REIS

A High-Performance and Energy-Efficient Retrieval System with In-Storage Processing

Kangqi Chen
Nika Mansouri Ghiasi
Jisung Park
Rakesh Nadig
Yu Liang
Mohammad Sadrosadati
Manos Frouzakis
Haiyu Mao
Onur Mutlu

Presented by: Andreas Kosmas Kakolyris

SAFARI ETH zürich KING'S College LONDON POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY 1986 포항공과대학교

Executive Summary

Problem:

- **Retrieval Augmented Generation** (RAG) pipelines incur **data movement overheads** between the **host** and the **storage** system
- Existing **In-Storage Processing** (ISP) solutions only accelerate the **Nearest Neighbor Search** stage of RAG and introduce significant **area** and **power overheads**

Goal: Alleviate the **I/O data movement** overheads of RAG by accelerating **both** Nearest Neighbor Search and document retrieval **without modifications** to storage

REIS introduces:

- A **lightweight** scheme that **links embeddings** to **documents**
- An **ISP-tailored** data-placement algorithm
- An **Approximate Nearest Neighbor Search** (ANNS) engine that uses the **existing** components within the storage system

Key Results:

- **Improves** throughput (energy efficiency) by **15x (55x)** over CPU
- **Reduces** the I/O data movement overheads of RAG to **<0.2%** of total runtime
- **Outperforms** an ISP-based ANNS accelerator by up to **2.5x**

Outline

Typical LLM Inference Pipeline

Large Language Models:

- Can **only** generate responses from data in their training sets
- Require **expensive** retraining to memorize new data

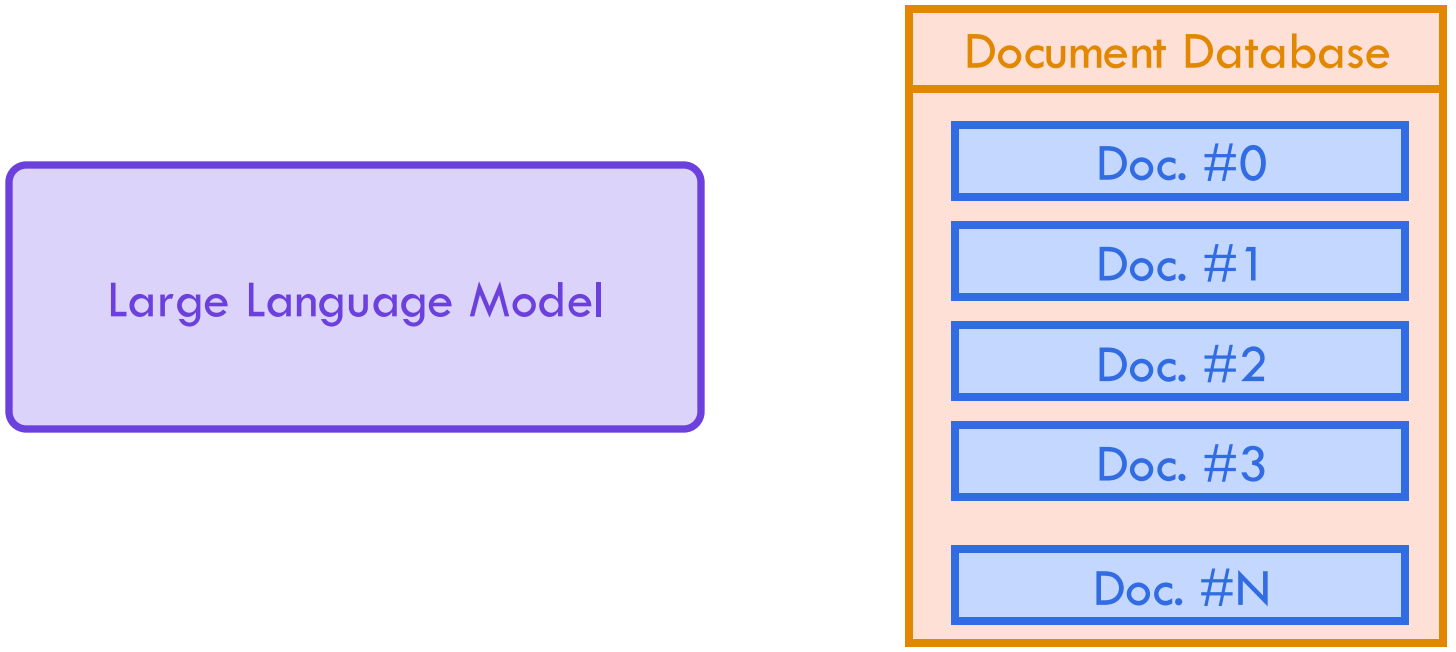
Limits their effectiveness in **real-time** and **domain-specific applications**

Retrieval Augmented Generation (RAG)

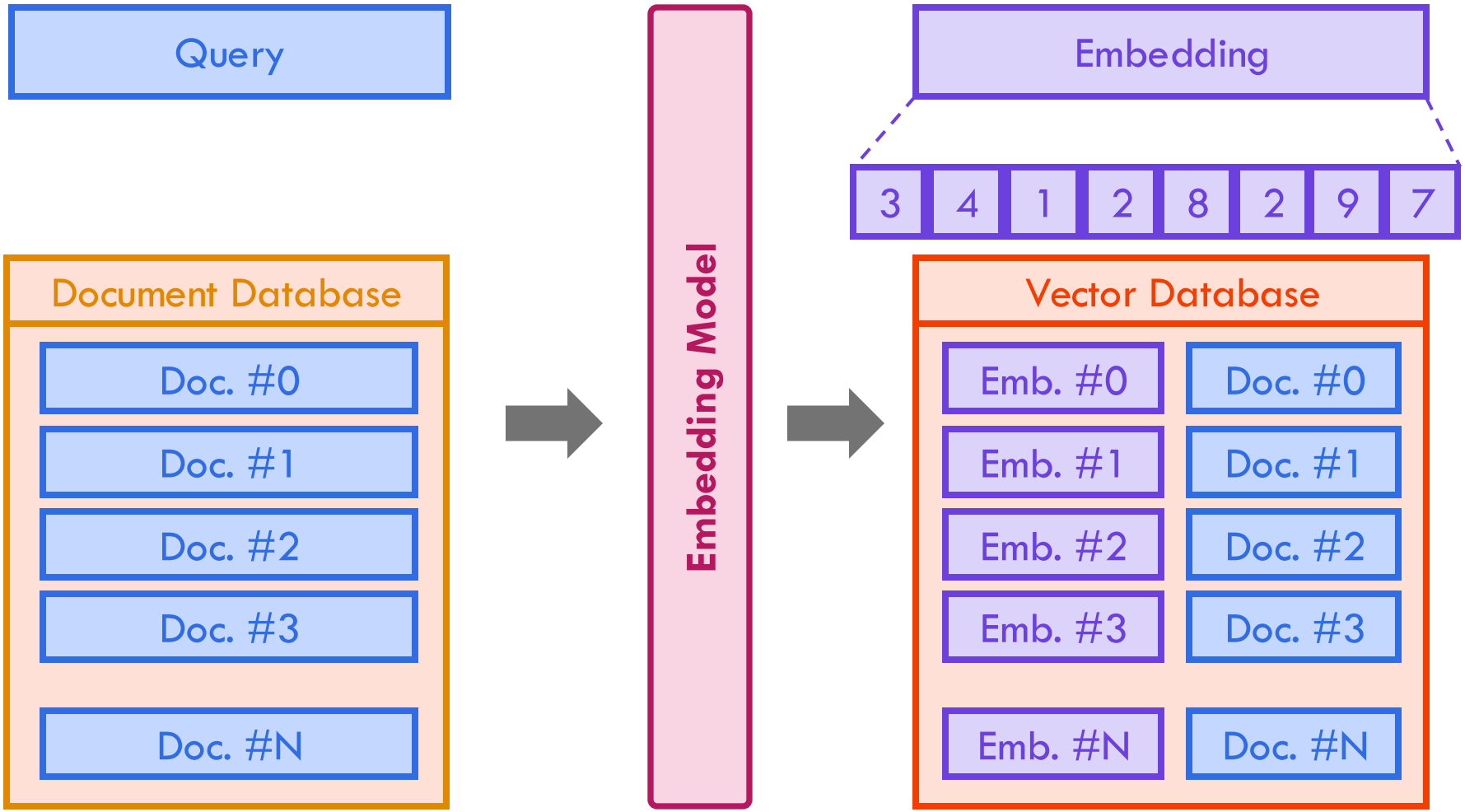
RAG combines:

