BeaconGNN: Large-Scale GNN Acceleration with Asynchronous In-Storage Computing

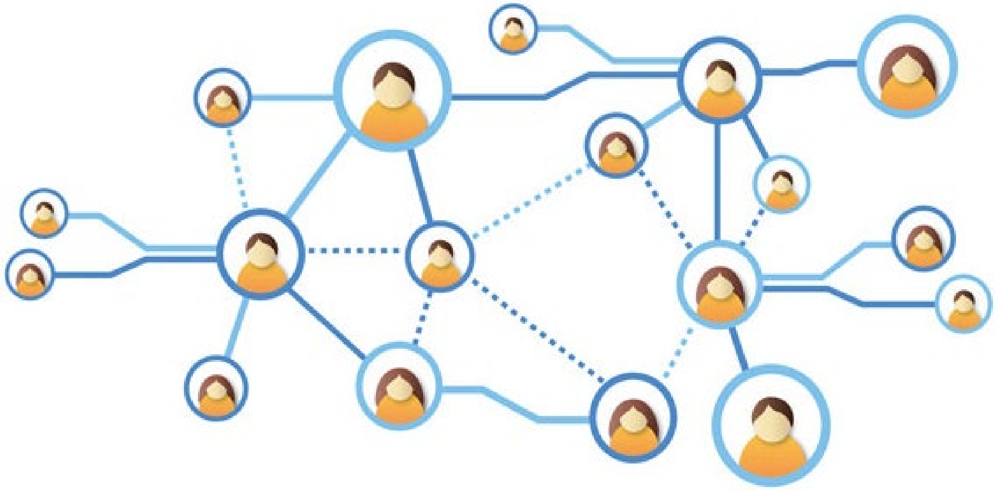
Yuyue Wang[1], Xiurui Pan[2], Yuda An[2],
Jie Zhang[2], **Glenn Reinman**[1]

1 UCLA Samueli School of Engineering

2 PEKING UNIVERSITY

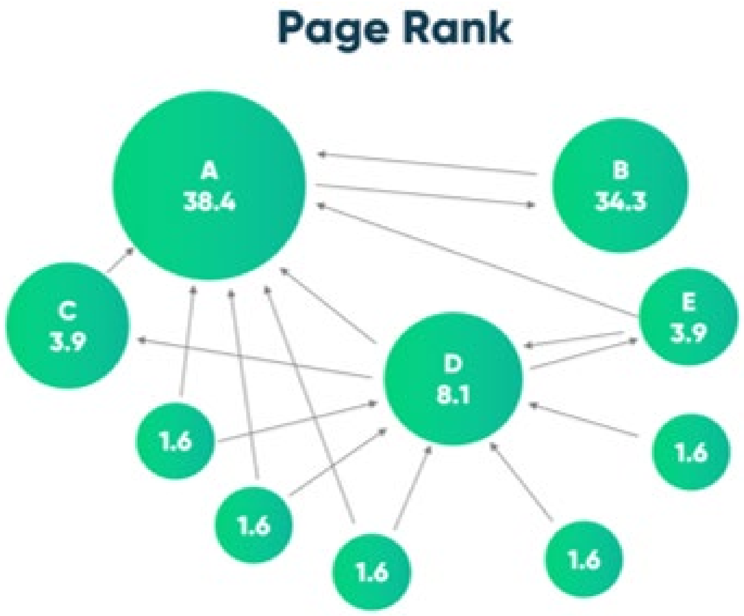
What is GNN, why does it matters

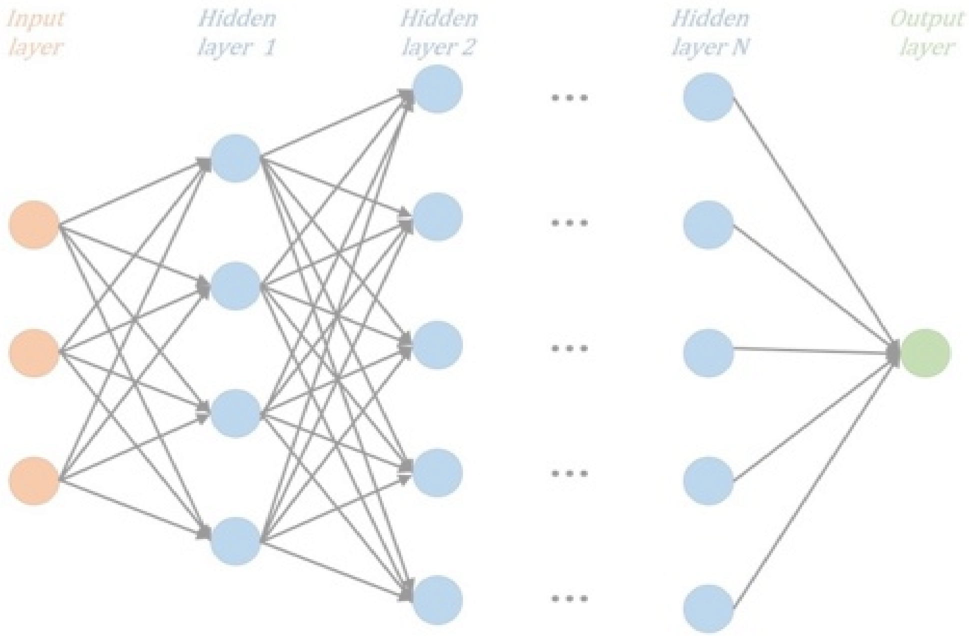
- Graph, a universal structure
  - Social network
  - Recommendation system
  - Pandemic…
- Graph information
  - Node: a vector of feature
  - Edge: relation between nodes



Nodes and edges provide rich information to analyze

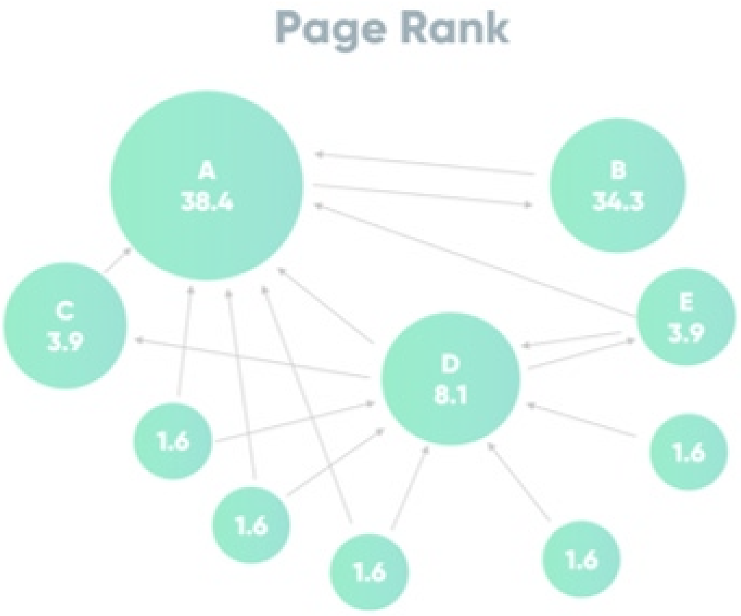
They used to be processed in separate

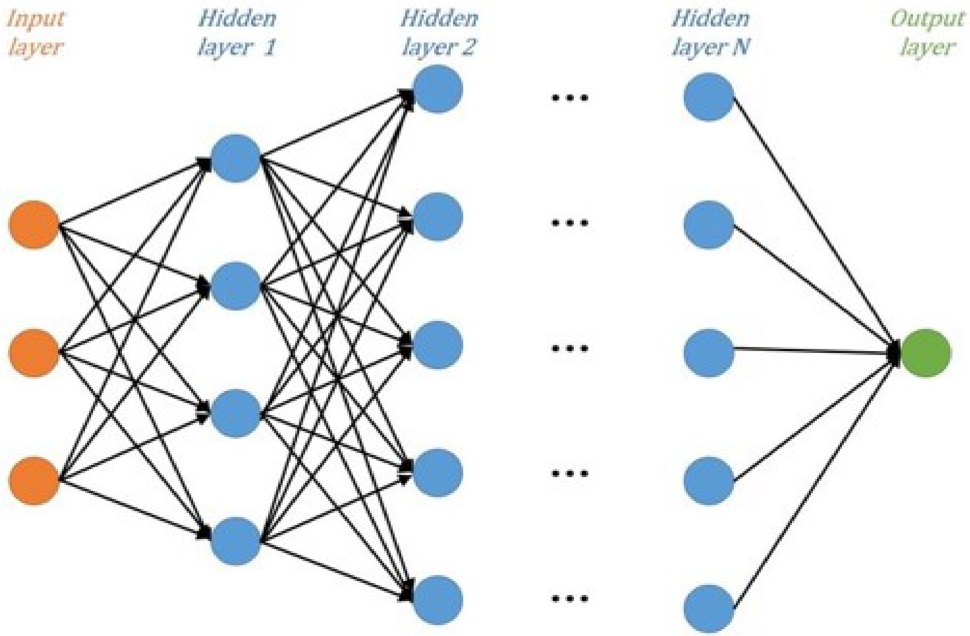




Data type	Representation	Analysis method
Edge (connection)	Adjacency matrix, …	Classical graph analytics algorithms (e.g. page rank)
Node	Feature vectors	Machine learning to extract high level features

