Ariadne: A Hotness-Aware and Size-Adaptive Compressed Swap Technique for Fast Application Relaunch and Reduced CPU Usage on Mobile Devices

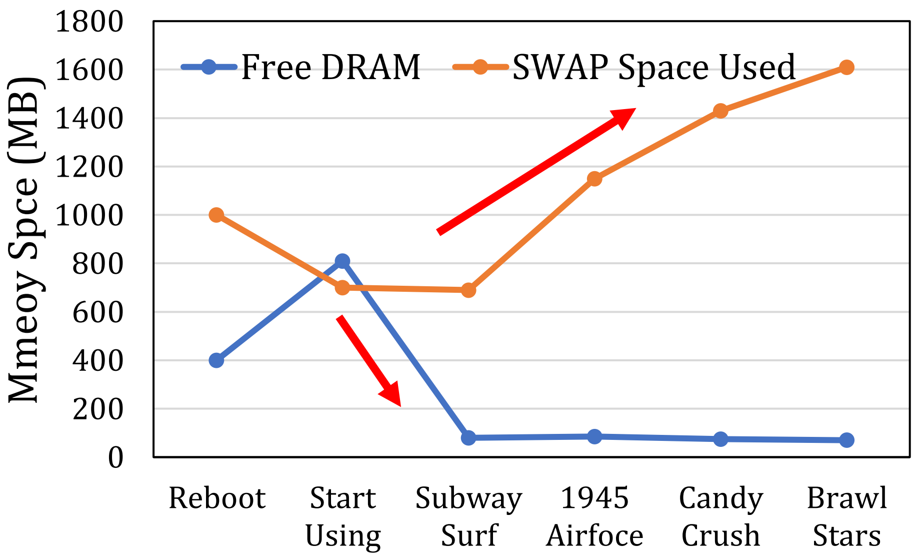
Yu Liang, Aofeng Shen, Chun Jason Xue, Riwei Pan, Haiyu Mao

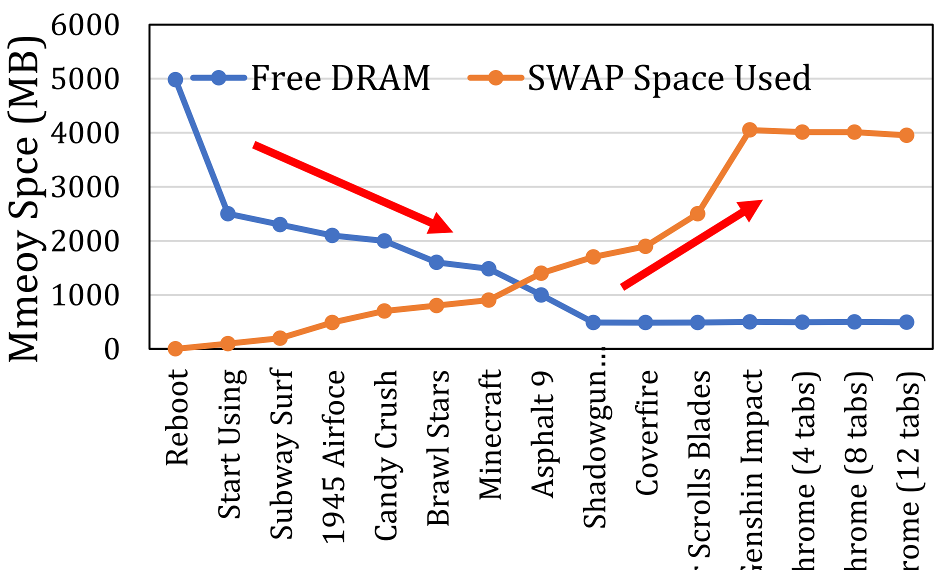
Nika Mansouri Ghiasi, Qingcai Jiang, Rakesh Nadig, Lei Li

Rachata Ausavarungnirun, Mohammad Sadrosadati, Onur Mutlu

Available Memory on Mobile Devices is Scarce

- **Memory demands of individual mobile applications increase**
- **Number of concurrently running applications grows**
- **DRAM capacity cannot be increased accordingly**
  - **Due to constrained power budget of mobile devices**



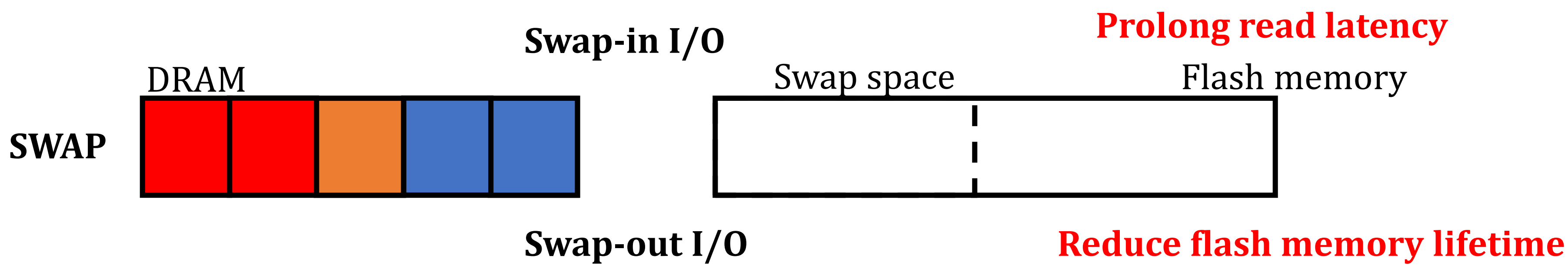


The available memory is usually **insufficient**
to support **multiple applications** running concurrently

[Android Authority, 2025]

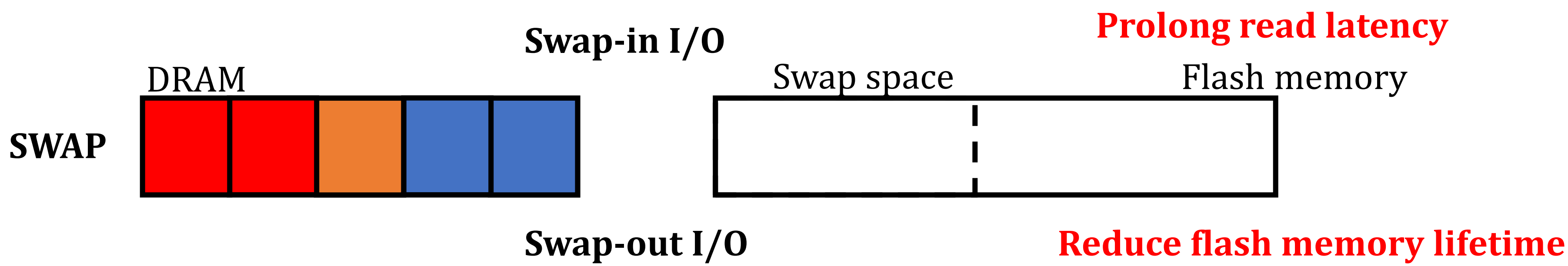
Memory Swap Schemes on Mobile Devices

- **When available memory is insufficient:**
  - **SWAP: Swap data to secondary storage (i.e., flash memory)**

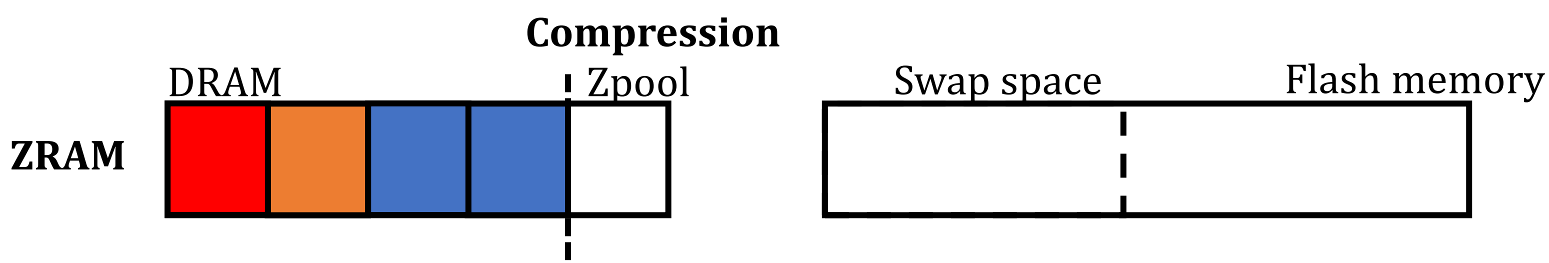


Memory Swap Schemes on Mobile Devices

- When available memory is **insufficient**:
  - SWAP: Swap data to secondary storage (i.e., flash memory)



  - ZRAM: Compress data and store it in a specific area in DRAM (i.e., Zpool)



ZRAM **increases flash memory lifetime** and **reduces read latency** since decompression latency is shorter than the data swap latency

Executive Summary

Problem: **ZRAM prolongs application relaunch latency** and **wastes CPU usage** because it does ***not*** differentiate **data hotness levels** or leverage different **compression chunk sizes** and **data locality**