- **Text generation capabilities** of **Large Language Models**
- **Information retrieval techniques** from **document databases**

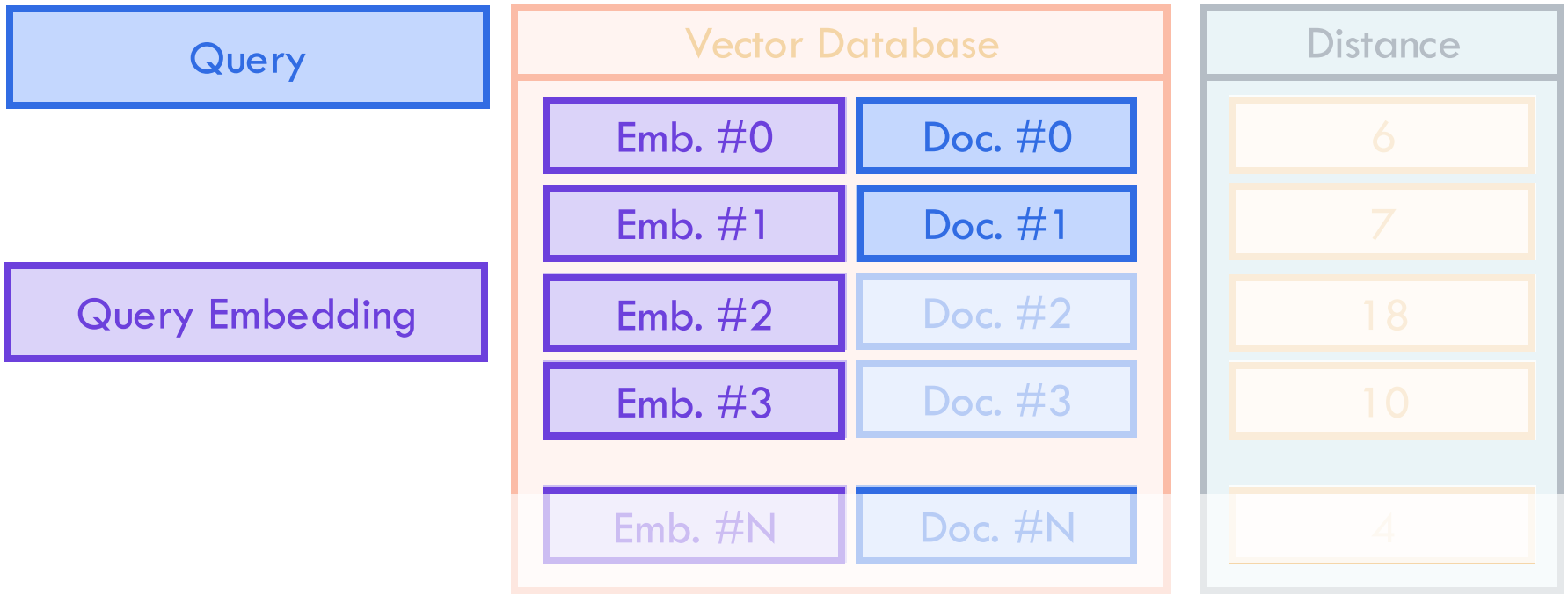
RAG **identifies** the most **relevant** documents to the query to generate a **high-quality response.**



Retrieval Augmented Generation (RAG)



Nearest Neighbor Search



Nearest Neighbor Search creates significant data movement

Large embeddings lead to expensive distance calculations

Accelerating Nearest Neighbor Search

Multiple approaches reduce Nearest Neighbor Search overheads

Quantization:

- Integer Quantization
- Binary Quantization
- Product Quantization

In-Storage Processing:

- Employs the computational resources within storage
- Takes advantage of the large internal bandwidth of SSDs

Outline

Background

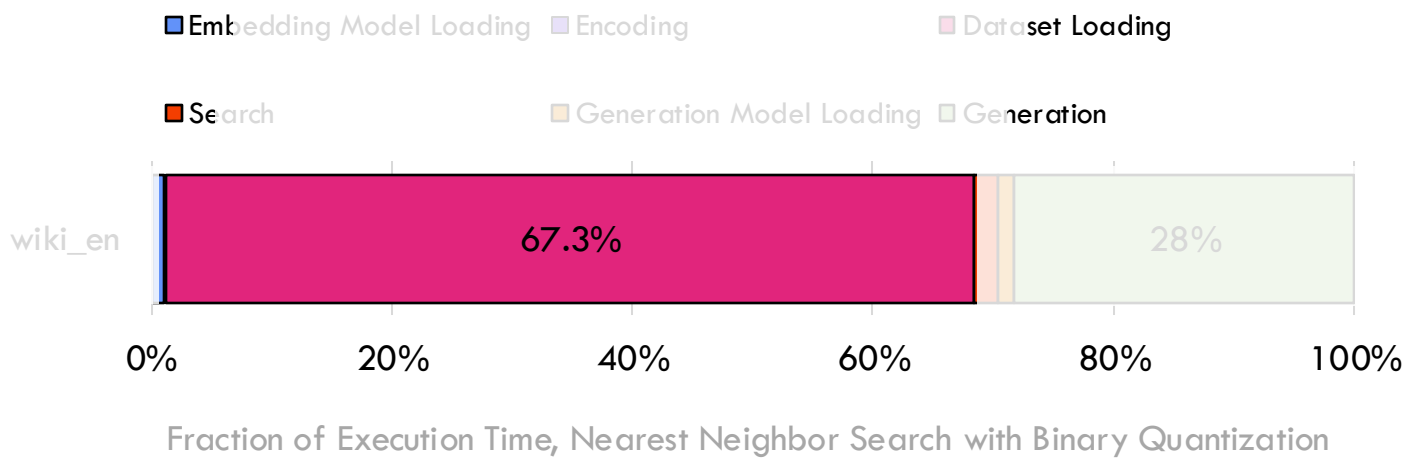
Motivation

REIS

Evaluation

Conclusion

Latency Breakdown of RAG

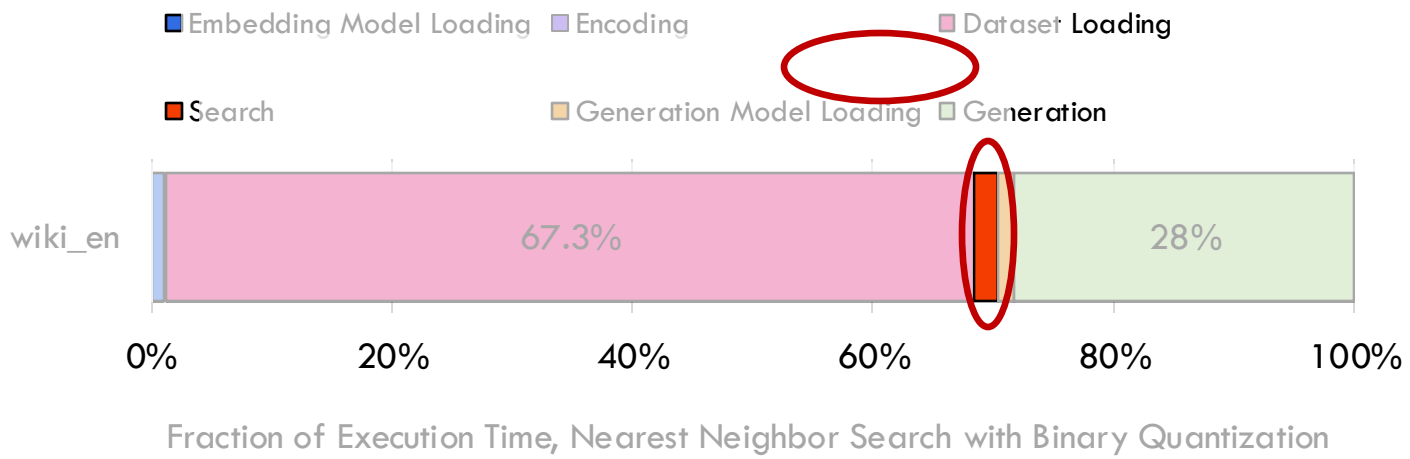


Fraction of Execution Time, Nearest Neighbor Search with Binary Quantization

Dataset Loading dominates total latency for large datasets

Quantization does not alleviate data movement overheads

In-Storage Nearest Neighbor Search Accelerators



Fraction of Execution Time, Nearest Neighbor Search with Binary Quantization

Existing designs **only accelerate** the **search** process

Prior works **introduce** significant **modifications** to the SSDs

Our Goal

Fundamentally **alleviate** the I/O data movement **bottlenecks** of RAG through an **ISP design** that **does not** require **modifications** to perform computation inside the storage system.