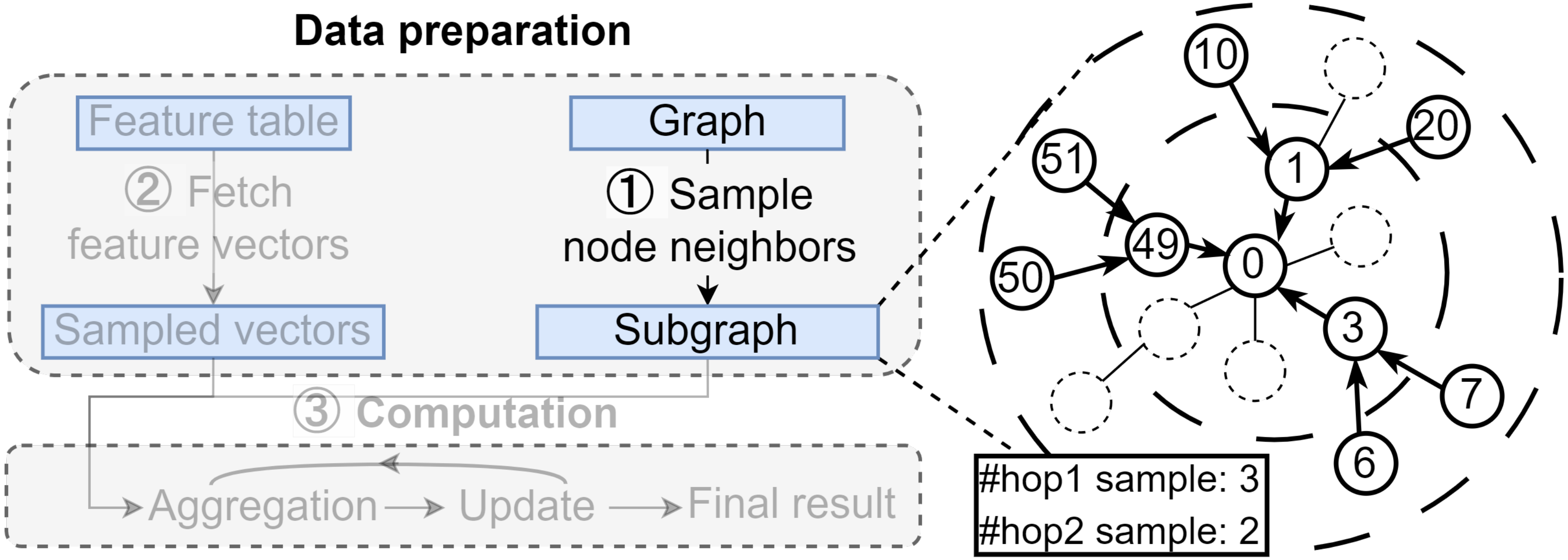
They used to be processed in separate



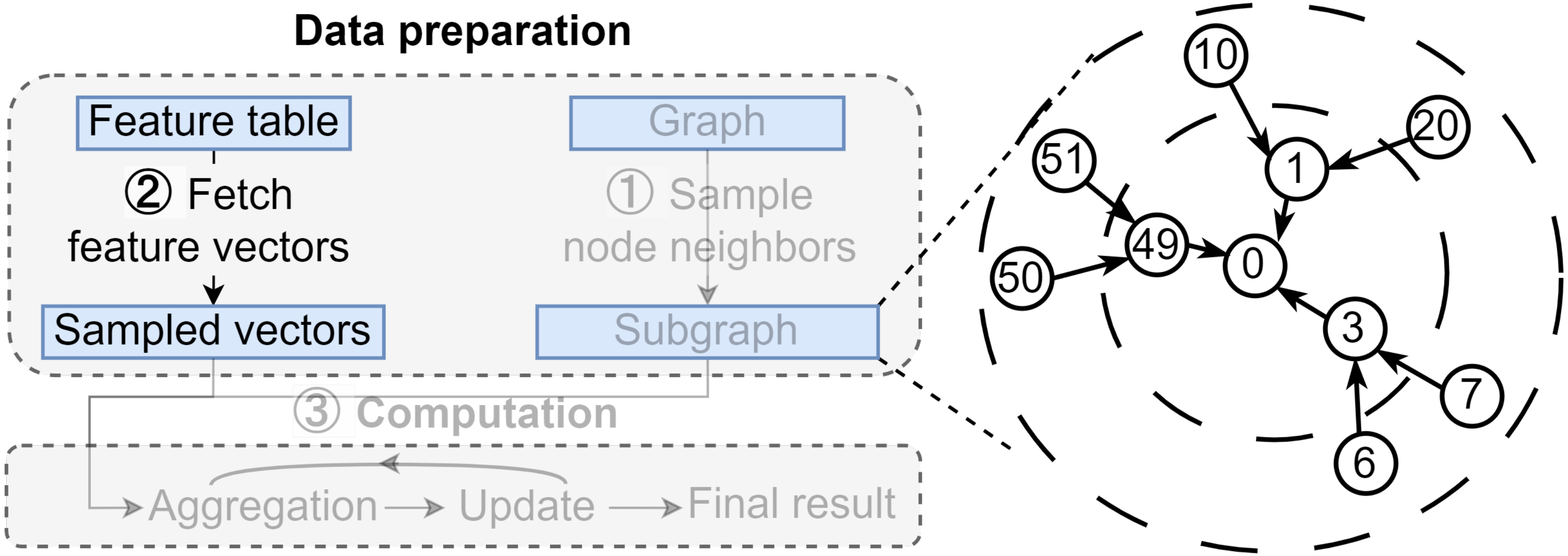


Data type	Representation	Analysis method
Edge (connection)	Adjacency matrix, …	Classical graph analytics algorithms (e.g. page rank)
Node	Feature vectors	Machine learning to extract high level features

Graph neural network (GNN) bridges the two domains

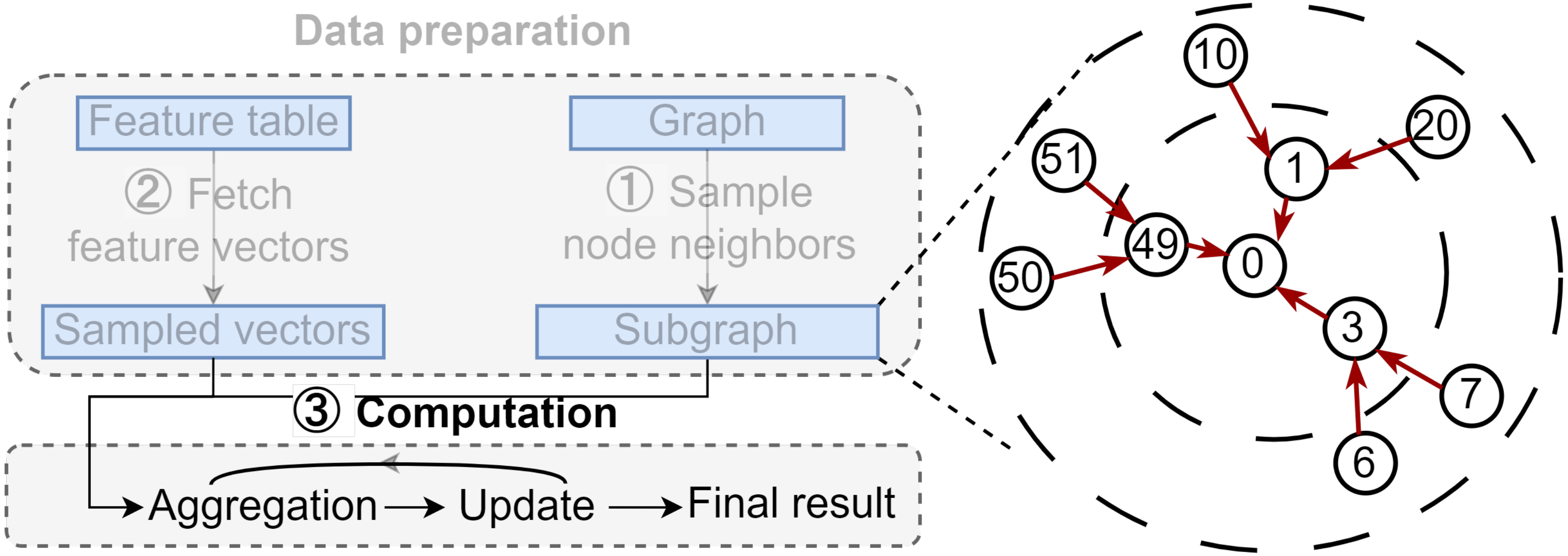


Graph neural network (GNN) bridges the two domains



$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ \ldots \\ \\ \end{bmatrix}$$

Graph neural network (GNN) bridges the two domains



Data preparation

Feature table

② Fetch feature vectors

Sampled vectors

Graph

① Sample node neighbors

Subgraph

③ Computation

Aggregation → Update → Final result

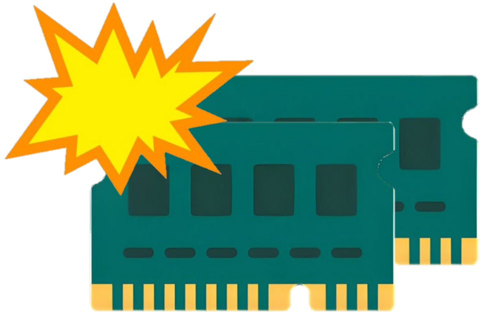
GNN extracts both *graph structure* and *node features*

System-level challenge of GNN

- The dataset is getting larger and larger

# Node	Feature length	# Edge	Total size
500 Million	200 (Int16)	50 Billion	(200 + 400) GB

- Easily exceeds the Server DIMM Capacity



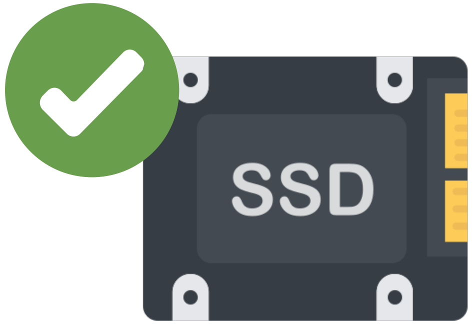
Several hundreds of GB

System-level challenge of GNN

- The dataset is getting larger and larger

# Node	Feature length	# Edge	Total size
500 Million	200 (Int16)	50 Billion	(200 + 400) GB

- But entirely fits into a single Solid-State Drive (SSD)!



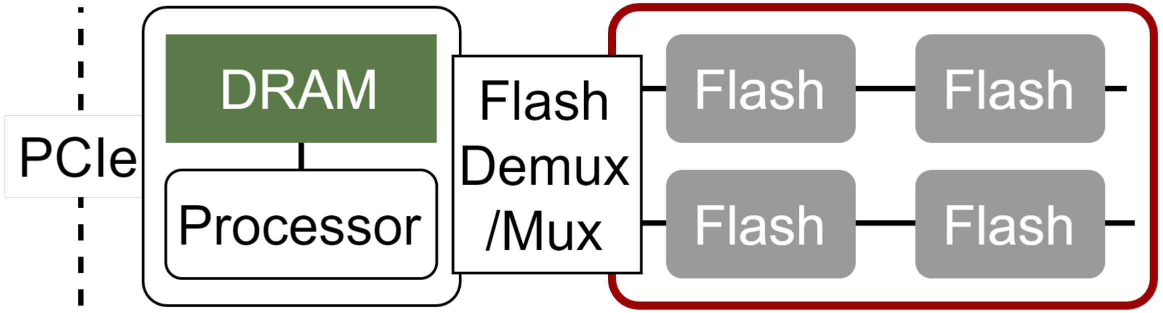
Several TBs

SSD: The way they were

Capacity

N x100 GB

Large number of flash chips



SSD internal architecture (simplified)

SSD: The way they were

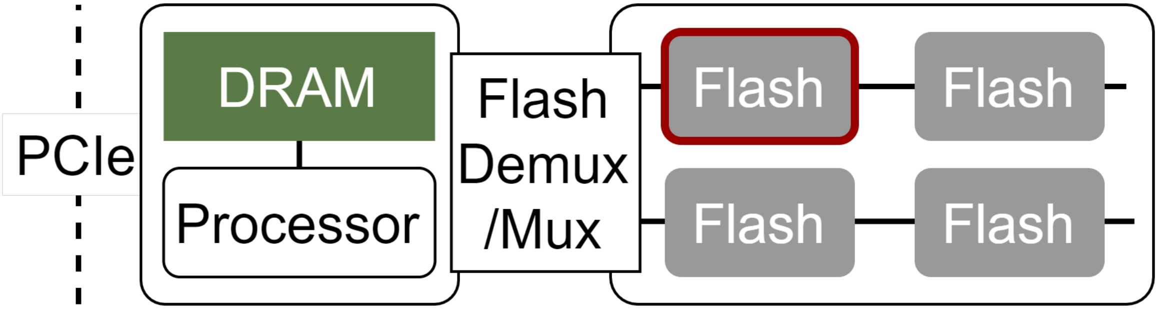
Capacity

N x100 GB

Latency

~40-100 us read

High flash sensing delay



SSD internal architecture (simplified)

SSD: The way they were

Capacity

N x100 GB

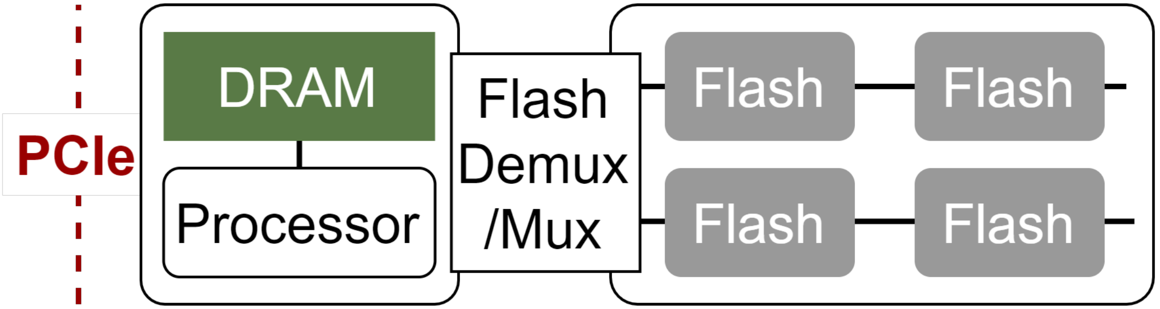
Latency

~40-100 us read

Throughput

< 4 GB/s read

Narrow PCIe 3.0 x4 bandwidth



SSD internal architecture (simplified)

SSD: The way they were

Capacity

N x100 GB

Latency

~40-100 us read

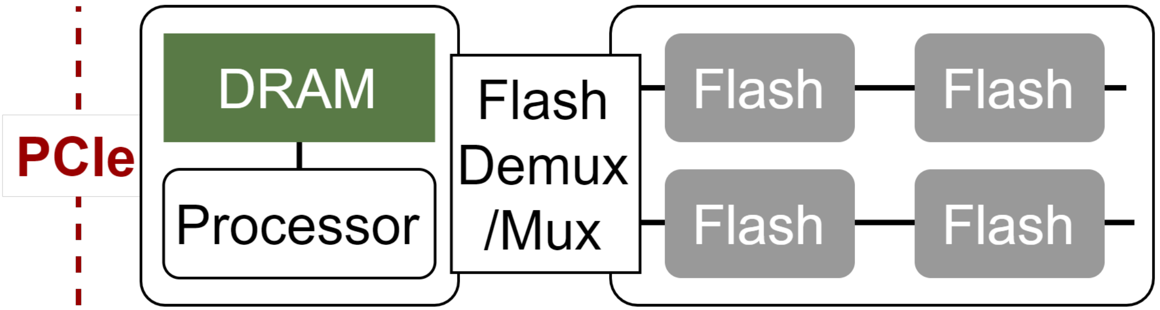
Throughput

<4 GB/s read

Interface

4 KB block

Block granular interface



SSD internal architecture (simplified)