Goal: To design a **new compressed swap scheme** for mobile devices that **reduces application relaunch latency** and **CPU usage**

Insights:
- Hot data is **similar** between **two consecutive relaunches**
- **Small-size compression** is fast, while **large-size compression** provides a better compression ratio
- There is **locality in data access** during application relaunch

Ariadne:
- Leverages **different compression chunk sizes** based on **data hotness level**
- Performs speculative **pre-decompression** based on **data locality**

Key Results: **Google Pixel 7** with **Android 14.** Compared to ZRAM, **Ariadne**
- Reduces application relaunch latency by 50%
- Decreases the CPU usage of compression and decompression procedures by 15%

Open source: https://github.com/CMU-SAFARI/Ariadne

Outline

Background

Problem and Motivation

New Insights into Mobile Workloads

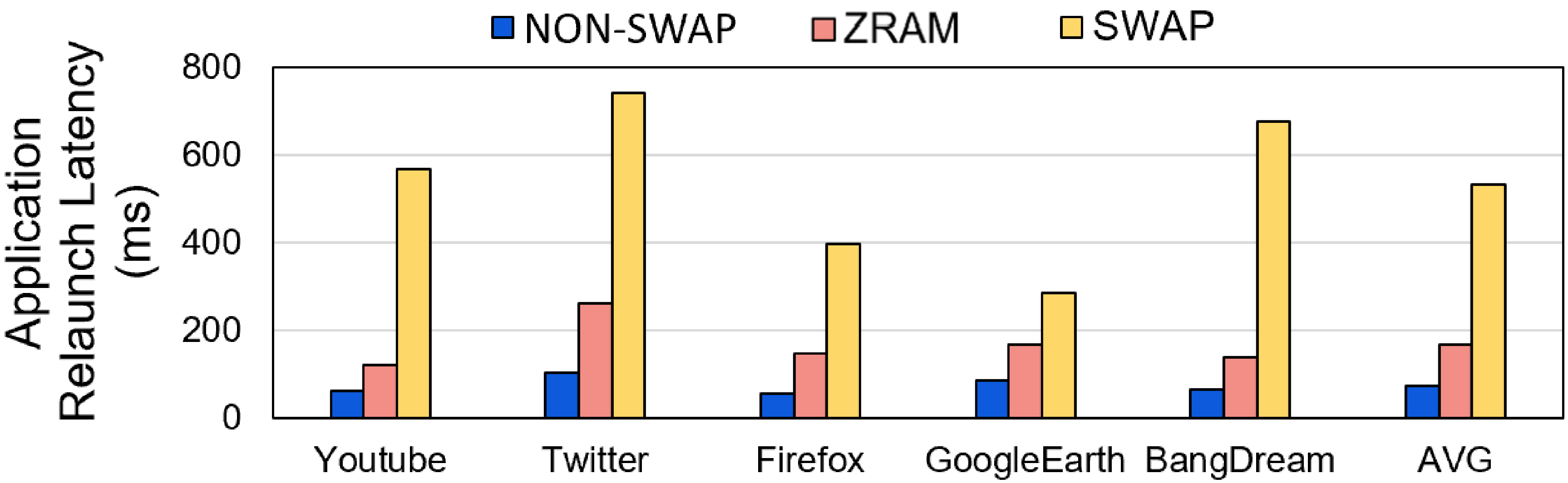
Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap

Evaluation

Conclusion

Problem 1: Delayed Application Relaunch

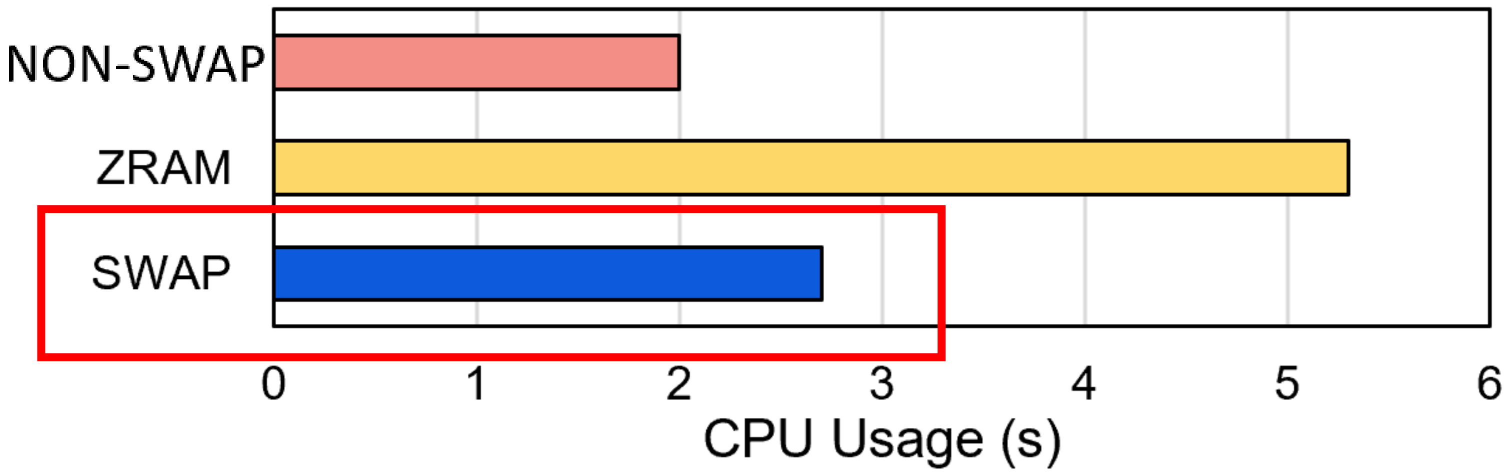
- **Application relaunch latency in three cases**



ZRAM outperforms the traditional SWAP scheme
but **increases relaunch latency** 2.1× compared to NON-SWAP

Problem 2: Higher CPU Usage

- **CPU usage of memory reclaim** **(i.e., freeing memory) in three cases**

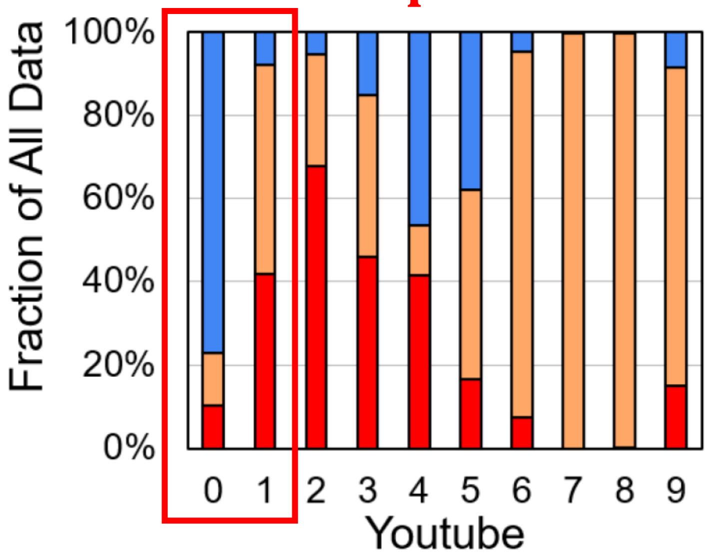


ZRAM has **significant CPU usage**
due to compression and decompression operations

Motivation:
ZRAM Does NOT Consider Data Hotness

- **We examine the fraction of hot, warm, and cold data in each part of compressed data**

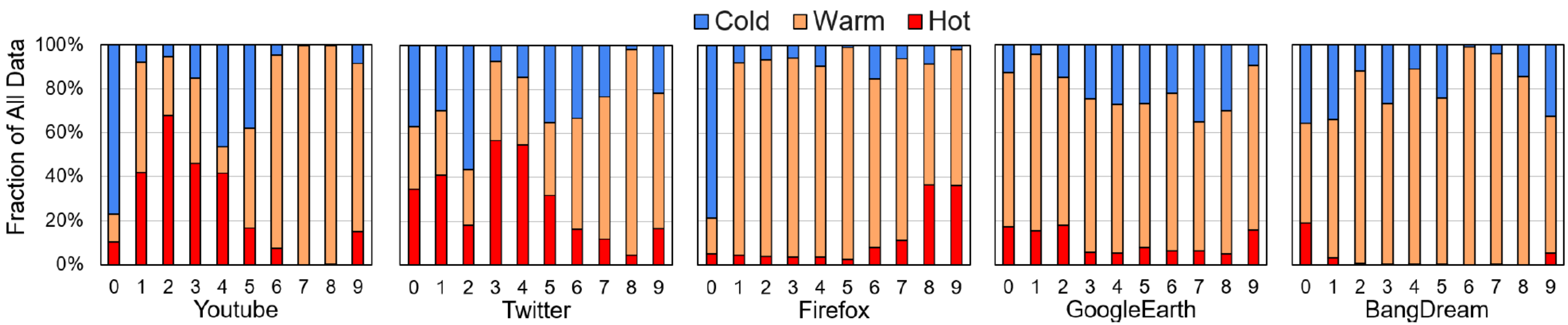
Optimal is all blue for parts 0 and 1



- Sort all compressed data in the order of compression and then divide it into ten equal parts (X-axis)
- The data in part 0 is the first to be compressed, that in part 9 is the last
- **To minimize compression operations, cold data should be compressed earlier (e.g., in parts 0 and 1) and hot data later (e.g., in parts 8 and 9)**

