NVMW'22

Hardware/Software Co-Programmable Framework for Computational SSDs to Accelerate Deep Learning Service on Large-Scale Graphs

Miryeong Kwon, Donghyun Gouk, Sangwon Lee, Myoungsoo Jung

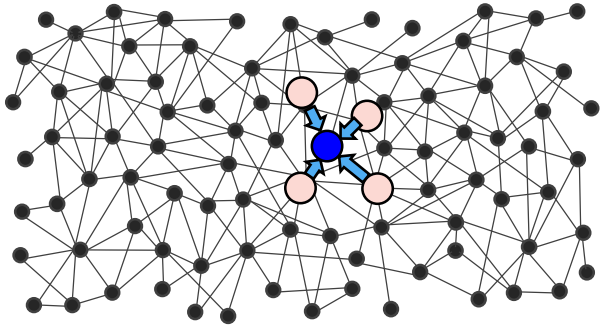
Computer Architecture and Memory systems Laboratory

KAIST CAMELab

First Step

High-level summary of talk

GNN have shown great success



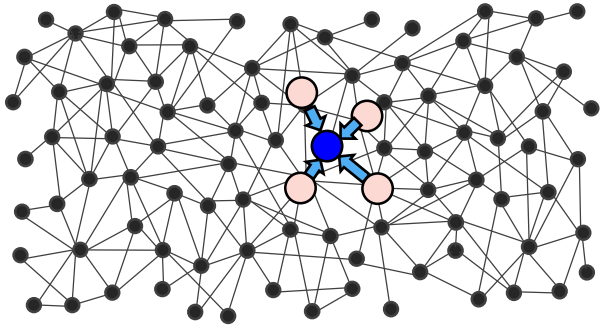
High accuracy

Well accelerated

First Step

High-level summary of talk

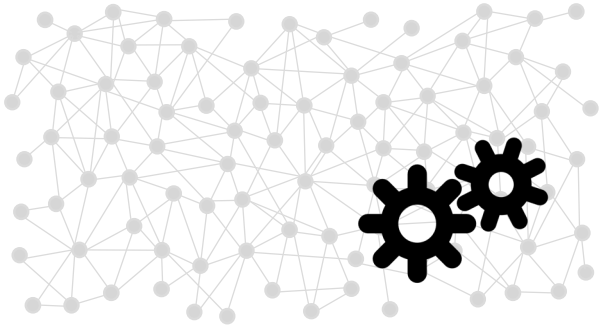
GNN have shown great success



High accuracy

Well accelerated

GNN preprocessing is missed out on

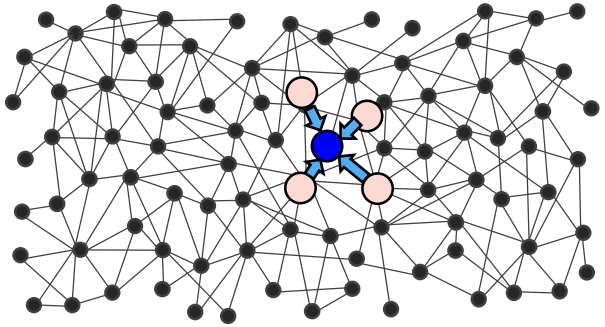


Current GNN works are only focusing on GNN algorithms

First Step

High-level summary of talk

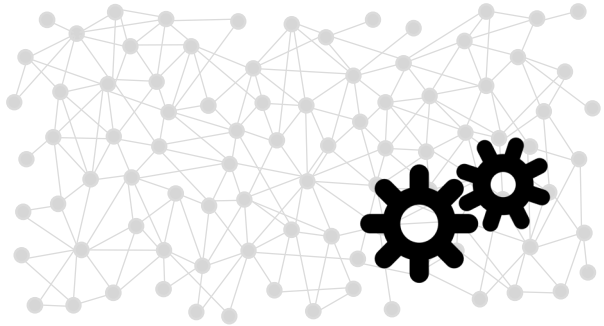
GNN have shown great success



High accuracy

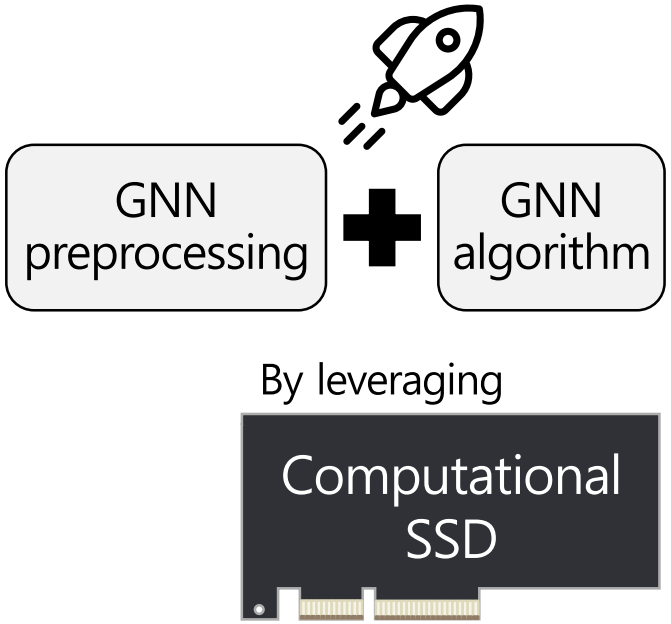
Well accelerated

GNN preprocessing is missed out on



Current GNN works are only focusing on GNN algorithms

Now, we need "HolisticGNN"



1. Background

2. Motivation and Design Considerations

3. Overview of HolisticGNN Framework

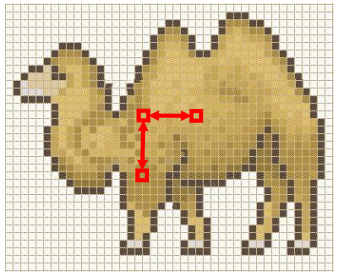
4. Details of HolisticGNN Components

5. Evaluation

Graph Neural Networks (GNN)