SSD: The way they were

Capacity

N x100 GB

Latency

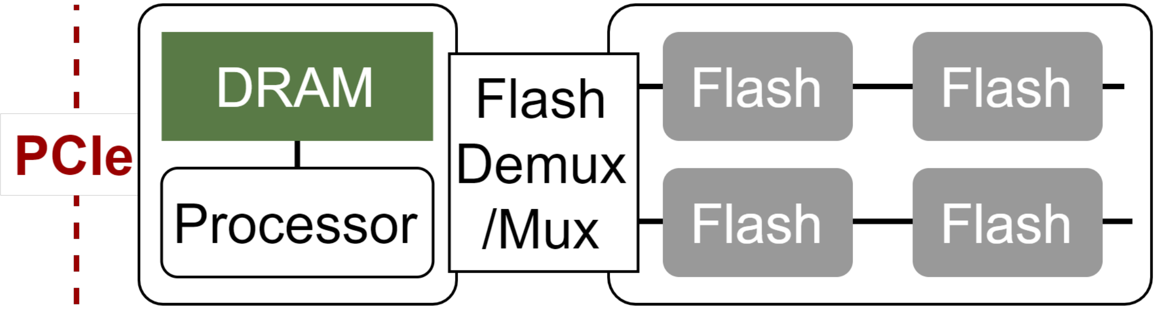
~40-100 us read

Throughput

<4 GB/s read

Interface

4 KB block

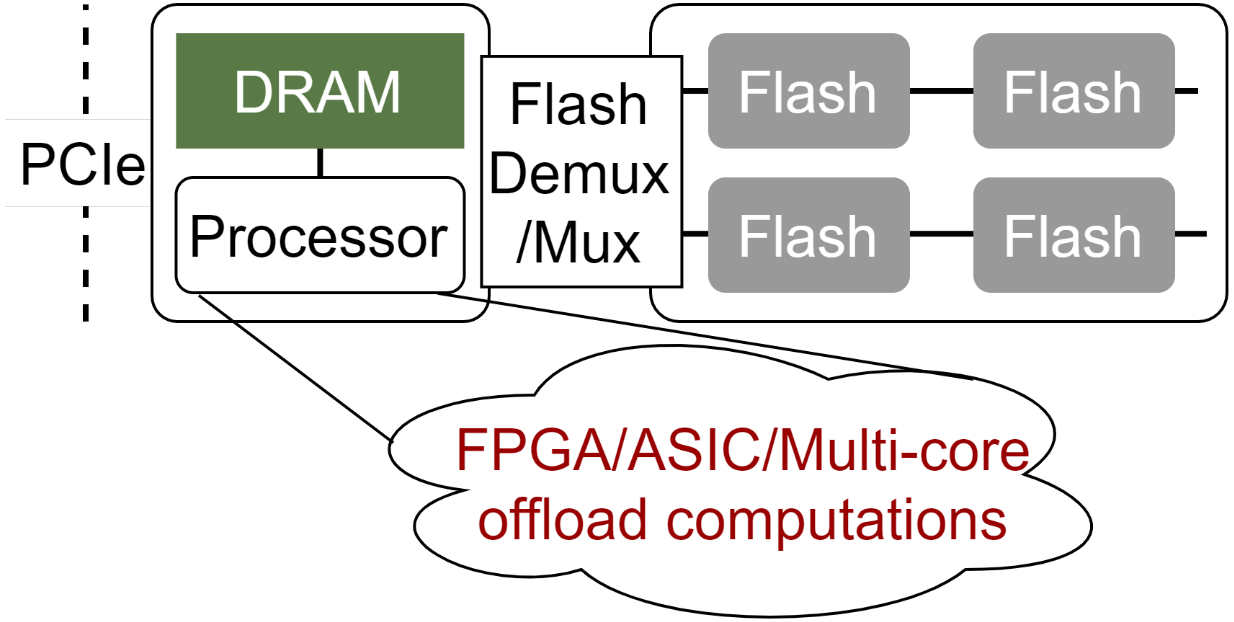


SSD internal architecture (simplified)

Transferring data outside SSD is slow, and causes read amplification!

UCLA

In-storage computing



SSD internal architecture w/ compute units (simplified)

Two types of offloads:

- Early predicate execution
  E.g. Database filter
- Compute in-situation
  E.g. Database aggregate

Both reduces data movement

SSD: The way they are

Slow PCIe interconnect?

- PCIe 4.0: 2GB/s per lane!
- PCIe 5.0: even faster
- Not a convincing motivation any longer

SSD: The way they are

Slow PCIe interconnect?

- PCIe 4.0: 2GB/s per lane!
- PCIe 5.0: even faster
- Not a convincing motivation any longer

Flash are high latency media?

- Ultra-low latency Z-SSD (3 μs flash read)
- More pressure to host storage stack (~ 10 us)

SSD: The way they are

Slow PCIe interconnect?

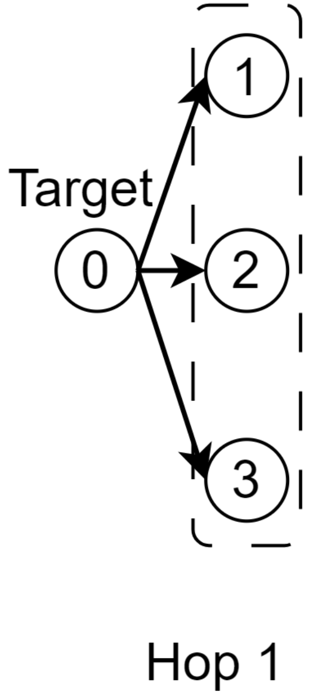
- PCIe 4.0: 2GB/s per lane!
- PCIe 5.0: even faster
- Not a convincing motivation any longer

Flash are high latency media?

- Ultra-low latency Z-SSD (3 µs flash read)
- More pressure to host storage stack (~ 10 us)

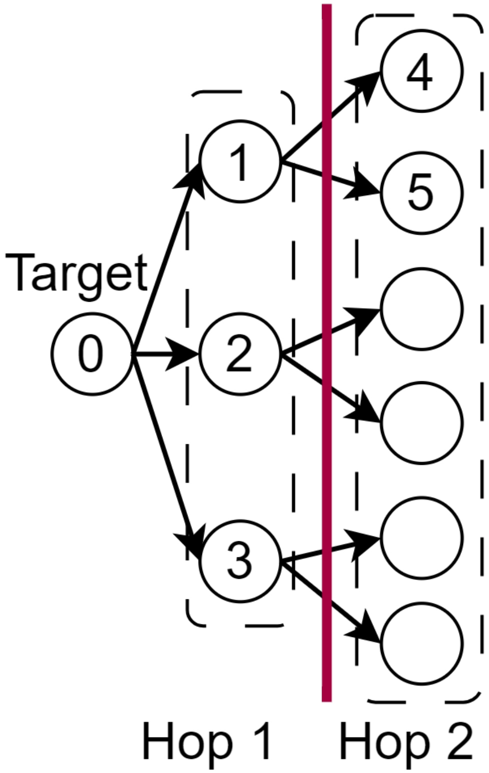
Technology shifts bring new challenges and opportunities!

Challenge 1: host-SSD communication



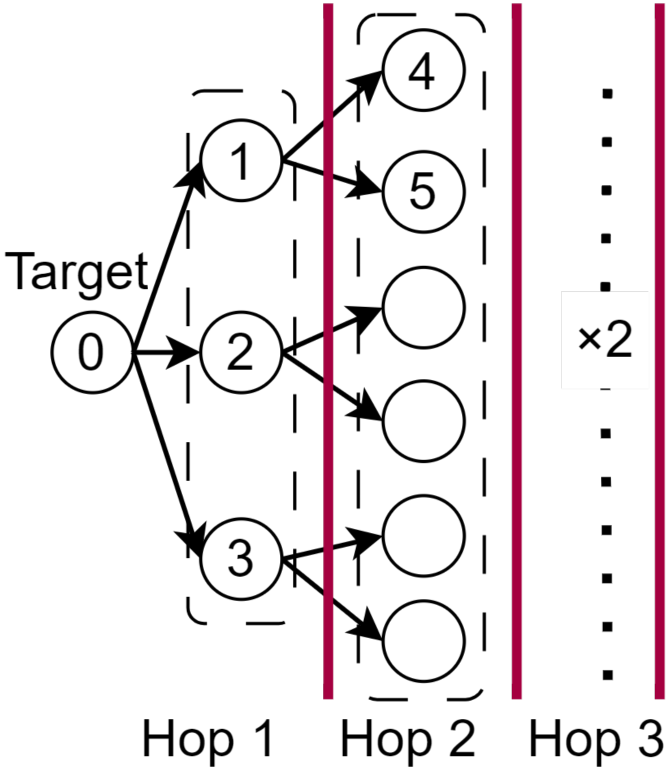
GNN subgraph generation:
iterations of node sampling

Challenge 1: host-SSD communication



GNN subgraph generation:
iterations of node sampling

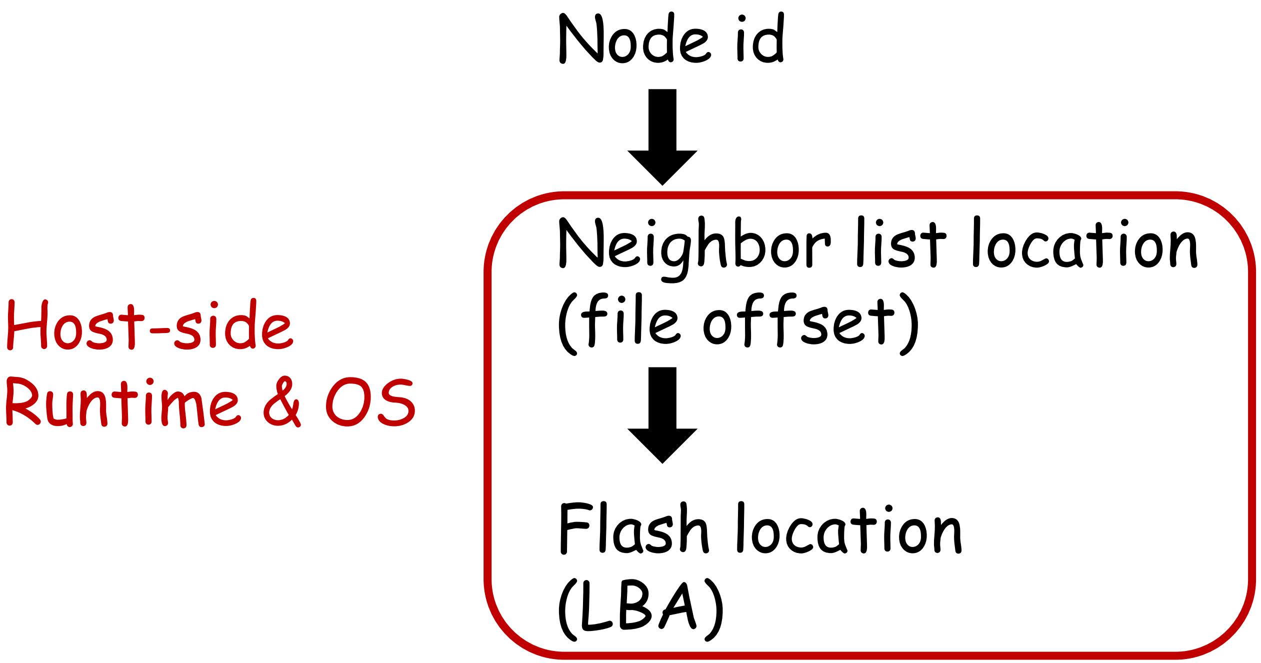
Challenge 1: host-SSD communication



GNN subgraph generation:
iterations of node sampling

Challenge 1: host-SSD communication

To sample a new hop: need the host to locate



Node id

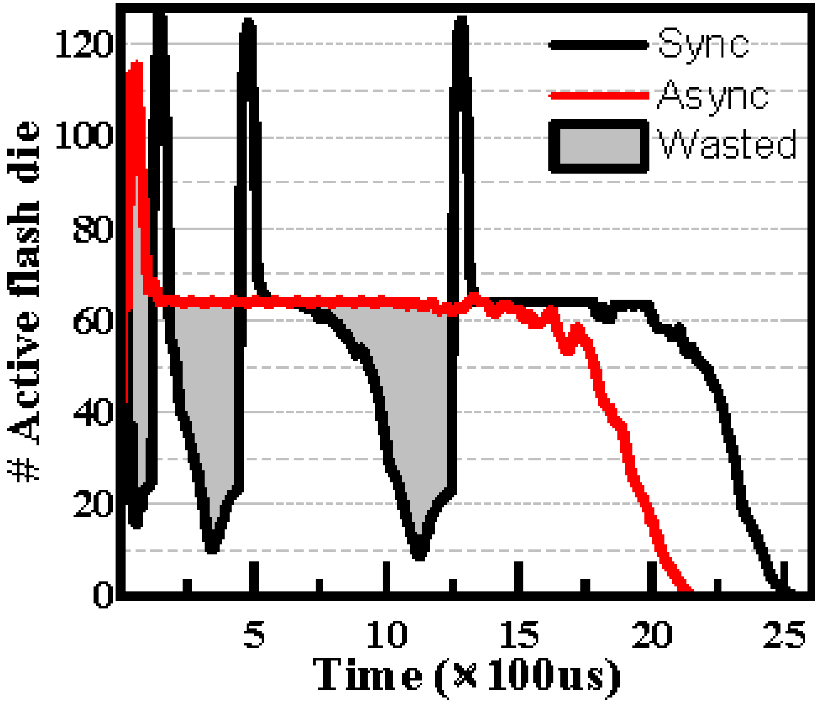
Host-side
Runtime & OS

Neighbor list location
(file offset)