Motivation: ZRAM Does NOT Consider Data Hotness

- **We examine the fraction of hot, warm, and cold data in each part of compressed data**



ZRAM does ***not*** differentiate between **data hotness levels**, causing **frequent compression and decompression**

Our Goal

To design a **new compressed swap scheme**
for mobile devices that reduces
application relaunch latency and **CPU usage**

Doing so requires reducing
the frequency and latency
of compression and decompression

Outline

Background

Problem and Motivation

New Insights into Mobile Workloads

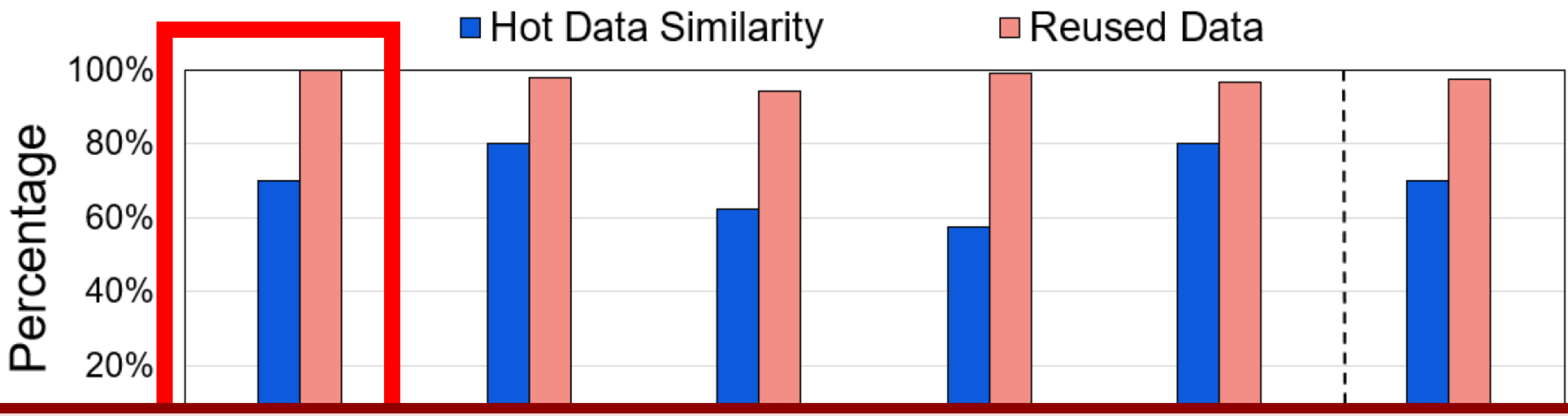
Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap

Evaluation

Conclusion

Insight 1: Data Reuse Across Relaunches

- We examine data reuse between two consecutive relaunches of each application

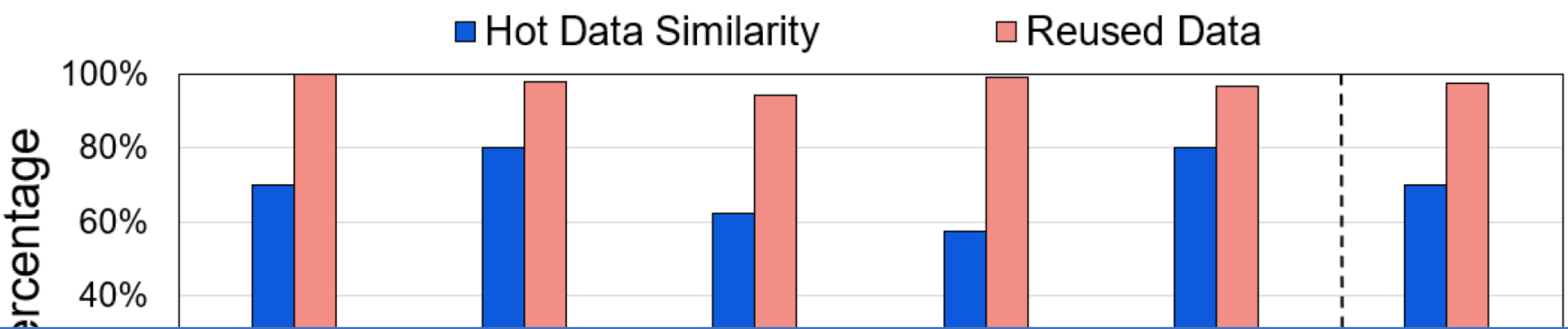


70% data used in first relaunch will be reused in next relaunch

Relaunch the same application could trigger the same activities

Insight 1: Data Reuse Across Relaunches

- We examine **data reuse** between two consecutive relaunches of each application

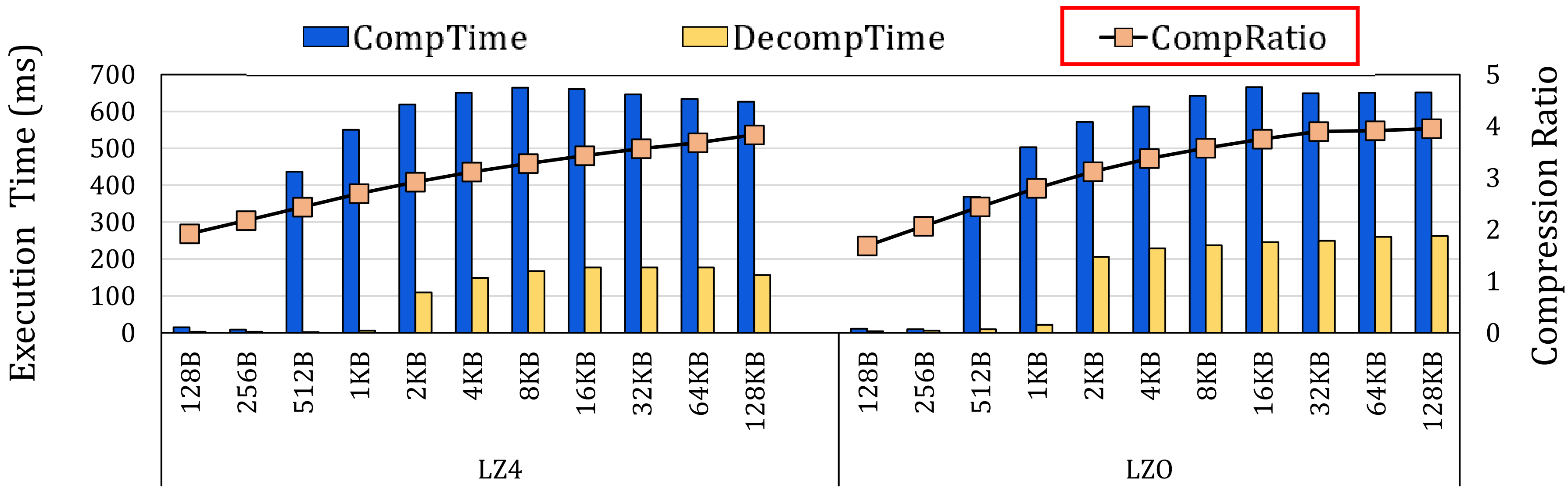


Insight 1: Hot data that is used during application relaunch is usually **similar** between two consecutive relaunches

Key idea 1: Identify hot data based *only* on the most recent relaunch to reduce the hotness identification overhead