Outline

Background

Motivation

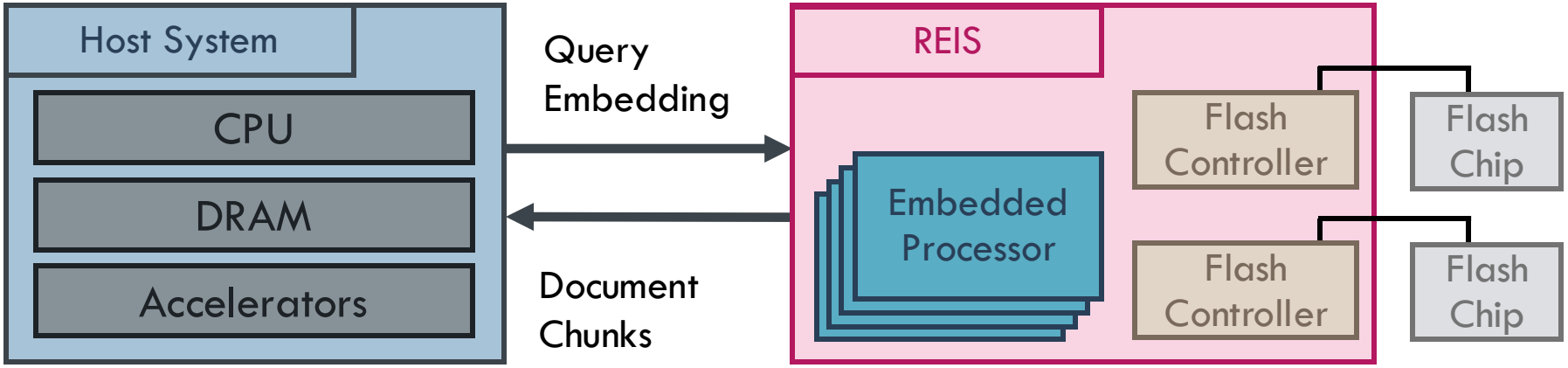
REIS

Evaluation

Conclusion

REIS: Retrieval System with In-Storage Processing

Key Idea: Leverage both **In-Storage** and **In-Flash Processing** to **exploit** the computational **parallelism** of Flash storage and **conserve** precious flash channel **bandwidth**.



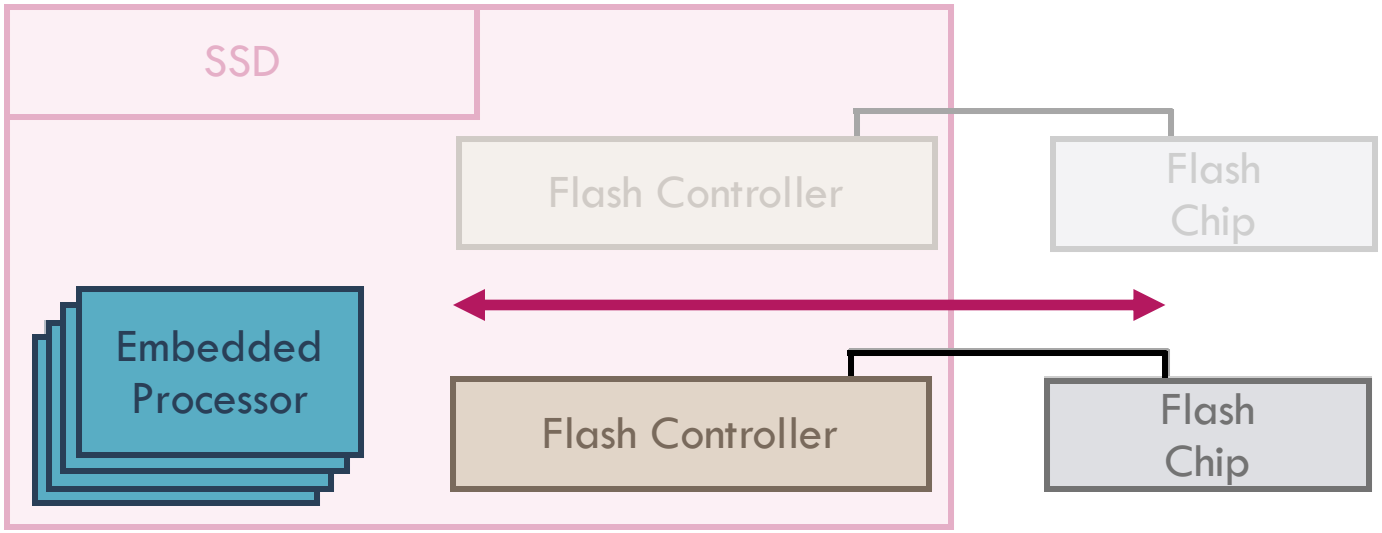
REIS employs **three key mechanisms**:

- An Efficient **ISP-tailored** implementation of the **Inverted File algorithm**
- **A low-cost** embedding-to-document **linkage mechanism**
- An **ANNS Engine** that **uses existing computational resources** within SSDs

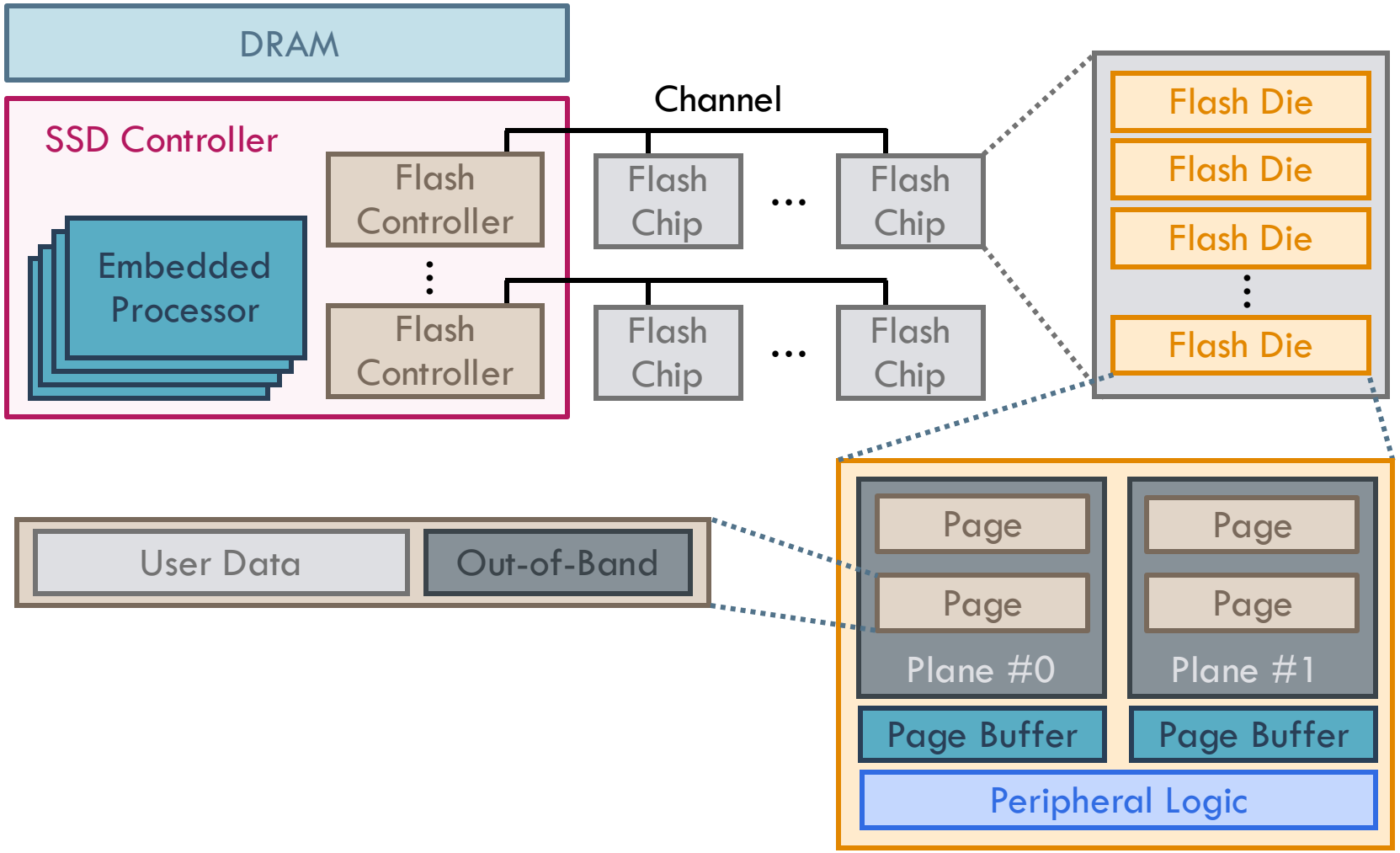
Challenges of In-Storage Processing

The design of the **storage** system creates **unique limitations:**

- Limited performance and functionality in the embedded cores
- Computation inside flash chips limited to bit-wise and bit-counting operations
- Error-correction limits ISP performance due to data movement overheads