Flash location
(LBA)

Challenge 1: host-SSD communication

- Resubmission requests traverse the whole OS stack
- Layer batch amortizes communication, but brings barriers

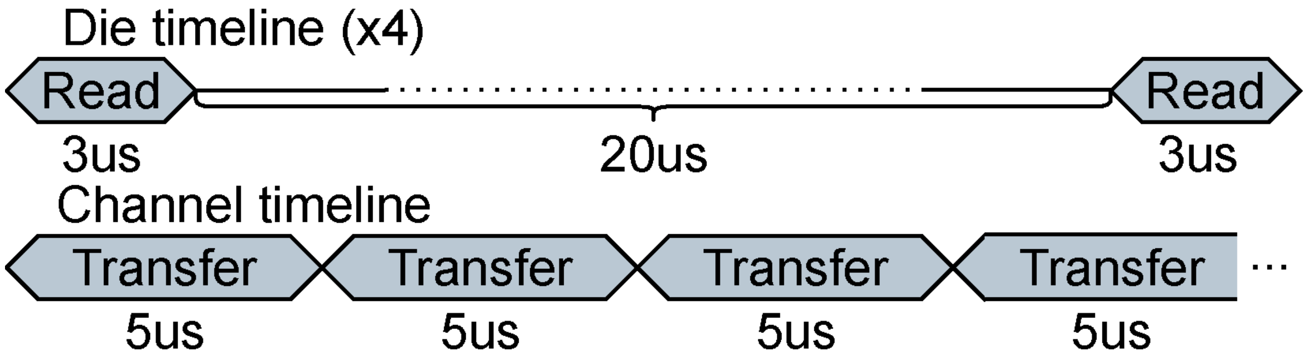


Challenge 2: SSD channel amplification

- Flash sense time: 3 µs
- Channel transfer rate: 800 MT/s

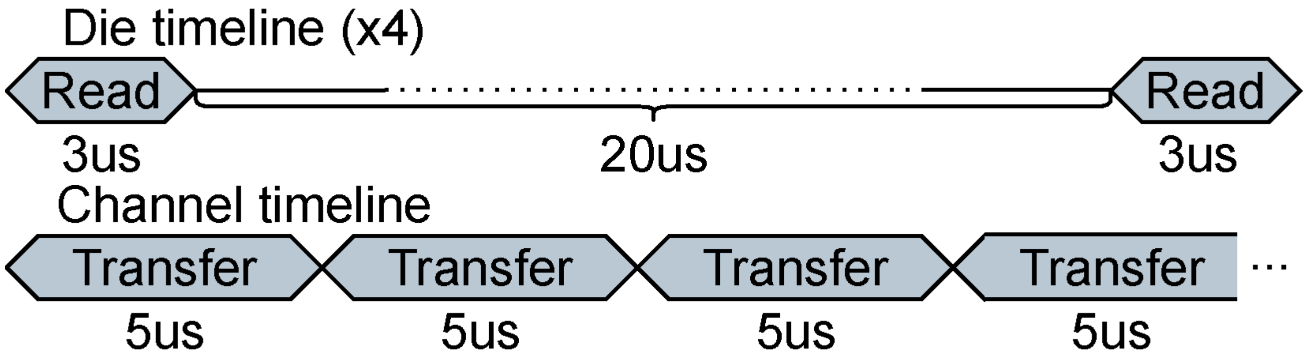
Challenge 2: SSD channel amplification

- Flash sense time: 3 µs
- Channel transfer rate: 800 MT/s



Challenge 2: SSD channel amplification

- Flash sense time: 3 µs
- Channel transfer rate: 800 MT/s

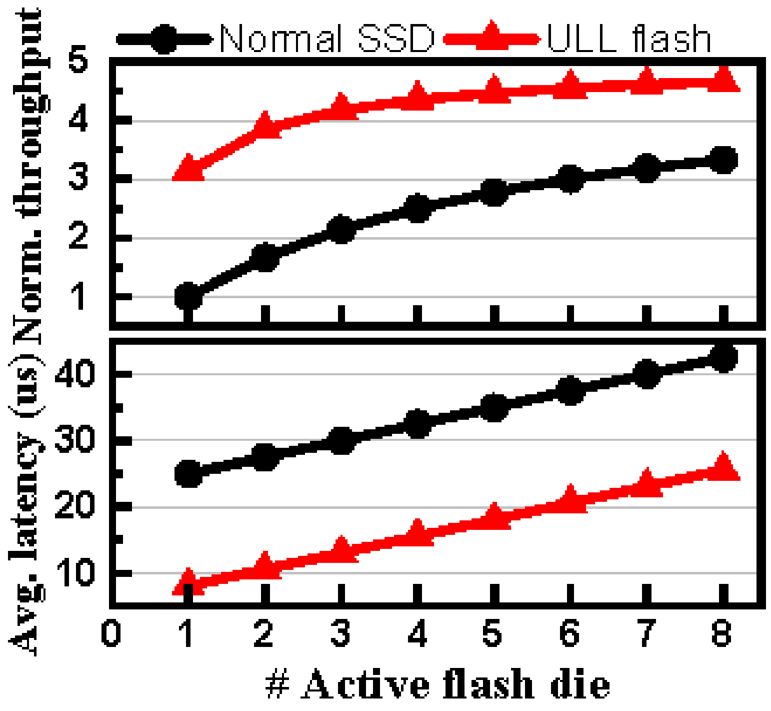


Die timeline (x4)

Read 3us — 20us — Read 3us

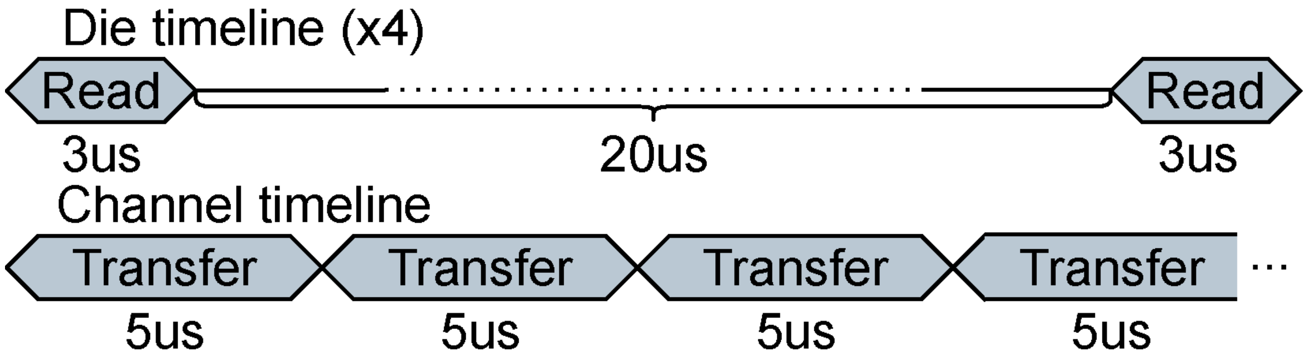
Channel timeline

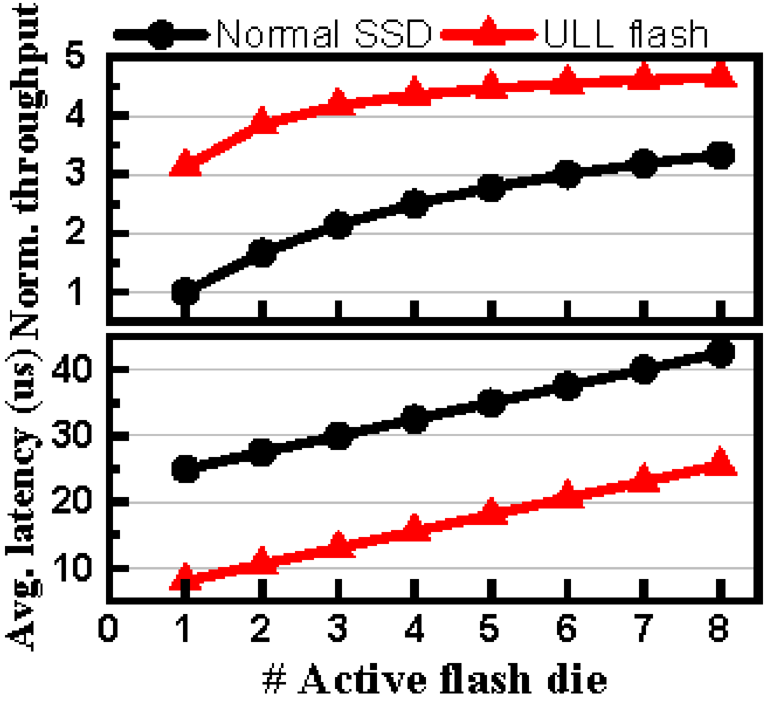
Transfer 5us | Transfer 5us | Transfer 5us | Transfer 5us …