Insight 2: Compression Chunk Size

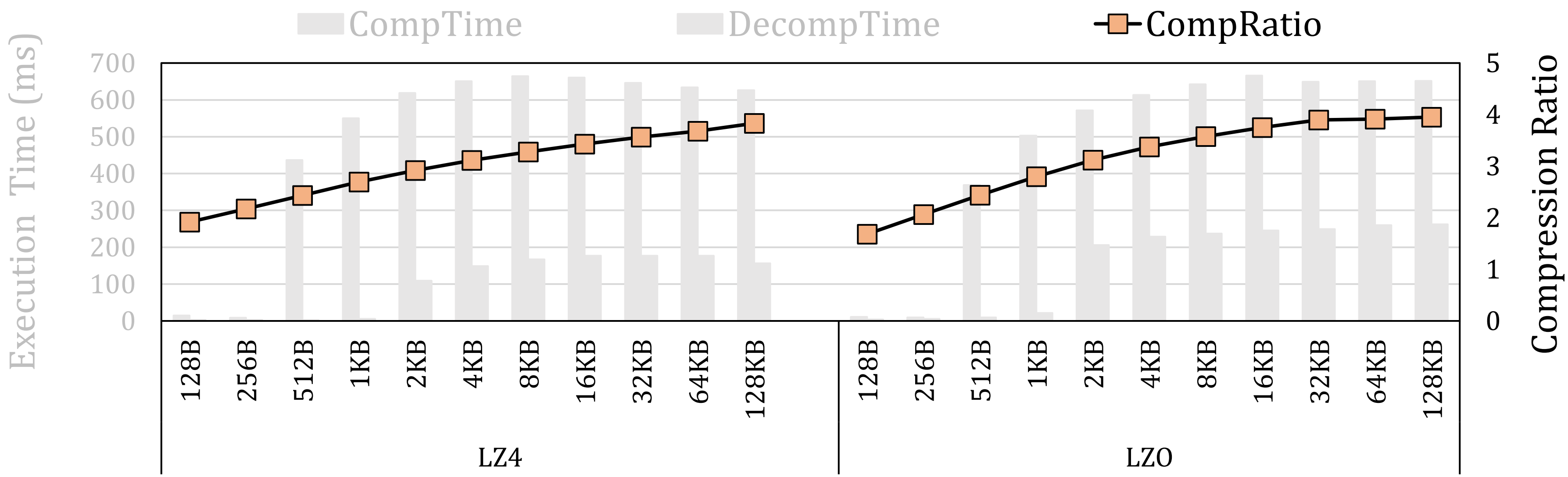
- **We examine compression performance with different compression chunk sizes**



- X-axis represents the compression chunk size

Insight 2: Compression Chunk Size

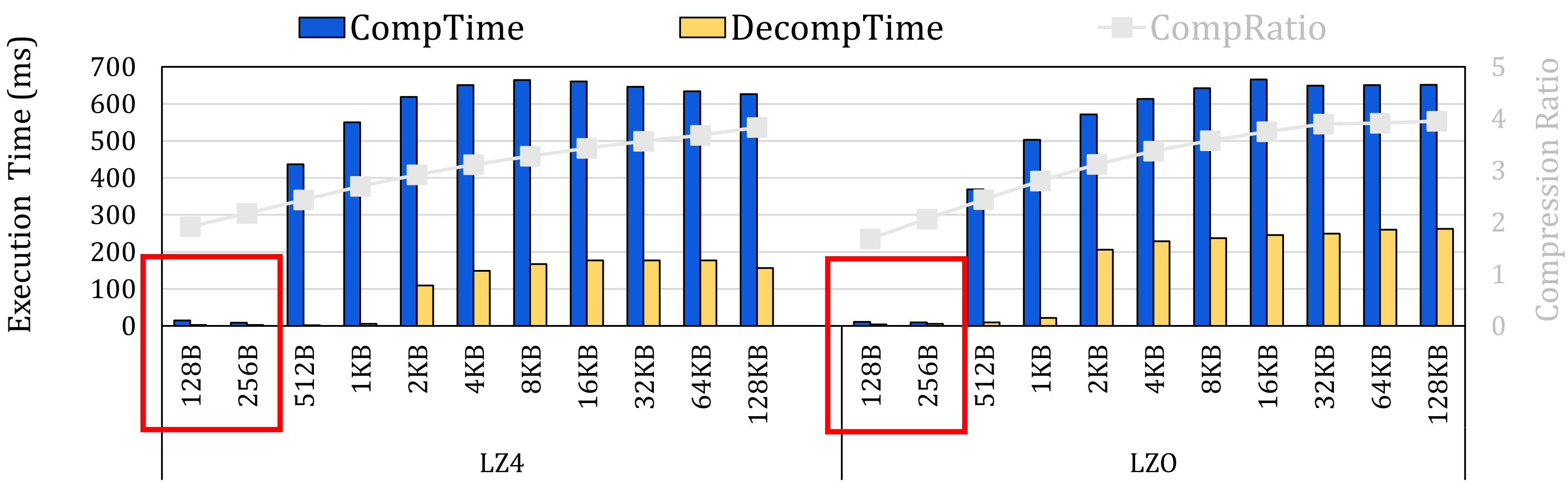
- **We examine compression performance with different compression chunk sizes**



Large-size compression provides a higher compression ratio

Insight 2: Compression Chunk Size

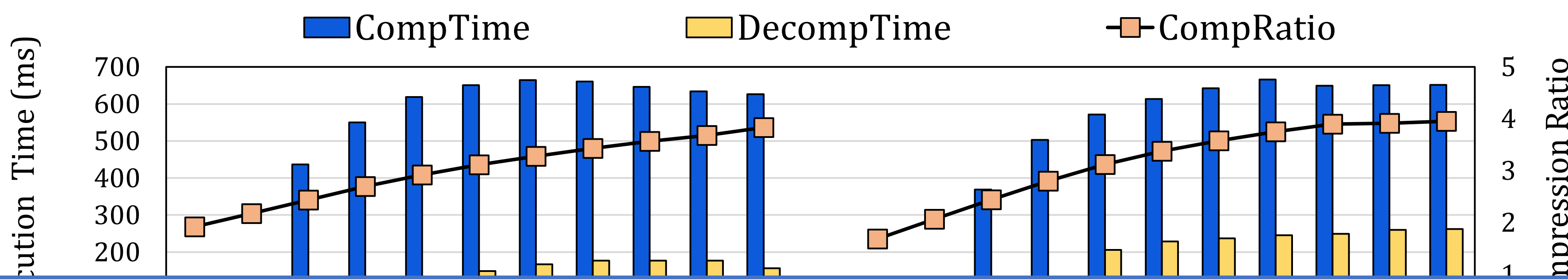
- **We examine compression performance with different compression chunk sizes**



Small-size compression provides short compression and decompression latencies

Insight 2: Compression Chunk Size

- We examine compression performance with different **compression chunk sizes**



Insight 2: Small-size compression is much faster than large-size but has lower compression ratio

Key idea 2: Use small-size on hot data for low latency and large-size on cold data for high compression ratio

Insight 3: Data Locality in Zpool

- **The probability of accessing multiple consecutive pages in Zpool for each evaluated application**

Probability of accessing two consecutive pages together

	Youtube	Twitter	Firefox	GoogleEarth	BangDream
2	0.86	0.81	0.69	0.77	0.61
4	0.72	0.61	0.43	0.54	0.33

Insight 3: There is data use locality in Zpool when decompress data during application relaunch

Key idea 3: Pre-decompress the next compressed data to hide decompression latency during application relaunch

Outline

Background

Problem and Motivation

New Insights into Mobile Workloads

Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap

Evaluation

Conclusion

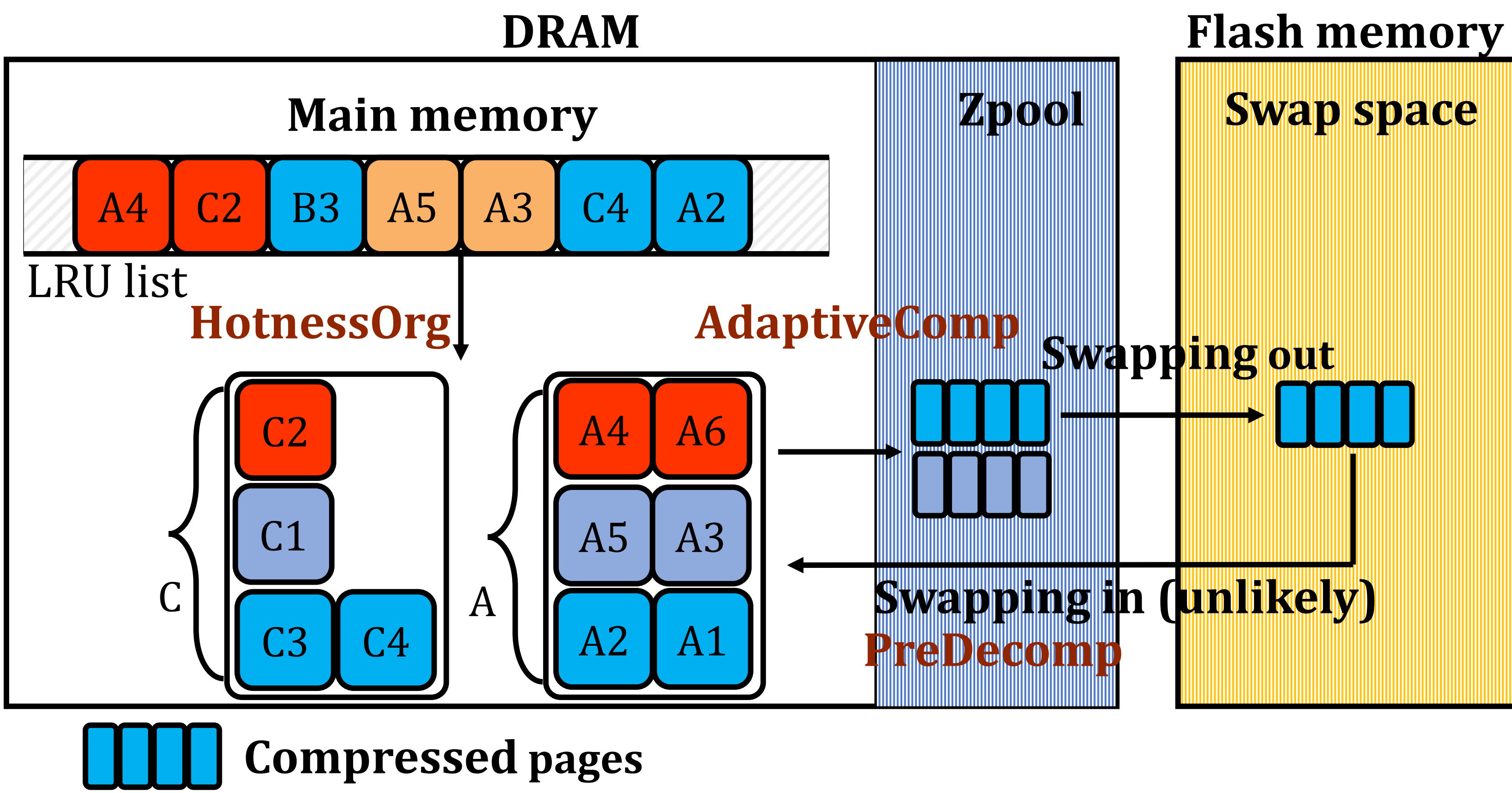
Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap Scheme

