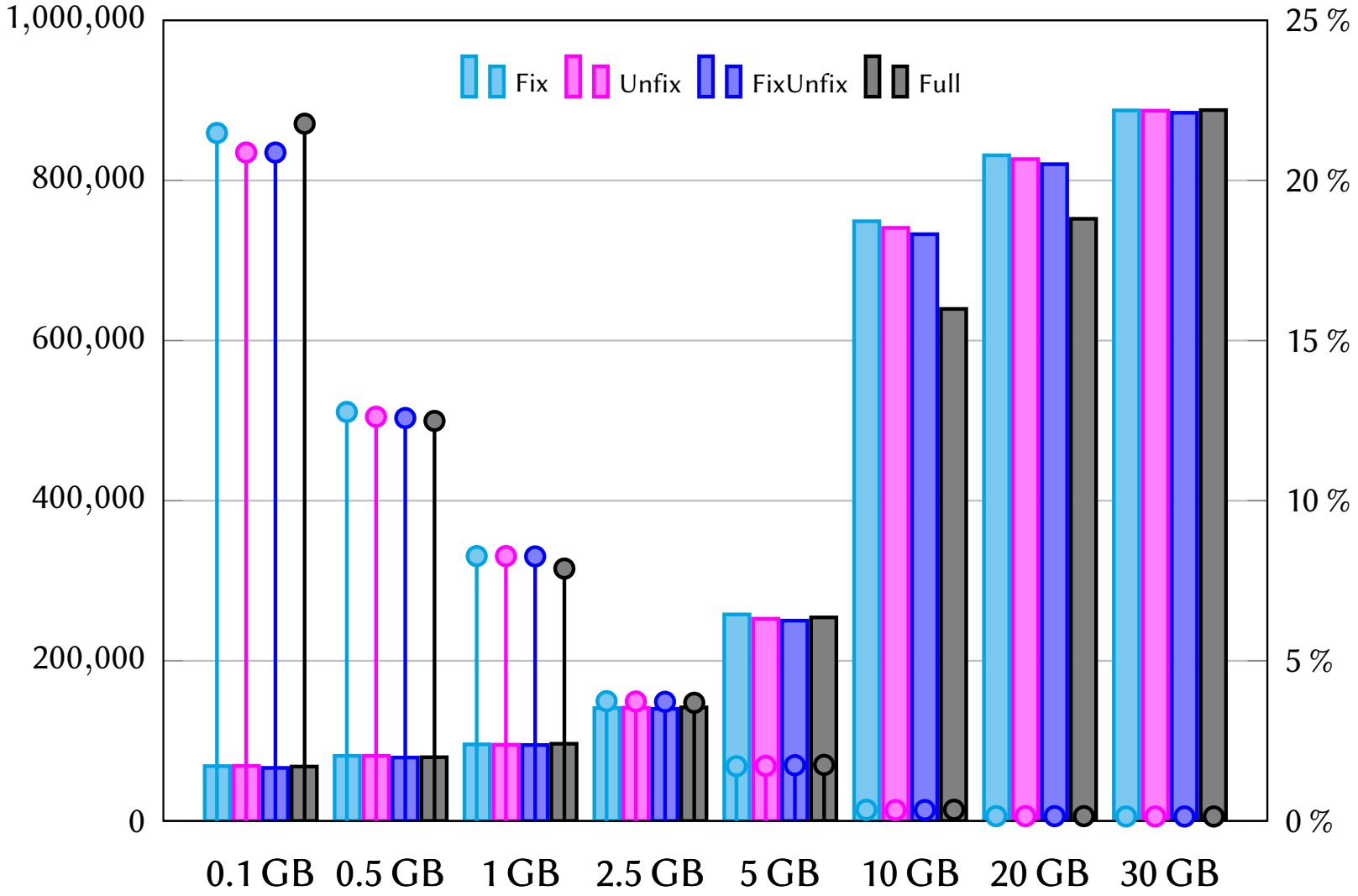
ZCLOCK Variants

Fix	Set usage-bit on *page fix*
Unfix	Set usage-bit on *page unfix*
FixUnfix	Set usage-bit on *page fix* and on *page unfix*
Full	Like FixUnfix but also set usage-bit when page cannot be evicted

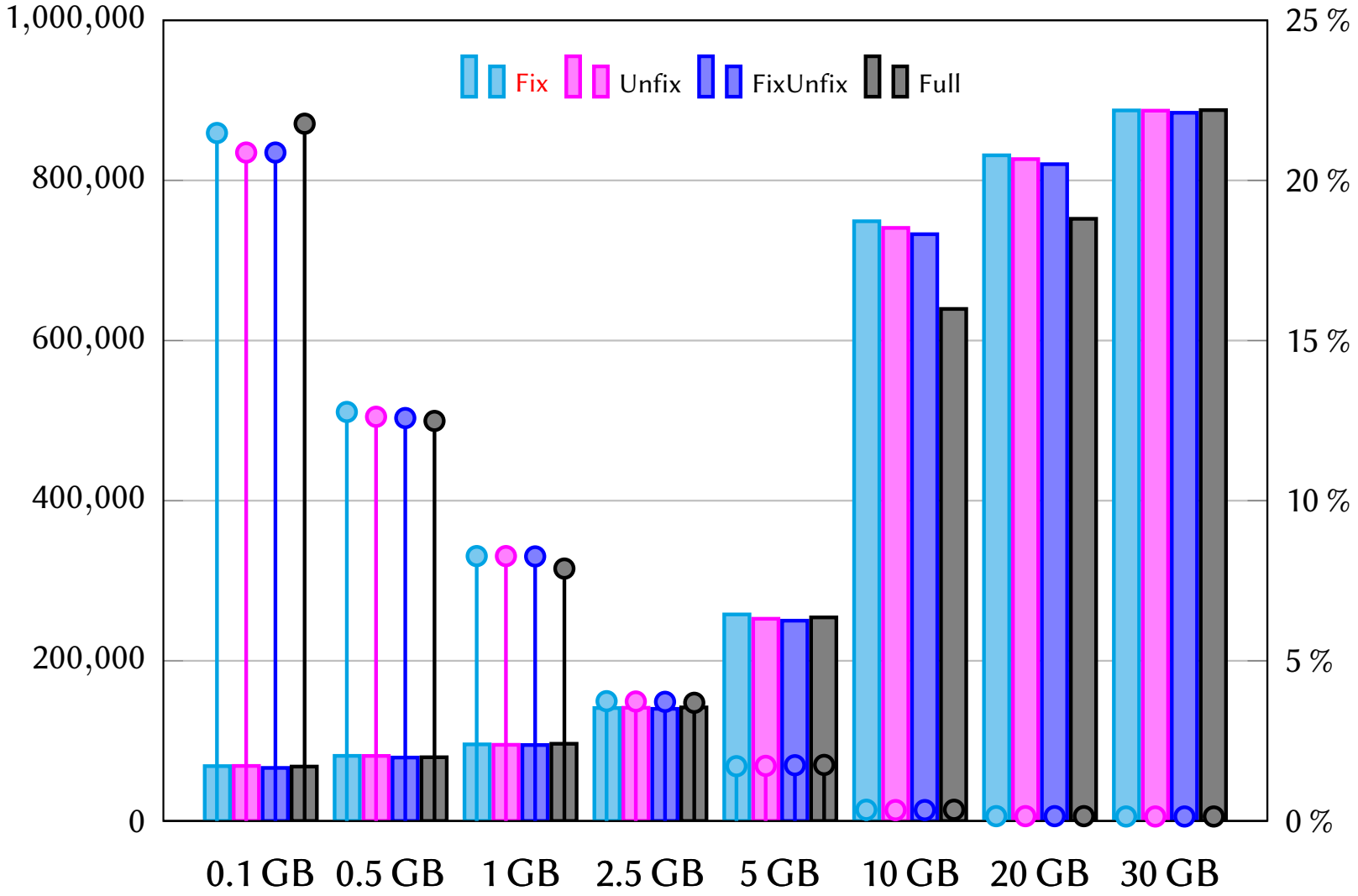
Performance Evaluation



Performance Evaluation



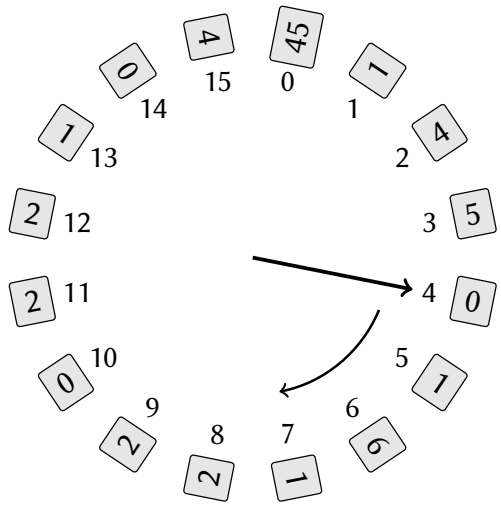
Performance Evaluation



GCLOCK-V1 Variants

Usage-Count

$$
\begin{aligned}
\text{(initially)} \quad & \mathrm{UC}(p) = F \\
\text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\
\text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1
\end{aligned}
$$

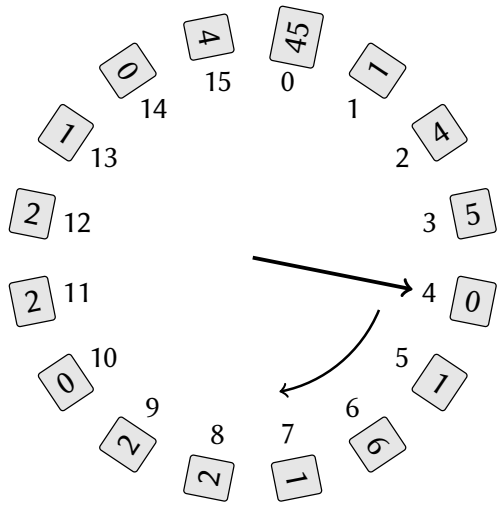


GCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*



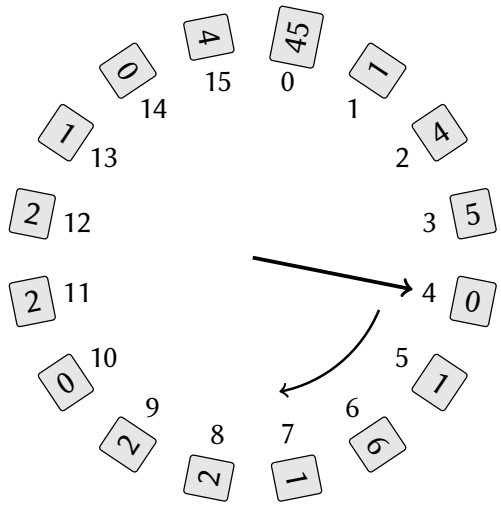
GCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted



GCLOCK-V1 Variants

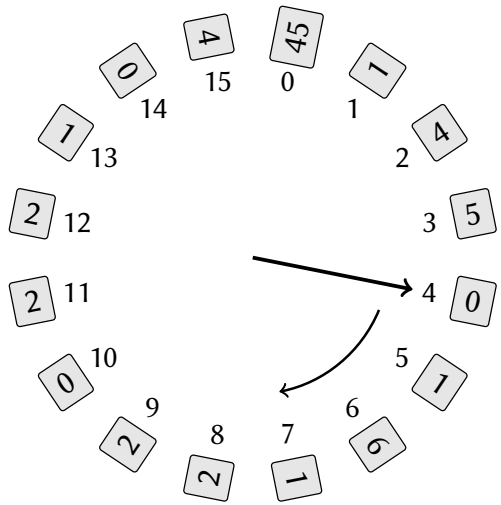
Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

Unfix Increase usage-count on *page unfix*



GCLOCK-V1 Variants

Usage-Count

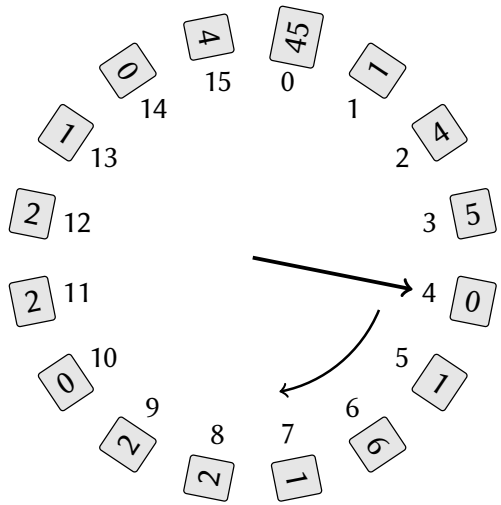
$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

Unfix Increase usage-count on *page unfix*

UnfixFull Like Unfix but also increase usage-count when page cannot be evicted



GCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

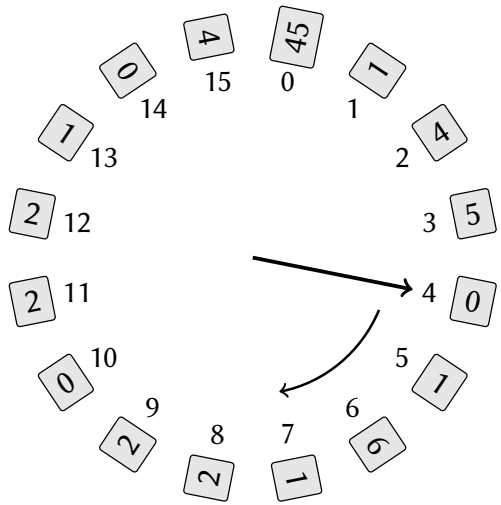
Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

Unfix Increase usage-count on *page unfix*

UnfixFull Like Unfix but also increase usage-count when page cannot be evicted

$F = 25, R = 5$



GCLOCK-V1 Variants

Usage-Count

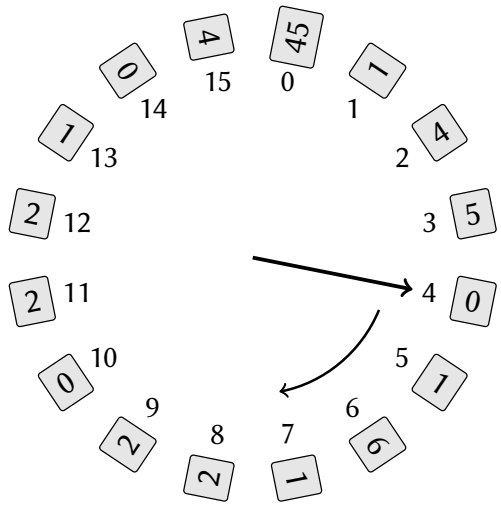
$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

Unfix Increase usage-count on *page unfix*

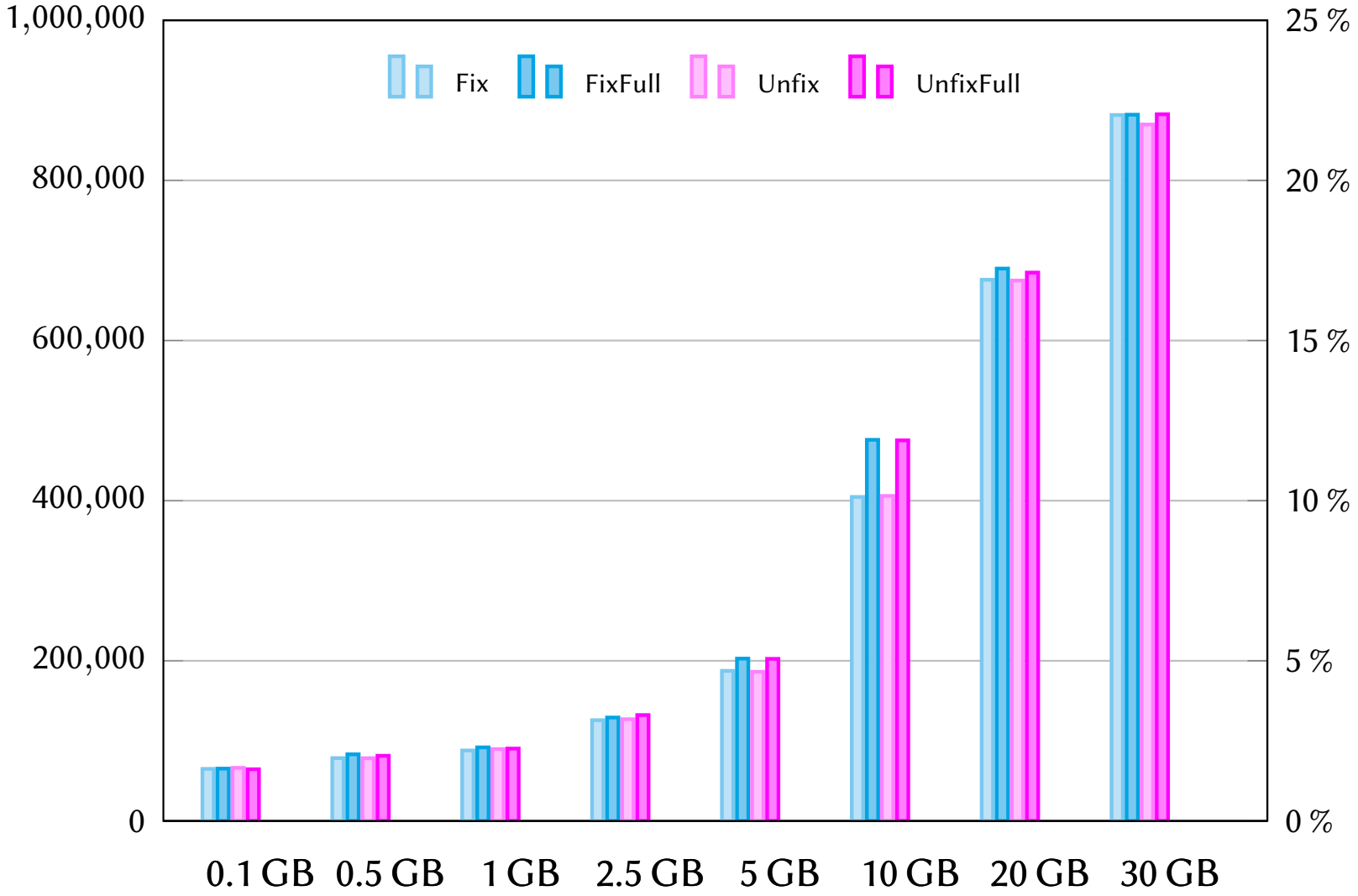
UnfixFull Like Unfix but also increase usage-count when page cannot be evicted