Challenge 2: SSD channel amplification

- Flash sense time: 3 µs
- Channel transfer rate: 800 MT/s



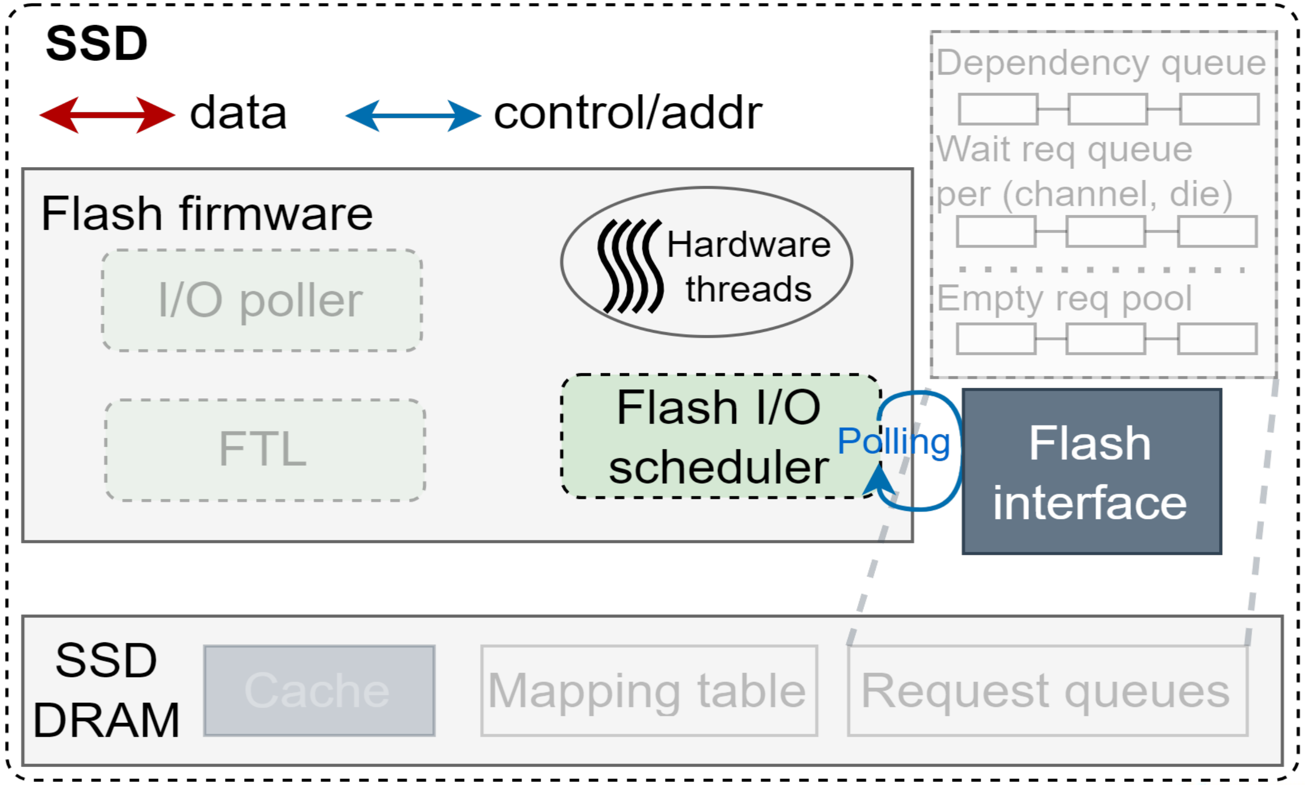


Flash dies are underutilized
Flash channels transfer useless data

→ Limited improvement

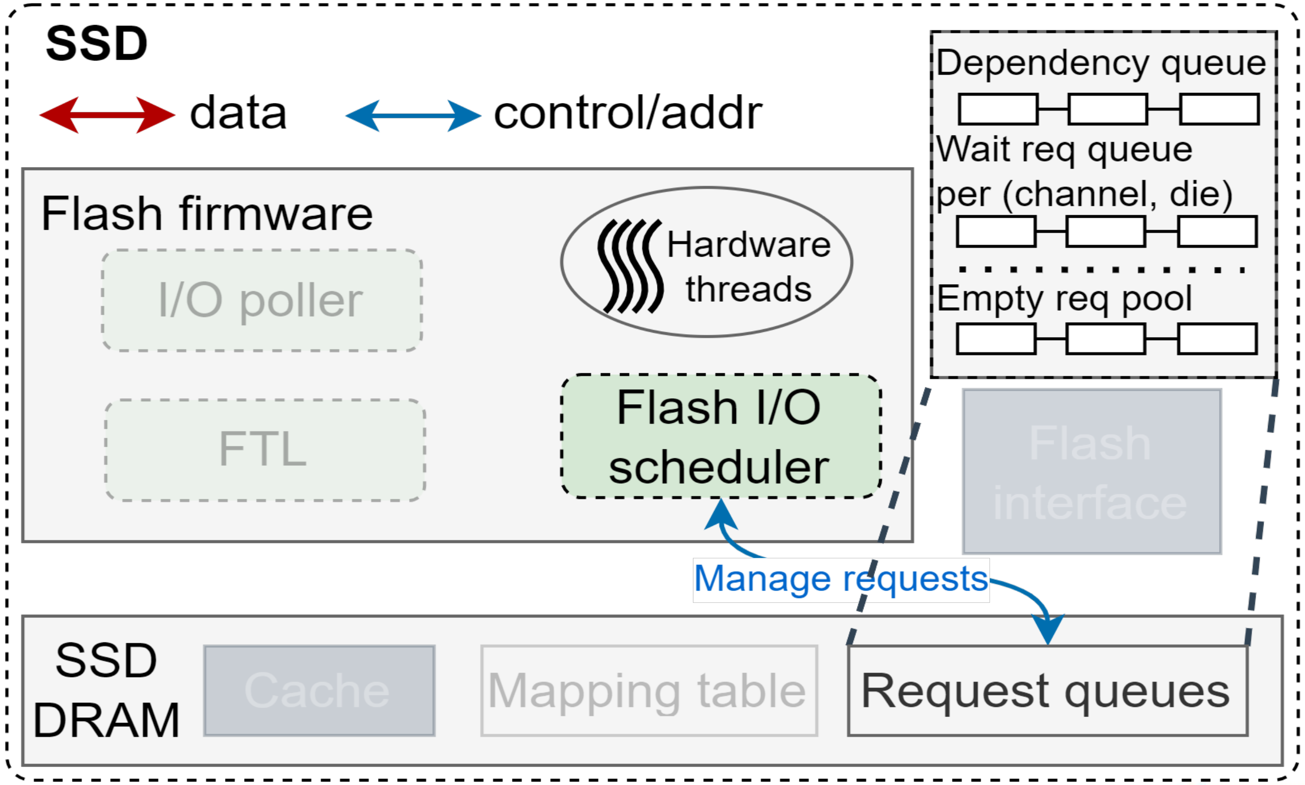
Challenge 3: Firmware-based backend I/O

- Scheduler polls I/O completion



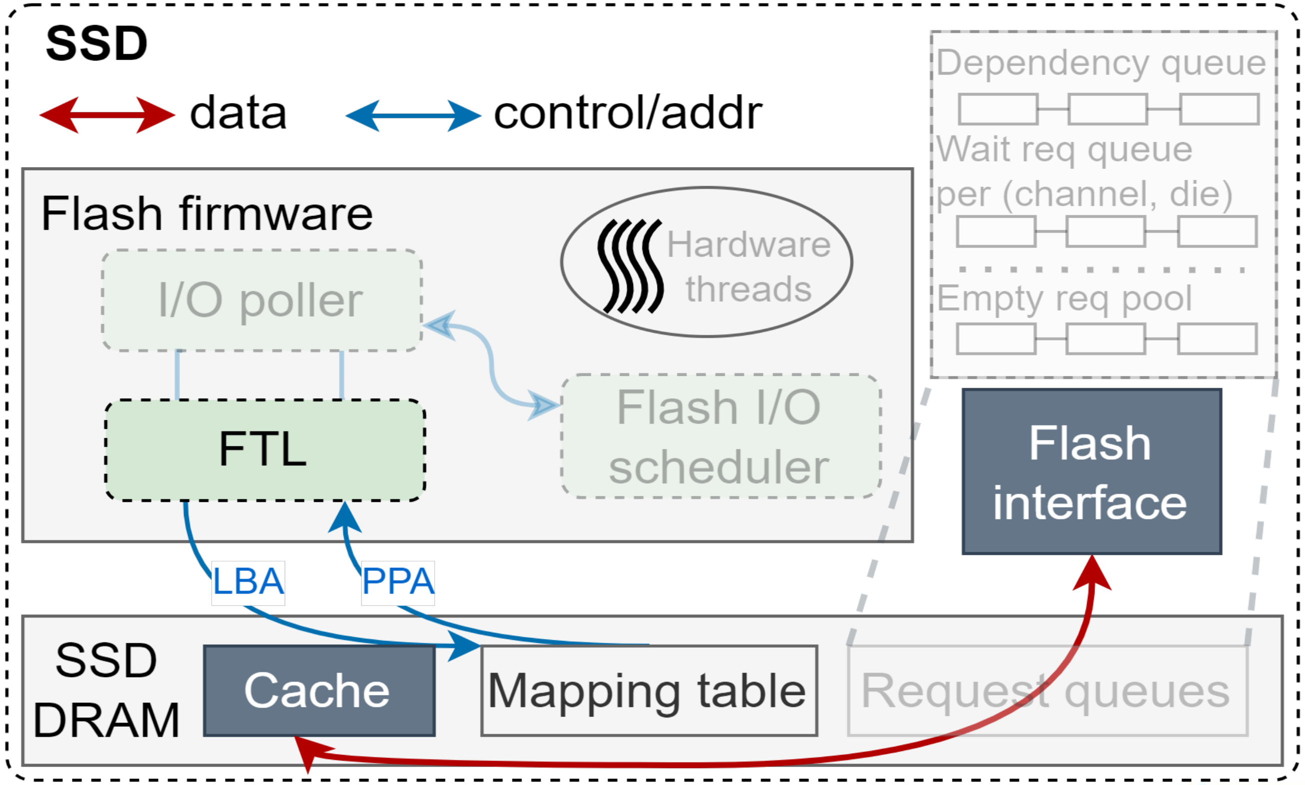
Challenge 3: Firmware-based backend I/O

- Scheduler polls I/O completion
- Manage request



Challenge 3: Firmware-based backend I/O

- Scheduler polls I/O completion
- Manage request
- Locate next request address

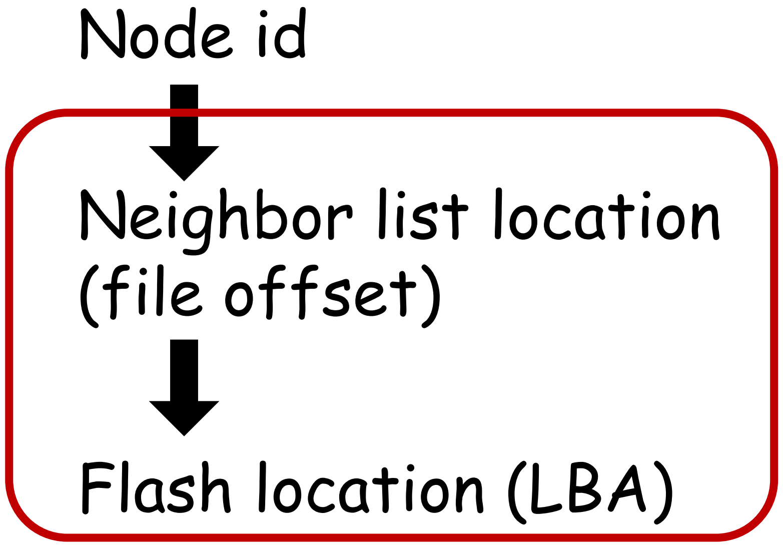


Challenge 3: Firmware-based backend I/O

Controller has 1-4 cores, while backend flash has about 100 dies in active

Huge mismatch!

Optimization 1: Address translation fusion



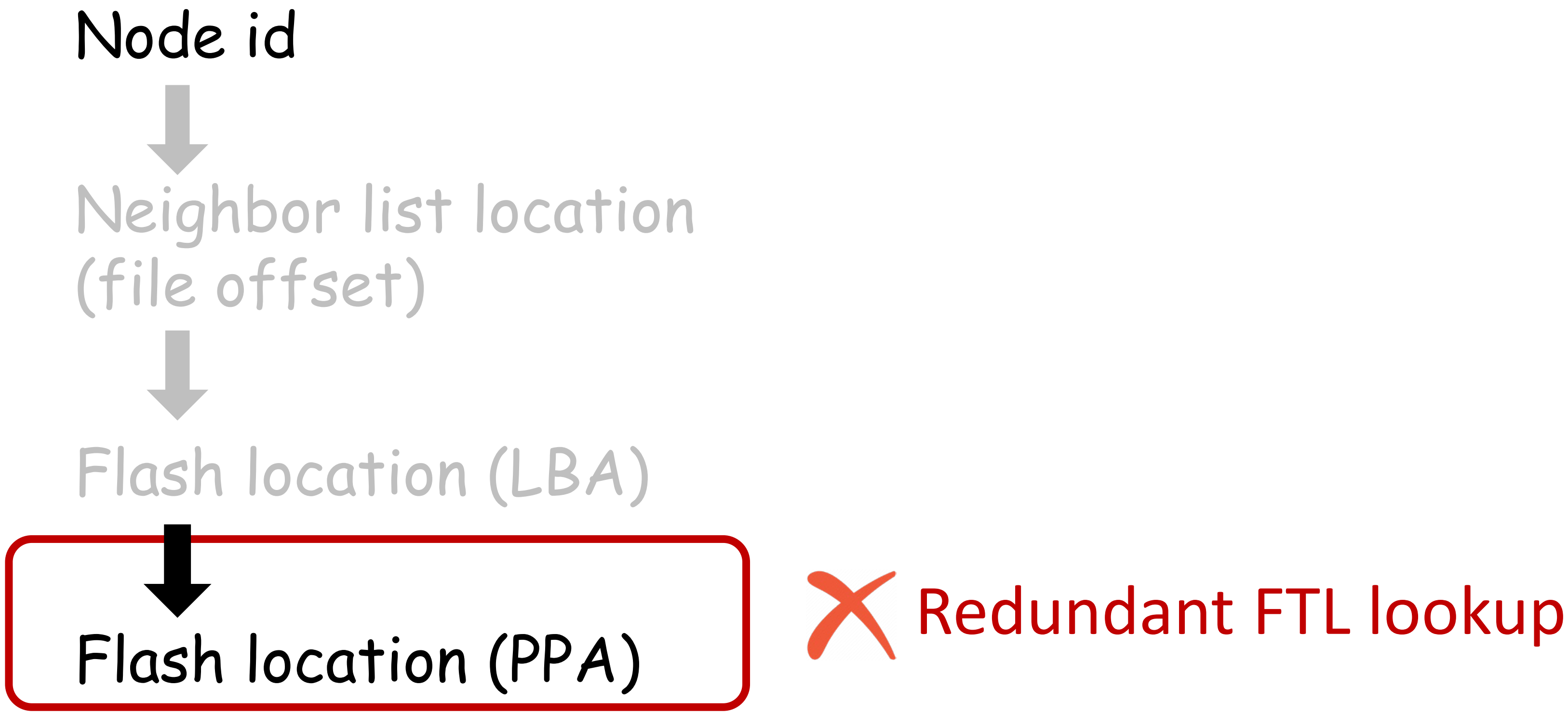
Node id

Neighbor list location (file offset)