- **Key Idea:** Reduce **the frequency and latency** of compression and decompression by
  - Leveraging **different compression chunk sizes** based on **data hotness level**
  - Performing speculative **pre-decompression** based on **data locality**

- **Ariadne:** Hotness-Aware and Size-Adaptive Compressed Swap Scheme
  - **HotnessOrg:** Quickly identifies the **hotness of data** based on **last relaunch**
  - **AdaptiveComp:** Uses different **compression chunk sizes** based on data hotness
    - Small compression for hot and warm data, leading to **fast decompression**
    - Large compression for cold data, leading to **high compression ratio**
  - **PreDecomp:** **pre-decompresses** the next compressed data

Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap Scheme

- **Overview**



Outline

Background

Problem and Motivation

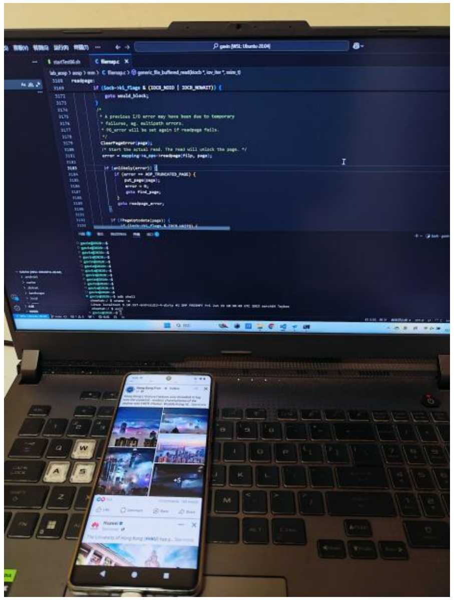
New Insights into Mobile Workloads

Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap

Evaluation

Conclusion

Evaluation Methodology

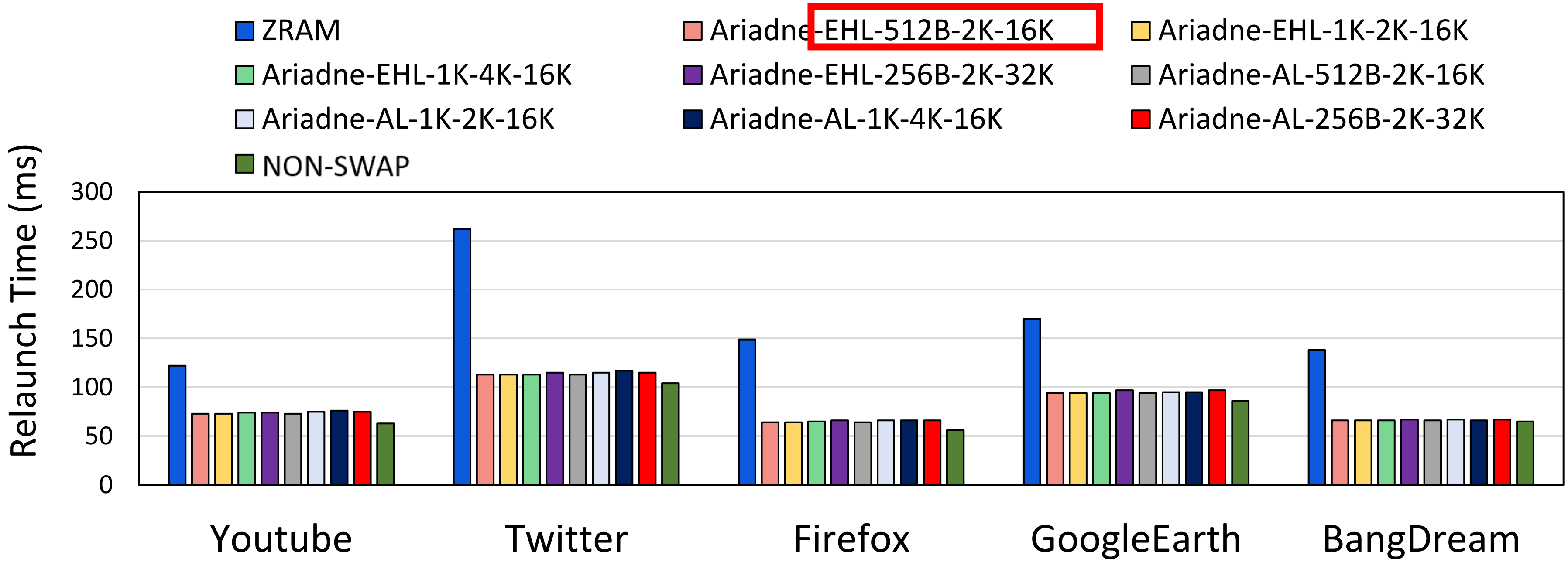


- **Platform**: Google Pixel 7
  - Eight core CPU
  - 12 GB DRAM LPDDR5
  - 128 GB SSD
  - Android 14 with Linux 5.10.157

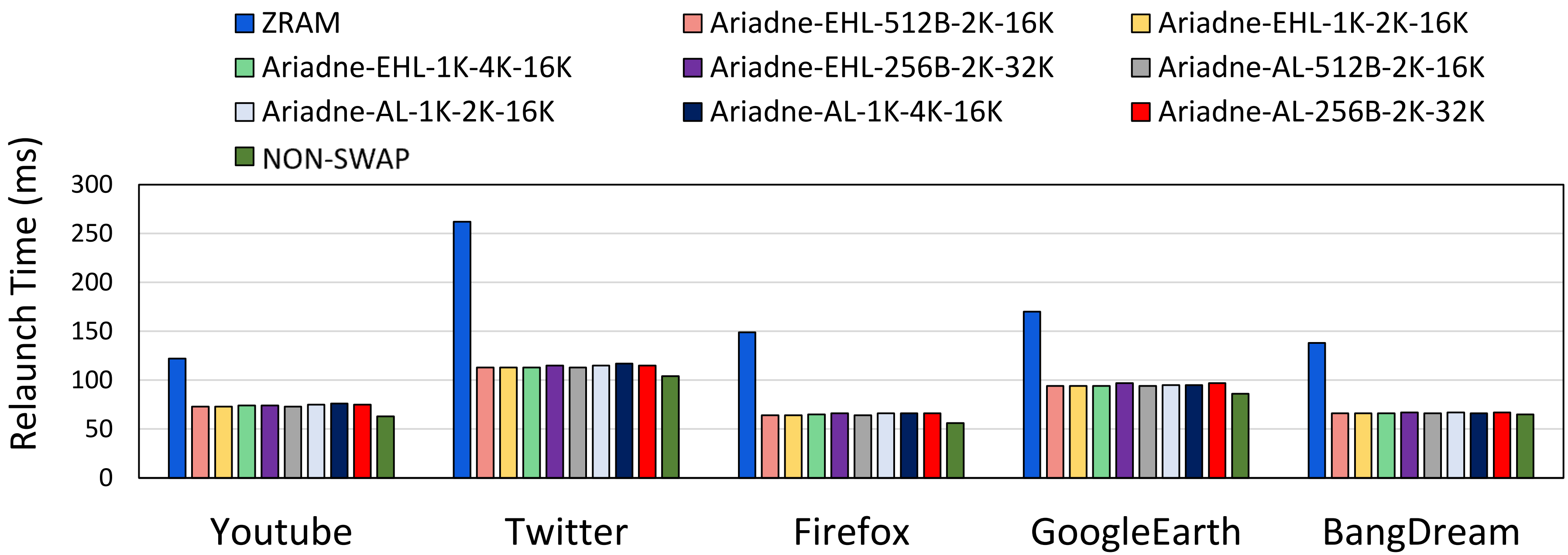
- **Evaluated Schemes**:
  - Baseline: state-of-the-art ZRAM
    - LRU list data organization, page-size compression, no pre-decompression
  - Ariadne: with different configurations
    - HotnessOrg, AdaptiveComp, PreDecomp