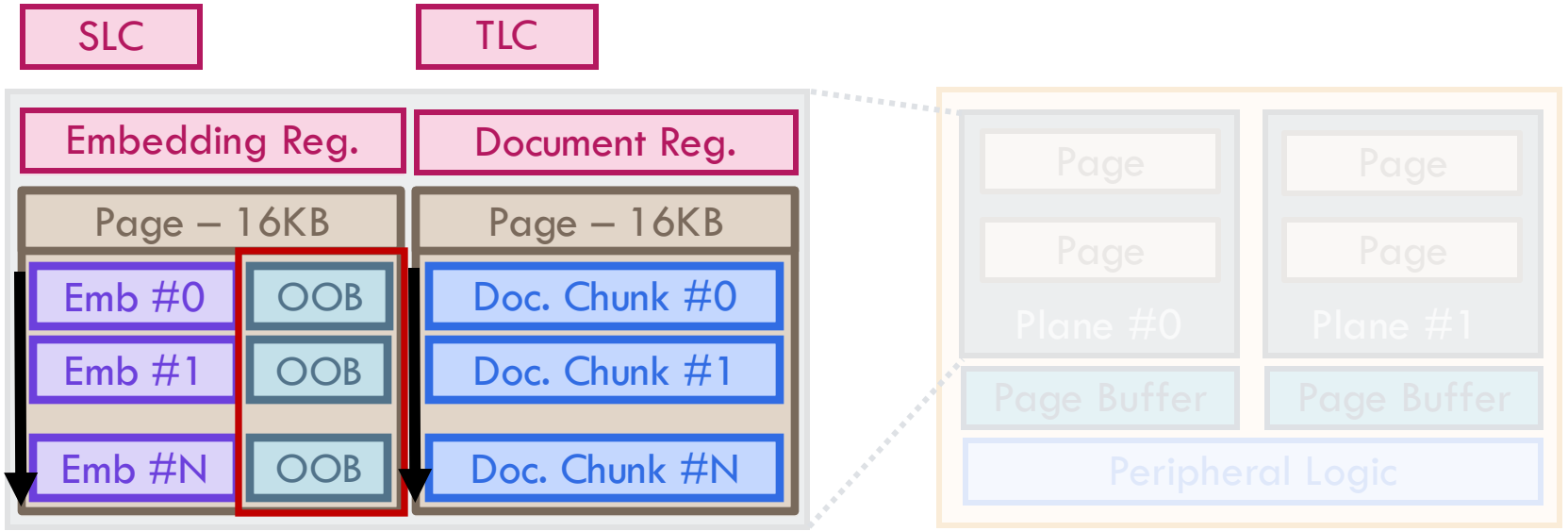
REIS: Overview



REIS: Database Layout

REIS' Database Layout:

- Creates separate regions for storing embeddings and documents.
- Stores Embeddings using Enhanced SLC Programming for robust ISP operations.
- Stores Documents in TLC to provide high storage density.
- Stores Embeddings and Documents contiguously to avoid address translation.
- Links Embeddings to Documents via the Out-of-Band (OOB) area in Flash pages.



Park et al, "Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory", in MICRO 2022

REIS: Inverted File Algorithm (IVF)

REIS employs the **Inverted File Algorithm** (IVF) with **Binary Quantization** to accelerate **Approximate Nearest Neighbor Search** with **In-Storage** and **In-Flash processing.**

IVF:

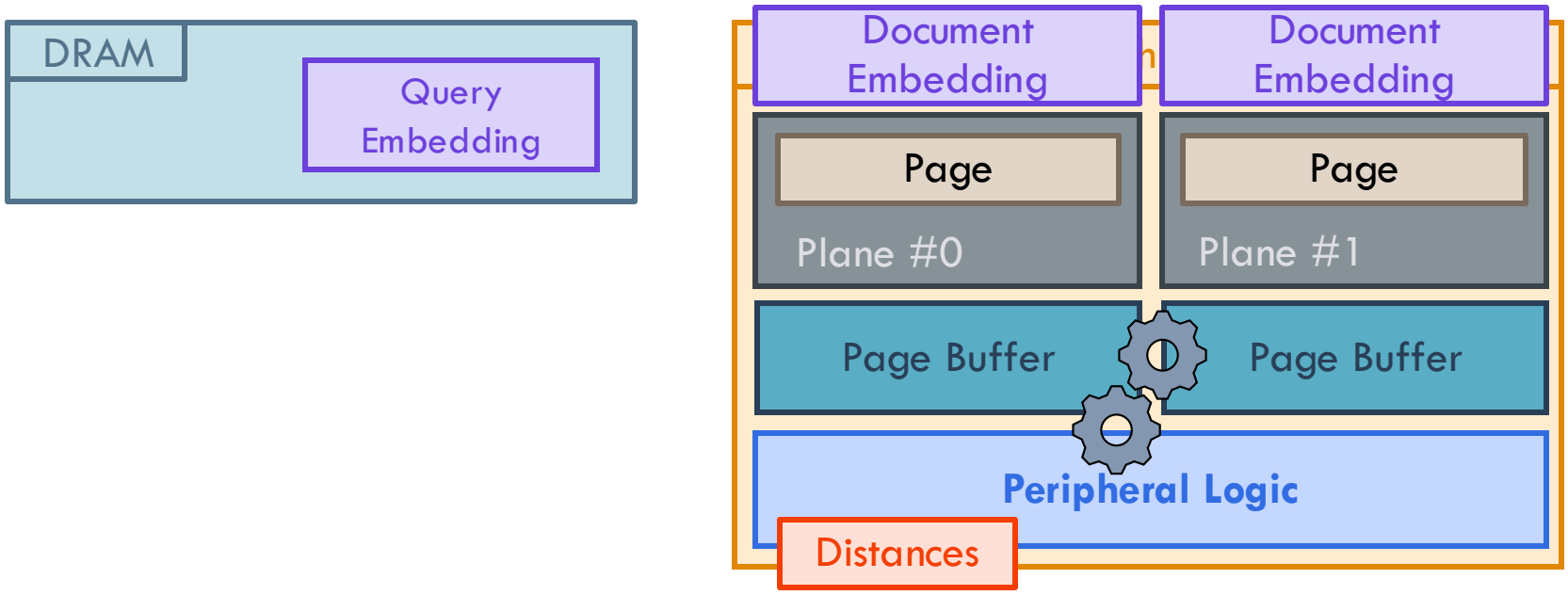
- **Organizes** embeddings into **clusters**
- Performs a **coarse-grained search** to identify the **nearest clusters**
- Performs a **fine-grained search** within the identified clusters to find the **nearest embeddings**

The **Inverted File Algorithm:**

- Provides high **performance** and **accuracy** with Binary Quantization
- Features characteristics that make it **suitable** to the **storage** system