Flash location (LBA)

✗ Cross host-SSD resubmit

Optimization 1: Address translation fusion



Node id

Neighbor list location (file offset)

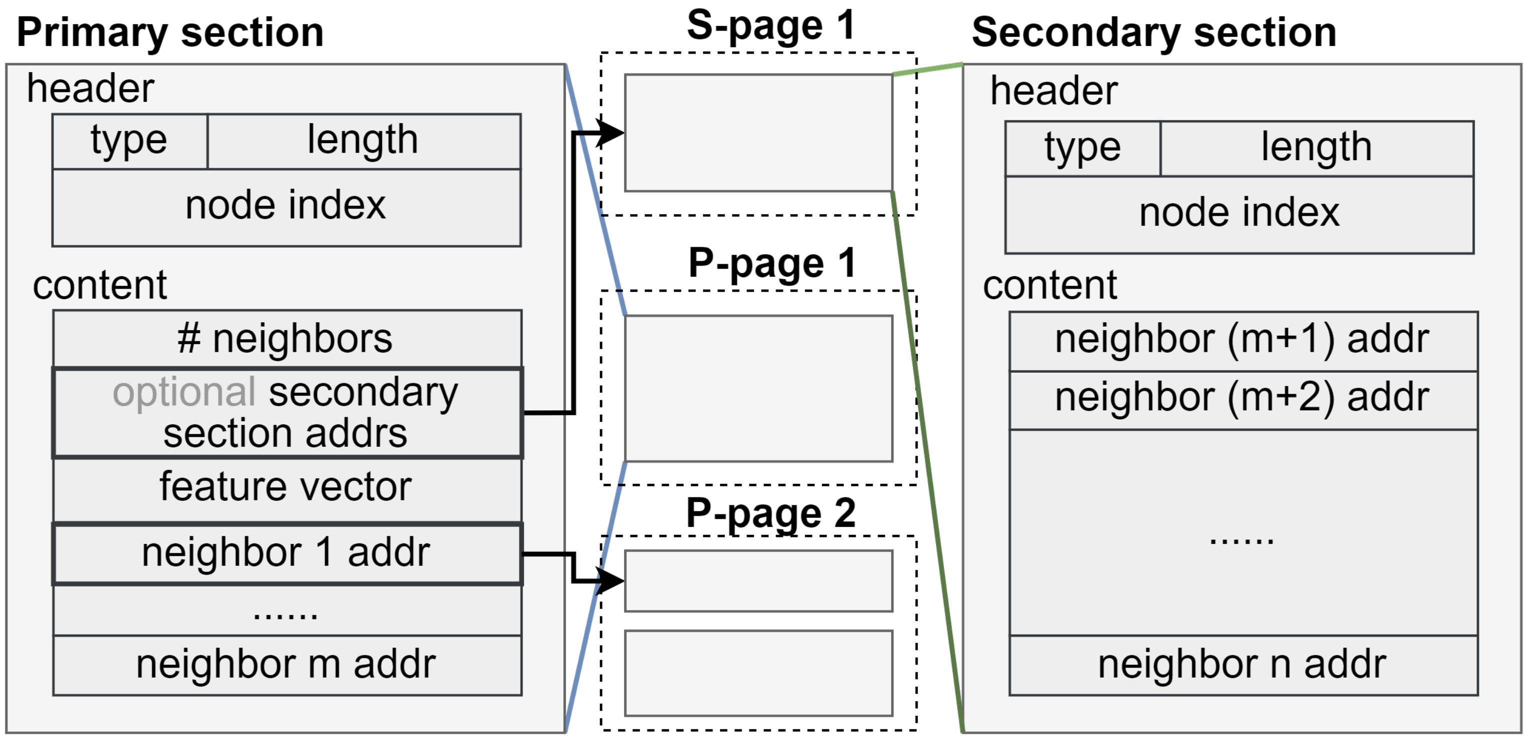
Flash location (LBA)

Flash location (PPA)

✗ Redundant FTL lookup

Optimization 1: Address translation fusion

Static graph with address mapping stored in flash

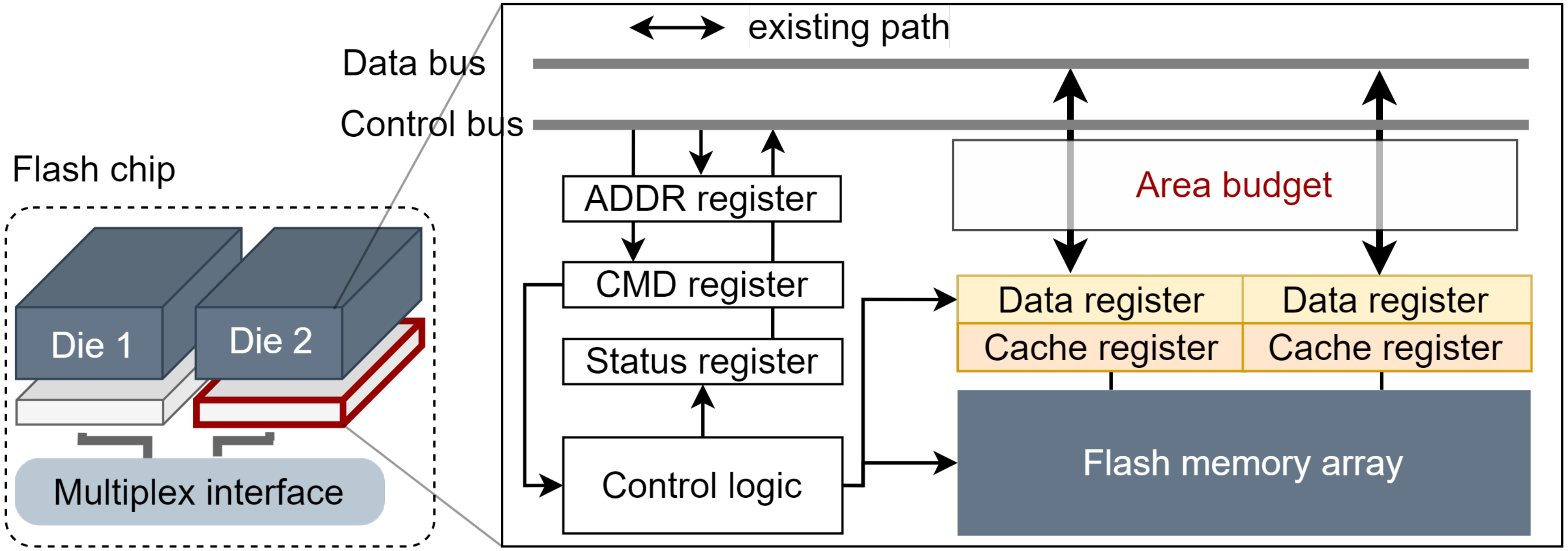


DirectGraph format

Optimization 2: In-flash sampling

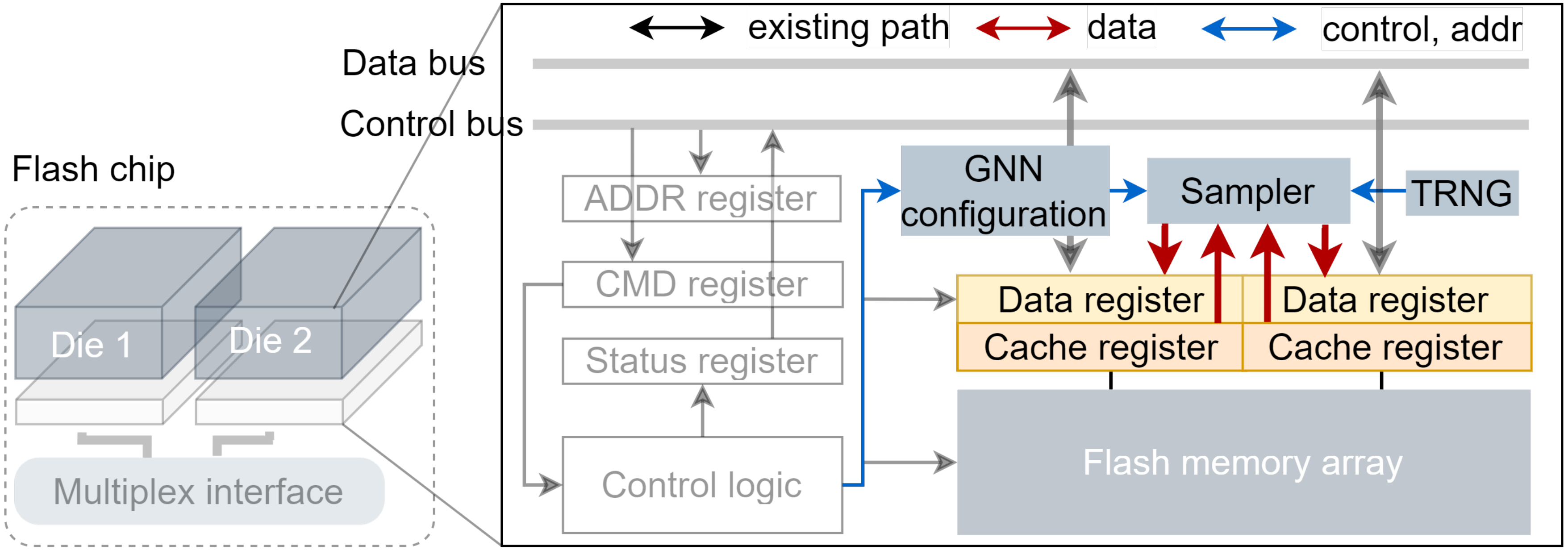
Flash dies area budget

- Add more control logic (offload sampling & vector retrieving)



Optimization 2: In-flash sampling

FSM to sample node features, generate resubmit request



Optimization 2: In-flash sampling

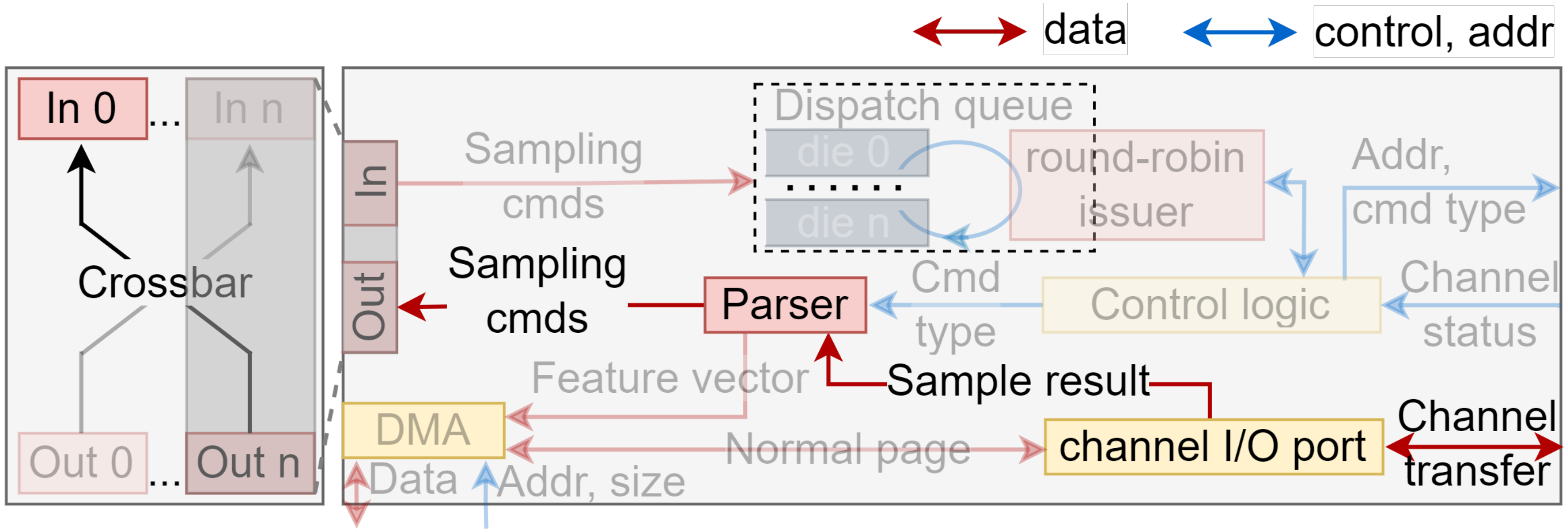
FSM to sample node features, generate resubmit request