Open Source: https://github.com/CMU-SAFARI/Ariadne

Effect on Application Relaunch Latency

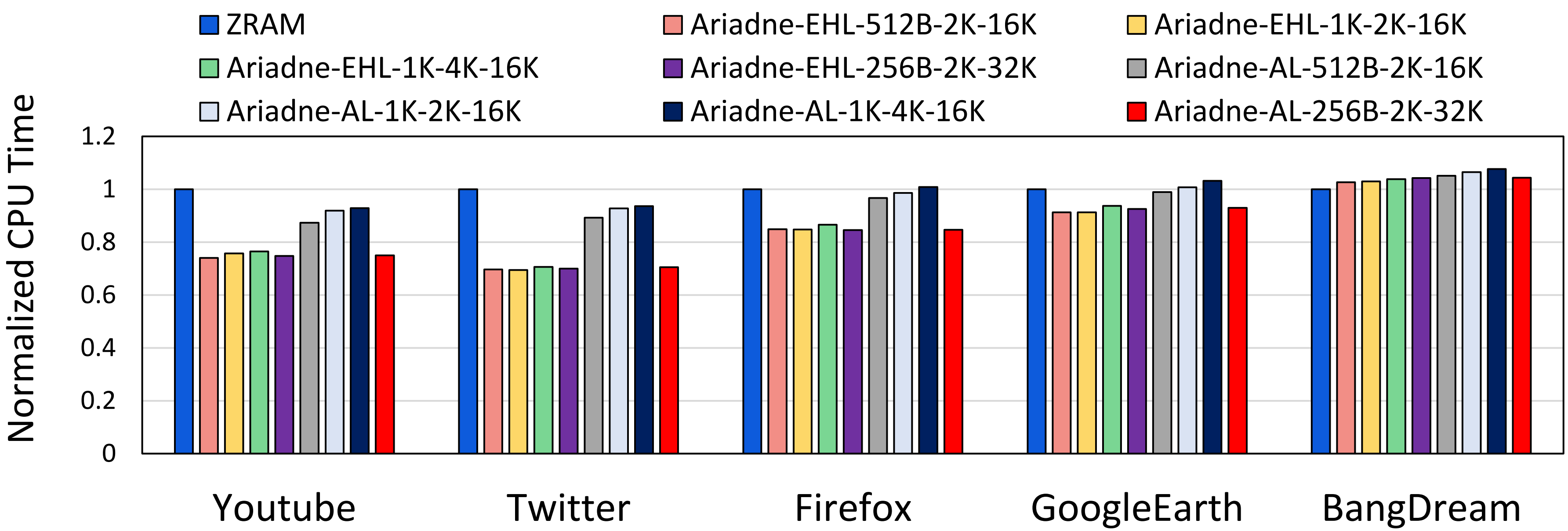


Effect on Application Relaunch Latency



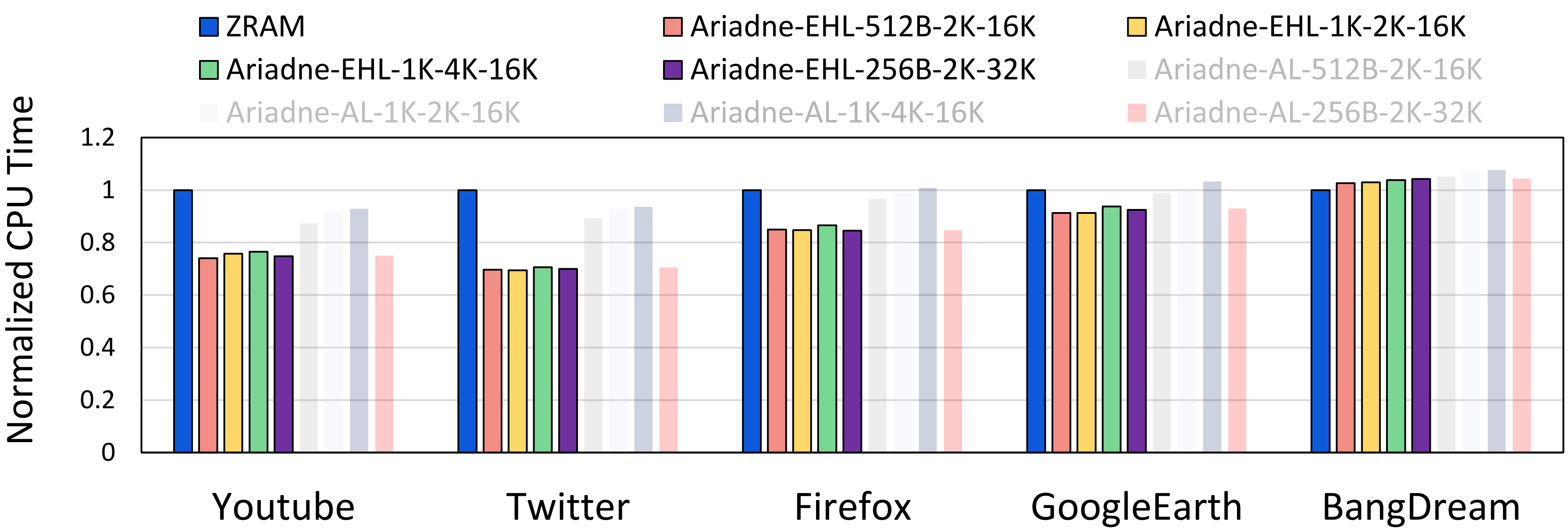
Ariadne reduces application relaunch latency by 50%
(over state-of-the-art ZRAM)

Effect on CPU Usage



Effect on CPU Usage

Hot data in the main memory



Ariadne reduces CPU usage of compression and decompression by 15% (over state-of-the-art ZRAM)

SAFARI

There is More in Our Paper

- Design details of Ariadne
- Ariadne implementation
- Evaluation methodology details
- More evaluation results
  - Compression and decompression latency
  - Compression ratio
  - Prediction accuracy and coverage
  - Sensitivity study with different parameters
- Overhead analysis
  - Small overhead in terms of computation and memory space

More in the Extended Version

Ariadne: A Hotness-Aware and Size-Adaptive Compressed Swap Technique for Fast Application Relaunch and Reduced CPU Usage on Mobile Devices

Yu Liang[§] Aofeng Shen[§] Chun Jason Xue[‡] Riwei Pan[†] Haiyu Mao[¶] Nika Mansouri Ghiasi[§]
Qingcai Jiang[𝒮] Rakesh Nadig[§] Lei Li[†] Rachata Ausavarungnirun[*] Mohammad Sadrosadati[§] Onur Mutlu[§]

[§]*ETH Zürich* [‡]*MBZUAI* [†]*City University of Hong Kong* [¶]*King's College London*
[𝒮]*University of Science and Technology of China* [*]*MangoBoost*

As the memory demands of individual mobile applications continue to grow and the number of concurrently running applications increases, available memory on mobile devices is becoming increasingly scarce. When memory pressure is high, current mobile systems use a RAM-based compressed swap scheme (called ZRAM) to compress unused execution-related data (called anonymous data in Linux) in main memory. This approach avoids swapping data to secondary storage (NAND flash memory) or terminating applications, thereby achieving shorter application relaunch latency.

their diverse needs [1–3]. To fulfill user expectations of seamless and rapid application relaunch, mobile systems preserve all execution-related data (called *anonymous data* in Linux [4]), such as stack and heap, in main memory. This practice, known as *keeping applications alive in the background* [1, 5–8], enables faster relaunches. However, it also results in significant main memory capacity requirements for each application.