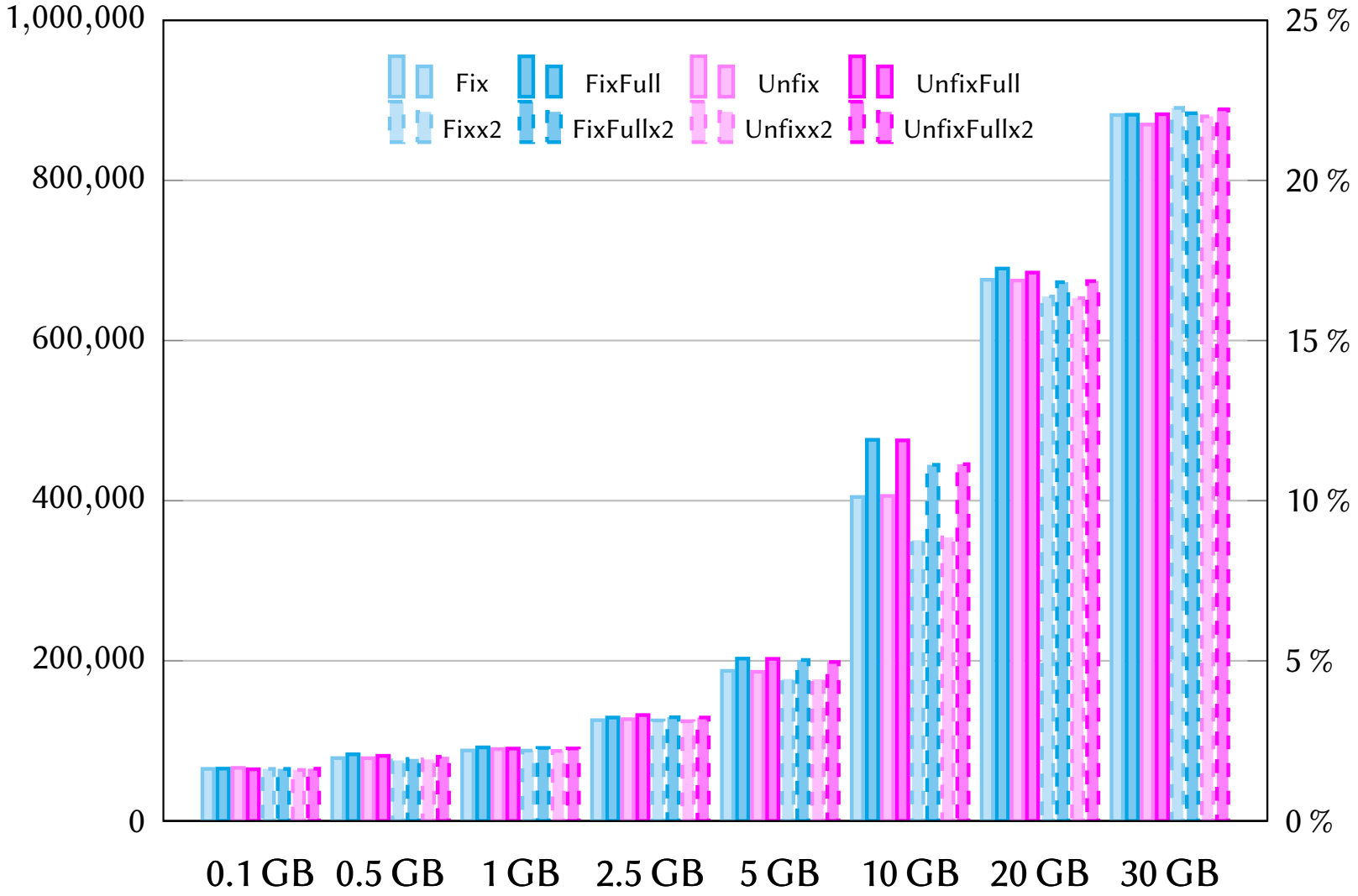
$F = 25, R = 5$

x2 $F = 50, R = 10$

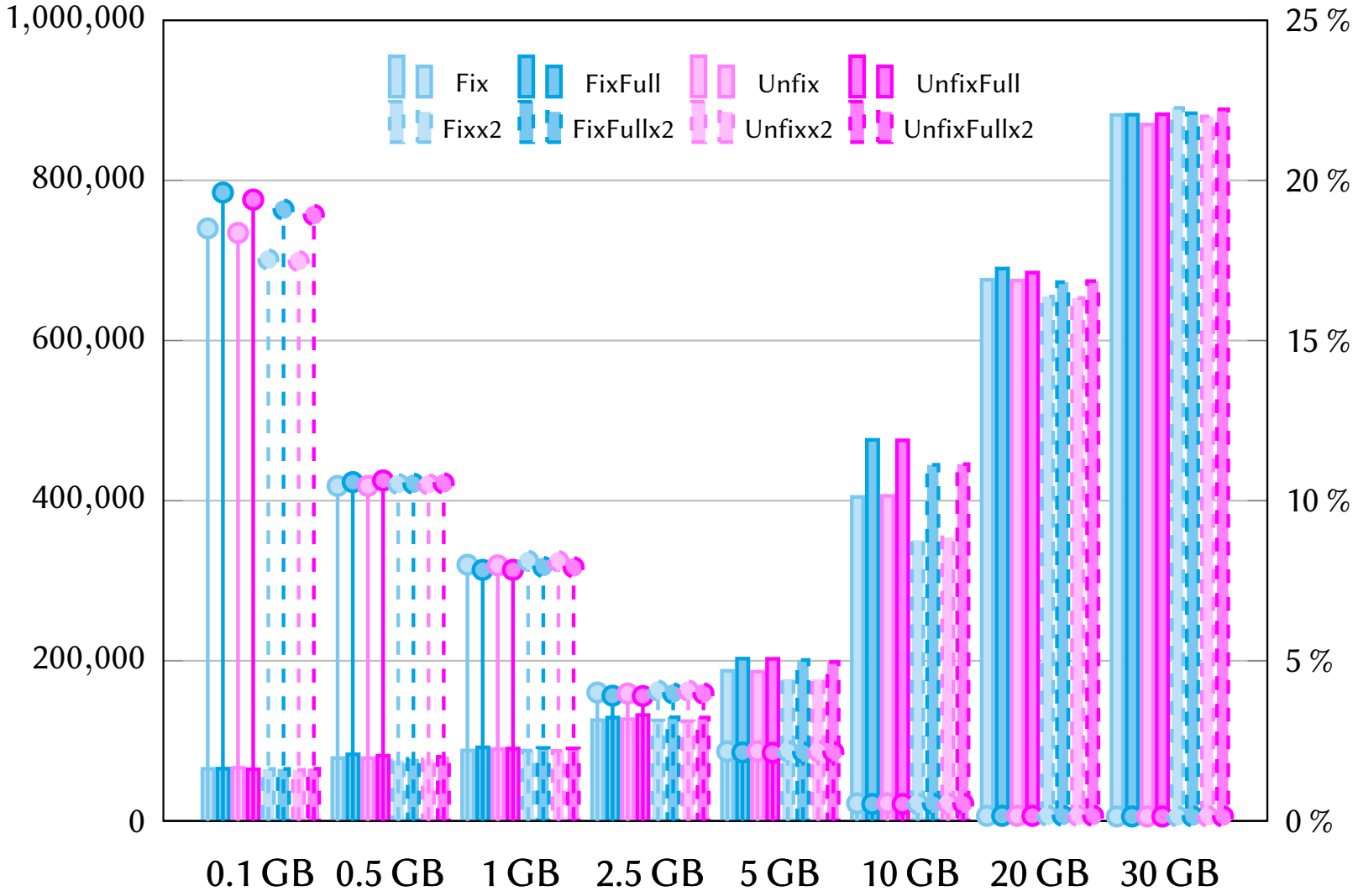
Performance Evaluation



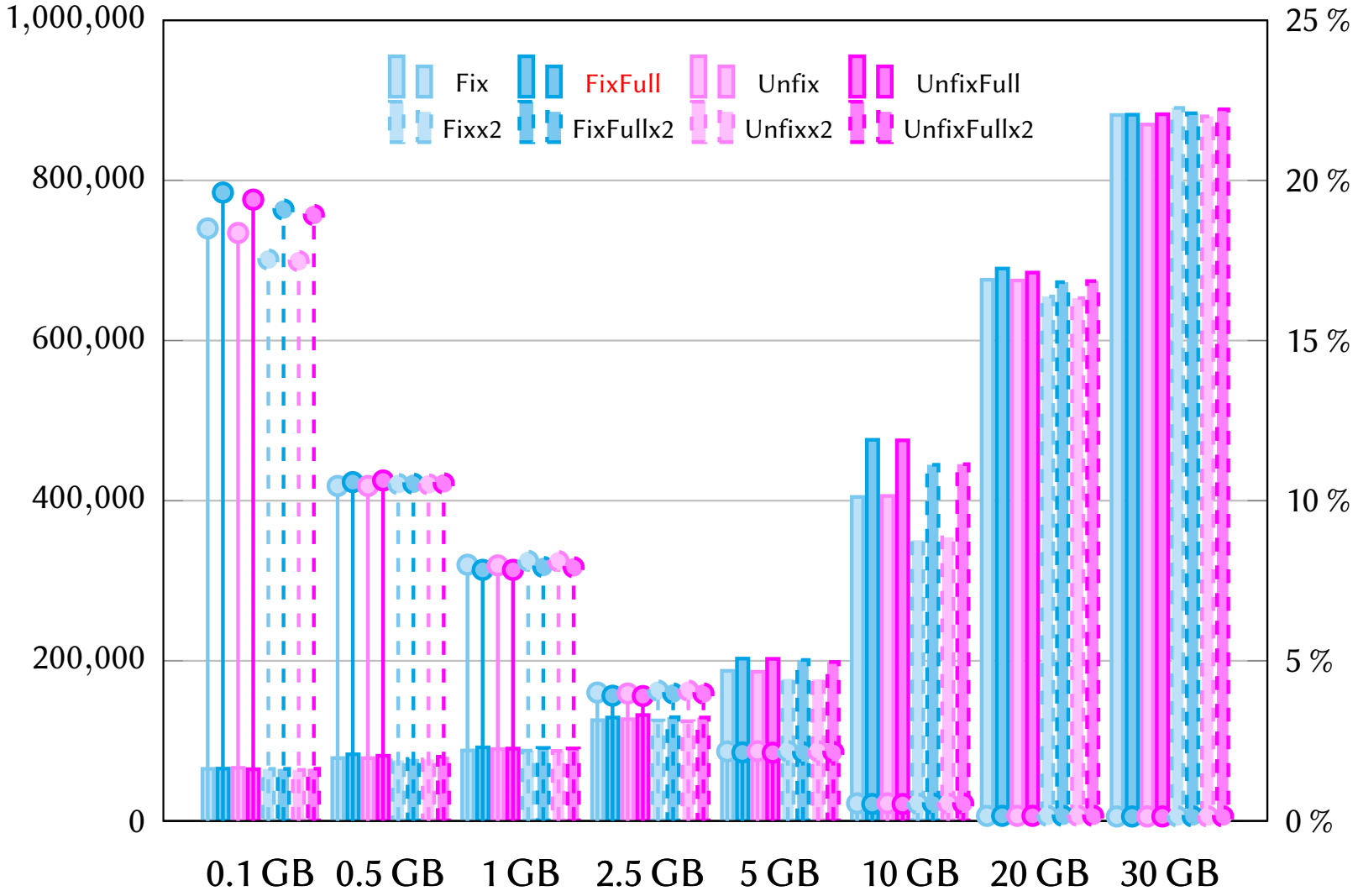
Performance Evaluation



Performance Evaluation



Performance Evaluation



GCLOCK-V2 Variants

Fix Set usage-count on *page fix*

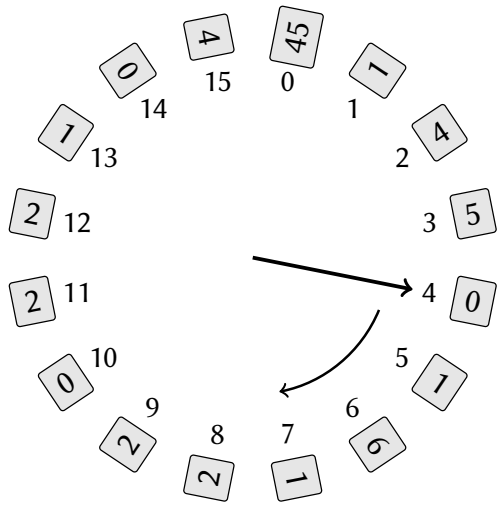
FixFull Like Fix but also set usage-count when page cannot be evicted

Unfix Set usage-count on *page unfix*

UnfixFull Like Unfix but also set usage-count when page cannot be evicted

$F = 25, R = 5$

x2 $F = 50, R = 10$



GCLOCK-V2 Variants

Usage-Count

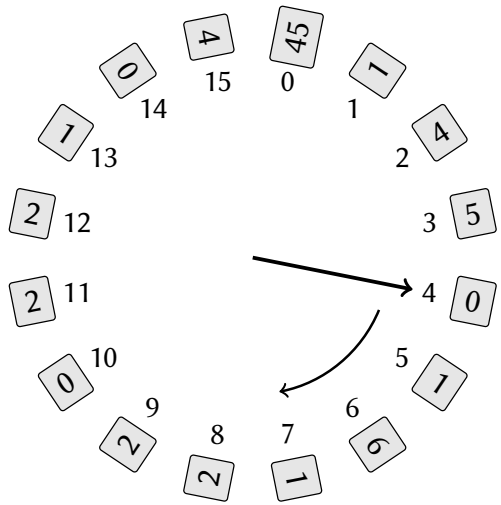
$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F \\ \text{(set)} \quad & \mathrm{UC}(p) = R \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Set usage-count on *page fix*

FixFull Like Fix but also set usage-count when page cannot be evicted

Unfix Set usage-count on *page unfix*

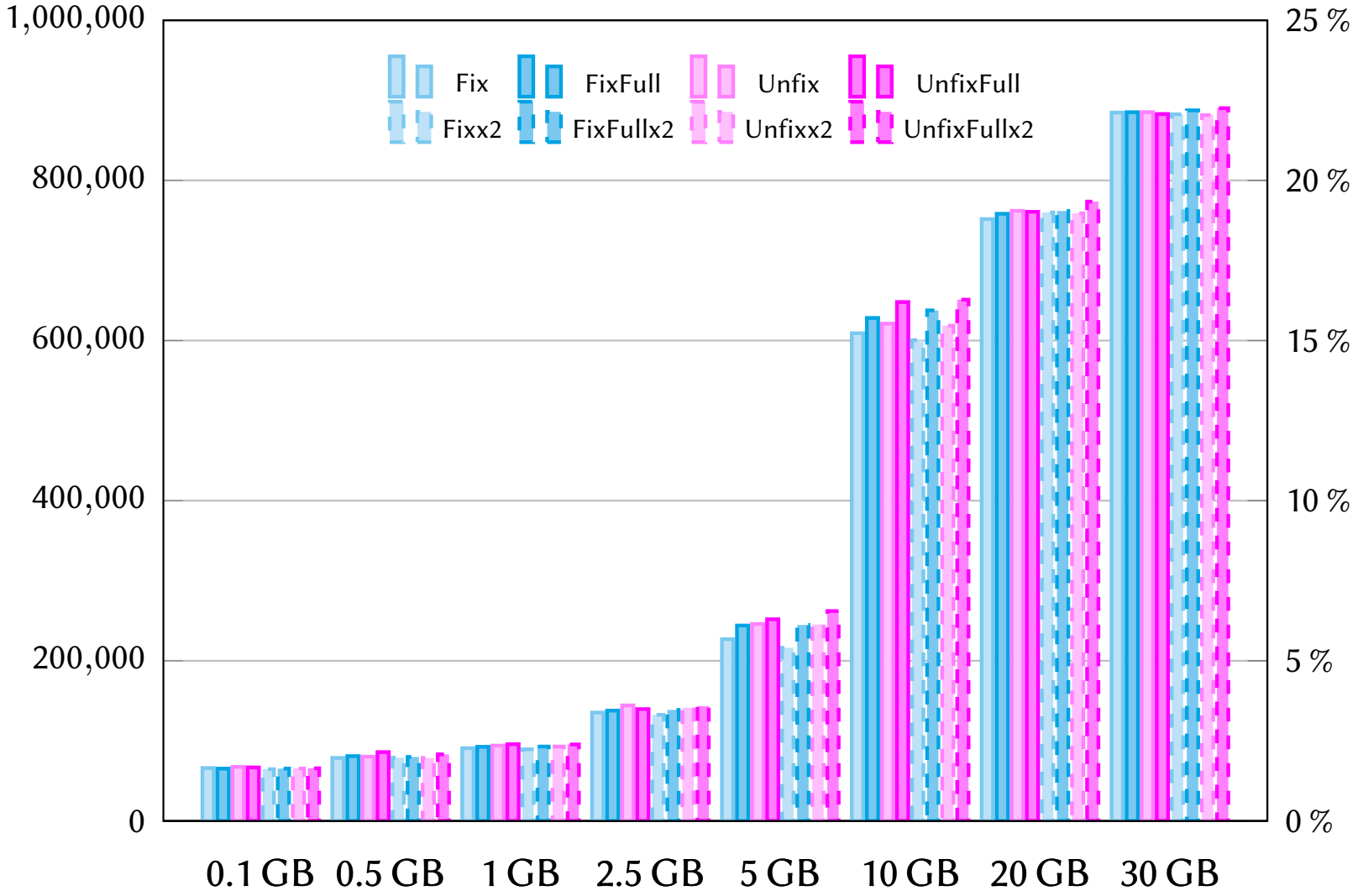
UnfixFull Like Unfix but also set usage-count when page cannot be evicted



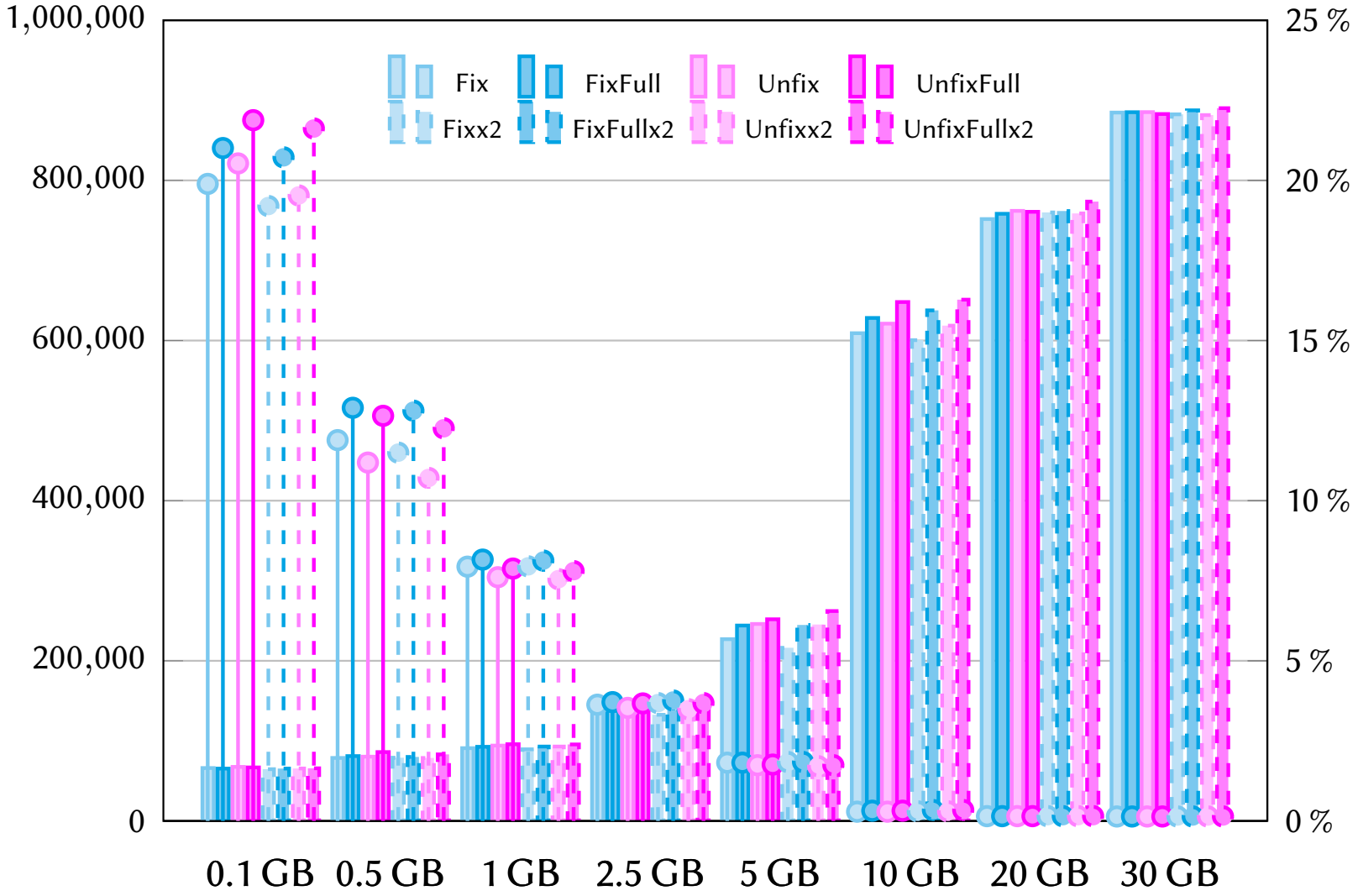
$F = 25, R = 5$

x2 $F = 50, R = 10$

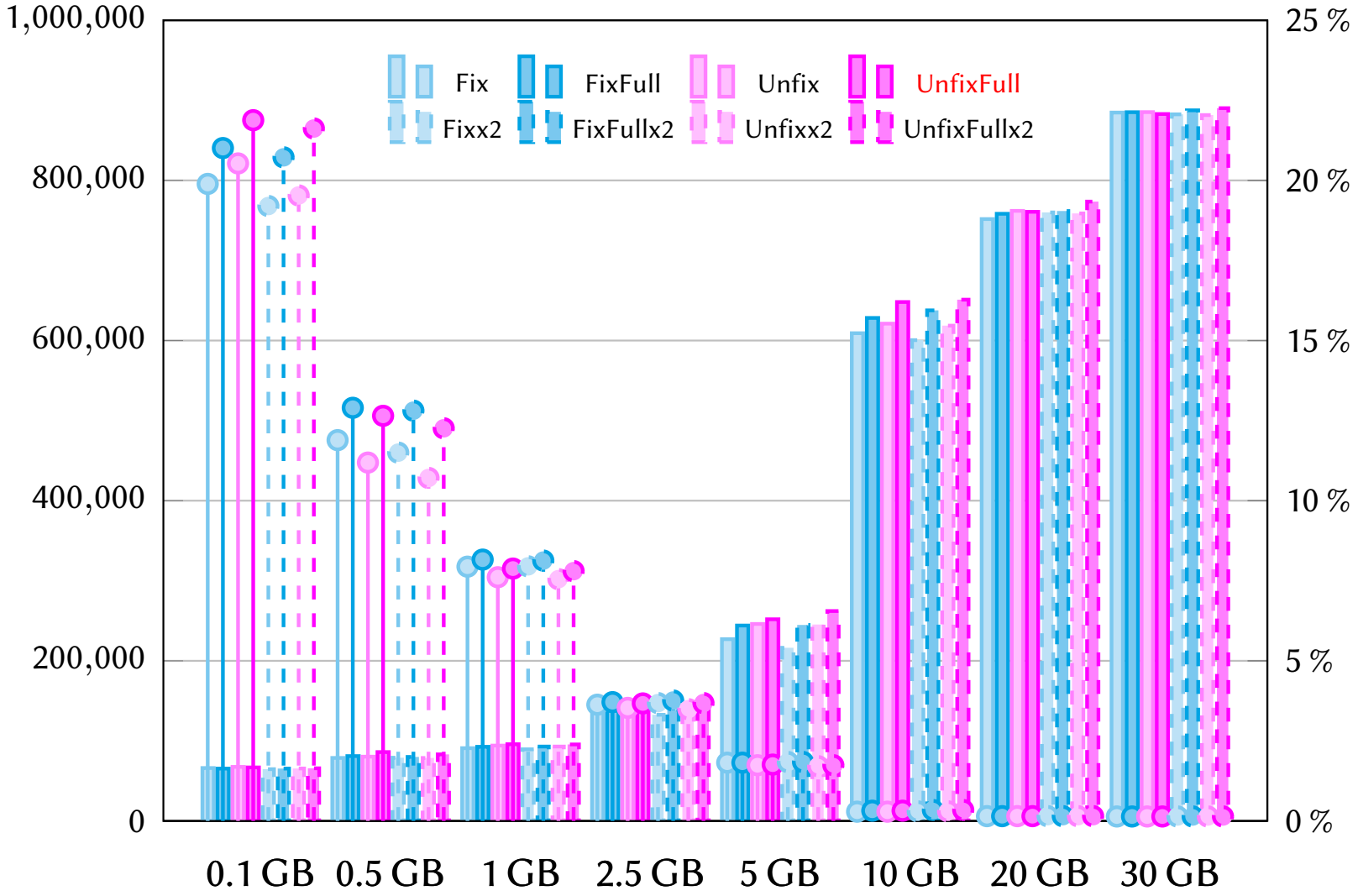
Performance Evaluation



Performance Evaluation



Performance Evaluation



DGCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F_p \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R_p \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

DGCLOCK-V1 Variants

Usage-Count

$$
\begin{aligned}
\text{(initially)} \quad & \mathrm{UC}(p) = F_p \\
\text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R_p \\
\text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1
\end{aligned}
$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

DGCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F_p \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R_p \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

$$\begin{aligned} F_{\text{non-B-tree}} &= 25, \; F_{\text{root}} = 25, \; F_{\text{root}-1} = 10, F_{\text{root}-\geq 2} = 5, \\ R_{\text{non-B-tree}} &= 5, \quad R_{\text{root}} = 5, \quad R_{\text{root}-1} = 2, \; R_{\text{root}-\geq 2} = 1 \end{aligned}$$

DGCLOCK-V1 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F_p \\ \text{(increase)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) + R_p \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