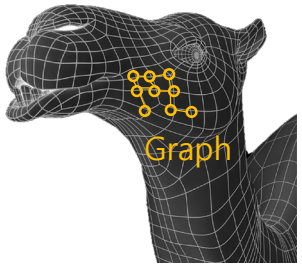
Why is it emerging?

Conventional CNN Model



Regular data in Euclidean space
(Learning information: "Euclidean distance")

Emerging GNN Model



Irregular data in non-Euclidean space
(Learning information: "Relationship")

Response of CNN model

"Women near the sofa"

Query image

Characteristic: "pain"

Response of GNN model

"pain"

Image source: Personalized Image Retrieval with Sparse Graph Representation Learning (KDD'20)

Graph Neural Networks (GNN)

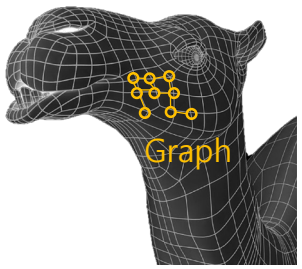
Why is it emerging?

Conventional CNN Model

Regular data in Euclidean space
(Learning information: "Euclidean distance")

How can GNN algorithm learn the relationship?

Emerging GNN Model



Irregular data in non-Euclidean space
(Learning information: "Relationship")

Response of CNN model

"Women near the sofa"

Query image

Characteristic: "pain"

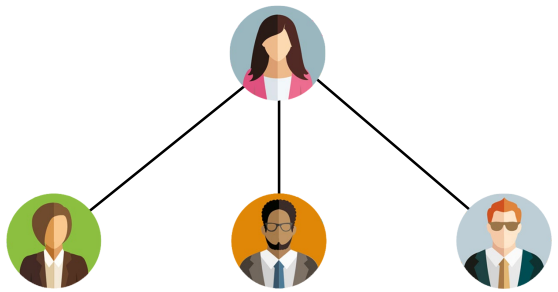
Response of GNN model

"pain"

Graph Neural Networks (GNN)

GNN algorithm

Input



Graph structure

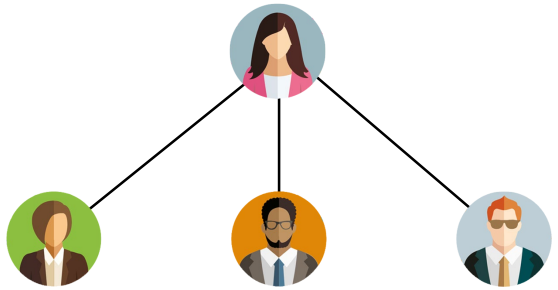
0.1	0.8	1	0.2	0	1	0.8	0.7	1
0	1	0.1	1	0.8	0.1	1	0.2	0
0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4
0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

Node embedding

Graph Neural Networks (GNN)