As the demand for memory capacity in mobile applications grows and the number of applications running simultaneously increases, available memory is becoming an increasingly scarce

https://arxiv.org/pdf/2502.12826

Outline

Background

Problem and Motivation

New Insights into Mobile Workloads

Ariadne: Hotness-Aware and Size-Adaptive Compressed Swap

Evaluation

Conclusion

Conclusion

Ariadne leverages **different compression chunk sizes** based on **data hotness level** and performs speculative **pre-decompression** based on **data locality**

Evaluation results on **Google Pixel 7** with **Android 14** show that Ariadne reduces ***both*** **application relaunch latency** and **CPU usage**, compared to ZRAM

Ariadne: A Hotness-Aware and Size-Adaptive Compressed Swap Technique for Fast Application Relaunch and Reduced CPU Usage on Mobile Devices

Yu Liang

Aofeng Shen Chun Jason Xue Riwei Pan Haiyu Mao

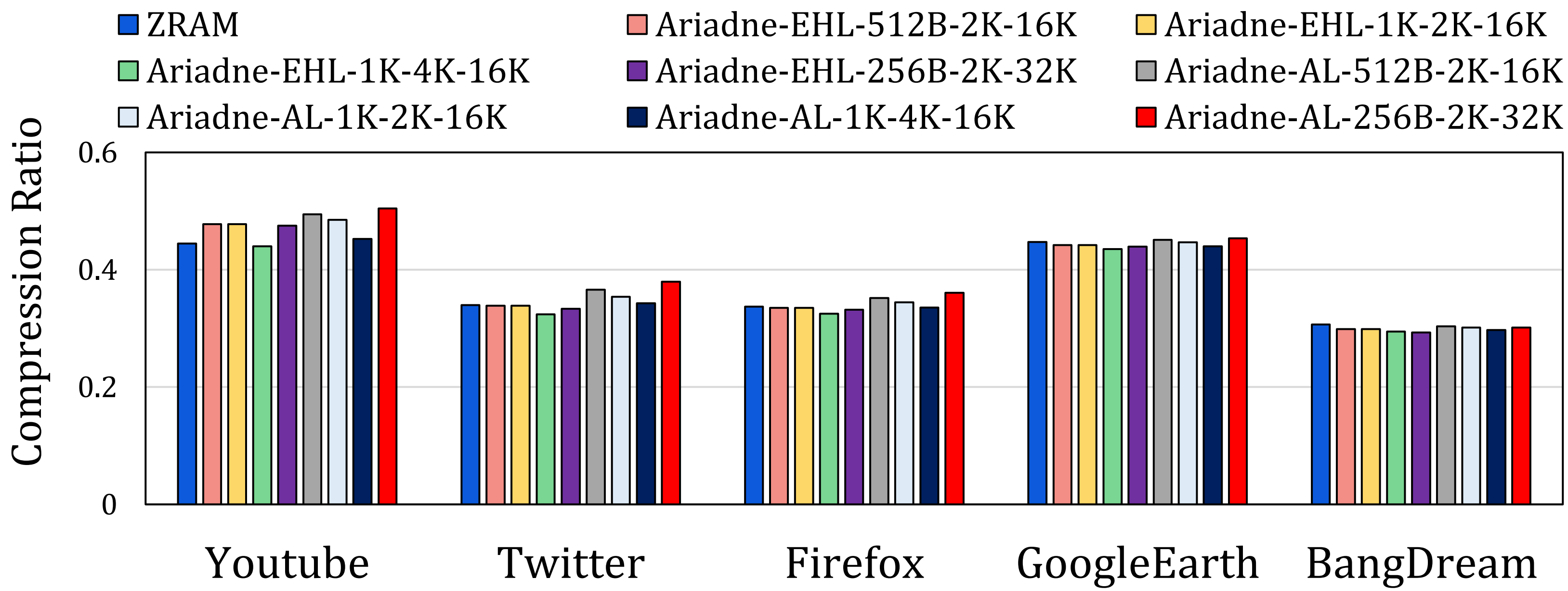
Nika Mansouri Ghiasi Qingcai Jiang Rakesh Nadig Lei Li