Example task: primary section $\xrightarrow{\text{sample}}$5 nodes

Get: 3 nodes (primary section sample request), 1 resubmit request to sample 2 nodes from a secondary section

Optimization 3: Hardware-based resubmission

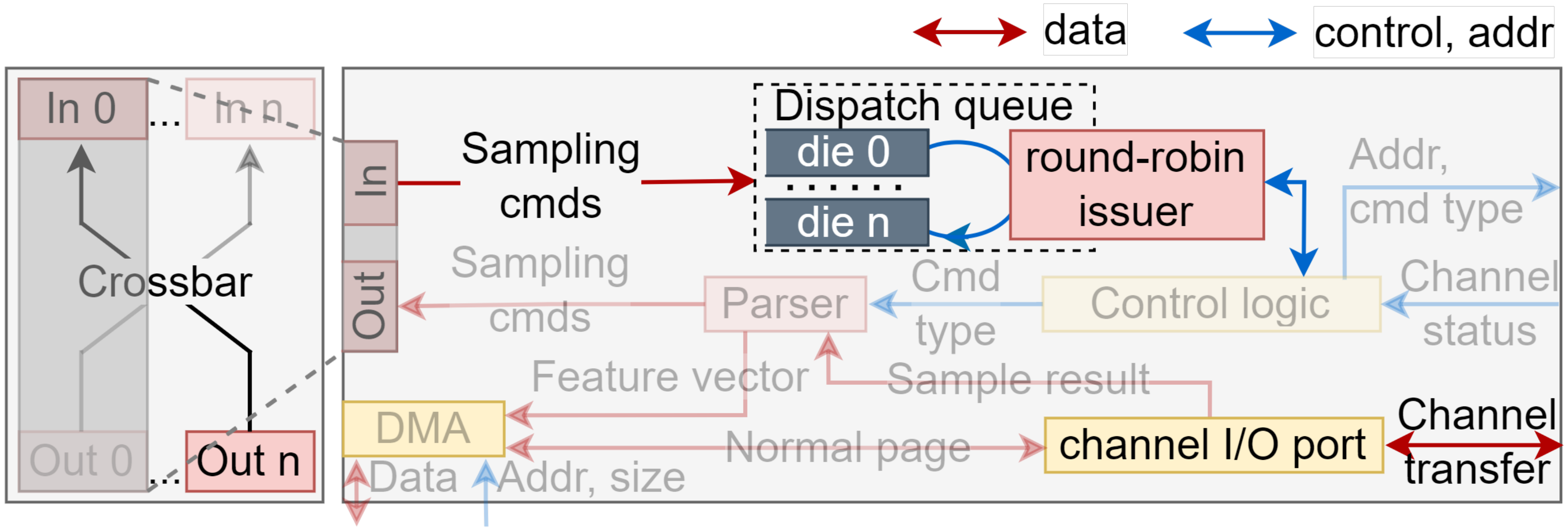
Route commands between channels (n → 0)



Router at channel N

Optimization 3: Hardware-based resubmission

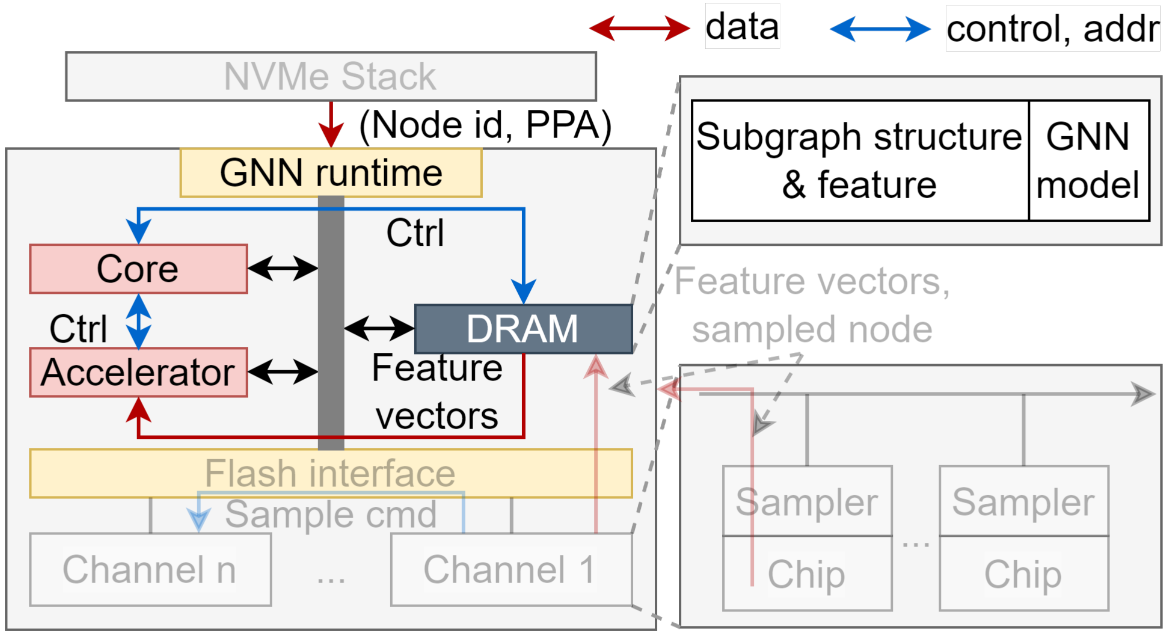
Route commands between channels (n → 0)



Router at channel 0

Overall architecture

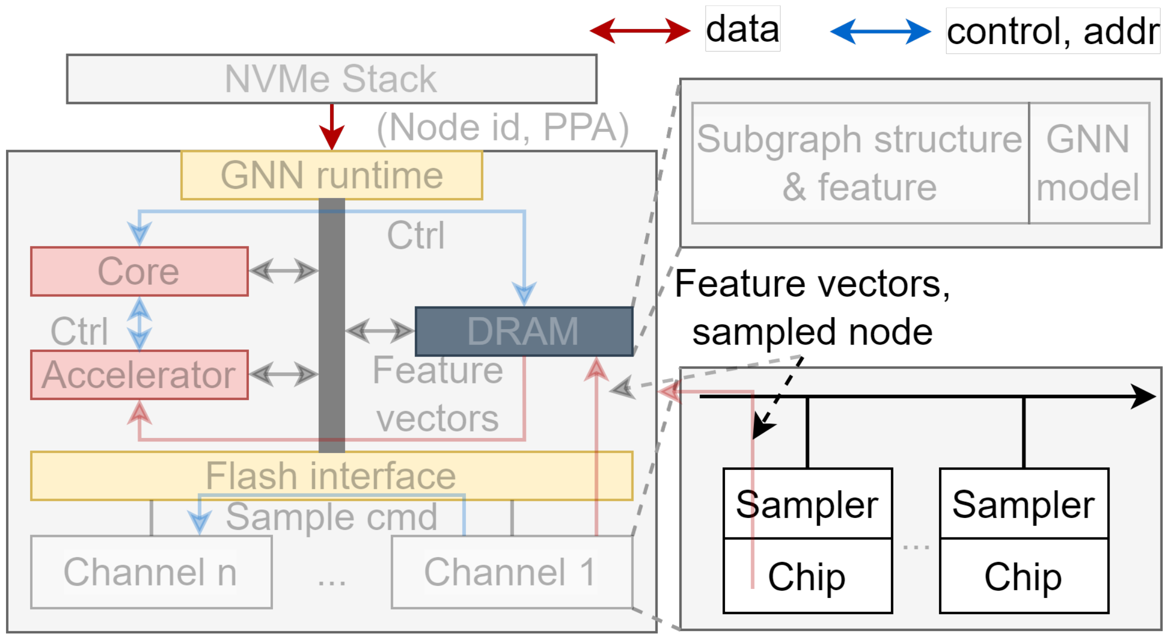
- GNN runtime
  - Interact w/ host
  - Submit flash request
  - Schedule DNN execution



Overall architecture

Overall architecture

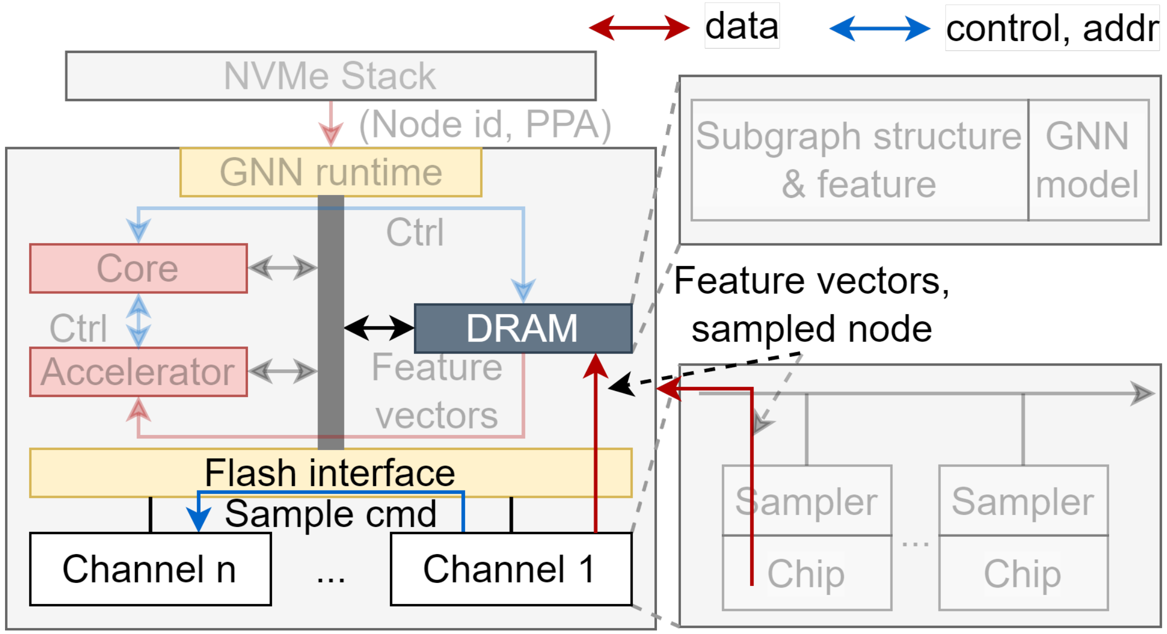
- GNN runtime
  - Interact w/ host
  - Submit flash request
  - Schedule DNN execution
- Flash die
  - Sample/Retrieval
  - Generate new requests



Overall architecture

Overall architecture

- GNN runtime
  - Interact w/ host
  - Submit flash request
  - Schedule DNN execution
- Flash die
  - Sample/Retrieval
  - Generate new requests
- Flash interface
  - Route resubmit commands



Overall architecture

Overall architecture

- GNN runtime
  - Interact w/ host
  - Submit flash request
  - Schedule DNN execution
- Flash die
  - Sample/Retrieval
  - Generate new requests
- Flash interface
  - Route resubmit commands