GNN algorithm

Input

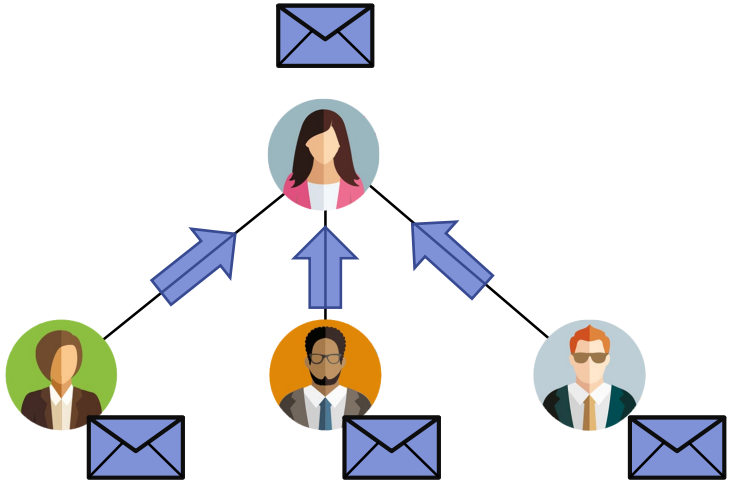


Graph structure

0.1	0.8	1	0.2	0	1	0.8	0.7	1
0	1	0.1	1	0.8	0.1	1	0.2	0
0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4
0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

Node embedding

#1: Aggregation



0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

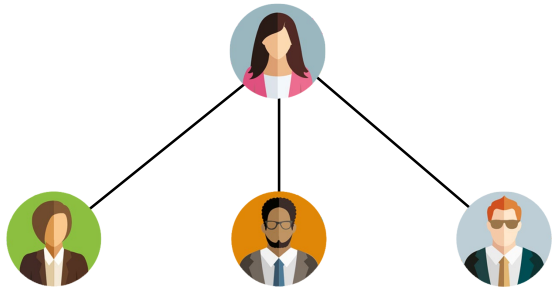
0.1	0.8	1	0.2	0	1	0.8	0.7	1

0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4

Graph Neural Networks (GNN)

GNN algorithm

Input

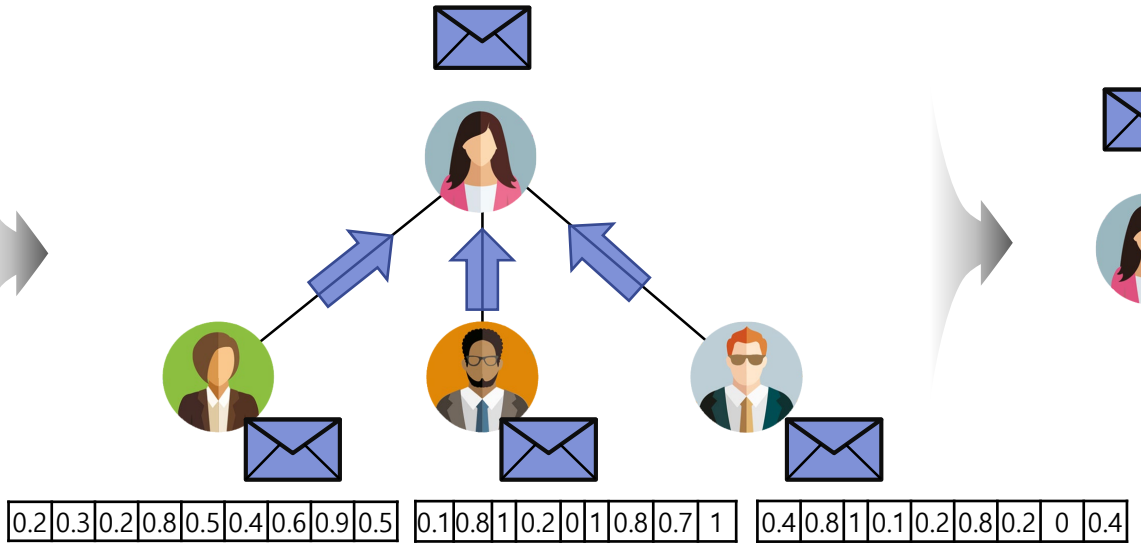


Graph structure

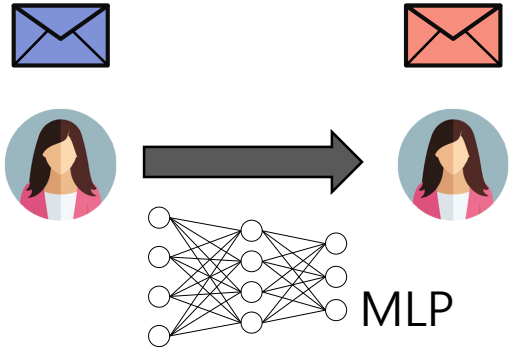
0.1	0.8	1	0.2	0	1	0.8	0.7	1
0	1	0.1	1	0.8	0.1	1	0.2	0
0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4
0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