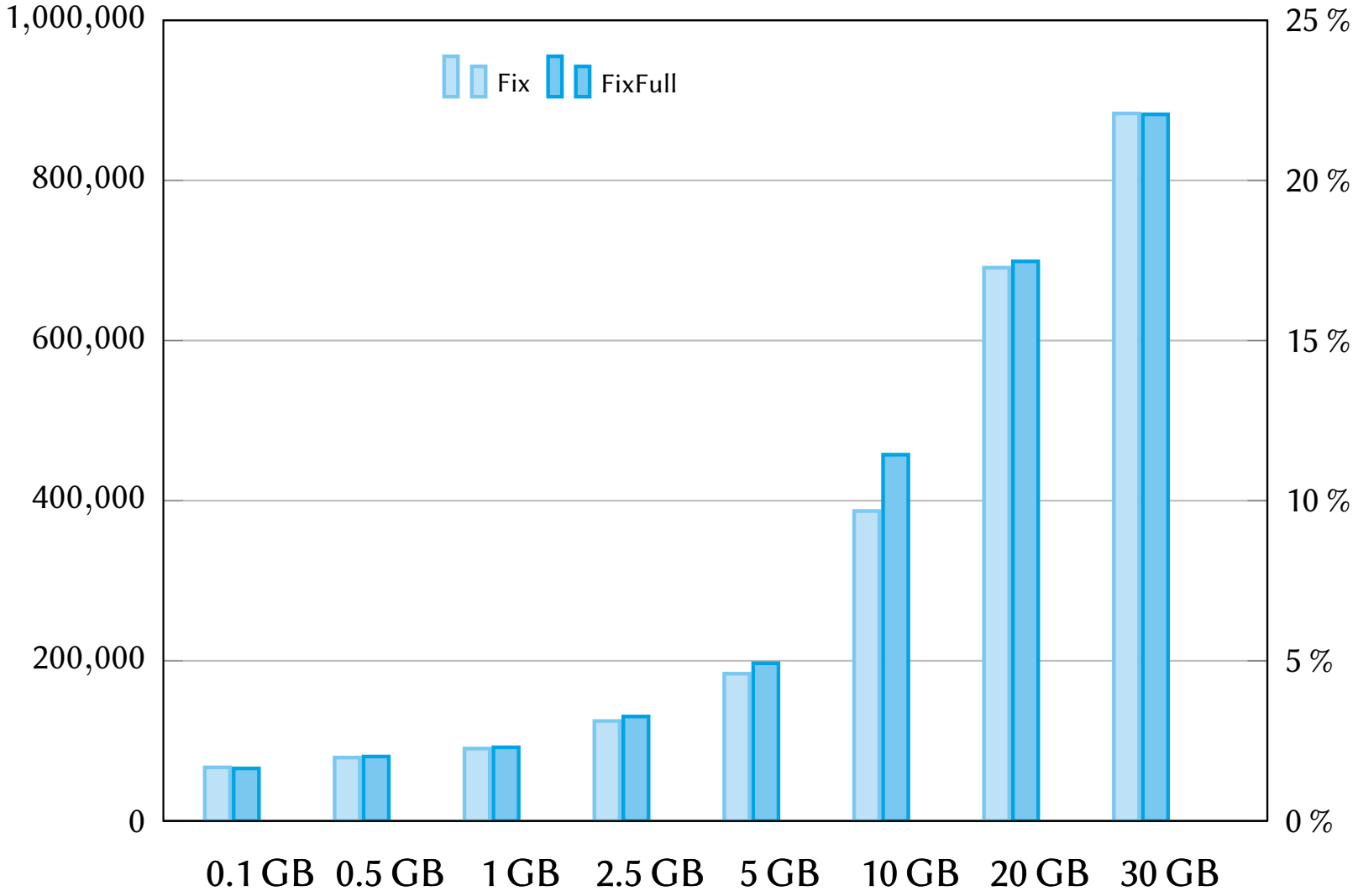
Fix Increase usage-count on *page fix*

FixFull Like Fix but also increase usage-count when page cannot be evicted

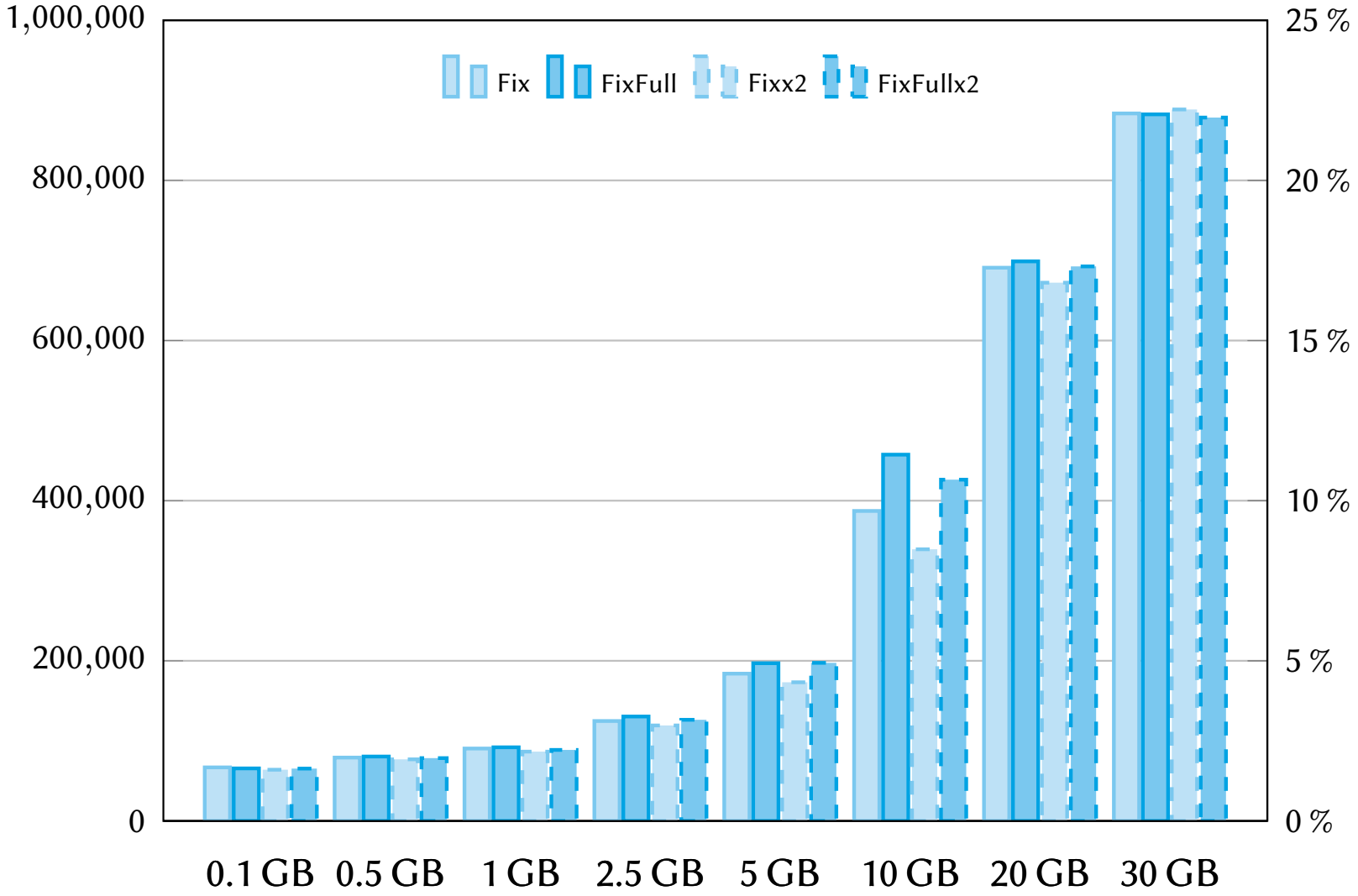
$$F_{\text{non-B-tree}} = 25,\ F_{\text{root}} = 25,\ F_{\text{root}-1} = 10, F_{\text{root}-\geq 2} = 5,$$
$$R_{\text{non-B-tree}} = 5,\ R_{\text{root}} = 5,\ R_{\text{root}-1} = 2,\ R_{\text{root}-\geq 2} = 1$$

x2
$$F_{\text{non-B-tree}} = 50,\ F_{\text{root}} = 50,\ F_{\text{root}-1} = 25, F_{\text{root}-\geq 2} = 10,$$
$$R_{\text{non-B-tree}} = 10,\ R_{\text{root}} = 10,\ R_{\text{root}-1} = 5,\ R_{\text{root}-\geq 2} = 2$$

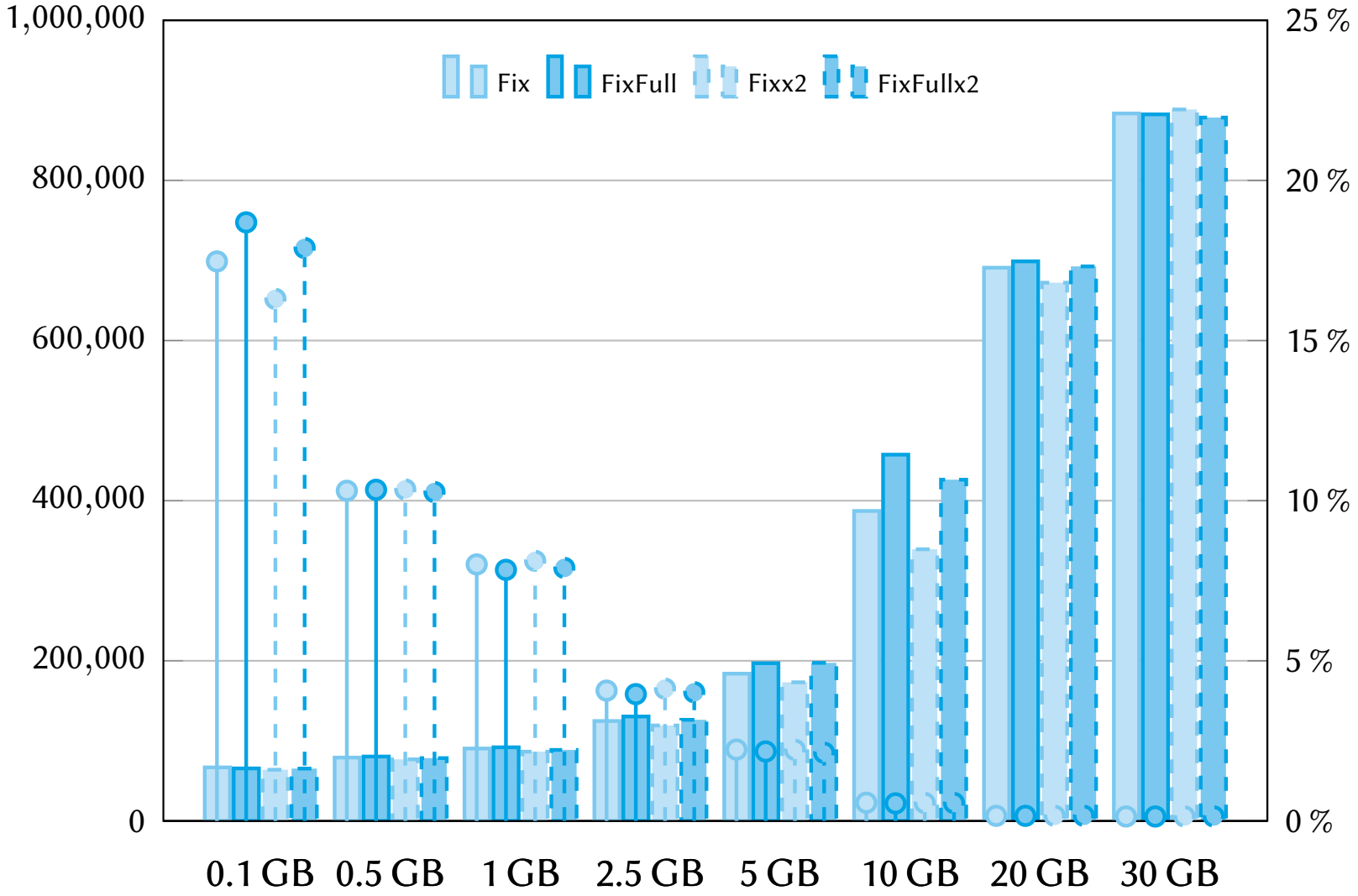
Performance Evaluation



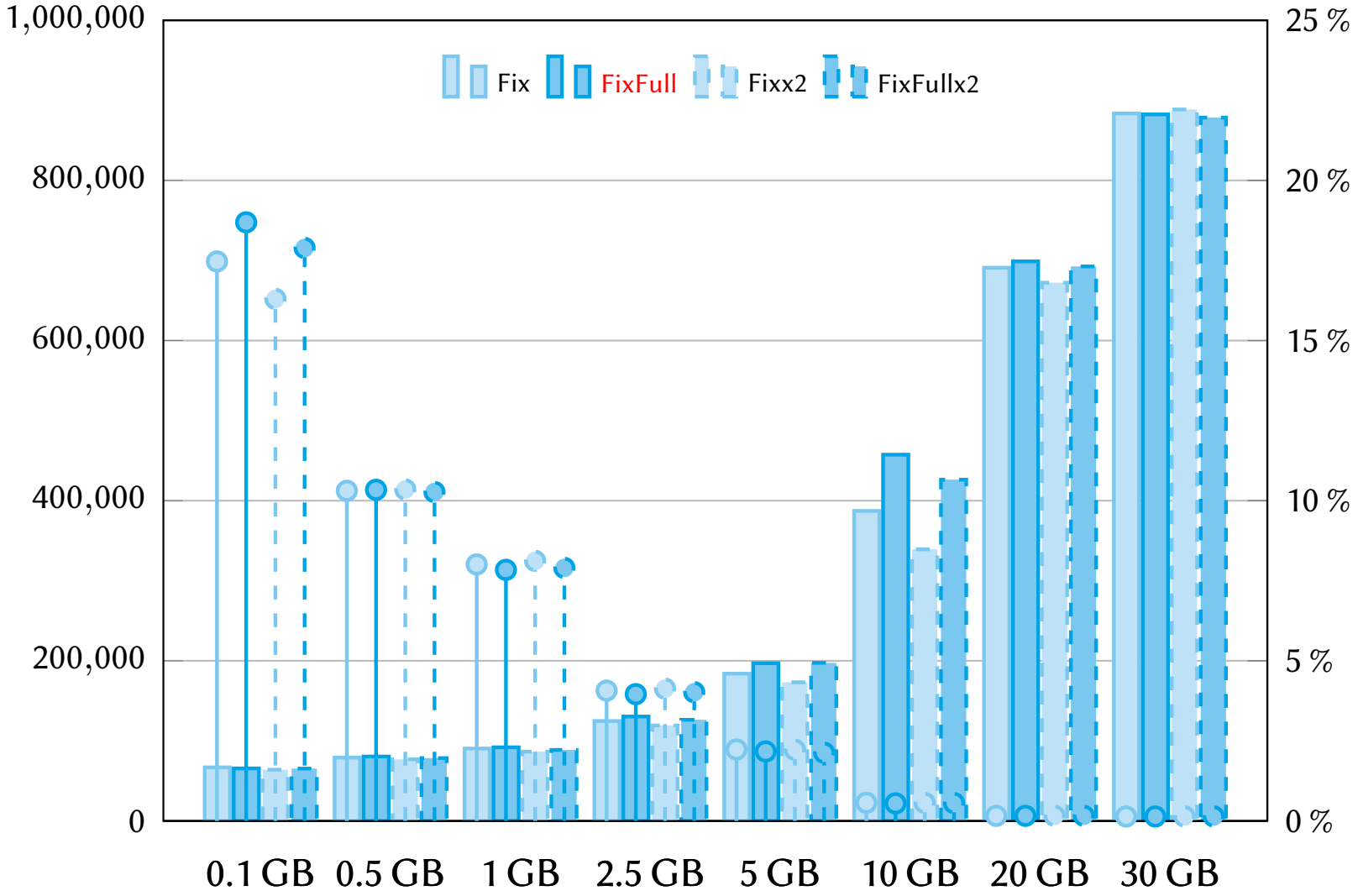
Performance Evaluation



Performance Evaluation



Performance Evaluation



DGCLOCK-V2 Variants

Fix Set usage-count on *page fix*

FixFull Like Fix but also set usage-count when page cannot be evicted

$$F_{\text{non-B-tree}} = 25,\ F_{\text{root}} = 25,\ F_{\text{root}-1} = 10, F_{\text{root}-\geq 2} = 5,$$
$$R_{\text{non-B-tree}} = 5,\ R_{\text{root}} = 5,\ R_{\text{root}-1} = 2,\ R_{\text{root}-\geq 2} = 1$$

x2
$$F_{\text{non-B-tree}} = 50,\ F_{\text{root}} = 50,\ F_{\text{root}-1} = 25, F_{\text{root}-\geq 2} = 10,$$
$$R_{\text{non-B-tree}} = 10,\ R_{\text{root}} = 10,\ R_{\text{root}-1} = 5,\ R_{\text{root}-\geq 2} = 2$$

DGCLOCK-V2 Variants

Usage-Count

$$\begin{aligned} \text{(initially)} \quad & \mathrm{UC}(p) = F_p \\ \text{(set)} \quad & \mathrm{UC}(p) = R_p \\ \text{(decrease)} \quad & \mathrm{UC}(p) = \mathrm{UC}(p) - 1 \end{aligned}$$

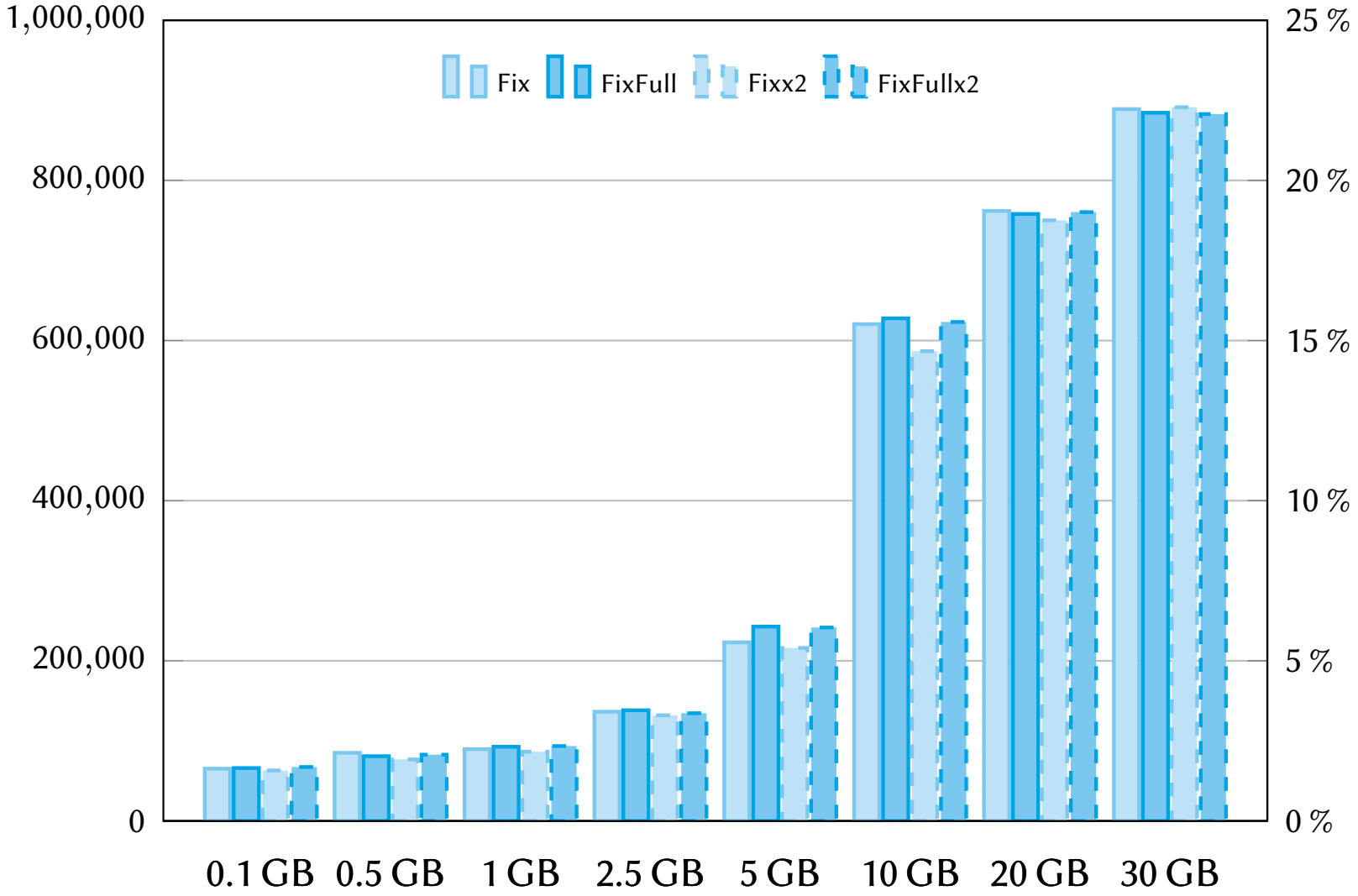
Fix Set usage-count on *page fix*

FixFull Like Fix but also set usage-count when page cannot be evicted

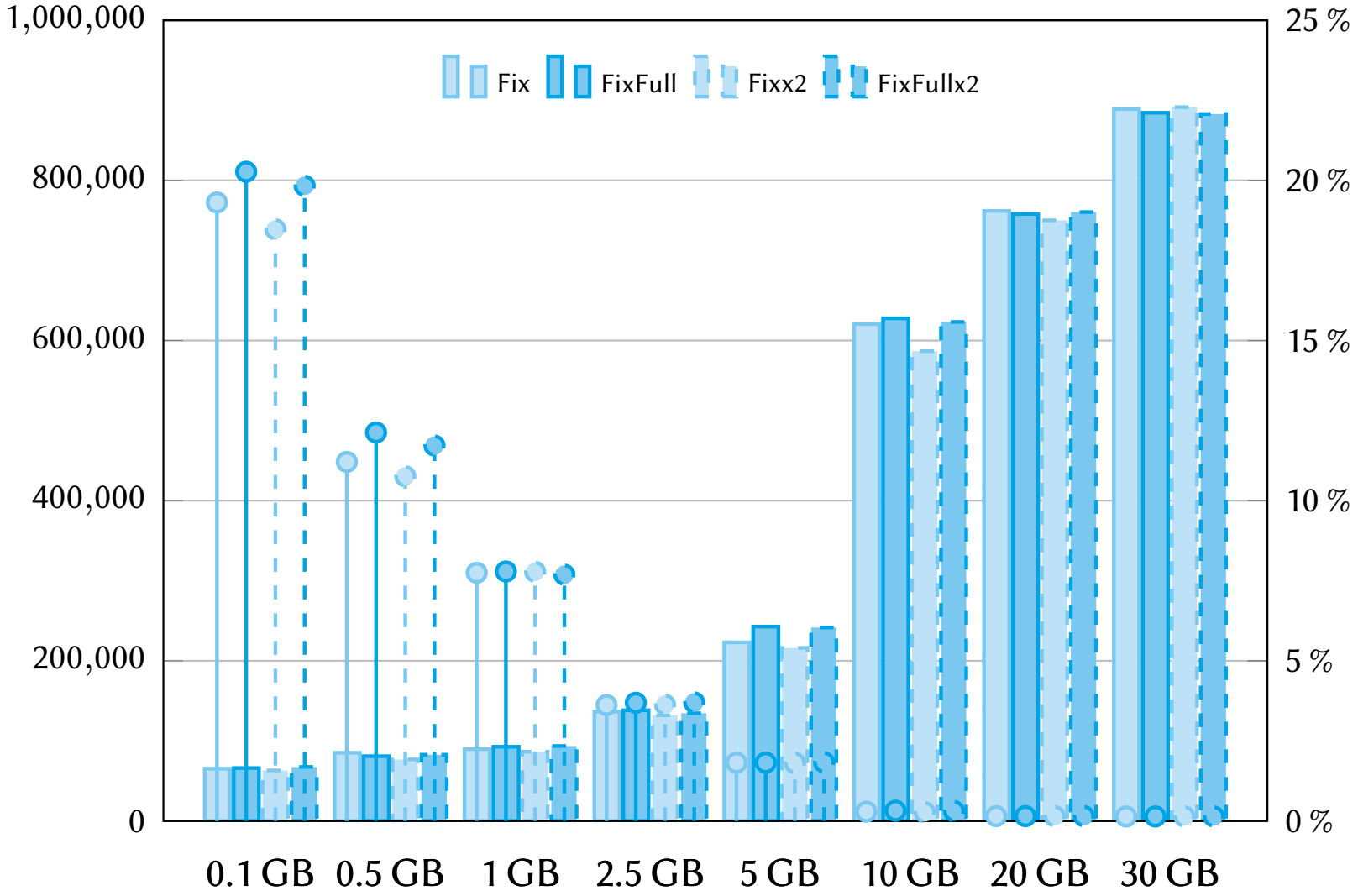
$$F_{\text{non-B-tree}} = 25,\ F_{\text{root}} = 25,\ F_{\text{root}-1} = 10, F_{\text{root}-\geq 2} = 5,$$
$$R_{\text{non-B-tree}} = 5,\ R_{\text{root}} = 5,\ R_{\text{root}-1} = 2,\ R_{\text{root}-\geq 2} = 1$$

x2
$$F_{\text{non-B-tree}} = 50,\ F_{\text{root}} = 50,\ F_{\text{root}-1} = 25, F_{\text{root}-\geq 2} = 10,$$
$$R_{\text{non-B-tree}} = 10,\ R_{\text{root}} = 10,\ R_{\text{root}-1} = 5,\ R_{\text{root}-\geq 2} = 2$$