We thus design an **In-Storage ANNS engine** that **accelerates** IVF

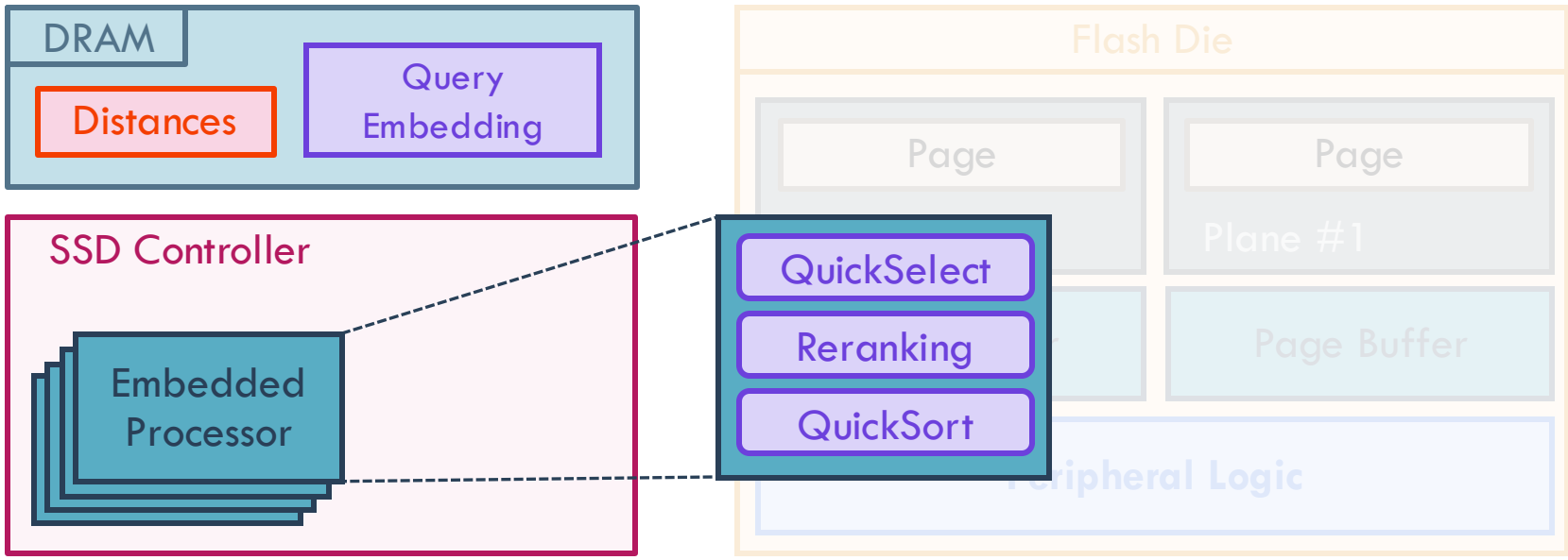
REIS: In-Storage ANNS Engine



For the coarse- and fine-grained stages of IVF, the ANNS engine:

- **Copies** the **Query Embeddings** to the Page Buffer
- **Moves** the **Document Embeddings** to the Page Buffer
- **Calculates distances** using logic in the page buffer and the peripheral circuitry
- **Transfers** the **distances** back to the SSD's DRAM

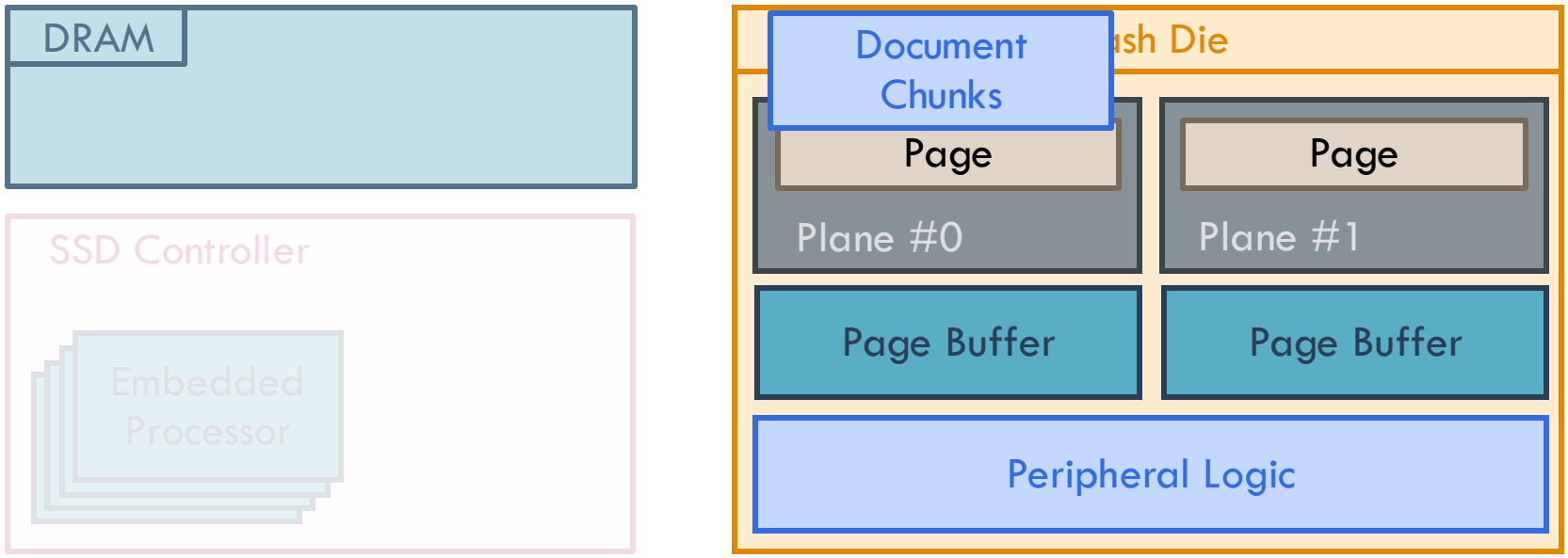
REIS: In-Storage ANNS Engine



The Embedded Processor:

- Performs QuickSelect to identify the Nearest Clusters and embeddings
- Performs Reranking and QuickSort to refine the search results.

REIS: In-Storage ANNS Engine



The ANNS Engine:

- Moves the identified document chunks to the SSD's DRAM
- Transfers them to the host

REIS: Optimizations

We improve the **In-Storage ANNS engine** with the following **optimizations:**

- **Distance Filtering (DF)**
  - Quickly **filters** out embeddings with **large** distances **without** transferring them to the embedded processor.
- **Multi-Plane Input Broadcasting (MPIBC)**
  - **Parallelly transfers** the query embedding to multiple planes.
- **Pipelining (PL)**
  - **Overlaps** flash page **reads** with distance **computations**.

Outline

Background

Motivation

REIS

Evaluation

Conclusion

Evaluated System Configurations & Baselines

Baselines

- **CPU-Real:** A CPU-based RAG retrieval system running IVF
- **No-I/O:** The same CPU-based system without I/O overheads
- **NDSearch:** A state-of-the-art ISP-based ANNS accelerator

Algorithms

- Exact Nearest Neighbor Search (BF)
- Inverted File Algorithm (IVF)

Evaluated Metrics

- **Throughput:** Queries-per-second (QPS)
- **Energy Efficiency:** QPS/ Watt
- **End-to-end Retrieval Performance**