Node embedding

#1: Aggregation

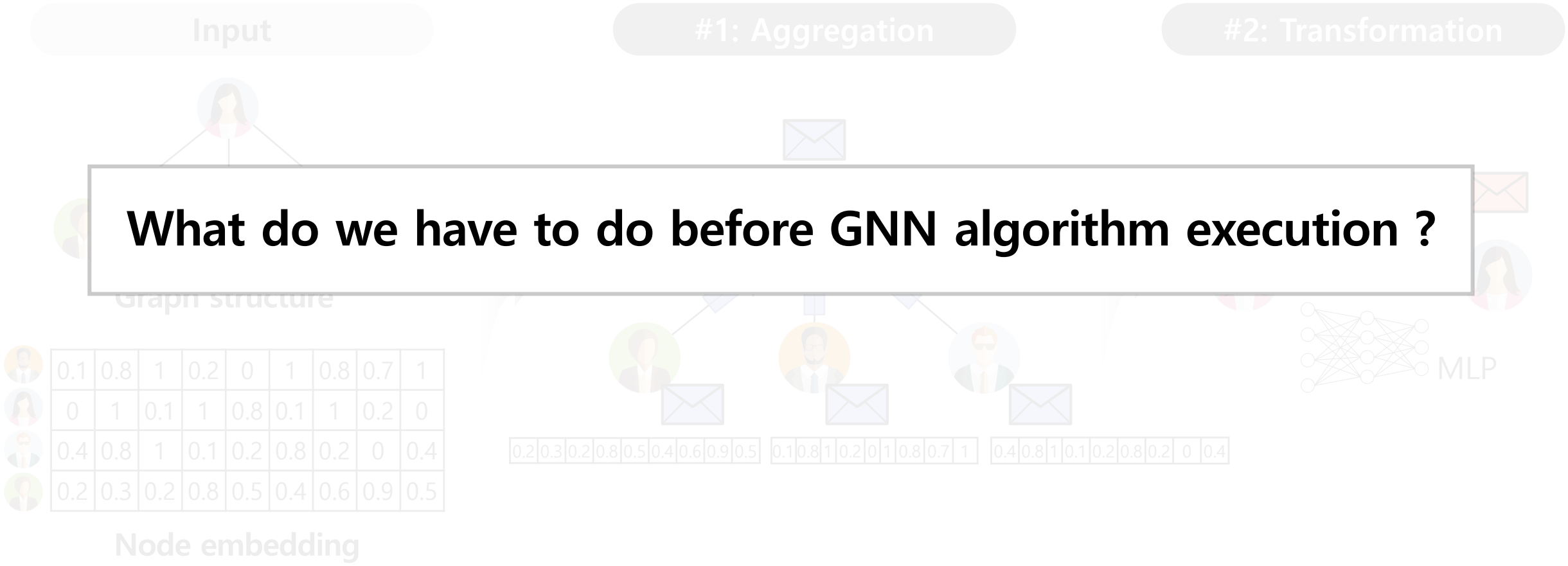


#2: Transformation



Graph Neural Networks (GNN)

GNN algorithm

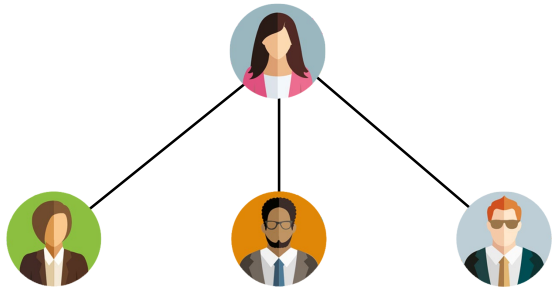


What do we have to do before GNN algorithm execution ?