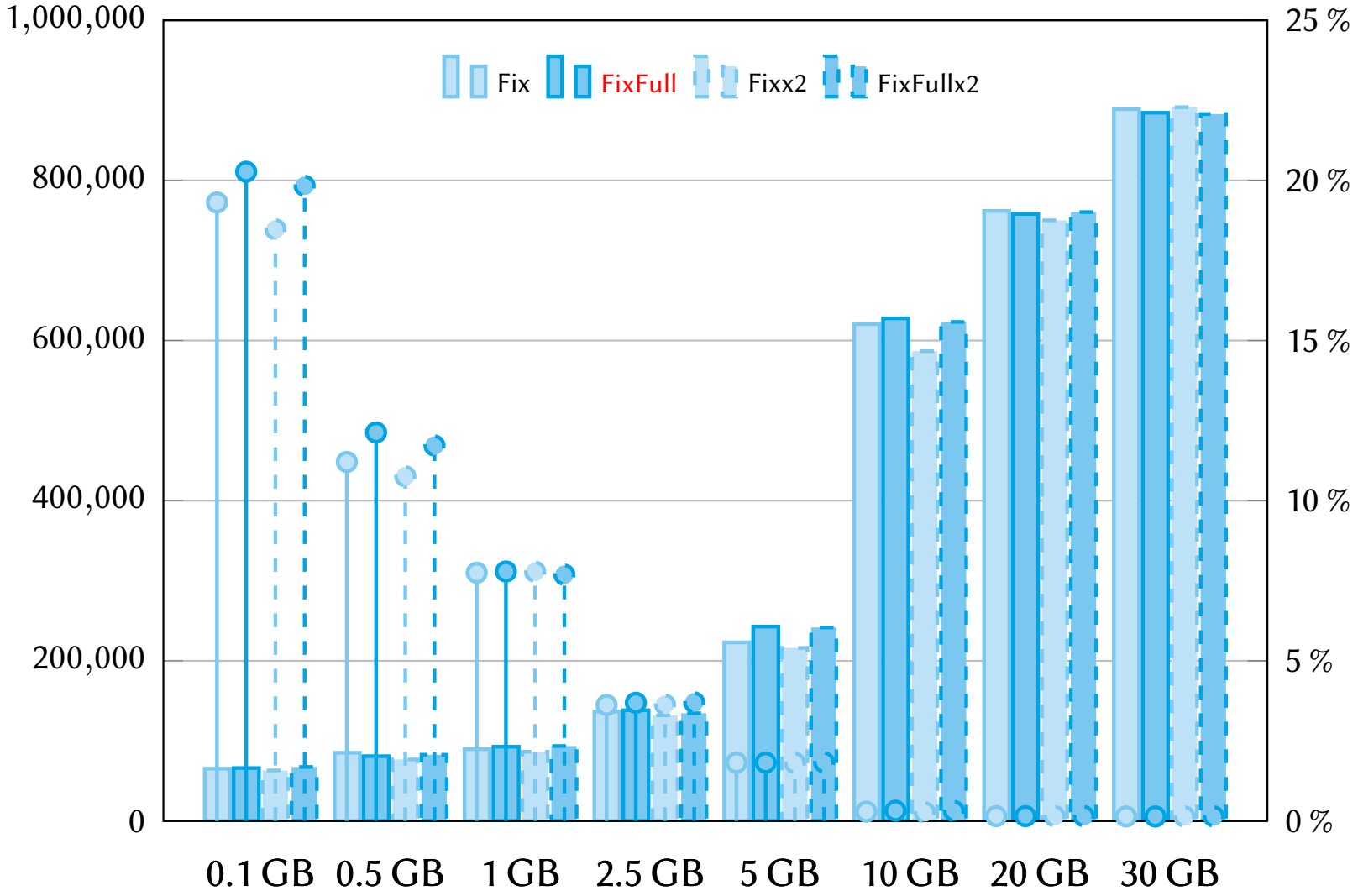
Performance Evaluation



Performance Evaluation



Performance Evaluation



LRD-V2 Variants

Reference Counts

$$\text{(aging)} \qquad \mathrm{UC}(p) = \mathrm{UC}(p) \circ x$$

$$\text{(increase)} \qquad \mathrm{RC}(p) = \mathrm{RC}(p) + 1$$

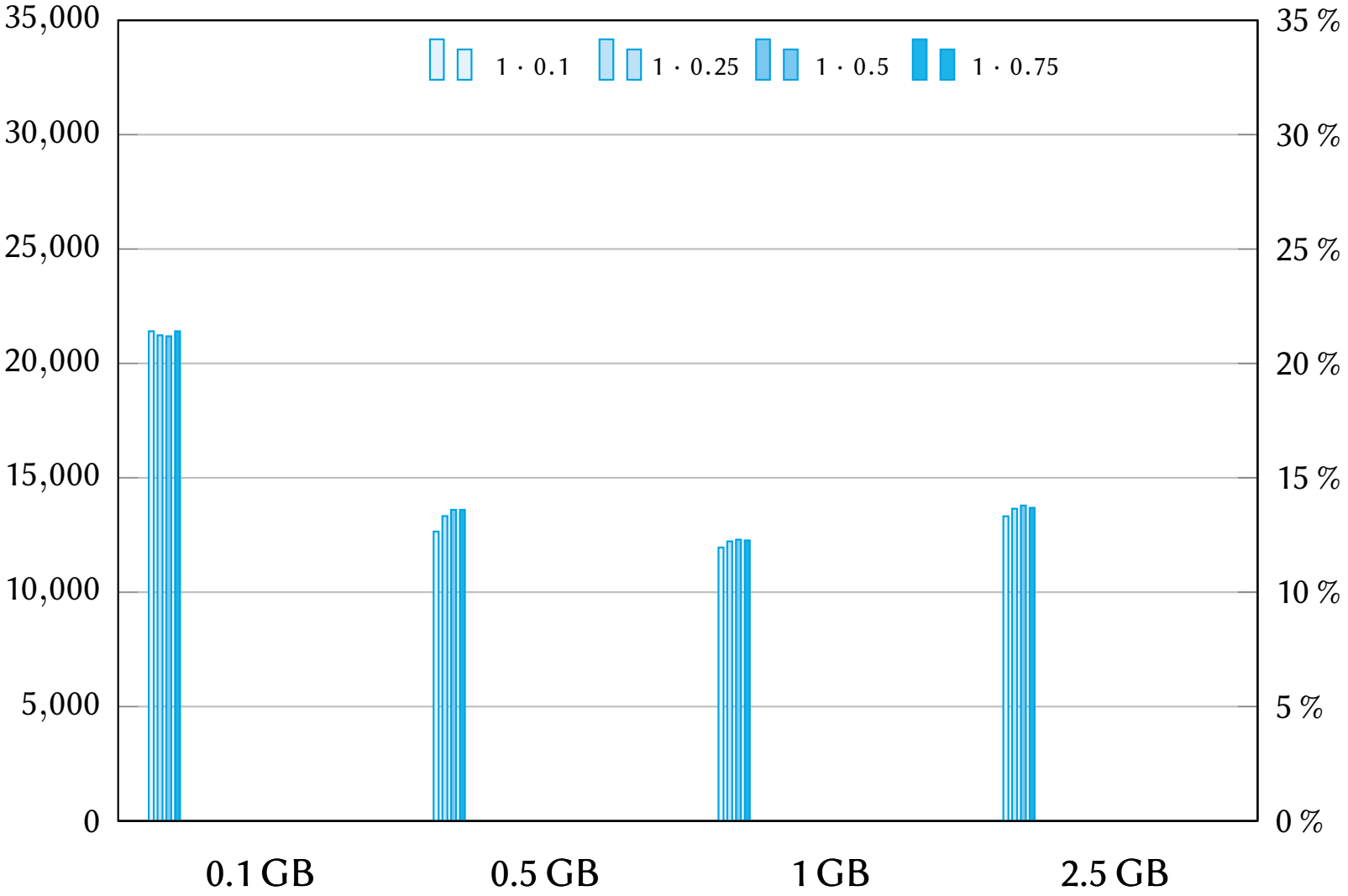
LRD-V2 Variants

Reference Counts

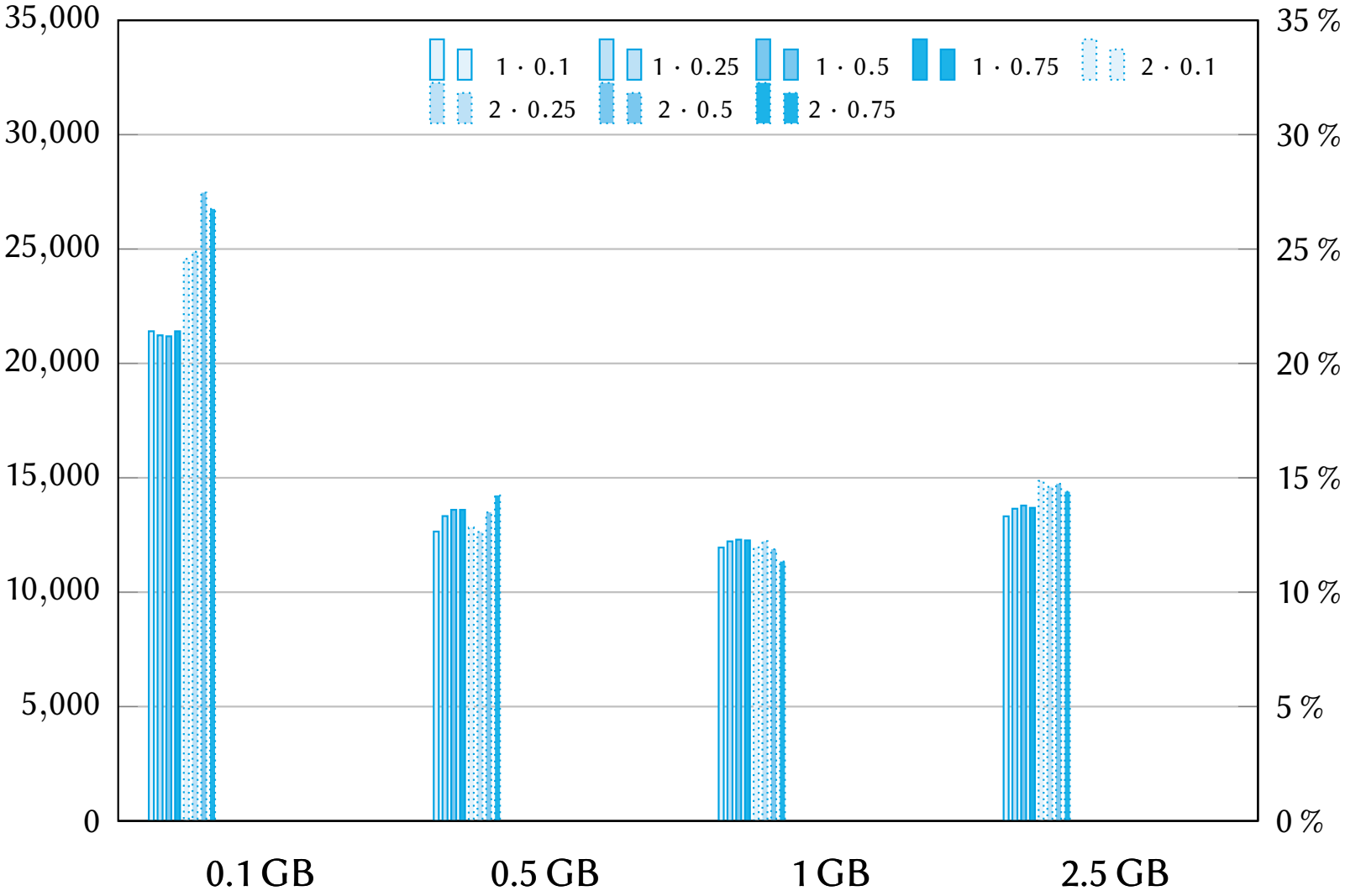
$$\text{(aging)} \quad \mathrm{UC}(p) = \mathrm{UC}(p) \circ x$$
$$\text{(increase)} \quad \mathrm{RC}(p) = \mathrm{RC}(p) + 1$$

$f \circ x$ Perform aging (using $\circ x$) on each reference count every $f \cdot \text{bufferPoolSize}$ global page references