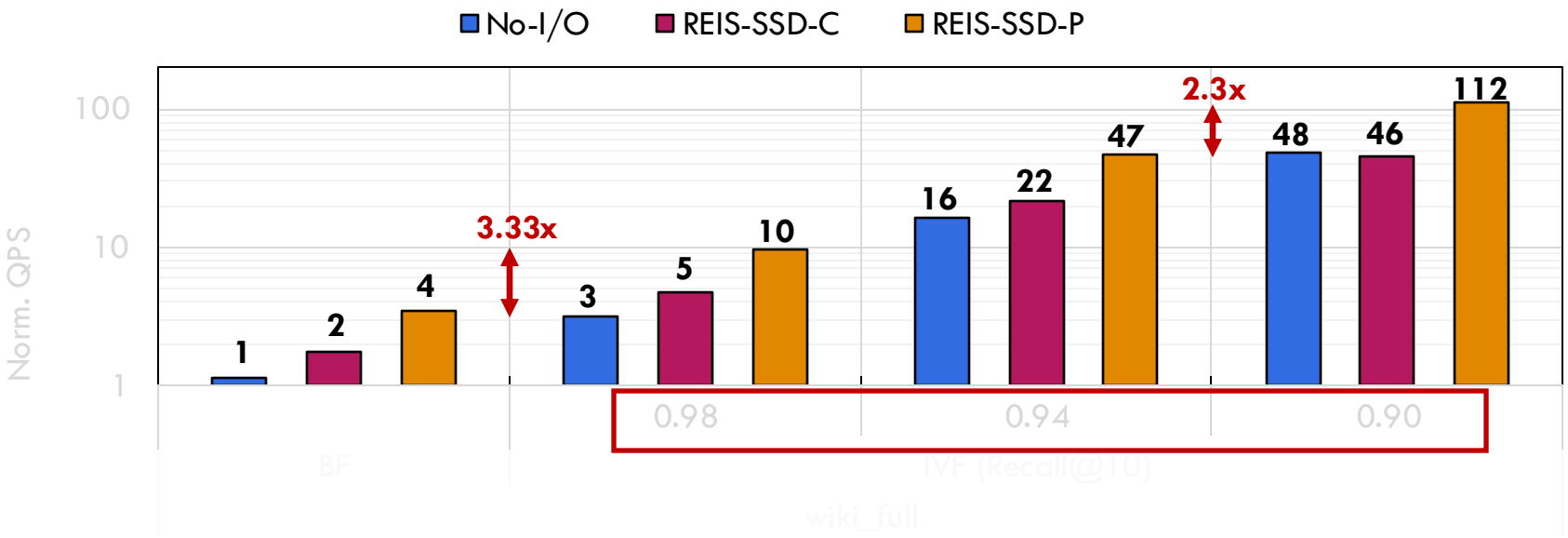
Datasets

- Wikipedia (Full)
- Wikipedia (English subset)
- Natural Questions (NQ)
- HotpotQA
- SIFT-1B
- DEEP-1B

SSD Configurations

- **SSD-C:** Cost-oriented
- **SSD-P:** Performance-oriented

Performance Evaluation

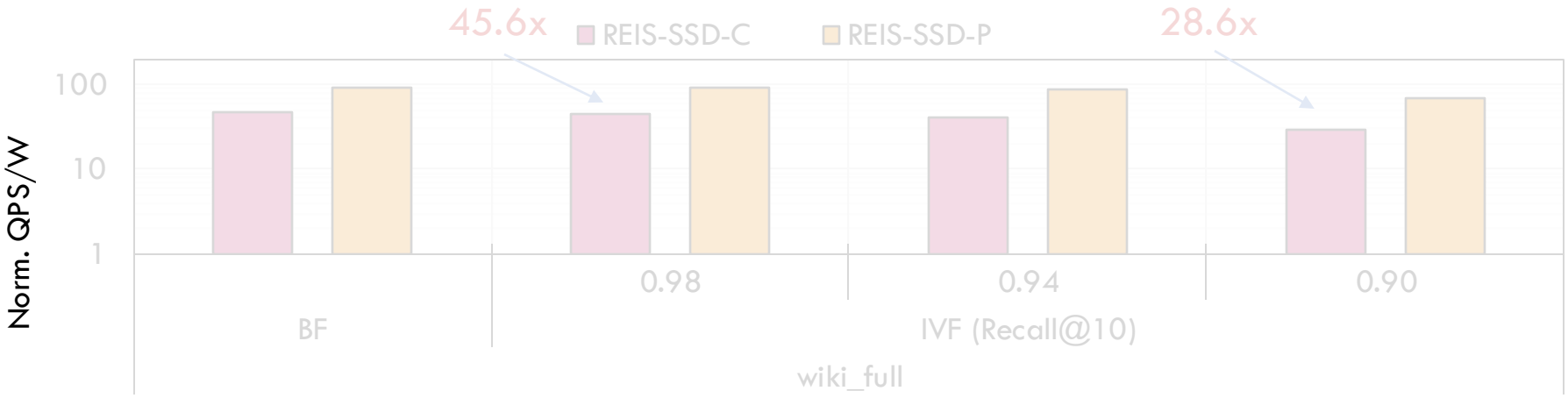


- We compare the throughput of **No-I/O** and REIS on **SSD-C** and **SSD-P**
- Throughput is **normalized** to that of CPU-Real
- We evaluate performance on **BF** and **IVF** across different recall values