Rachata Ausavarungnirun Mohammad Sadrosadati Onur Mutlu

SAFARI ETH zürich MBZUAI CityU UCL MANGOBOOST

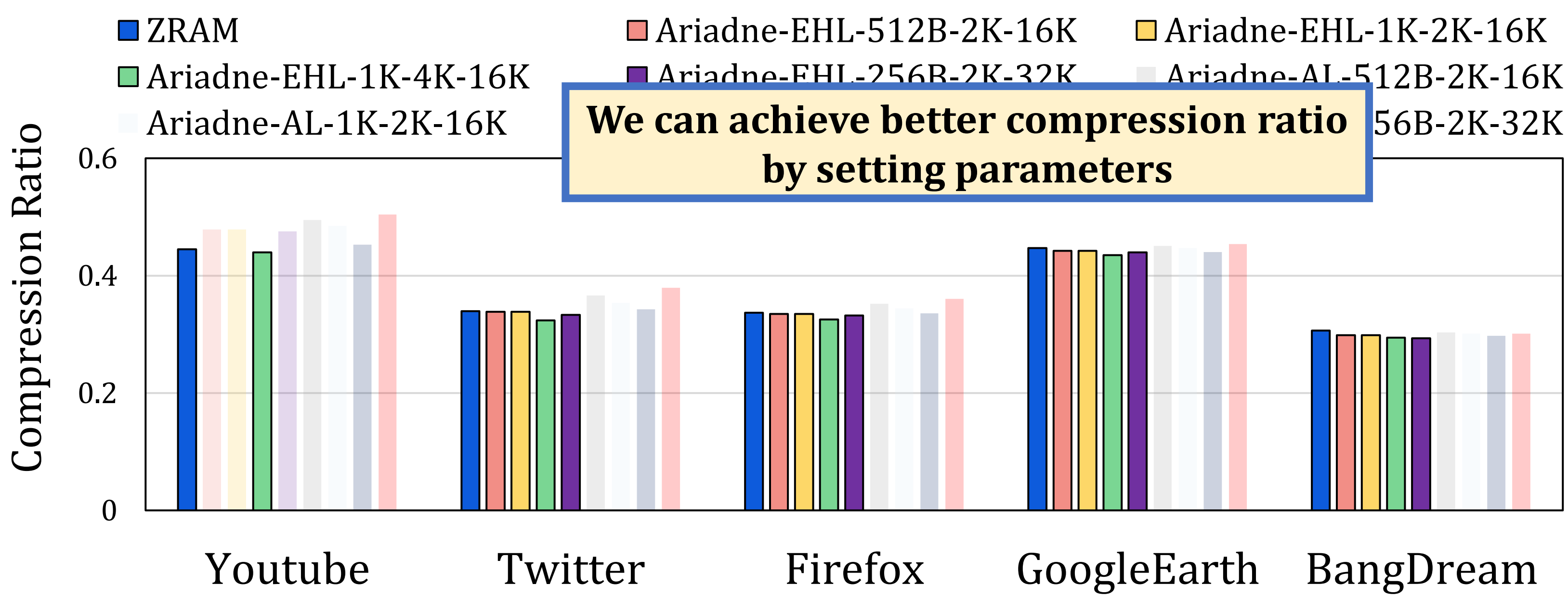
Compression Ratio

Compression ratio under different settings



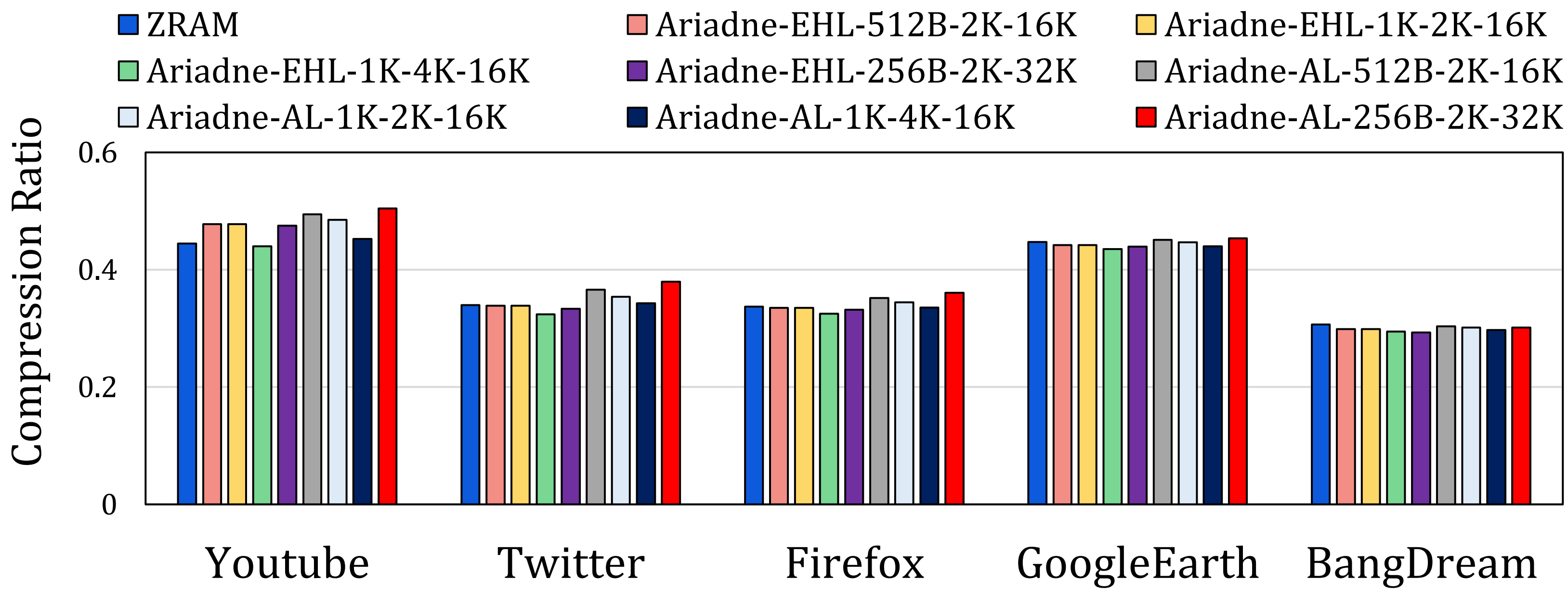
Compression Ratio

Compression ratio under different settings



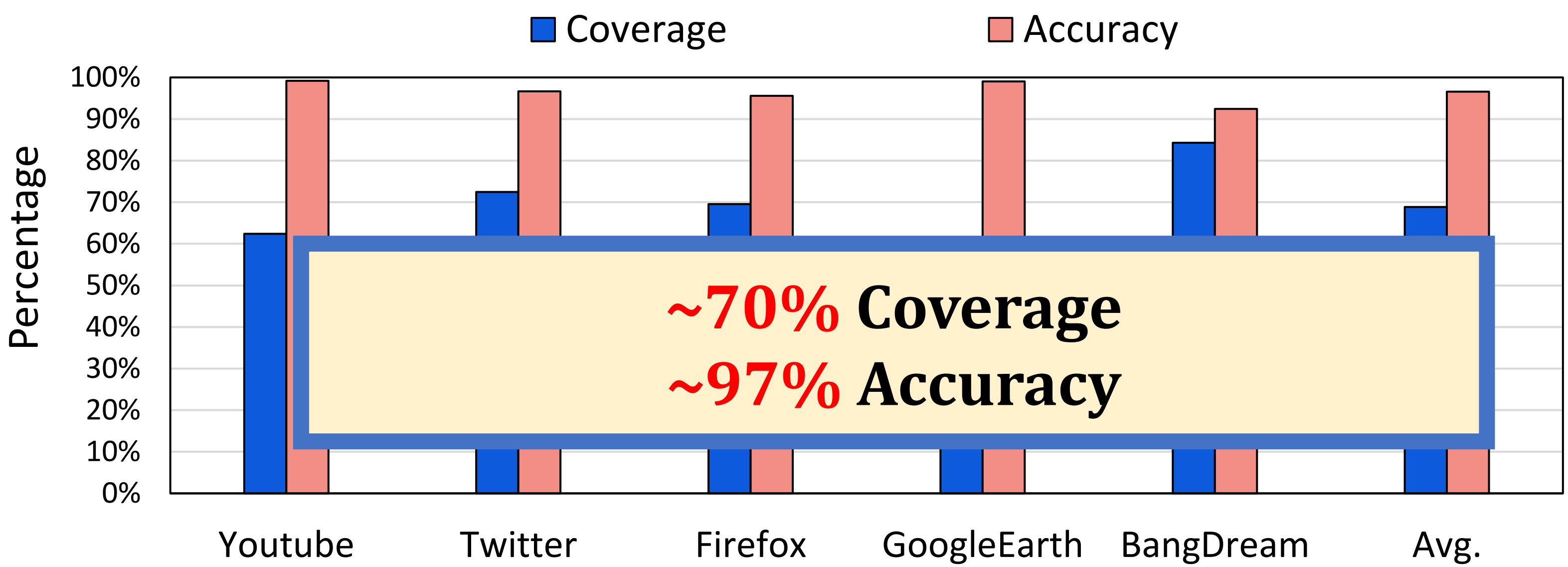
Compression and Decompression Latency

Compression and decompression latency under different settings



Prediction Accuray/Coverage

For hot data prediction



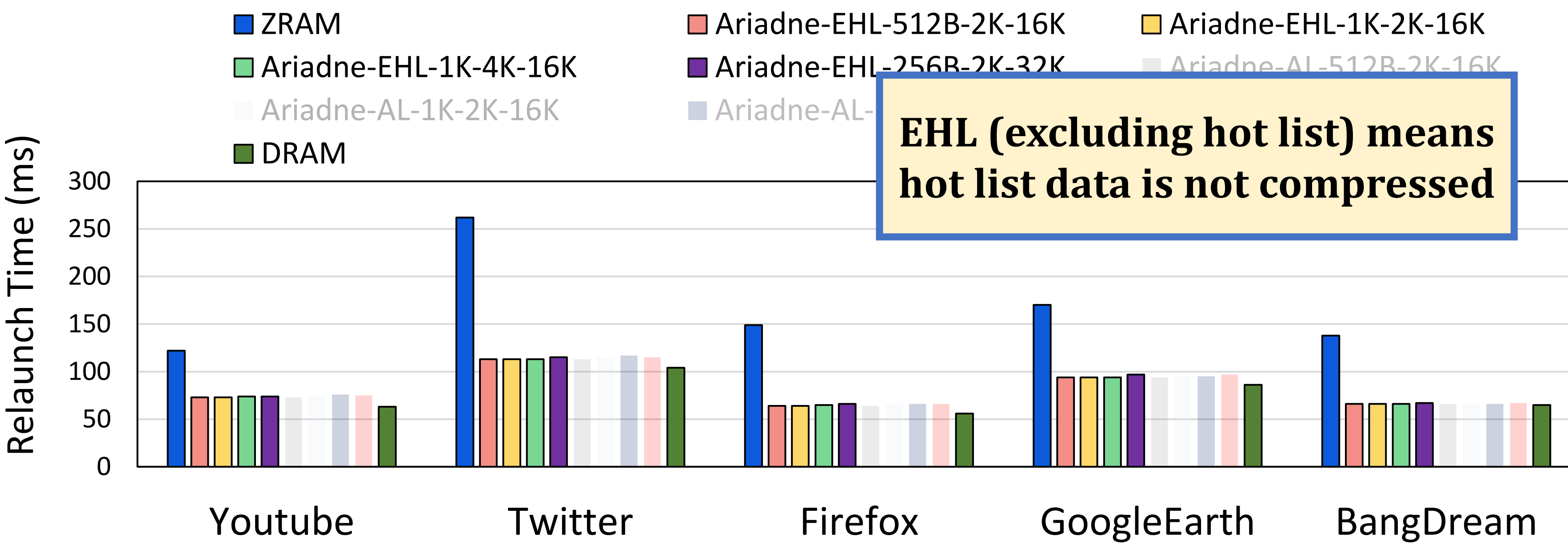
Overhead Analysis

- **HotnessOrg**:
  - Similar to existing LRU scheme, no overhead.

- **AdaptiveComp**:
  - Pages compressed together may used in different time, but locality shows impact is minimal.

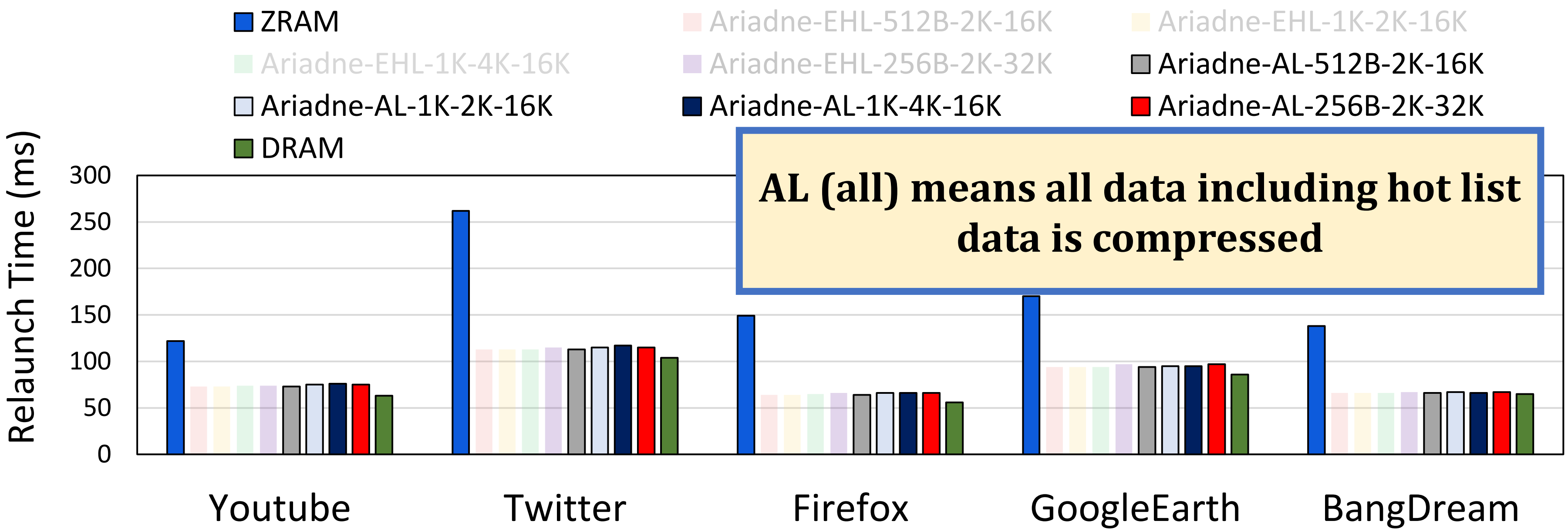
- **PreDecomp**:
  - May predict data wrongly, so only predict one page based on locality.

Implication on Application Relaunch Latency



EHL (excluding hot list) means hot list data is not compressed

Implication on Application Relaunch Latency



AL (all) means all data including hot list data is compressed