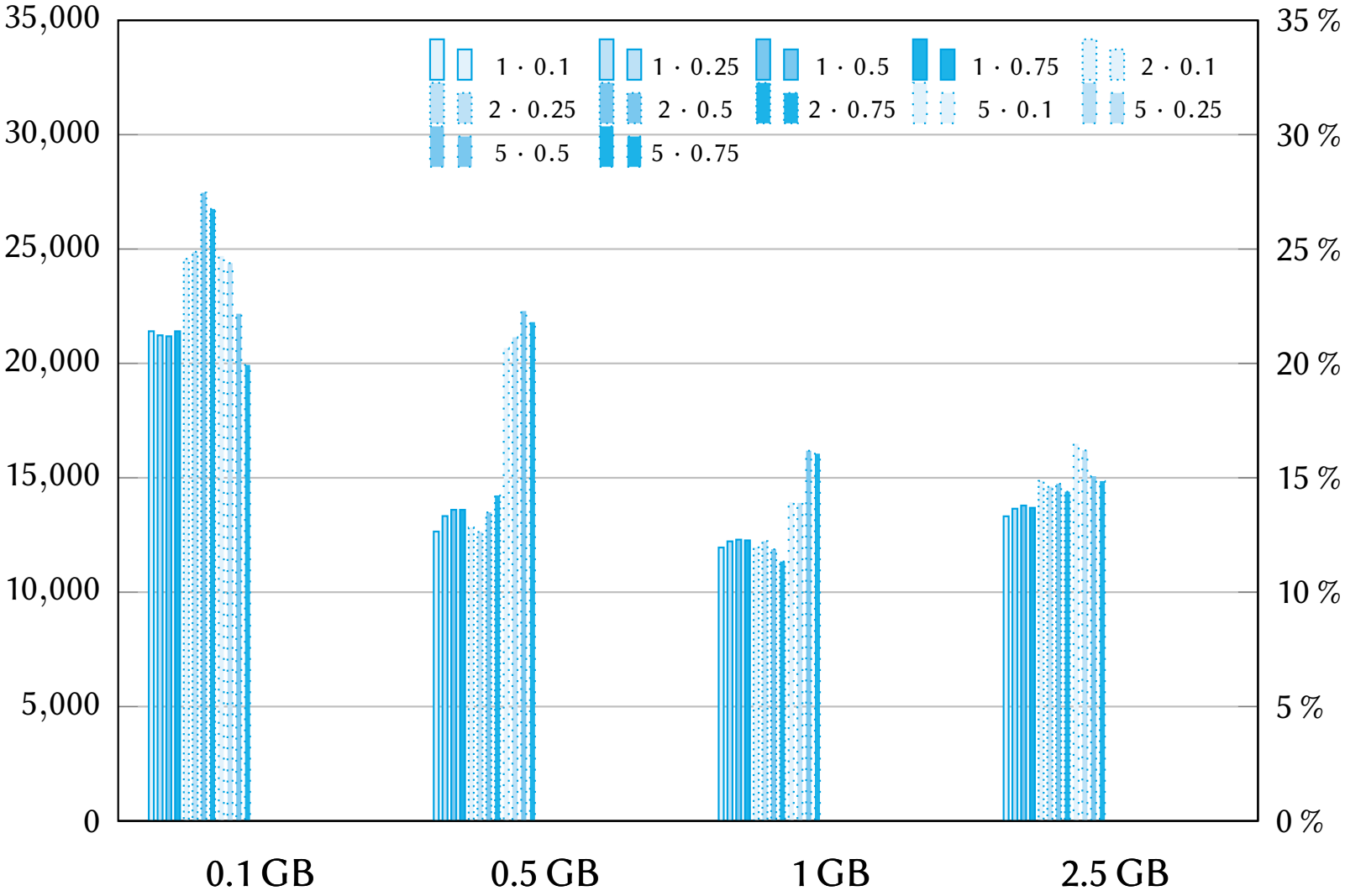
Performance Evaluation I



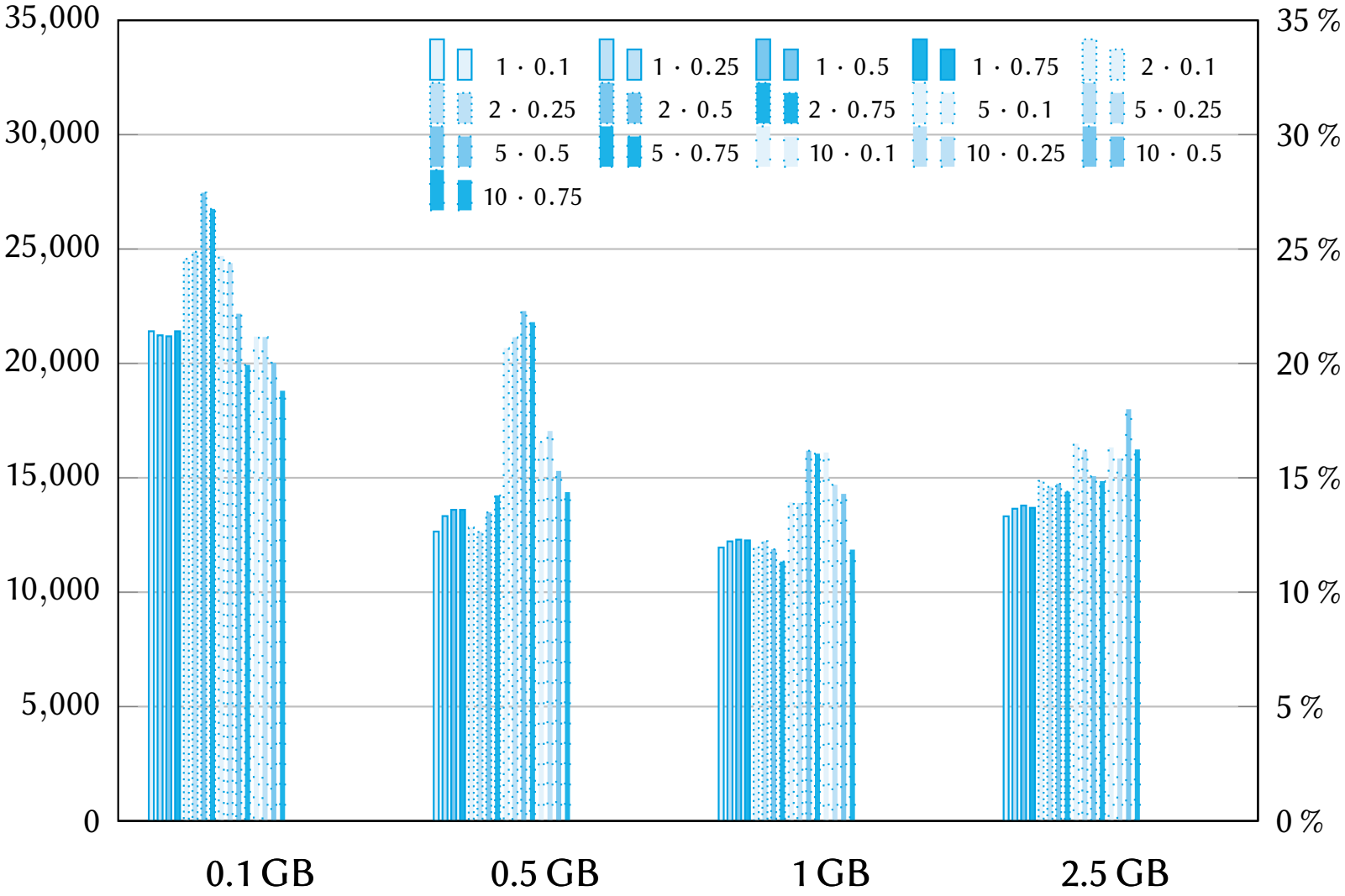
Performance Evaluation I



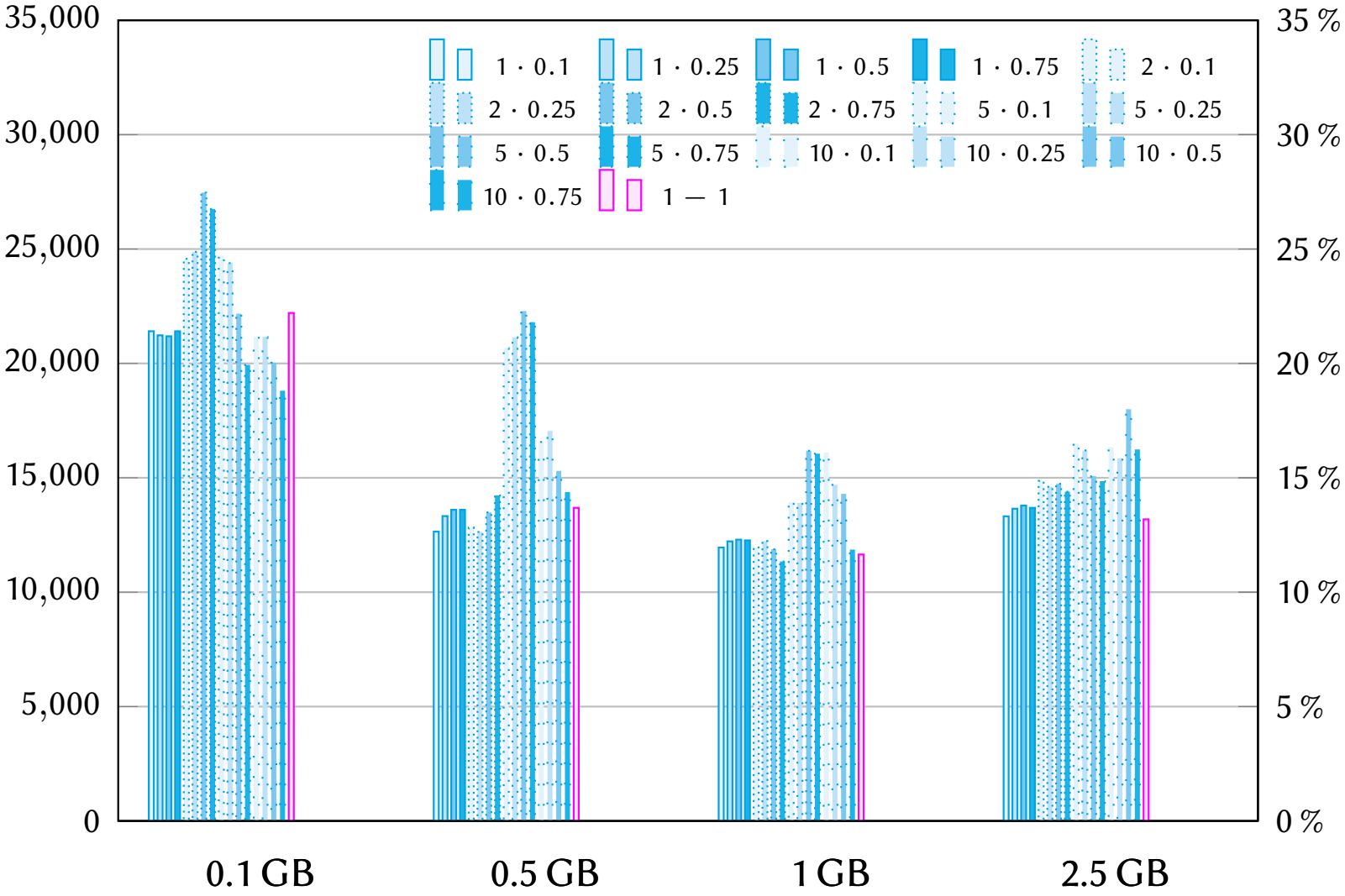
Performance Evaluation I



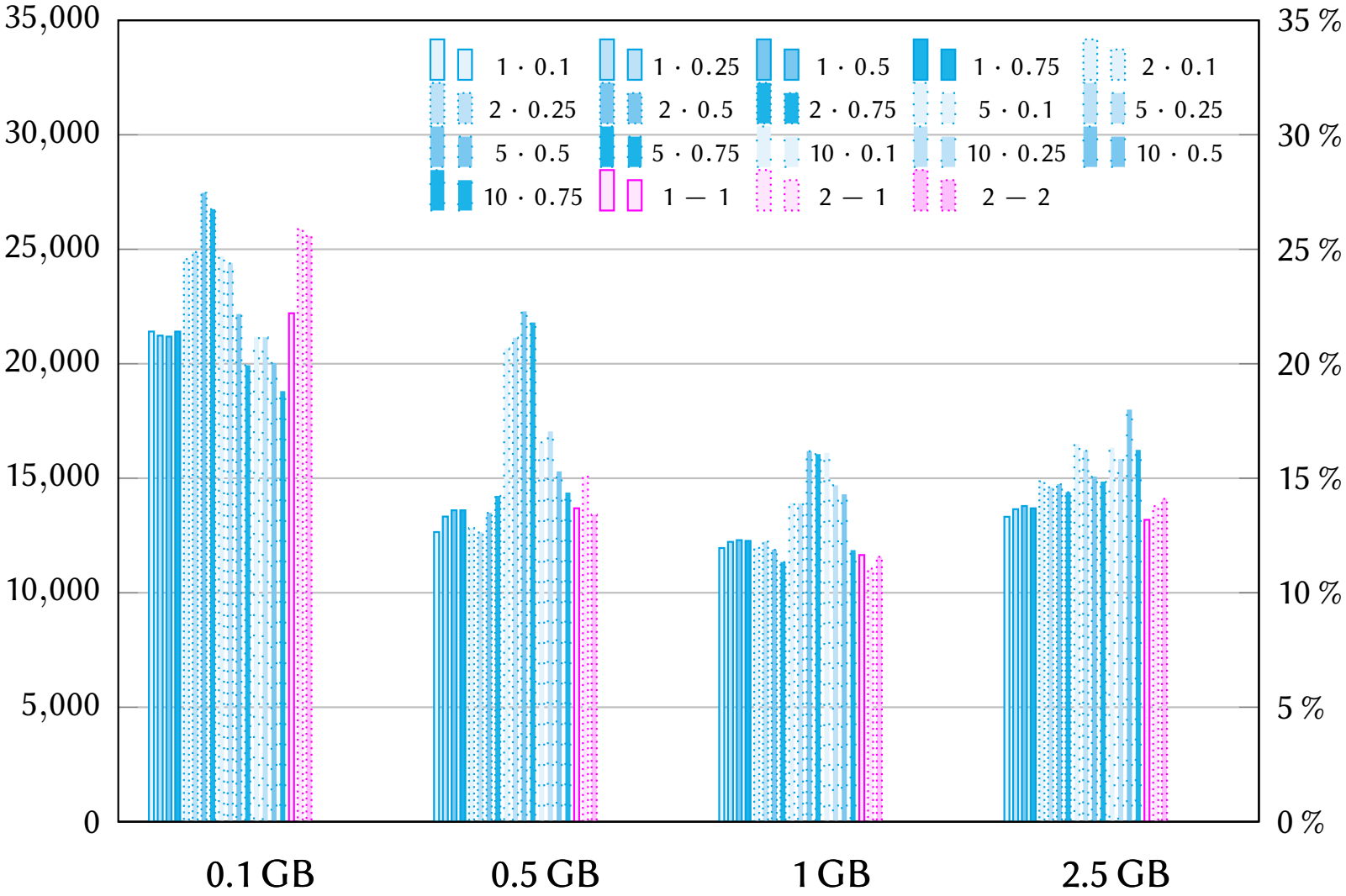
Performance Evaluation I



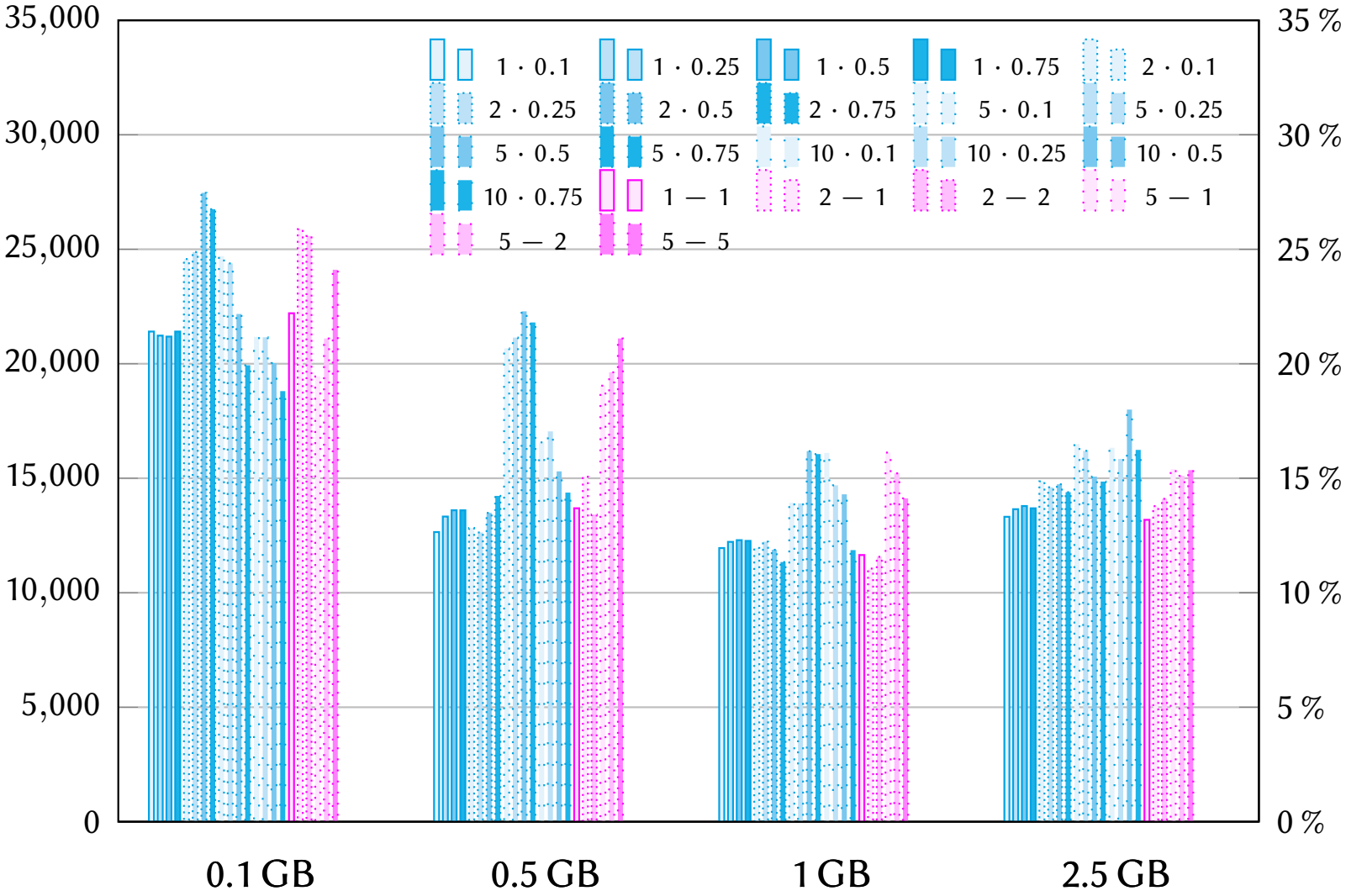
Performance Evaluation I



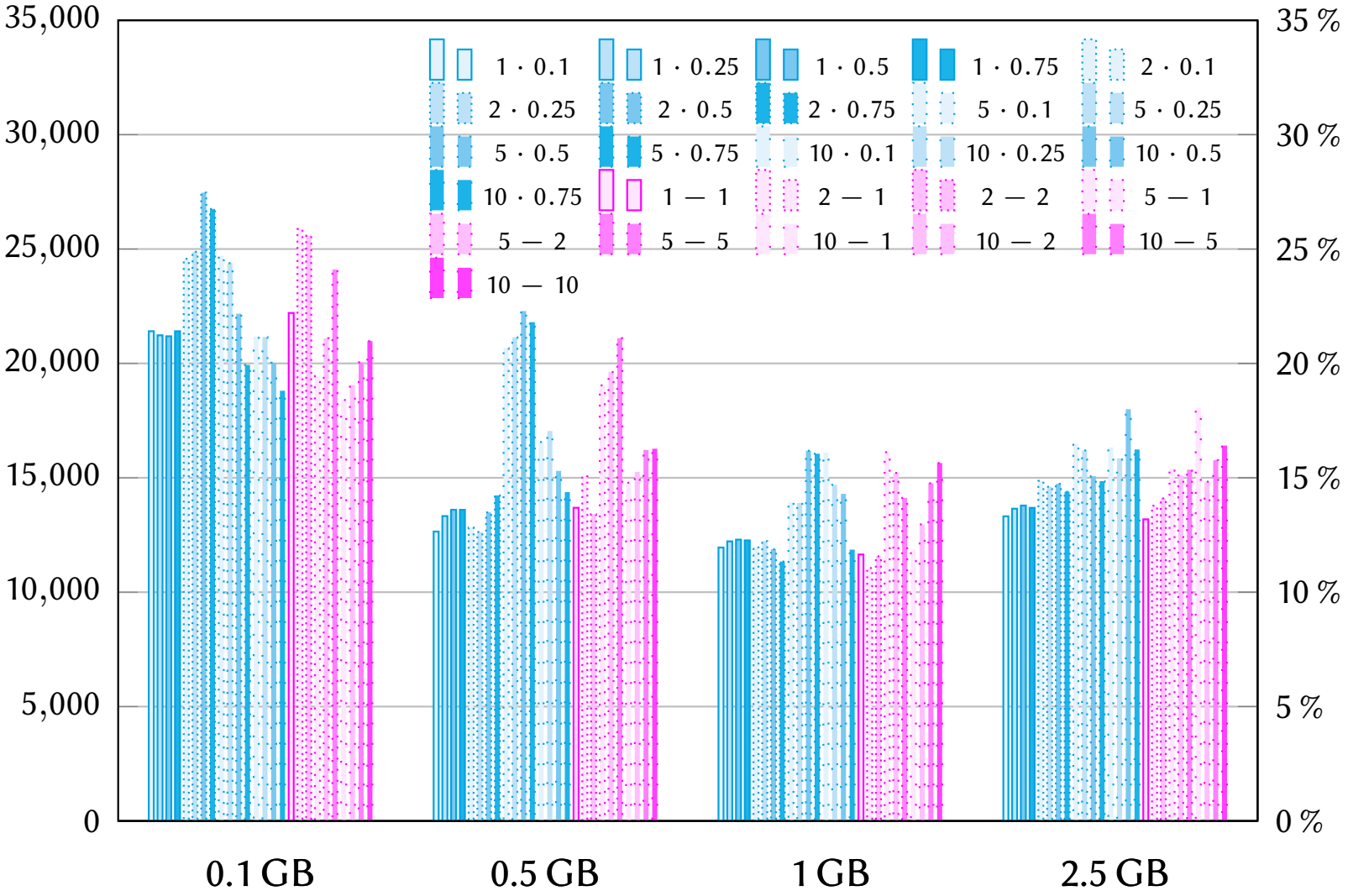
Performance Evaluation I



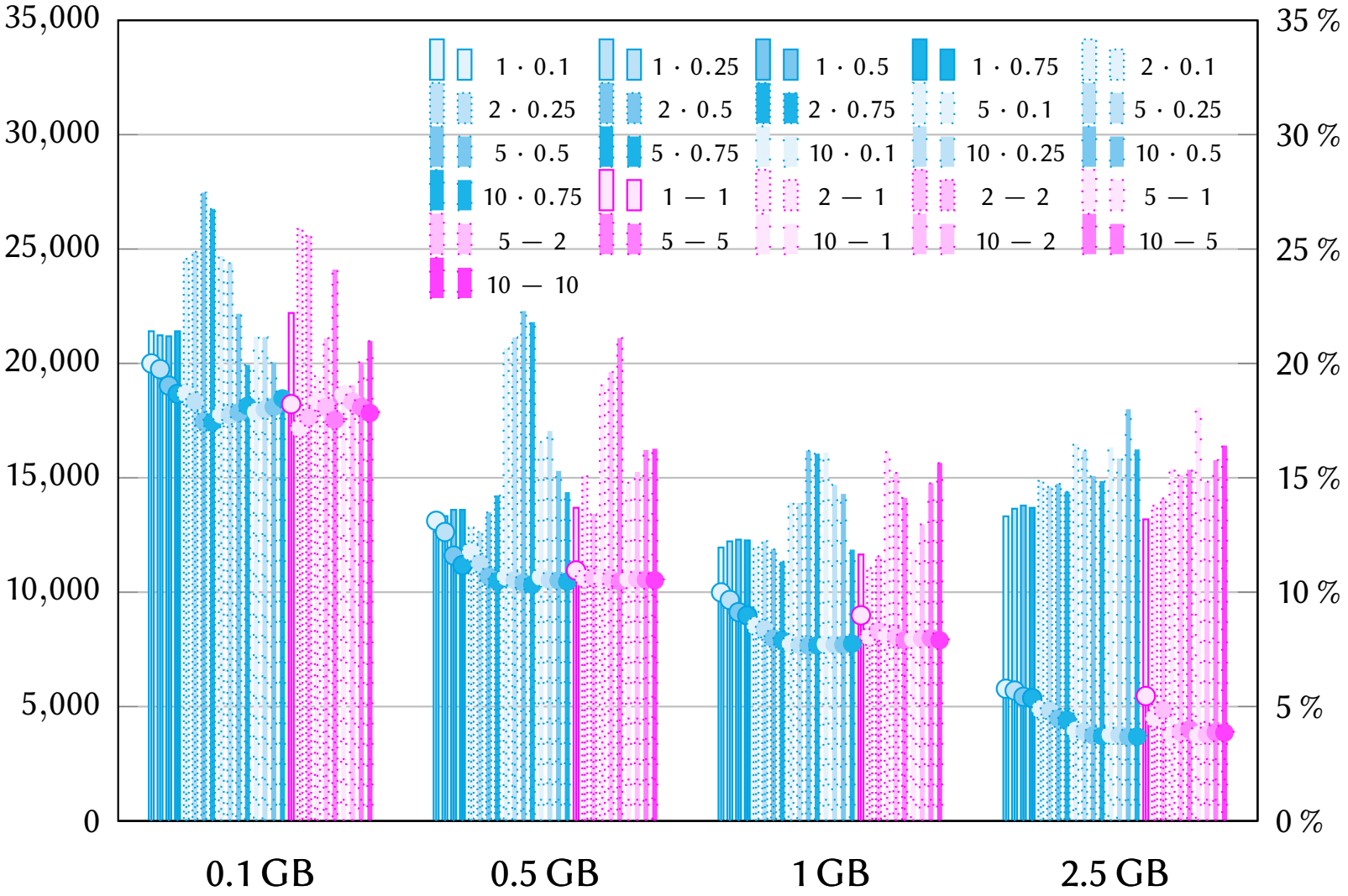
Performance Evaluation I



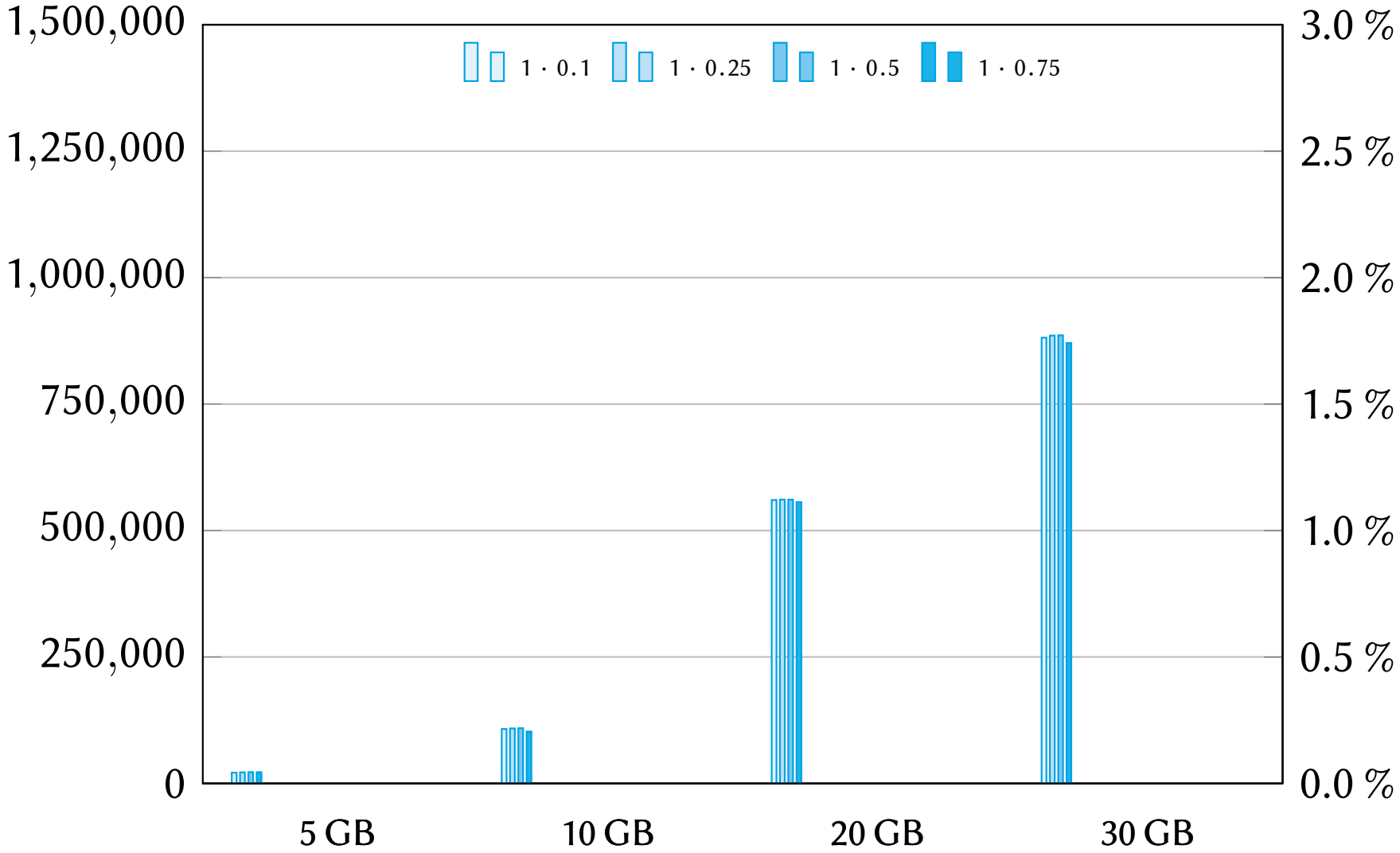
Performance Evaluation I