REIS outperforms **No-I/O due to** the **parallelism** of the SSD

Performance improvements increase at **higher recall values**

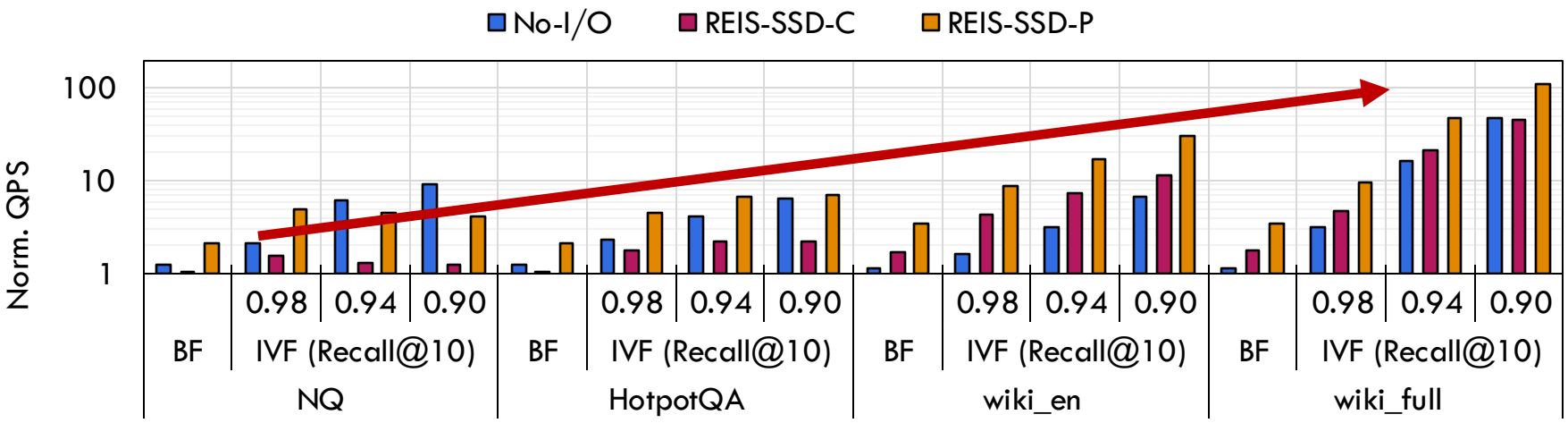
Energy Efficiency Evaluation



- Energy efficiency normalized to that of CPU-Real

Improvements in energy efficiency increase at higher recall values

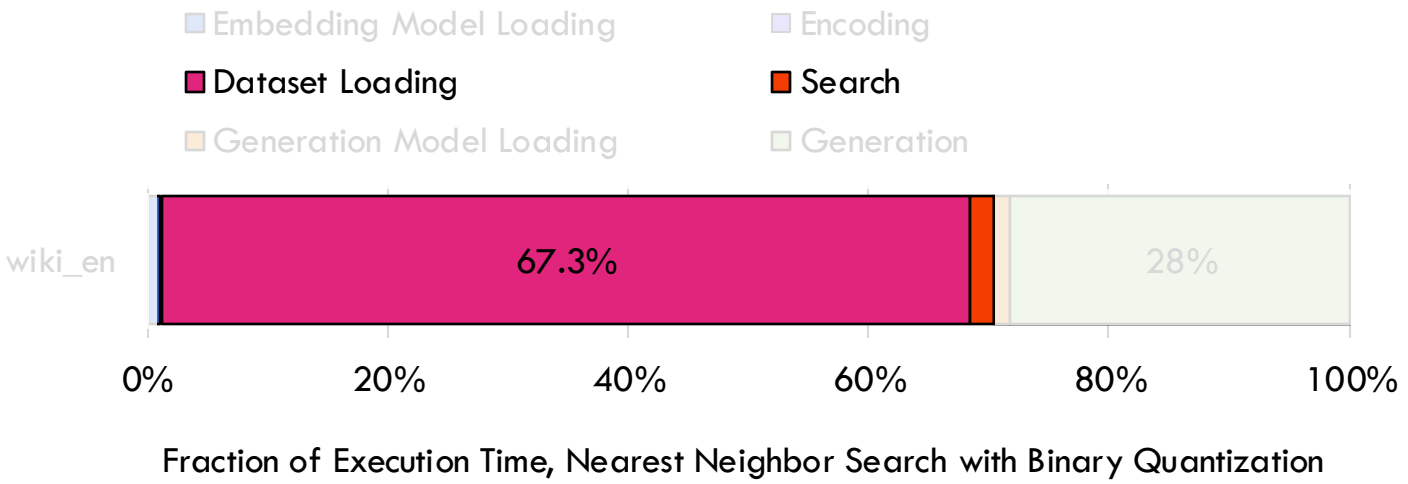
Evaluation Across Multiple Datasets



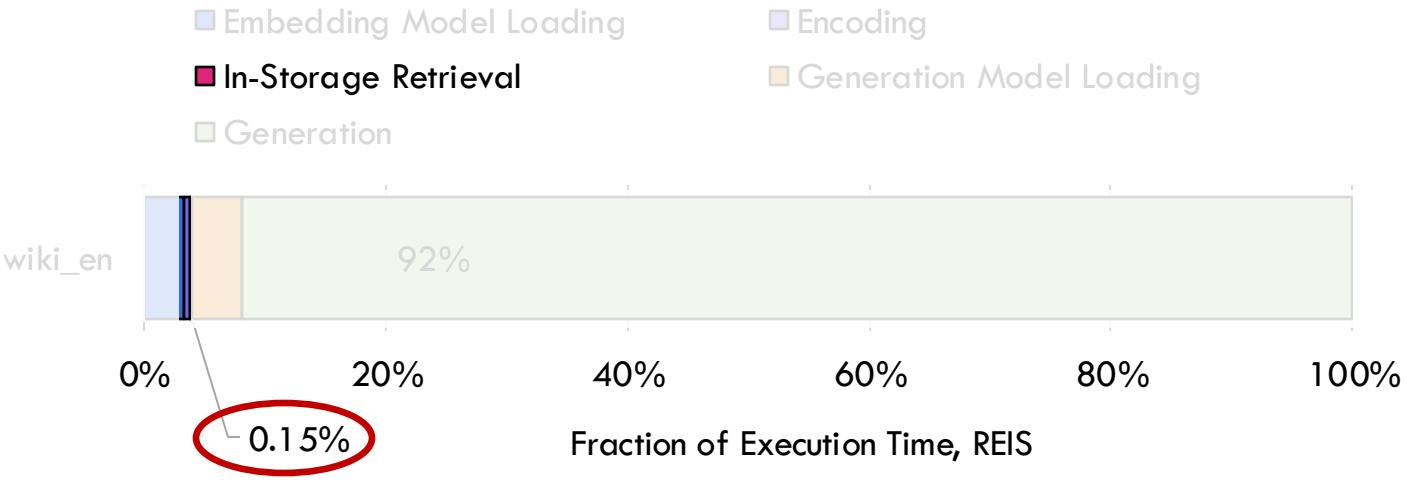
- **Similar trends** across all datasets
- **Performance** benefits **increase** with **larger datasets**

End-to-End Performance: Latency Breakdown

CPU Retrieval



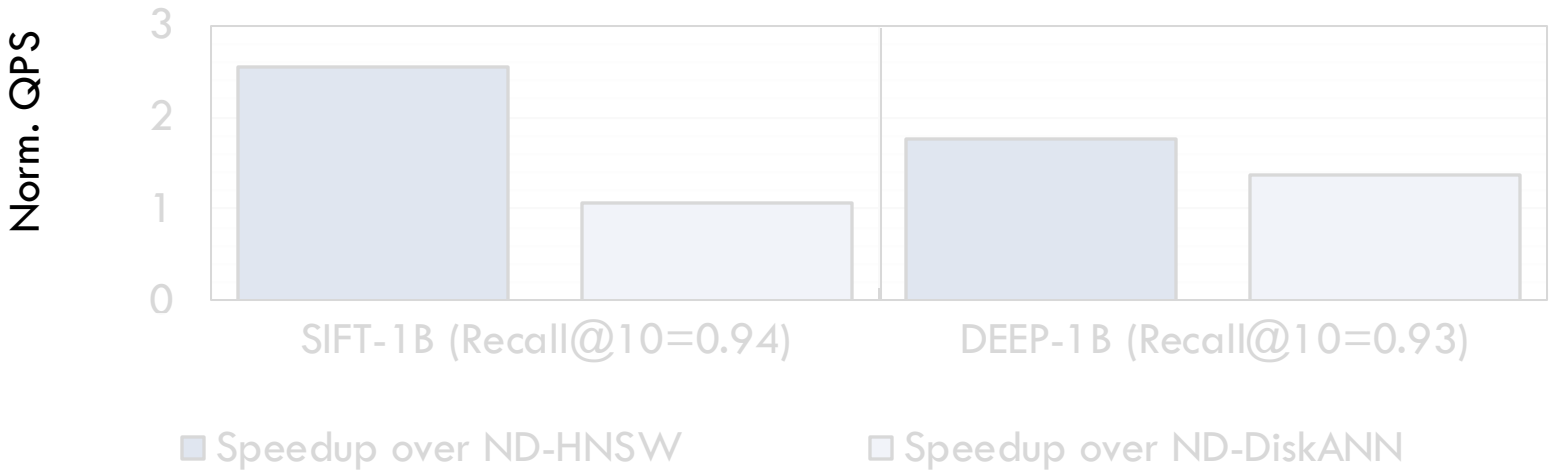
REIS



Sensitivity Study

- **Distance Filtering** provides a 5x throughput increase across all configurations
- **Pipelining** further improves performance by 1.4x on average
- **Multi-Plane Input Broadcasting** improves performance on **SSD-P** by 26% on average on top of Pipelining
- These optimizations offer a **combined performance improvement** of **12.9x** over the **NO-OPT** baseline.

Performance Comparison to NDSearch



- We normalize the throughput of REIS to that of NDSearch running HNSW and DiskANN
- We evaluate both REIS and NDSearch on SIFT-1B and DEEP-1B

REIS improves performance by:

- 2.5x and 1.8x over **NDSearch** with **HNSW**
- 1.05x and 1.36x over **NDSearch** with **DiskANN**

REIS with IVF features **streaming access patterns** that exploit the internal **bandwidth** of the storage system

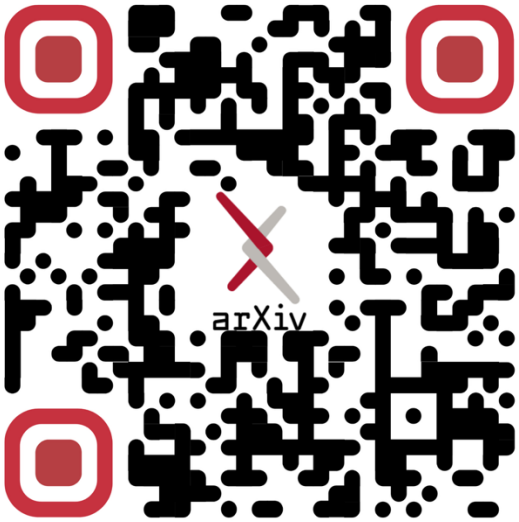
Wang et al, "NDSEARCH: Accelerating Graph-Traversal-Based Approximate Nearest Neighbor Search through Near Data Processing", in ISCA 2024

Available on arXiv

REIS: A High-Performance and Energy-Efficient Retrieval System with In-Storage Processing

Kangqi Chen[1] Andreas Kosmas Kakolyris[1] Rakesh Nadig[1] Manos Frouzakis[1]
Nika Mansouri Ghiasi[1] Yu Liang[1] Haiyu Mao[1,2]
Jisung Park[3] Mohammad Sadrosadati[1] Onur Mutlu[1]

ETH Zürich[1] King's College London[2] POSTECH[3]



https://arxiv.org/abs/2506.16444

Outline

Background

Motivation

REIS

Evaluation

Conclusion

Conclusion

RAG retrieval is **bottlenecked** by **I/O data transfer overheads** between the **host** and the **storage system**

REIS

The first system for RAG retrieval that **leverages In-Storage Processing using the existing computational resources** within the **storage** system

Key Results

- **Outperforms** CPU search by **15x on average**
- **Improves energy efficiency by 55x on average**
- **Reduces document** retrieval latency to **< 0.2%**
- **Outperforms NDSearch by up to 2.5x**

REIS:
A High-Performance and Energy-Efficient Retrieval System with In-Storage Processing

Kangqi Chen
Nika Mansouri Ghiasi
Jisung Park
Rakesh Nadig
Yu Liang
Mohammad Sadrosadati
Manos Frouzakis
Haiyu Mao
Onur Mutlu

Presented by: Andreas Kosmas Kakolyris

SAFARI ETH zürich KING'S College LONDON POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY 1986 포항공과대학교