Graph Neural Networks (GNN)

GNN algorithm

Input



Graph structure

0.1	0.8	1	0.2	0	1	0.8	0.7	1
0	1	0.1	1	0.8	0.1	1	0.2	0
0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4
0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

Node embedding

❶ We have to prepare **neighbor-oriented** graph structure

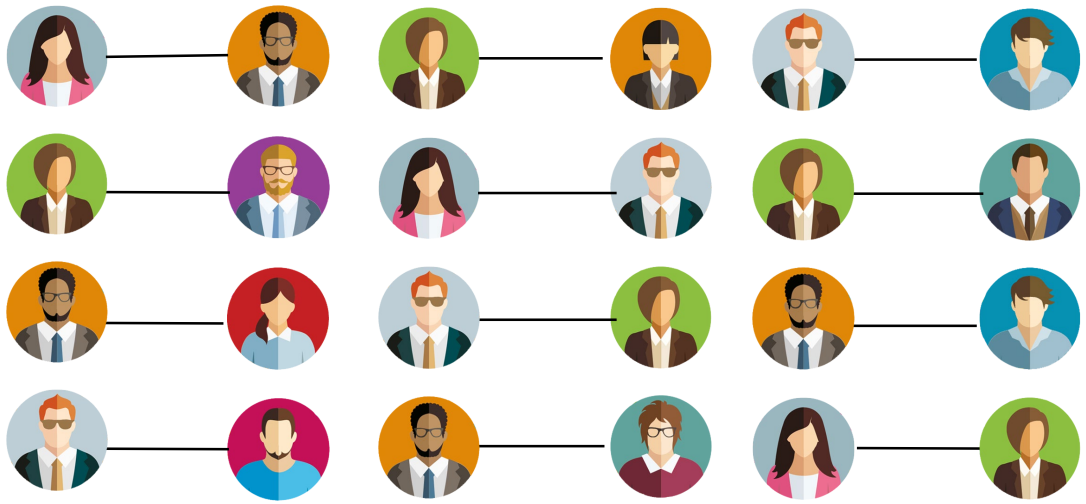
❷ We need small input data which **can be loaded** into accelerator memory

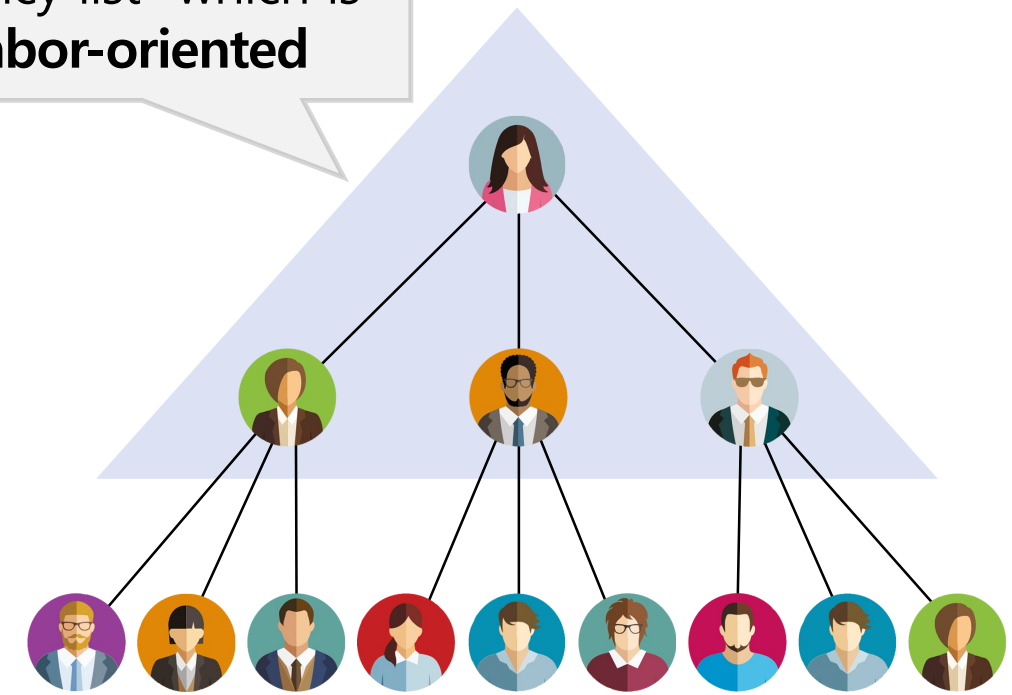
GNN Preprocessing

Graph preprocessing: to prepare *neighbor-oriented* graph structure

Graph structure is stored as "edge array" which is **update-friendly**

Graph preprocessing converts edge array to "adjacency list" which is **neighbor-oriented**