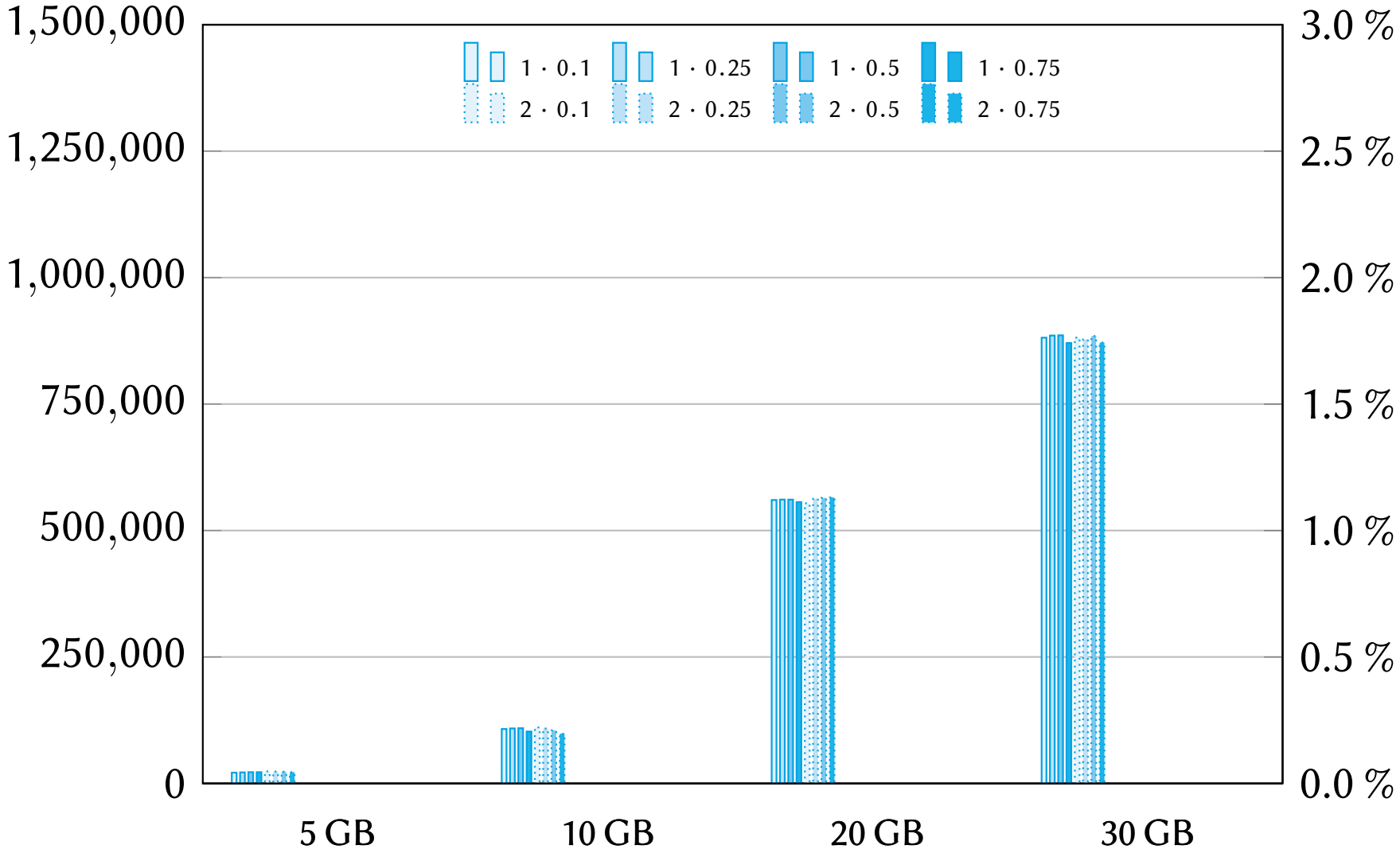
Performance Evaluation I



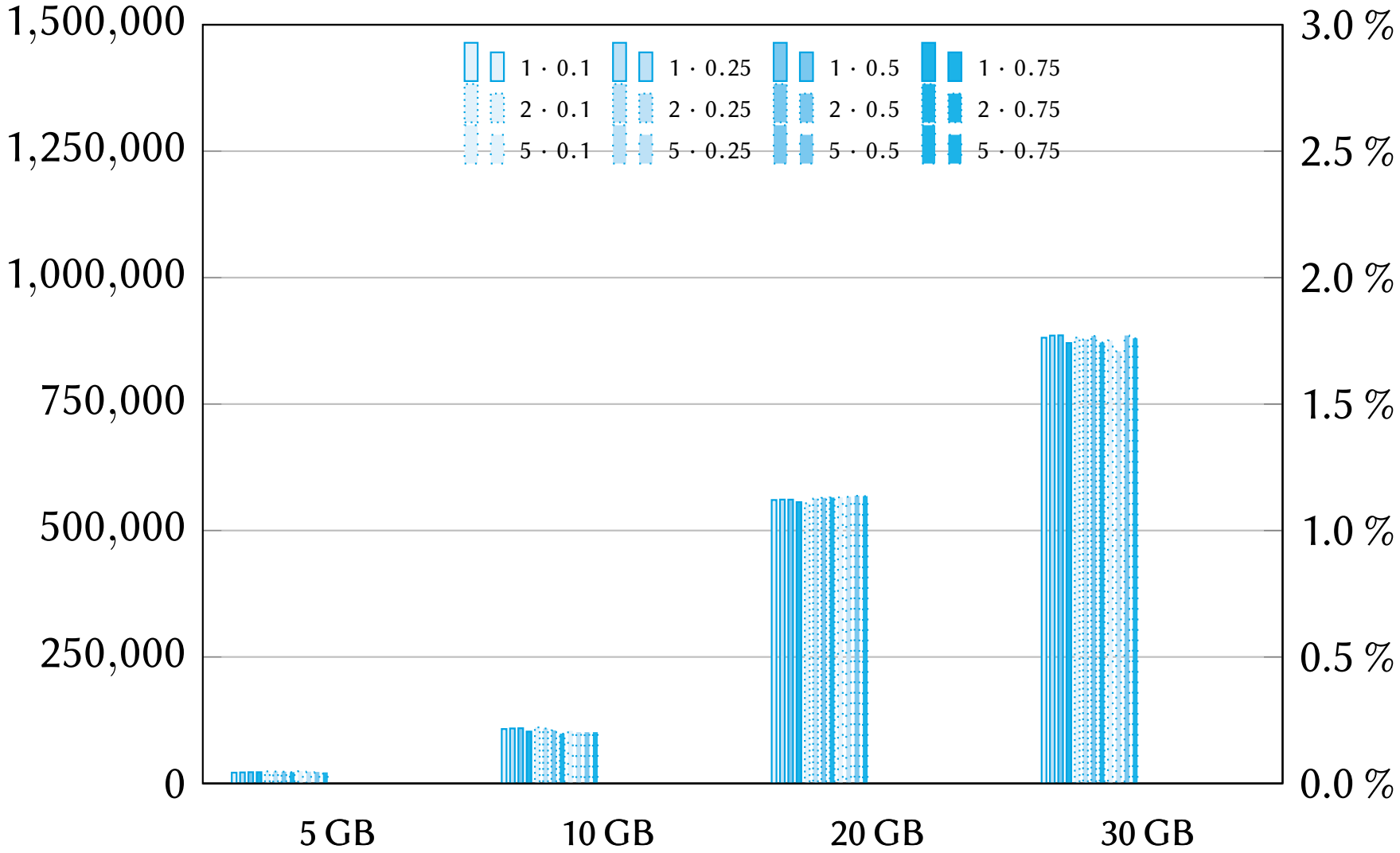
Performance Evaluation II



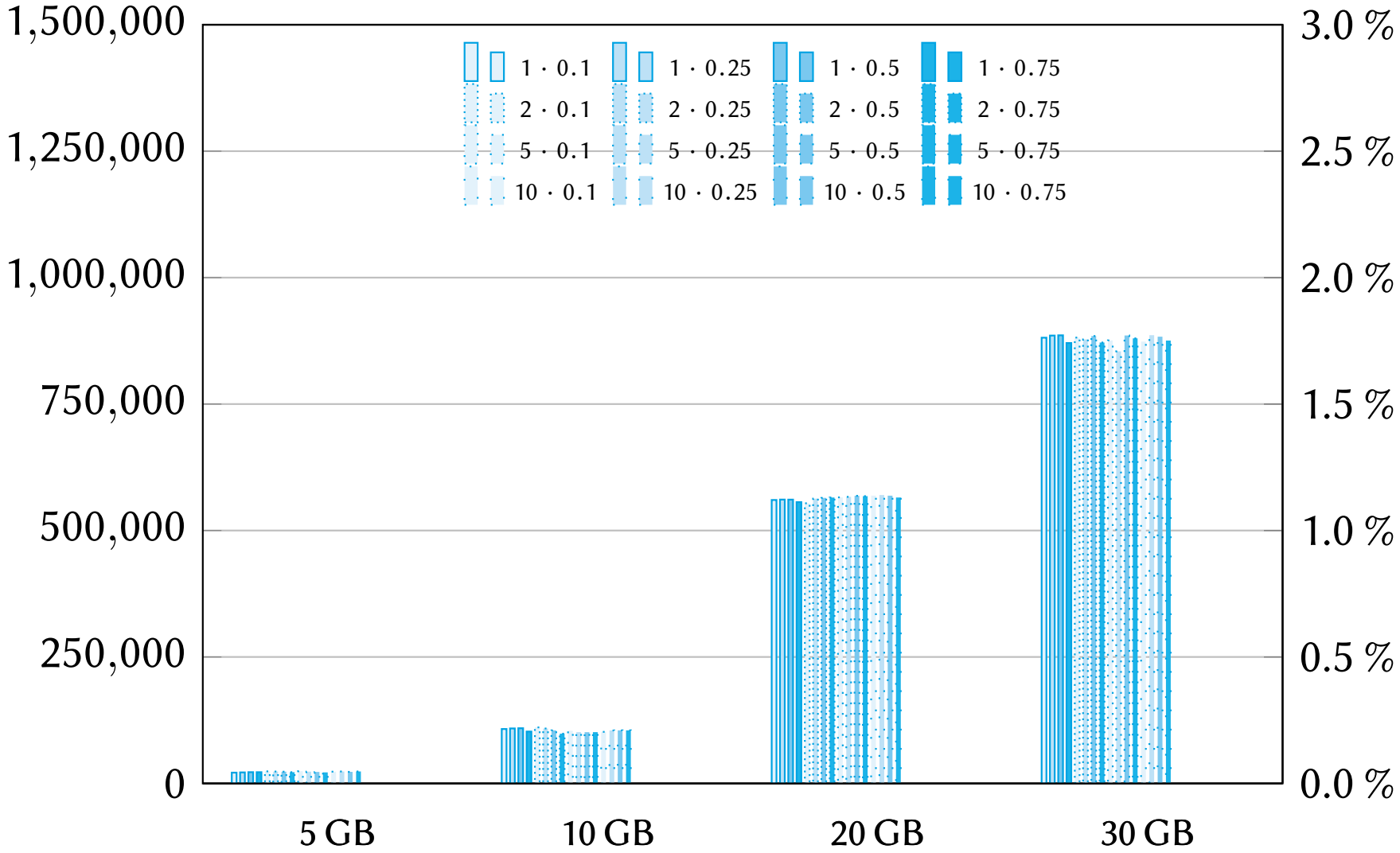
Performance Evaluation II



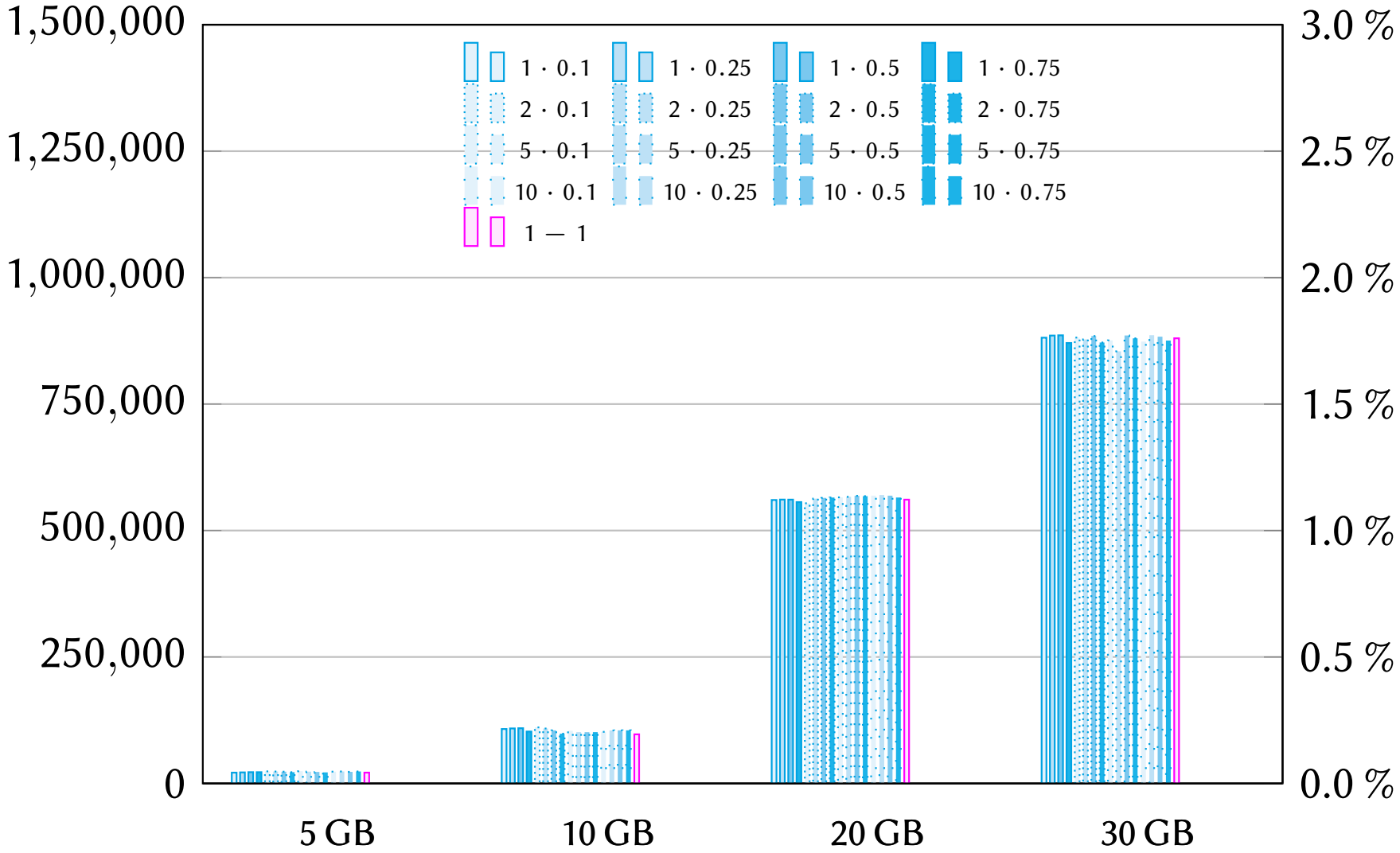
Performance Evaluation II



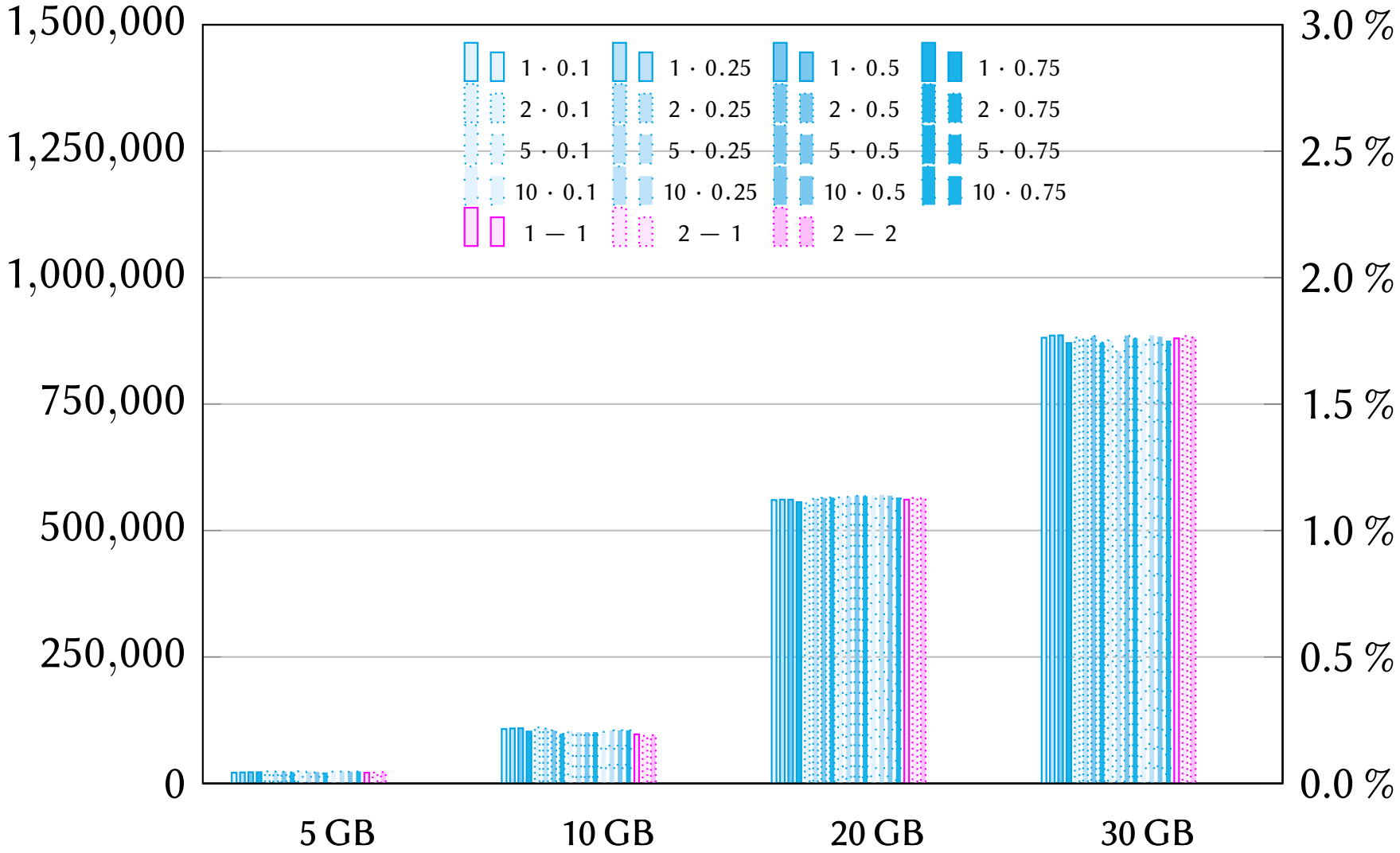
Performance Evaluation II



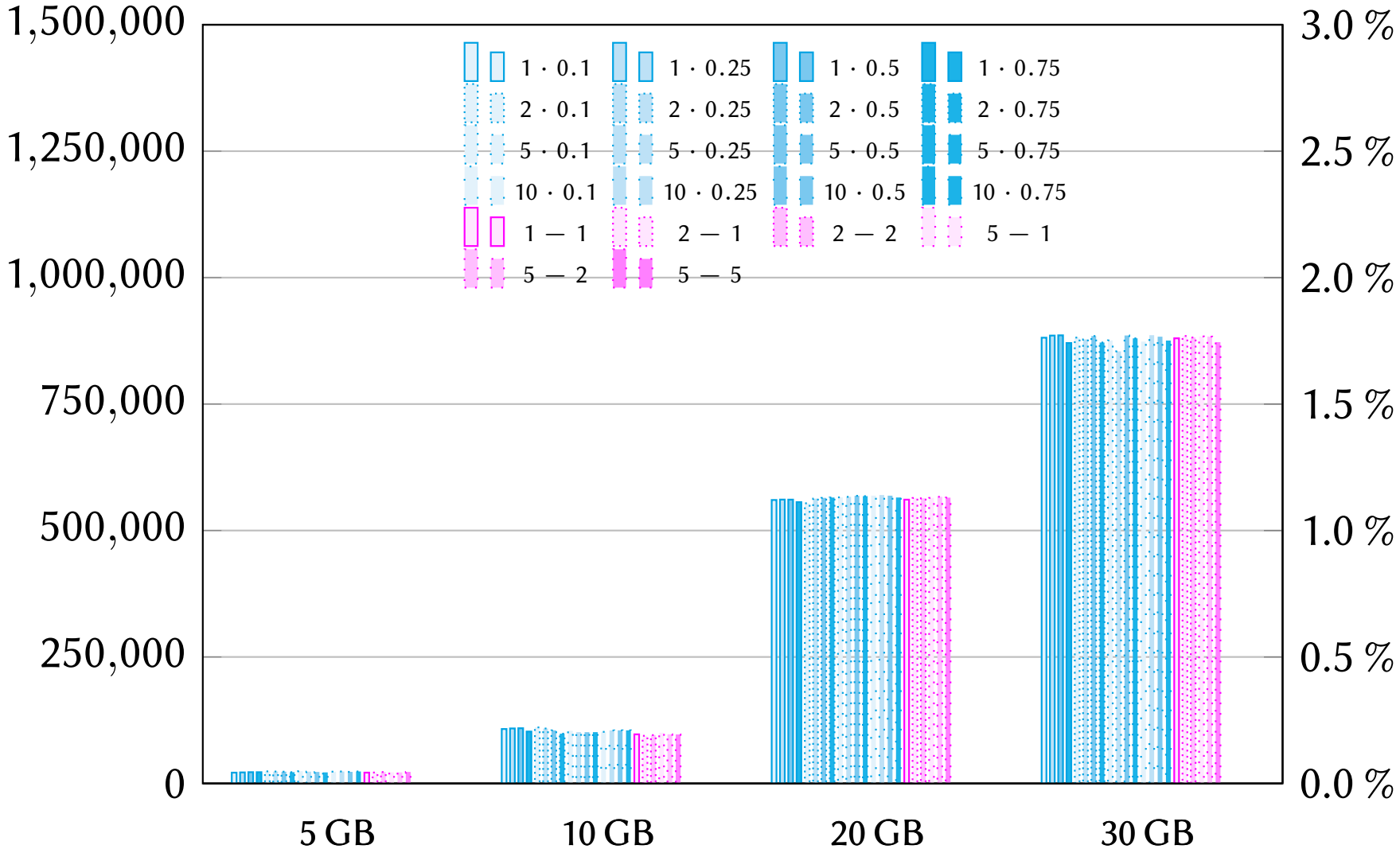
Performance Evaluation II



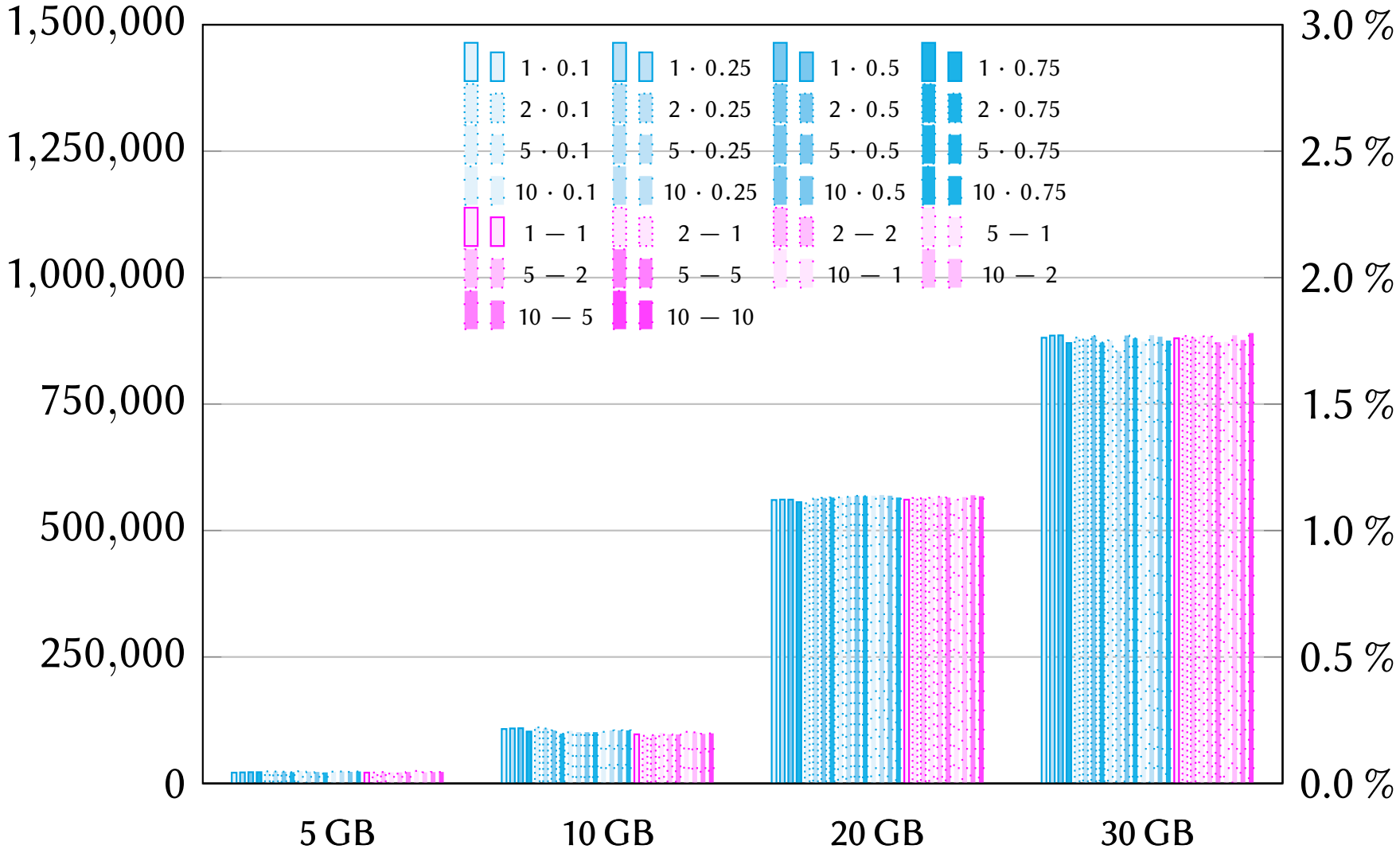
Performance Evaluation II



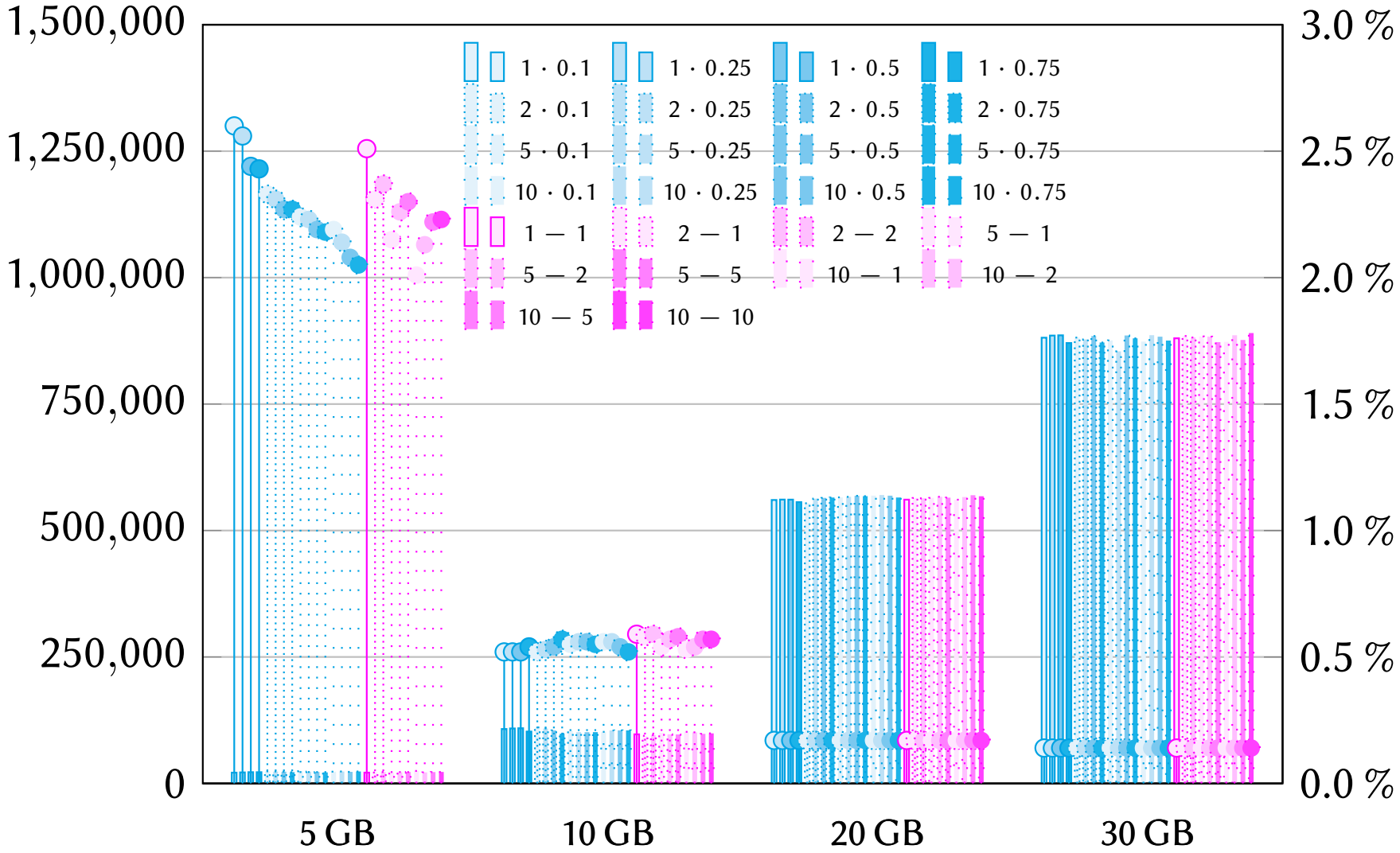