Hardware-based request resubmission

Evaluation

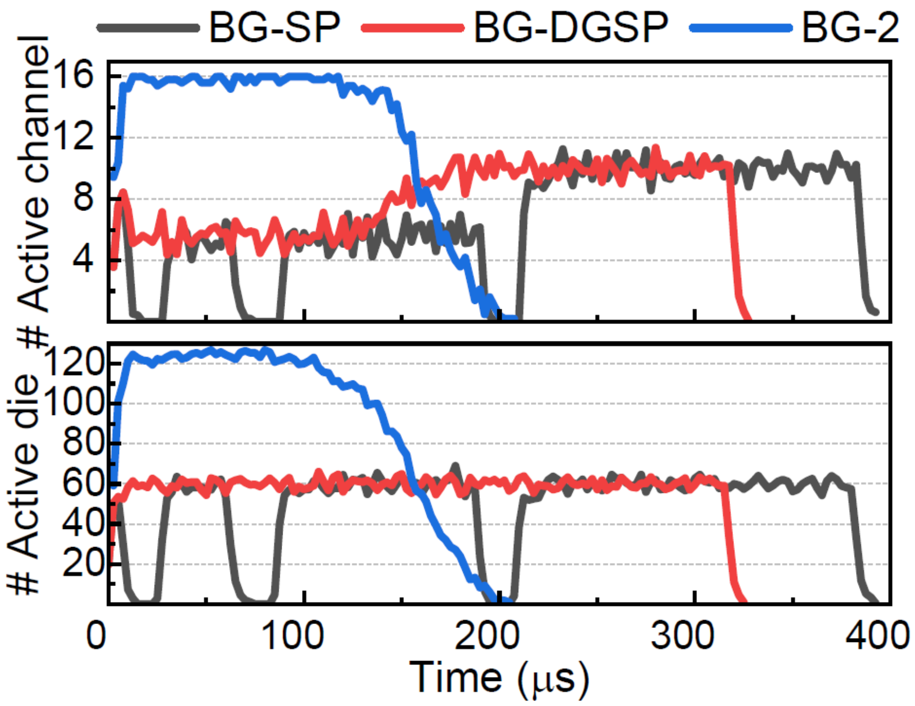
CC	CPU-centric architecture, with PCIe Accelerator 128x128 systolic array, 32 MB SRAM, 1 GHz
BG-1	Basic in-storage computing architecture
BG-DG	*BG-1* with DirectGraph GNN format
BG-SP	*BG-1* with in-flash node sampling and vector retrieving
BG-2	*BG-DGSP* with inter-channel hw-based command resubmission

Simulated platforms

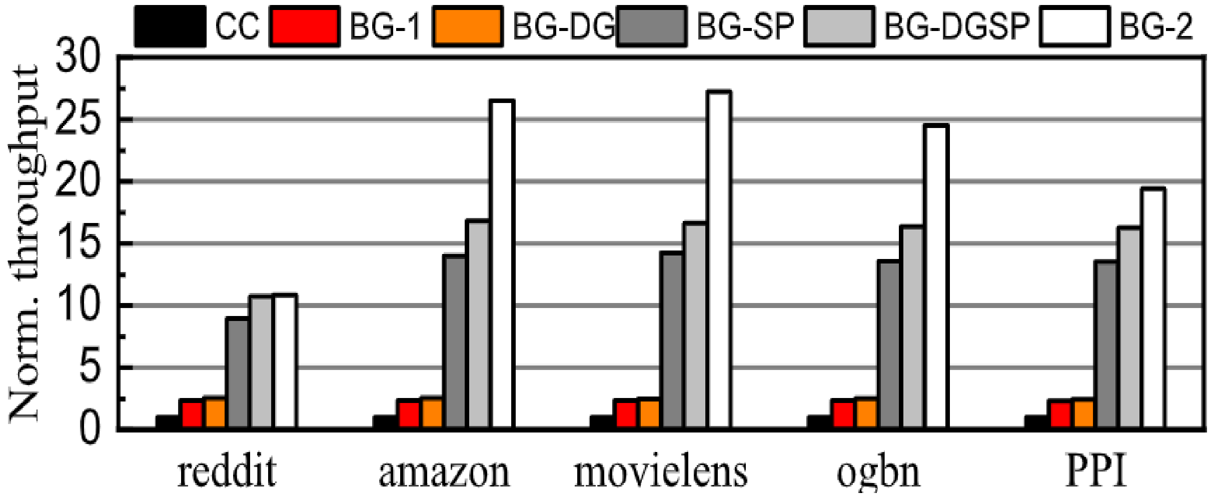
Interface	NVMe, PCIe 4.0 x4
Controller	4 ARM Cortex-A9 Cores
DRAM	DDR4-3200, 25.6 GB/s, 1 GB
Flash	16 Channel, 8 Die/Channel, 4 KB Page 3 us read, 800 MB/s channel transfer
ISC Accelerator	ISC: 64x64 systolic array 6 MB SRAM, 800 MHz

Default SSD configuration

Evaluation

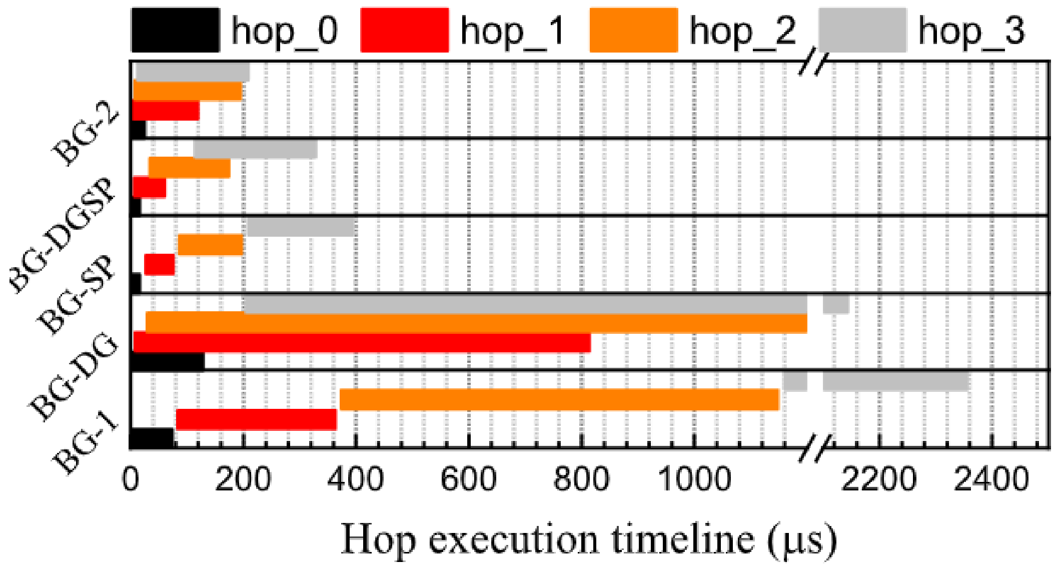


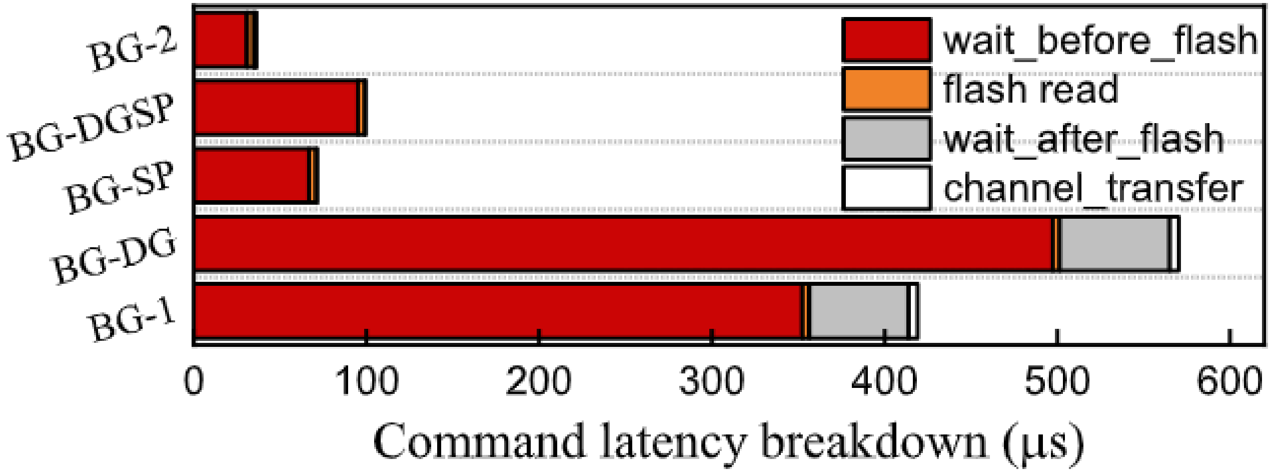
Flash utilization for *Amazon* (Nodes 265.9M, Avg. Degree 300, Feature 200)



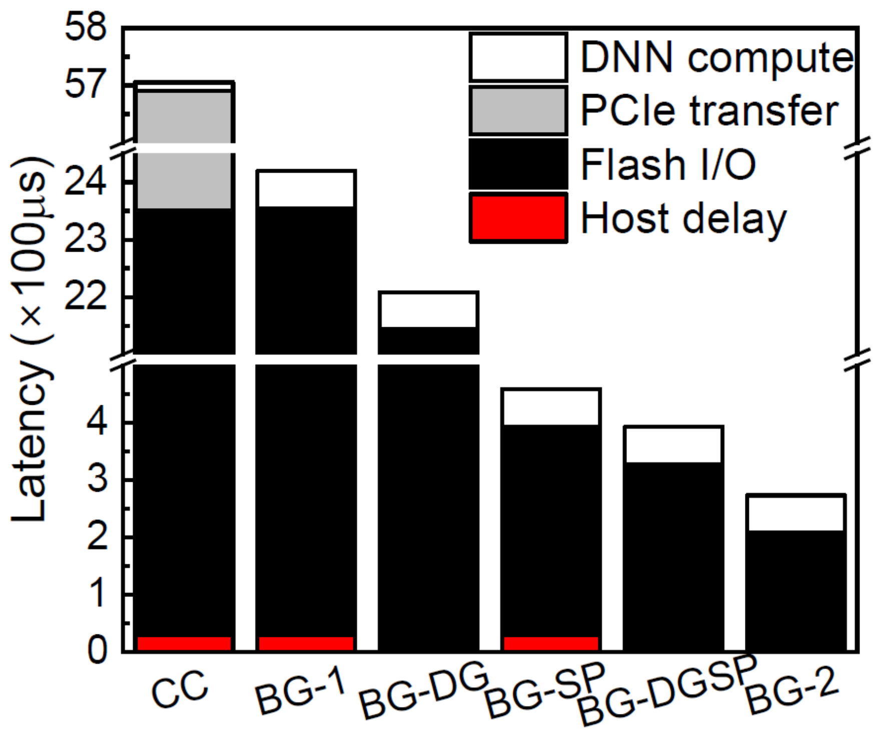
Throughput on five large-scale GNN dataset

Evaluation

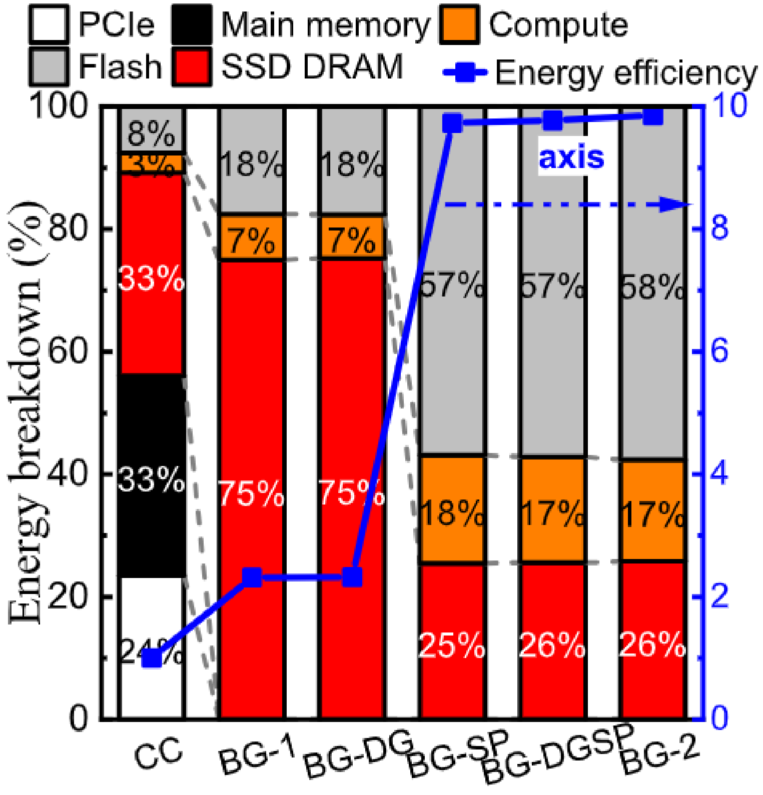

hop_0
hop_1
hop_2
hop_3
BG-2
BG-DGSP
BG-SP
BG-DG
BG-1
0
200
400
600
800
1000
2200
2400
Hop execution timeline (µs)


wait_before_flash
flash read
wait_after_flash
channel_transfer
BG-2
BG-DGSP
BG-SP
BG-DG
BG-1
0
100
200
300
400
500
600
Command latency breakdown (µs)

Evaluation



Latency breakdown on *amazon* dataset



Energy breakdown on *amazon* dataset

Takeaway

- Technical shifts, from both device and interconnect, break tradition of ISC design
- Control & Data path of traditional I/O can be a new bottleneck
- Automating such paths with hardware can offer huge performance benefit