REIS:
A High-Performance and Energy-Efficient Retrieval System with In-Storage Processing

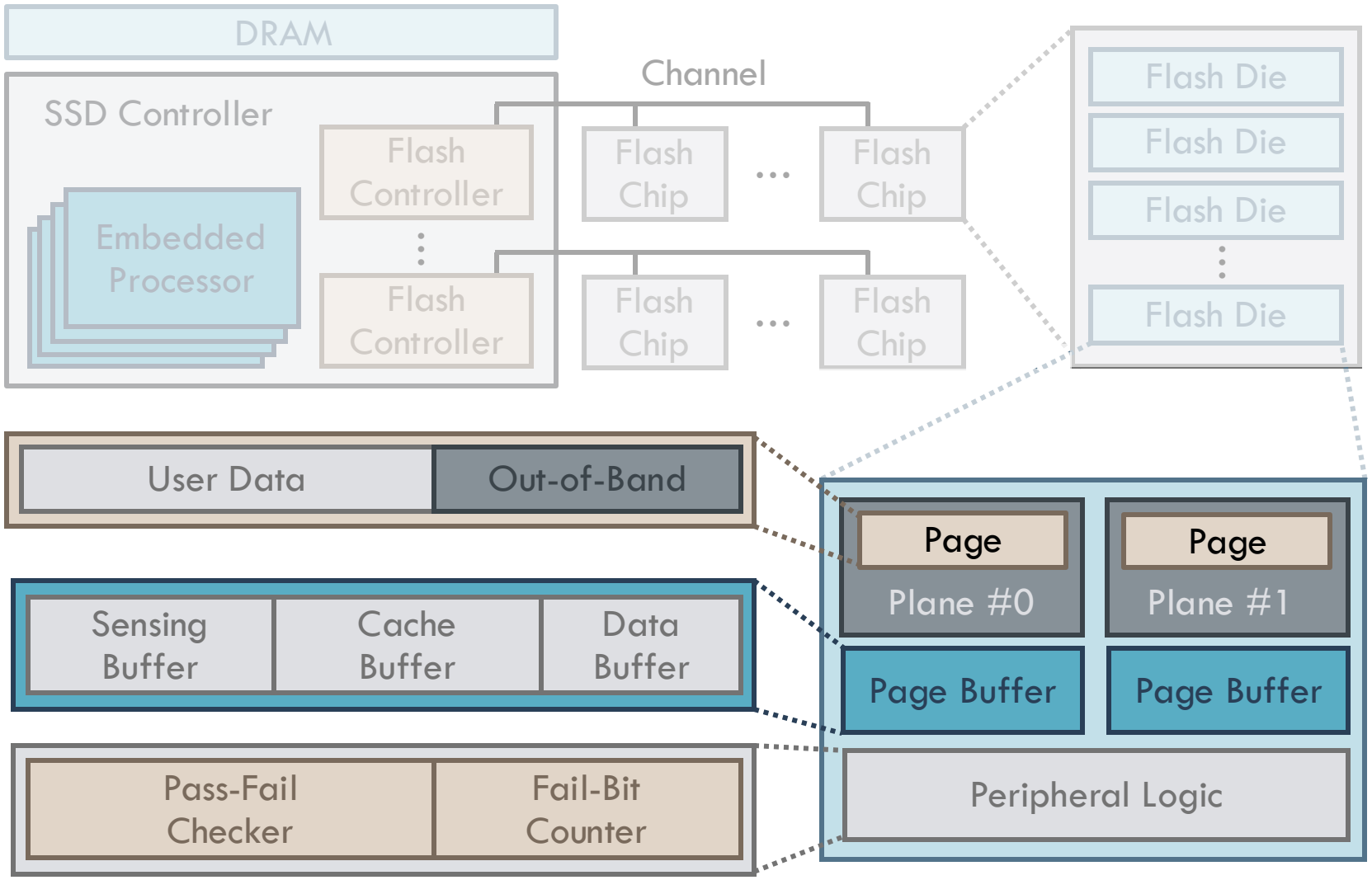
Backup Slides

Kangqi Chen
Nika Mansouri Ghiasi
Jisung Park
Rakesh Nadig
Yu Liang
Mohammad Sadrosadati
Manos Frouzakis
Haiyu Mao
Onur Mutlu

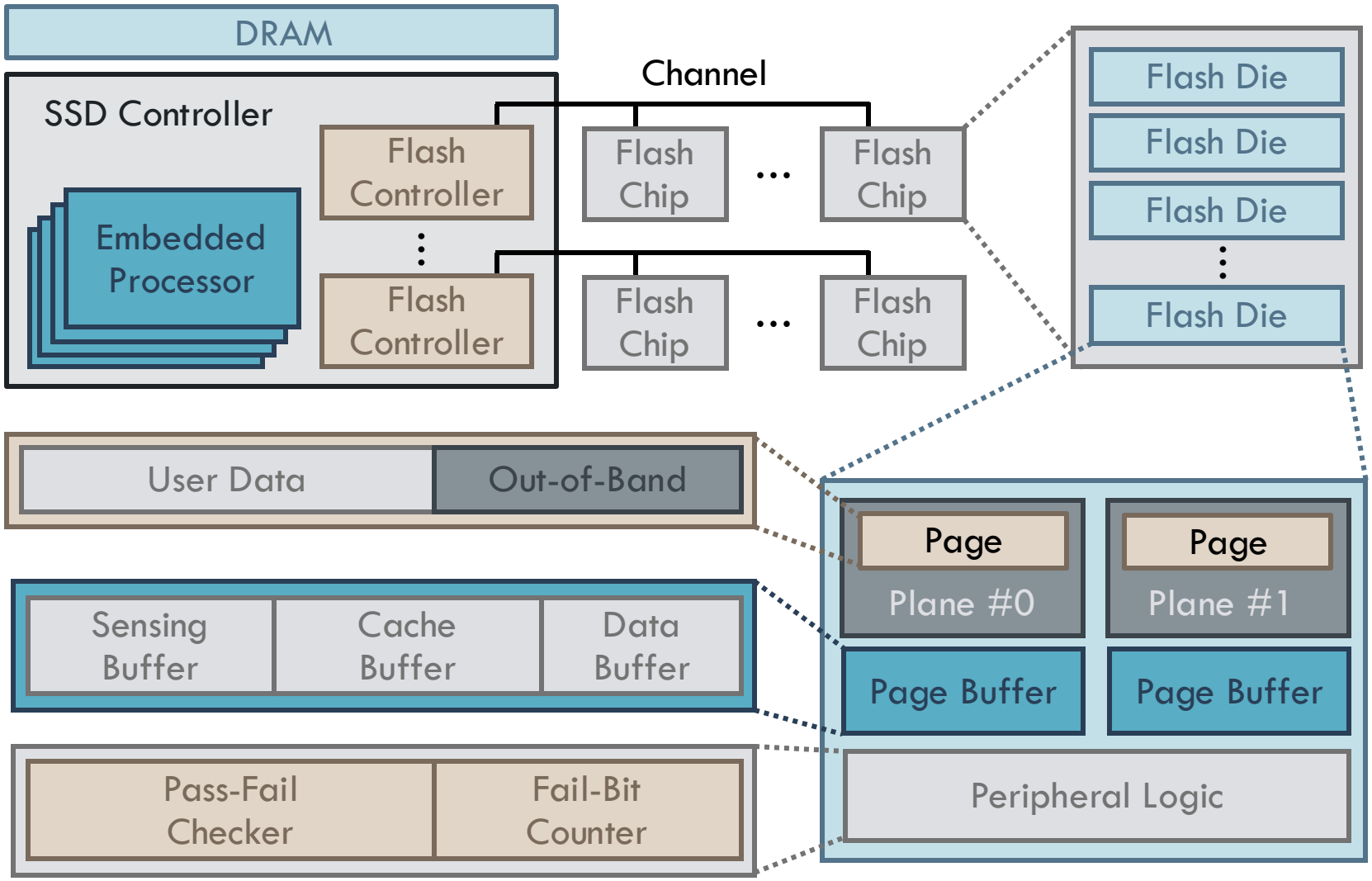
Presented by: Andreas Kosmas Kakolyris

SAFARI ETH zürich KING'S College LONDON POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY 1986 포항공과대학교

SSDs & NAND Flash Memory



SSDs & NAND Flash Memory



REIS: In-Storage ANNS Engine