Performance Evaluation II



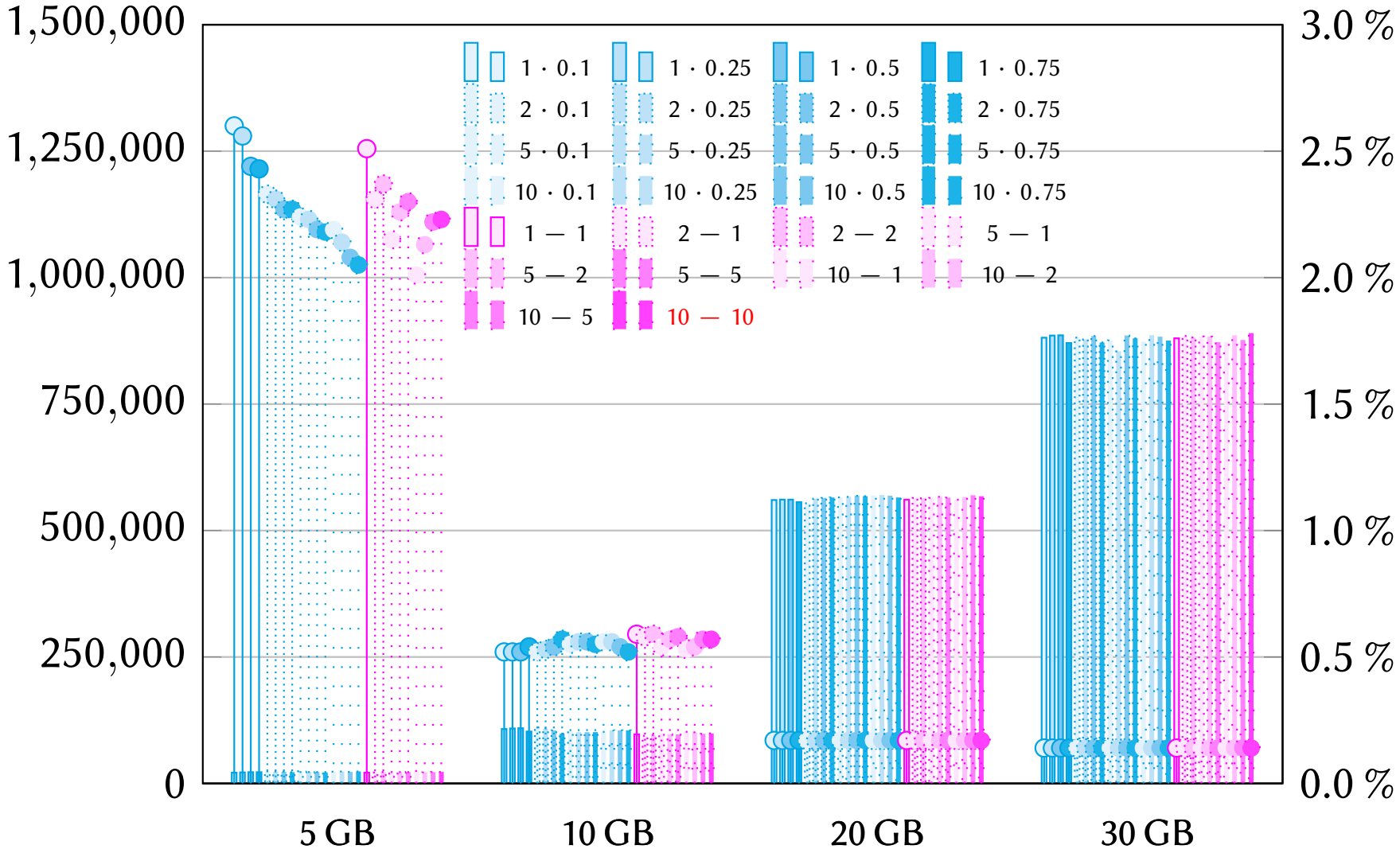
Performance Evaluation II



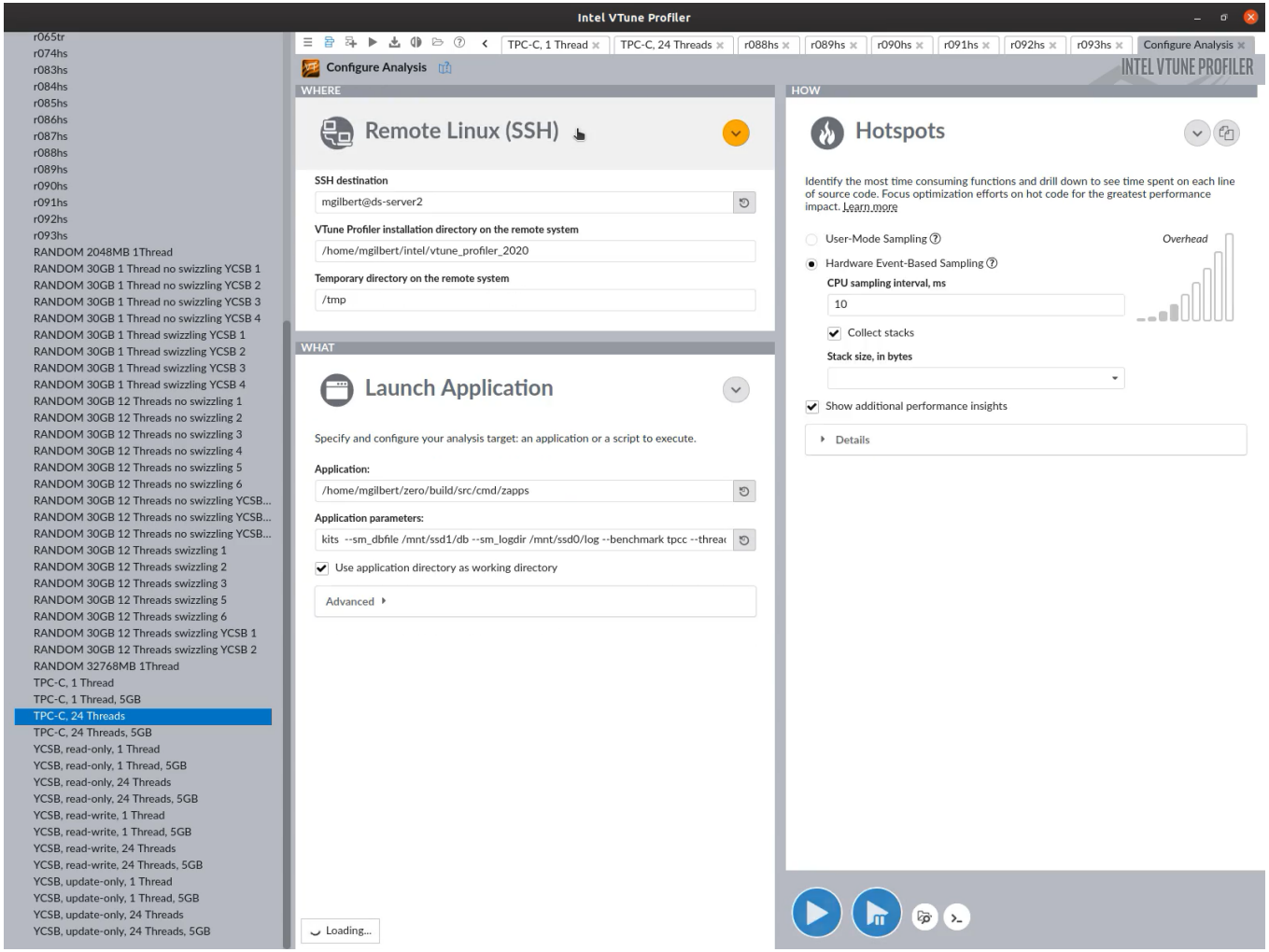
Performance Evaluation II



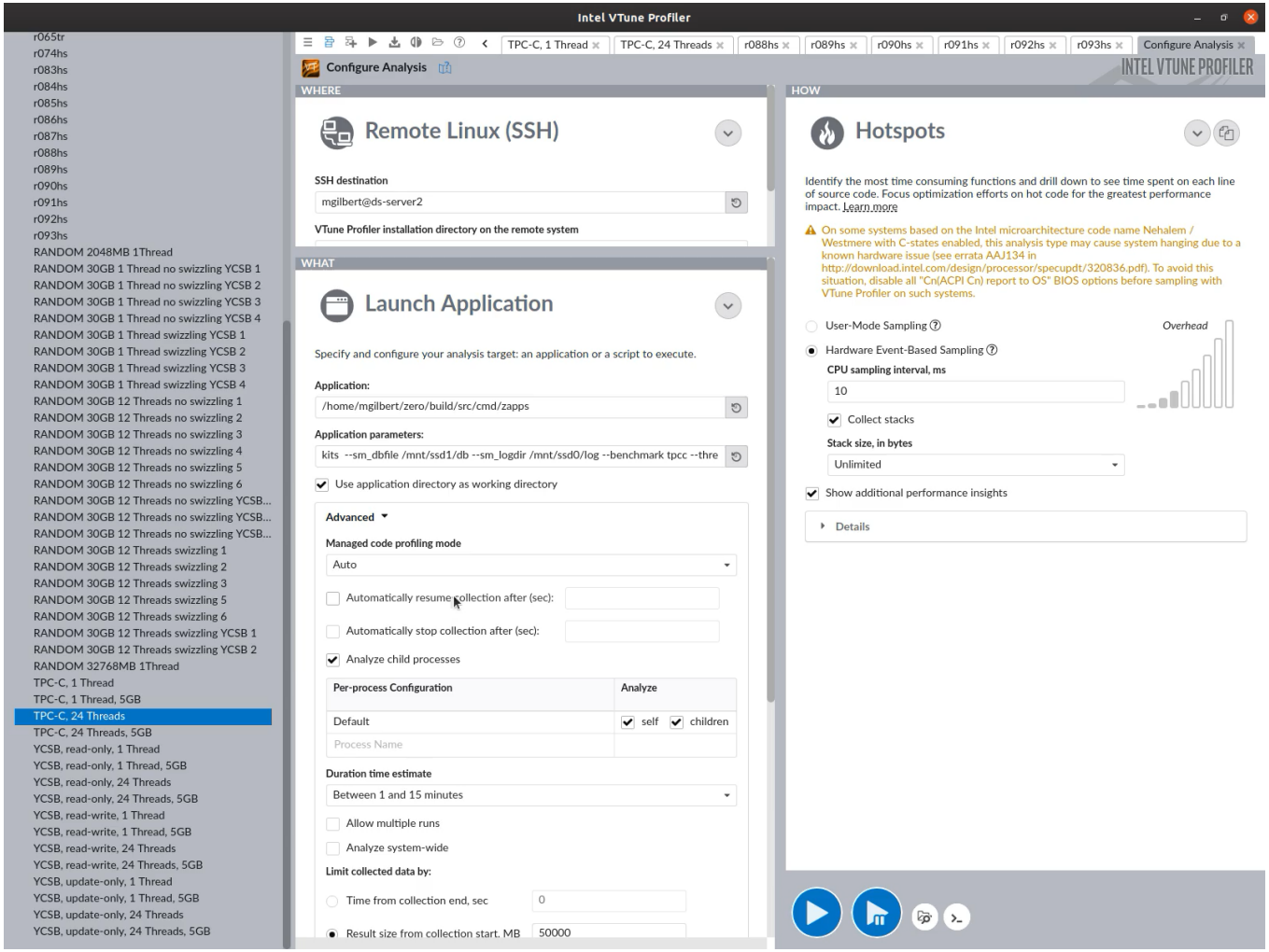
Performance Evaluation II



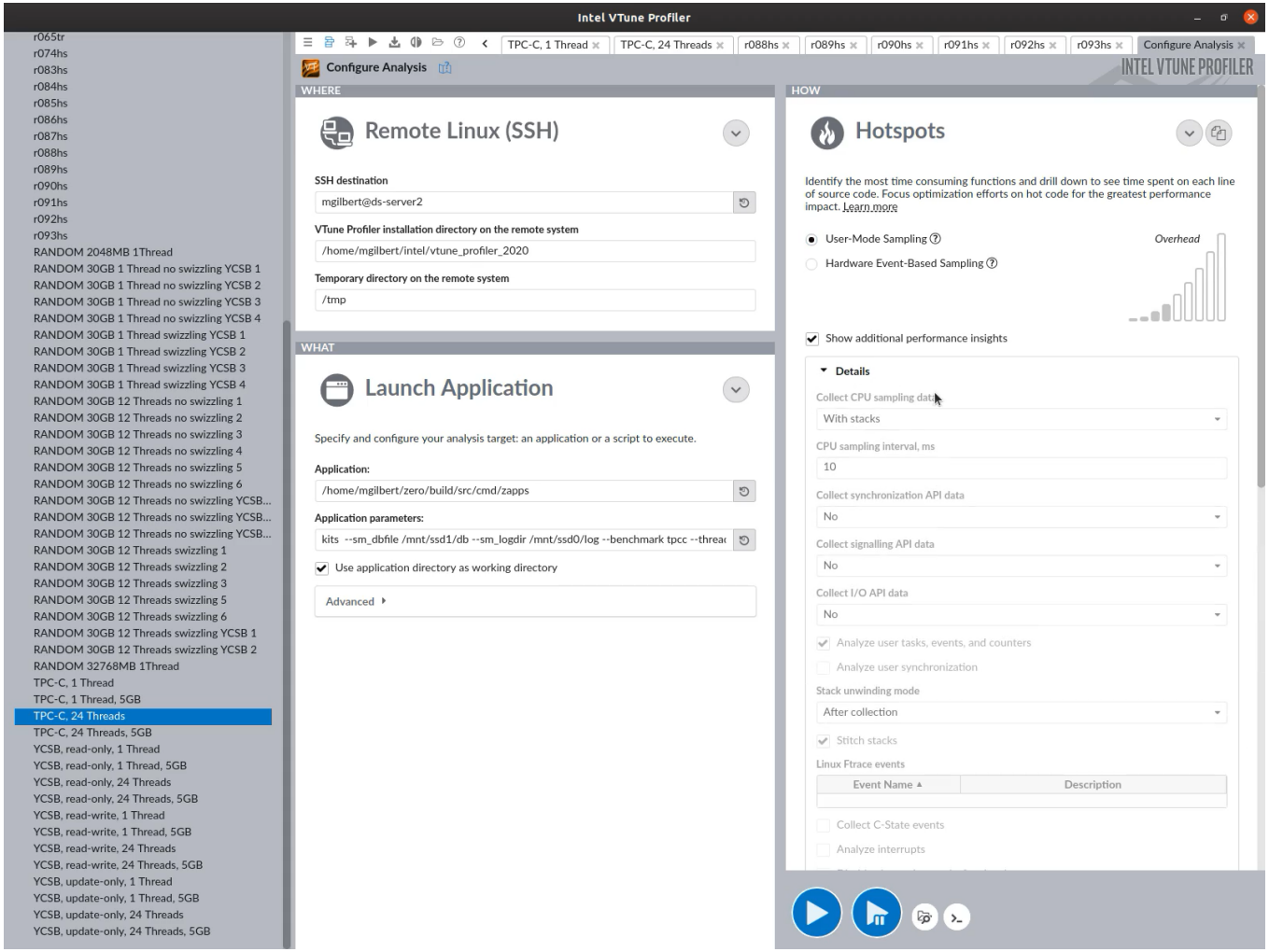
Analysis Configuration: Where?



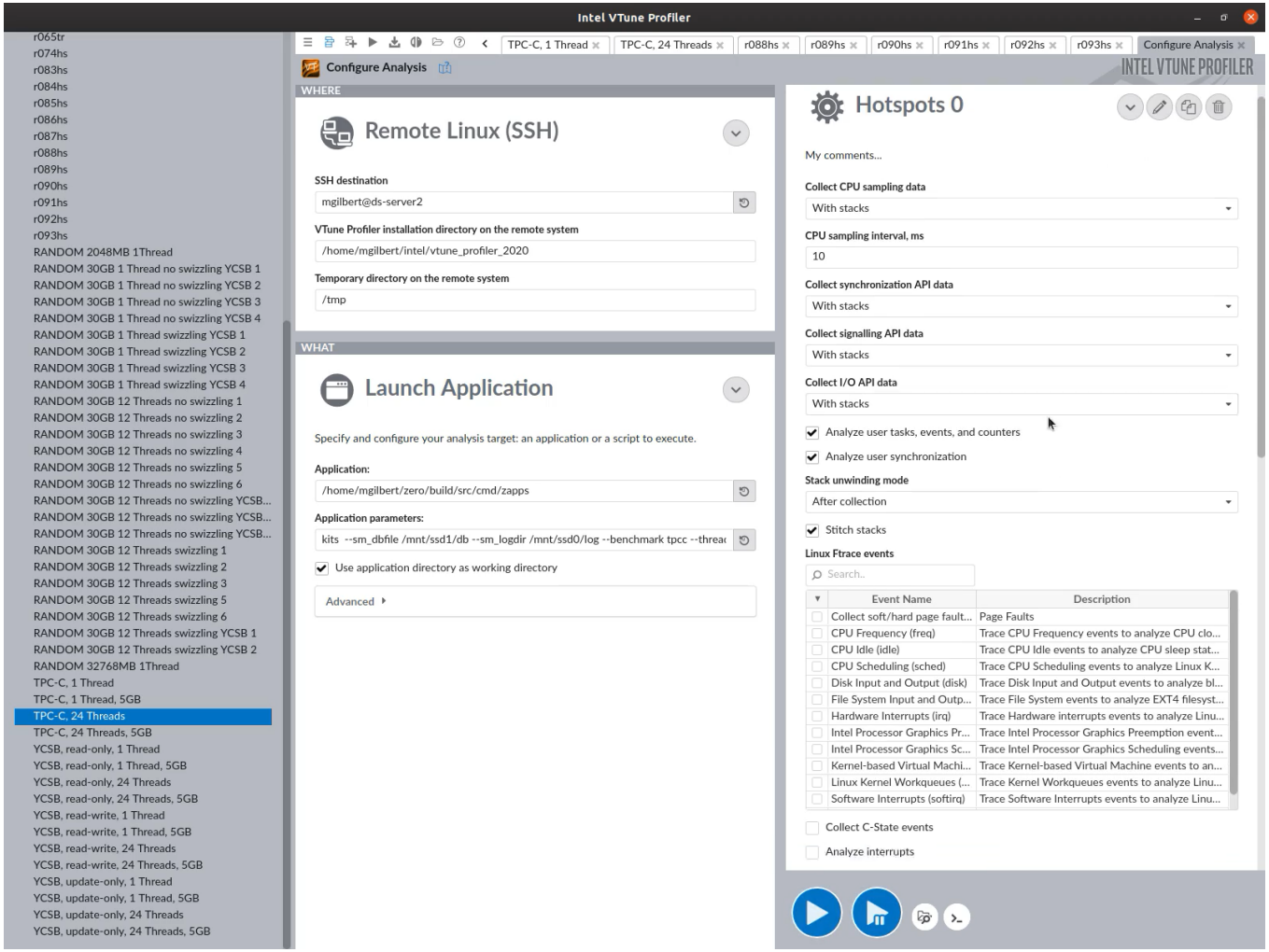
Analysis Configuration: What?



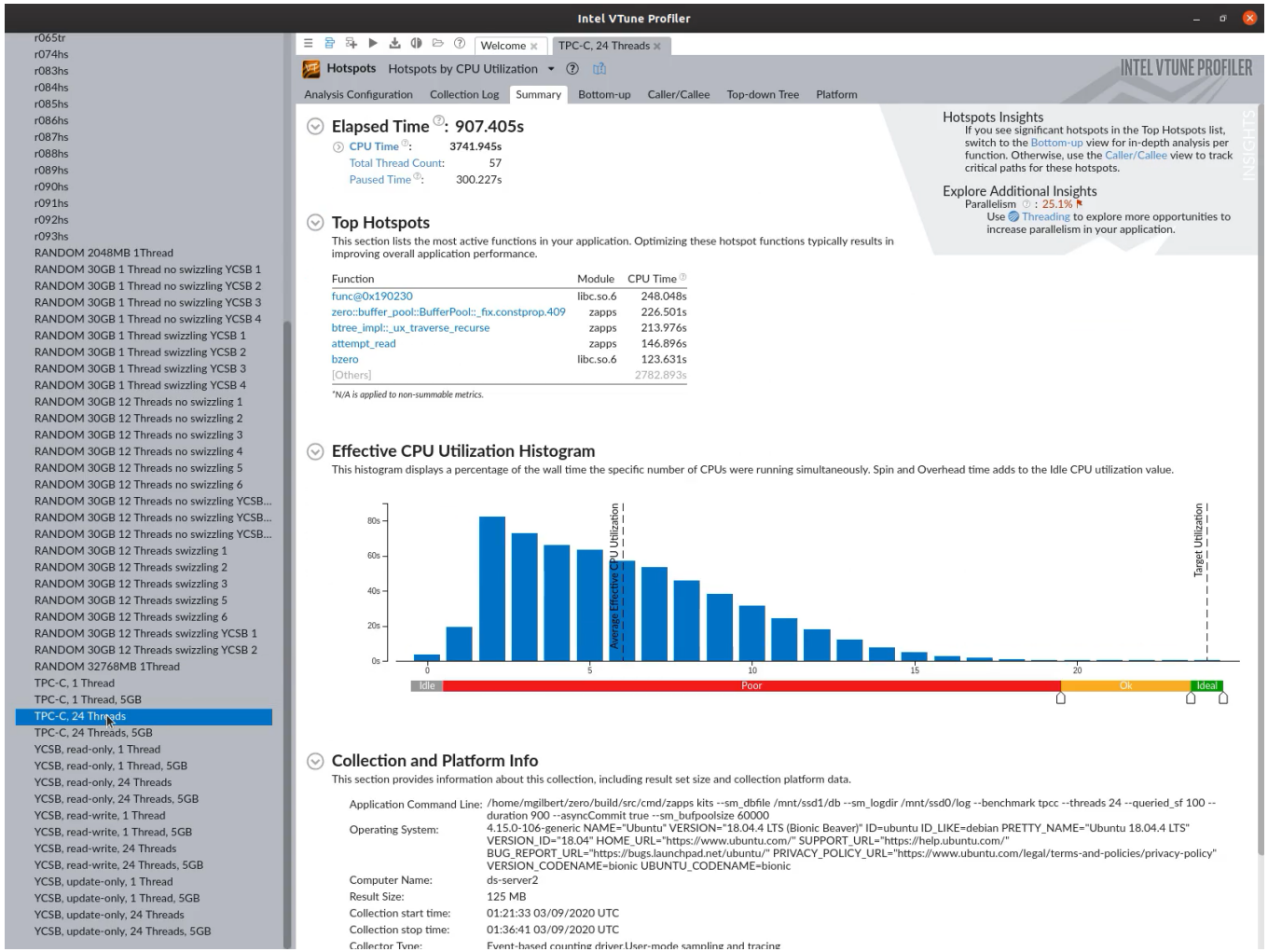
Analysis Configuration: How?



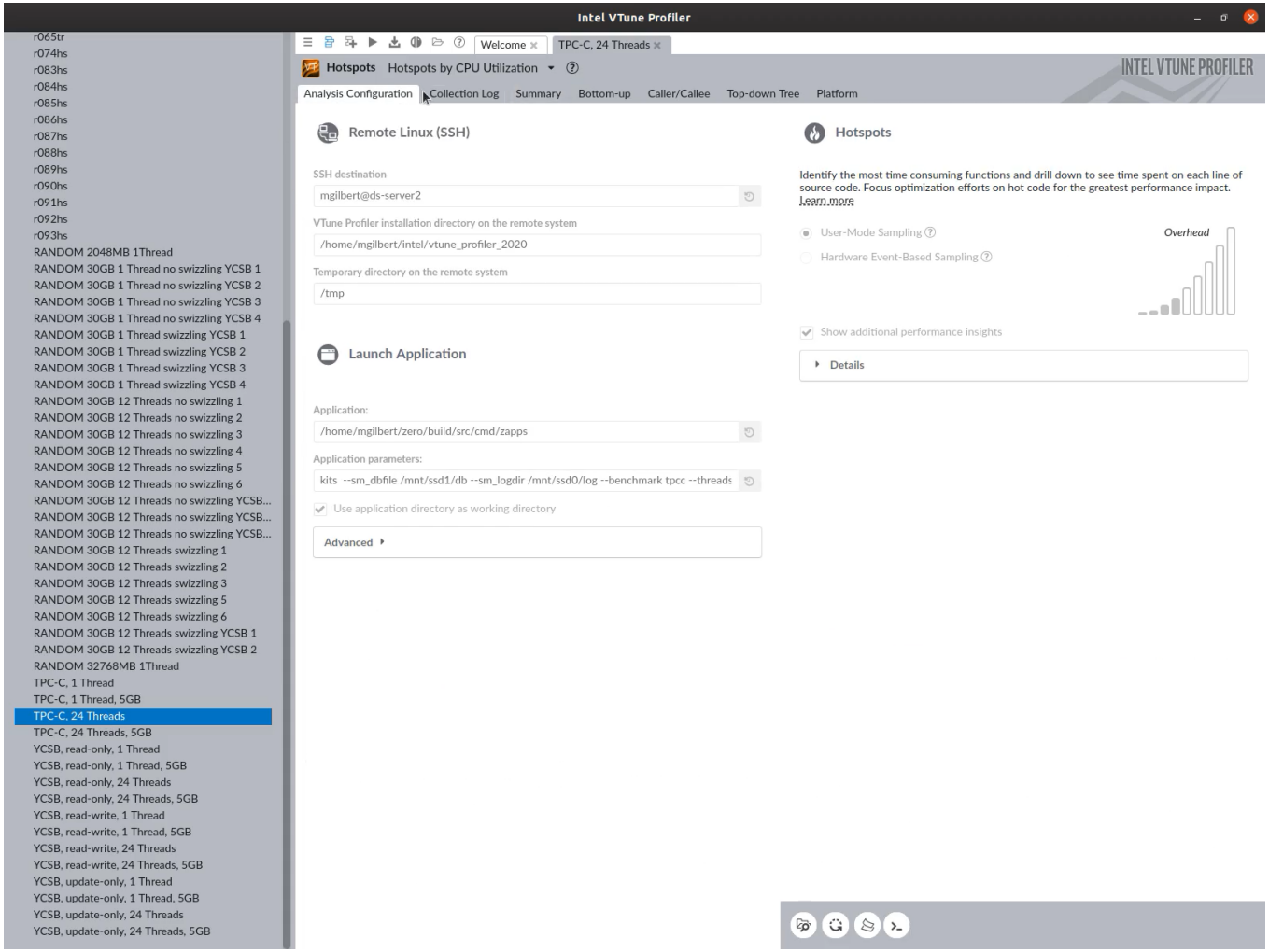
Analysis Configuration: How??



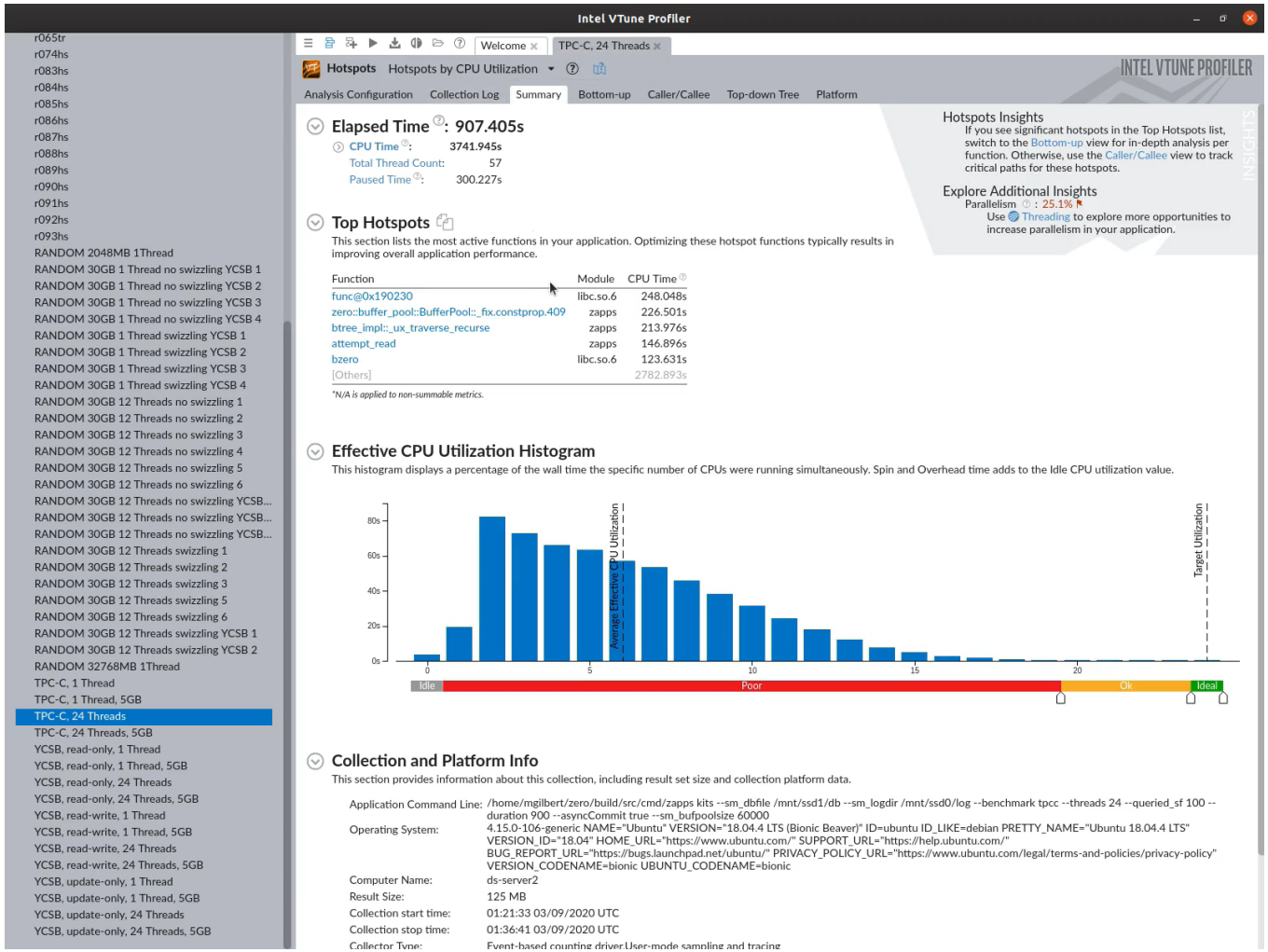
Analysis: Open



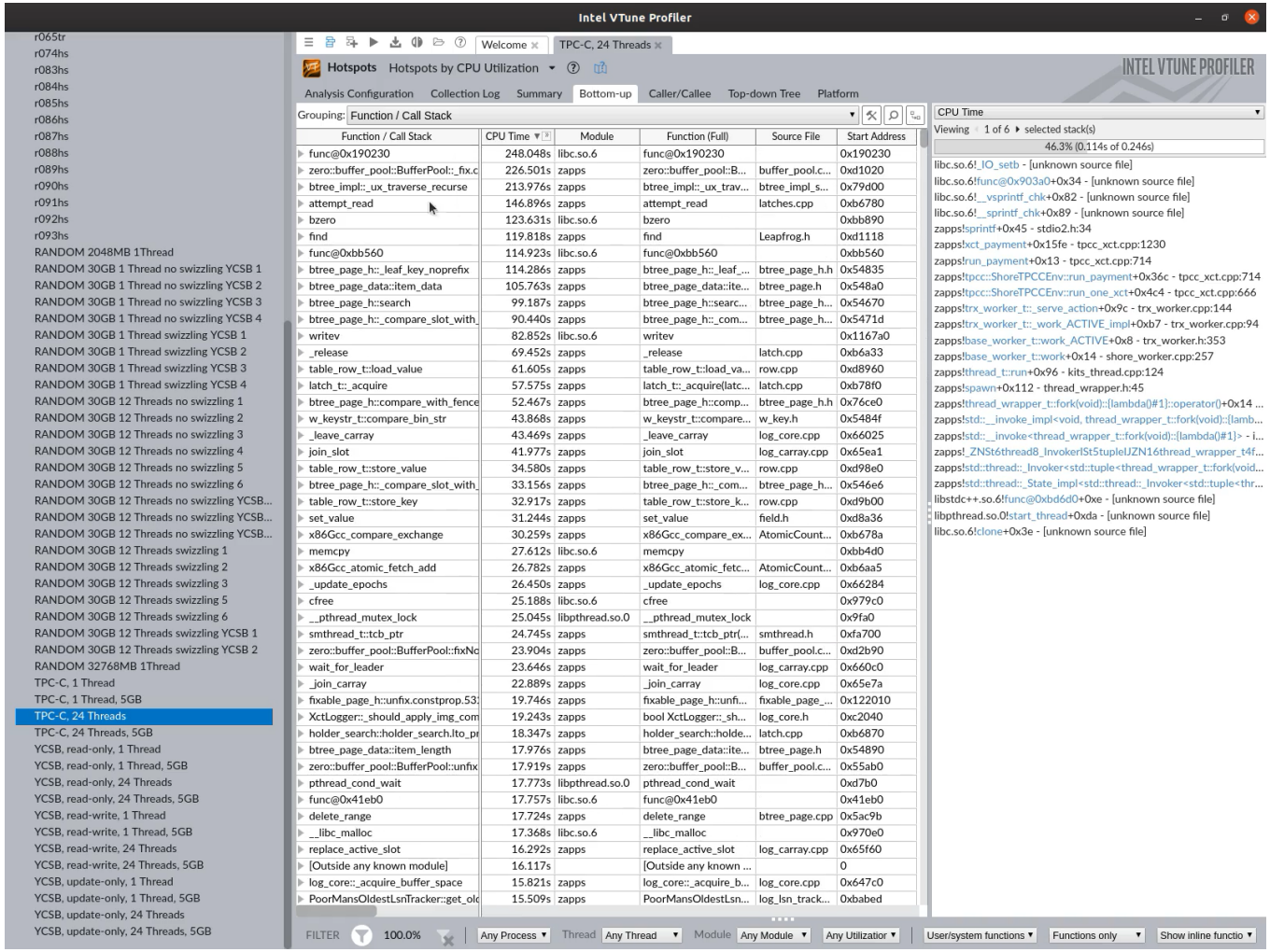
Analysis: Configuration & Collection Log



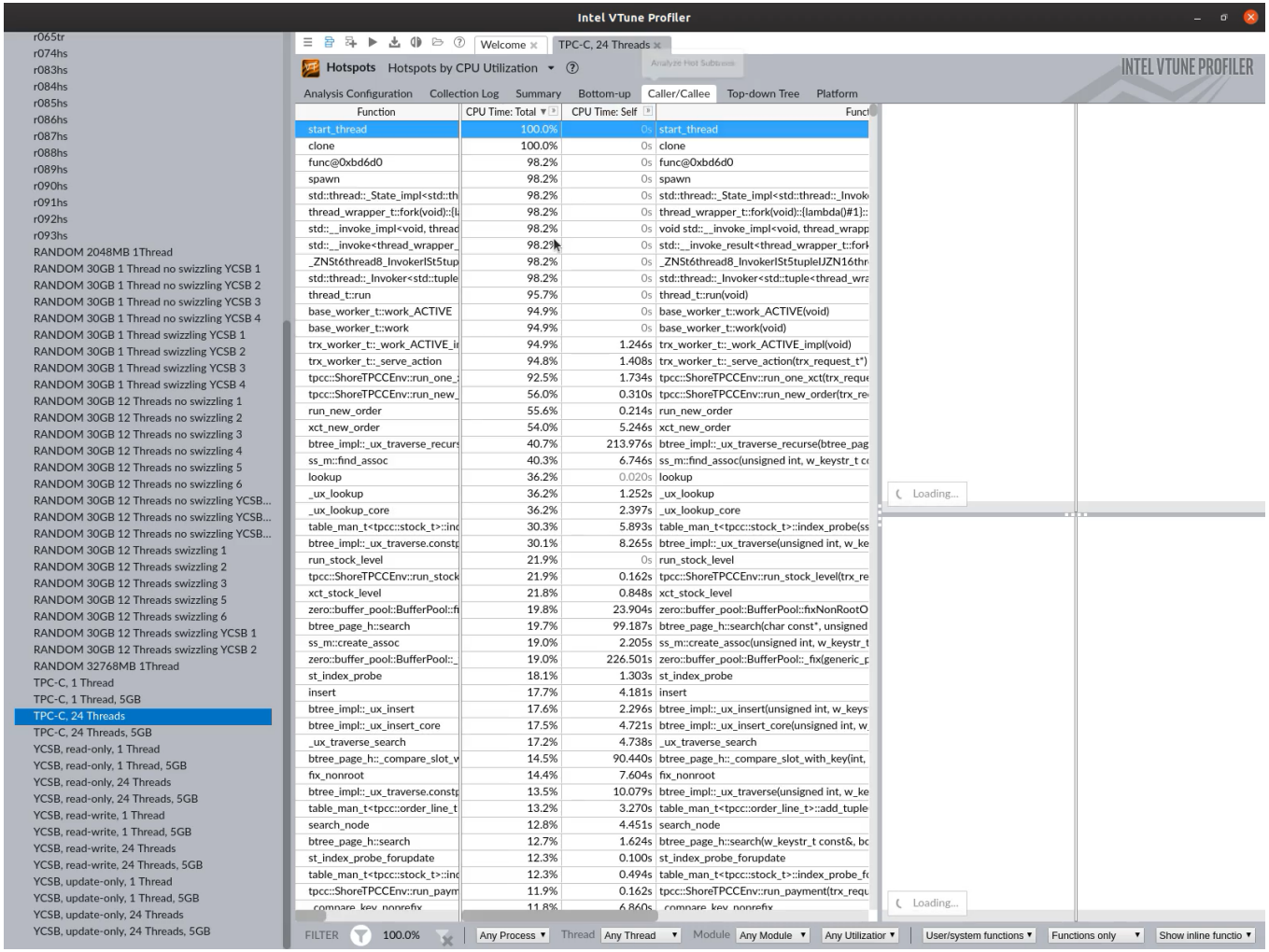
Analysis: Summary



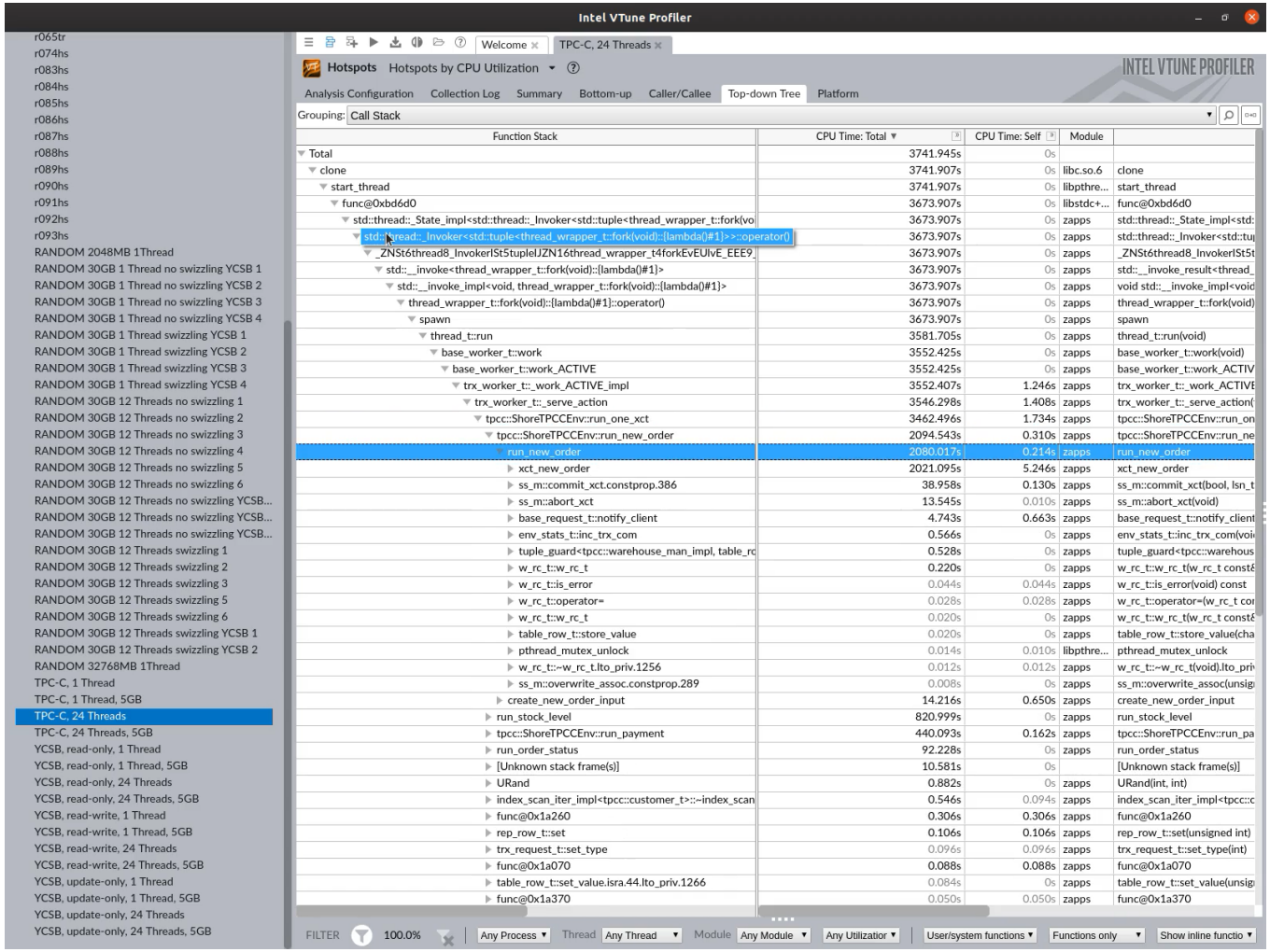
Analysis: Bottom-Up



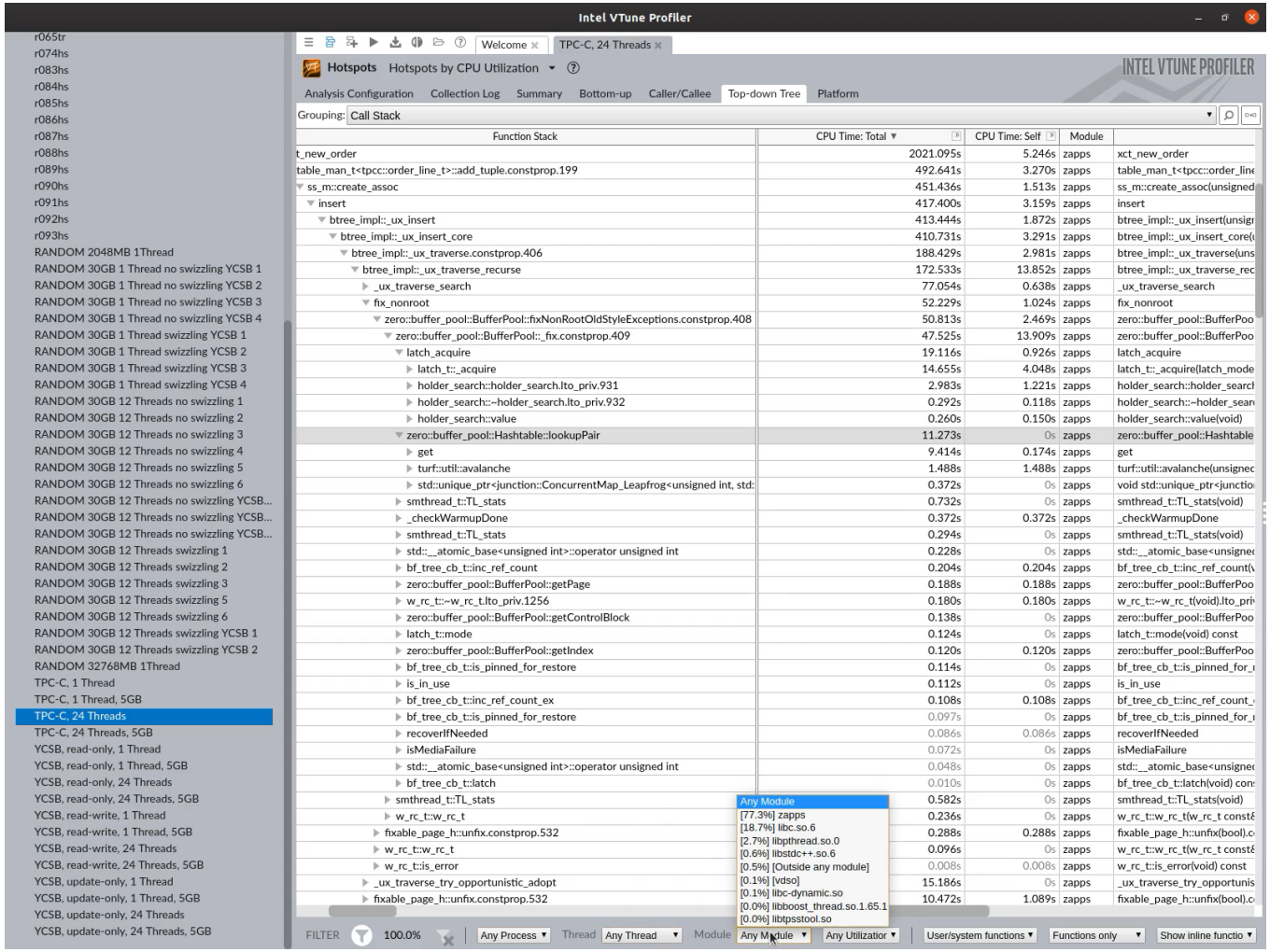
Analysis: Caller/Callee



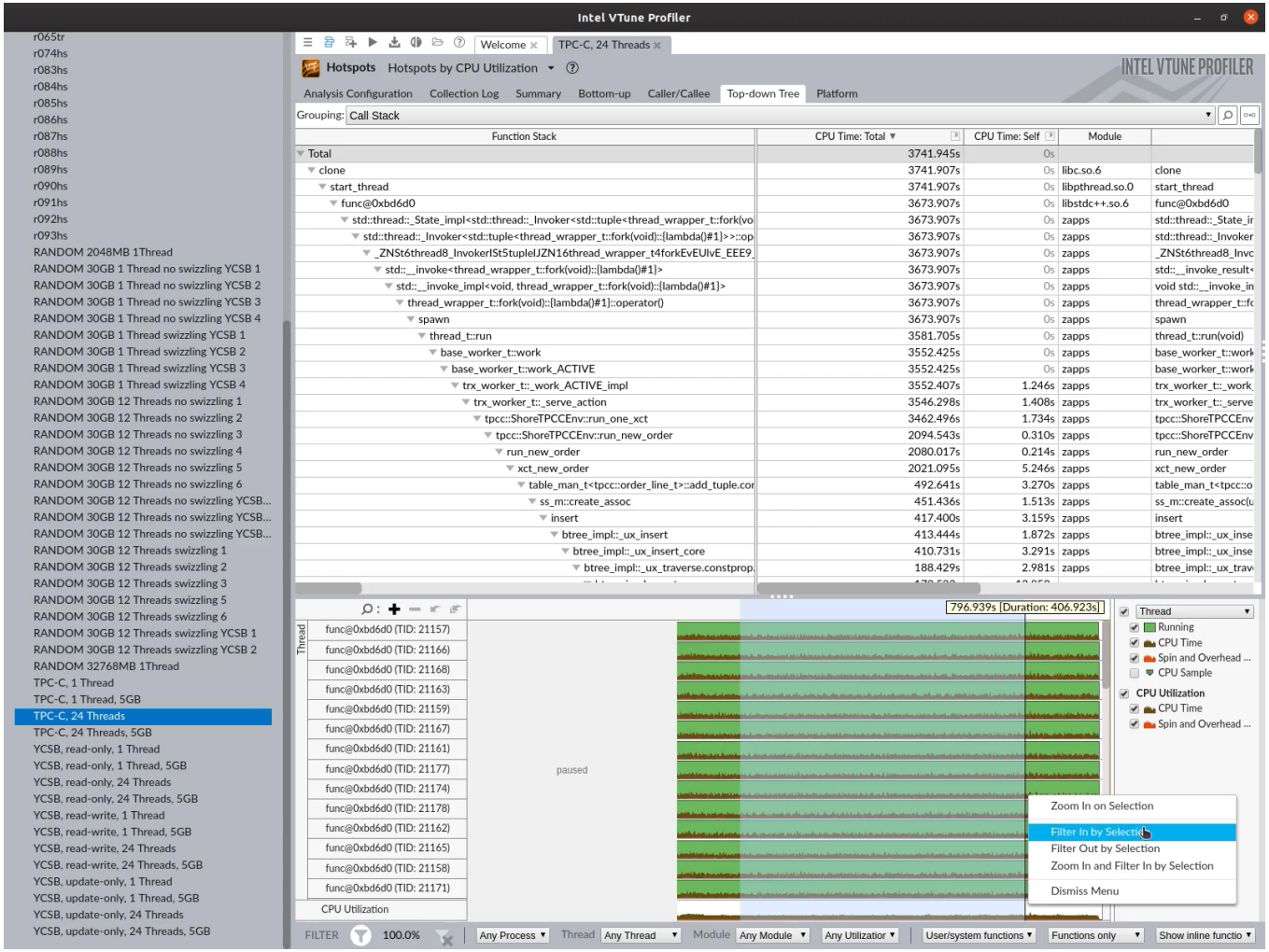
Analysis: Top-Down Tree



Analysis: Filter



Analysis: Filter in Time



Analysis: Platform