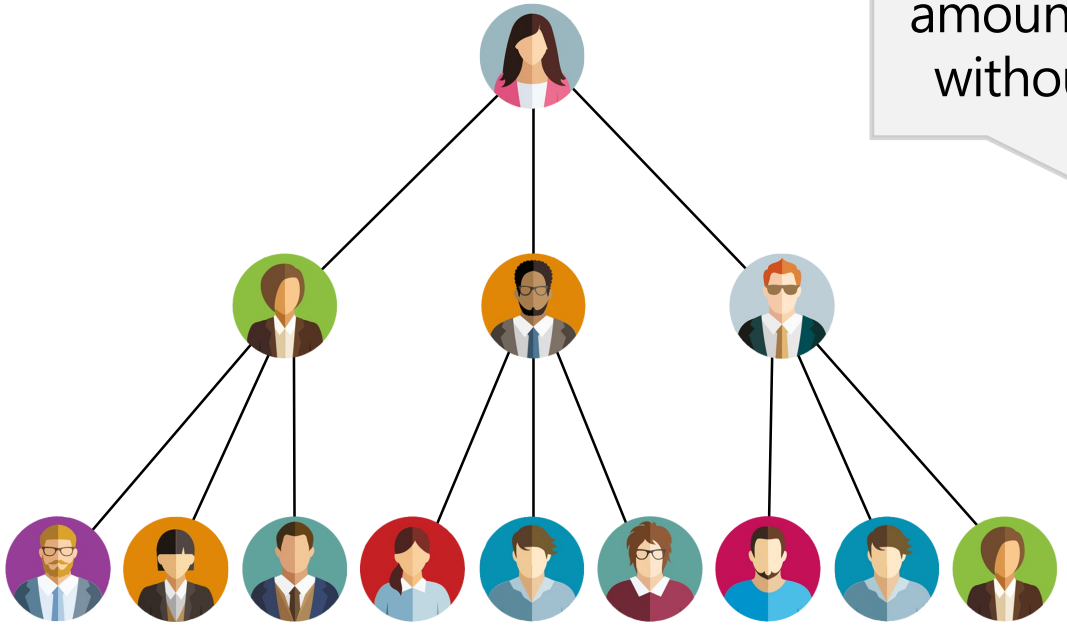




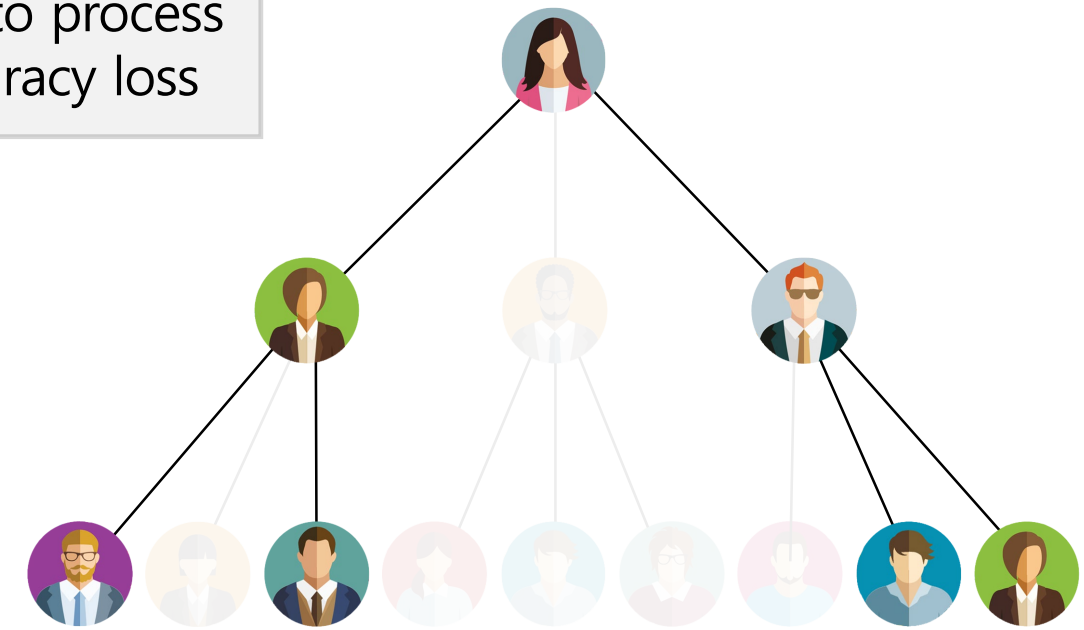
GNN Preprocessing

Batch preprocessing: to prepare *small graph*

Insight: "Node sampling" can significantly reduce the amount of data to process without an accuracy loss



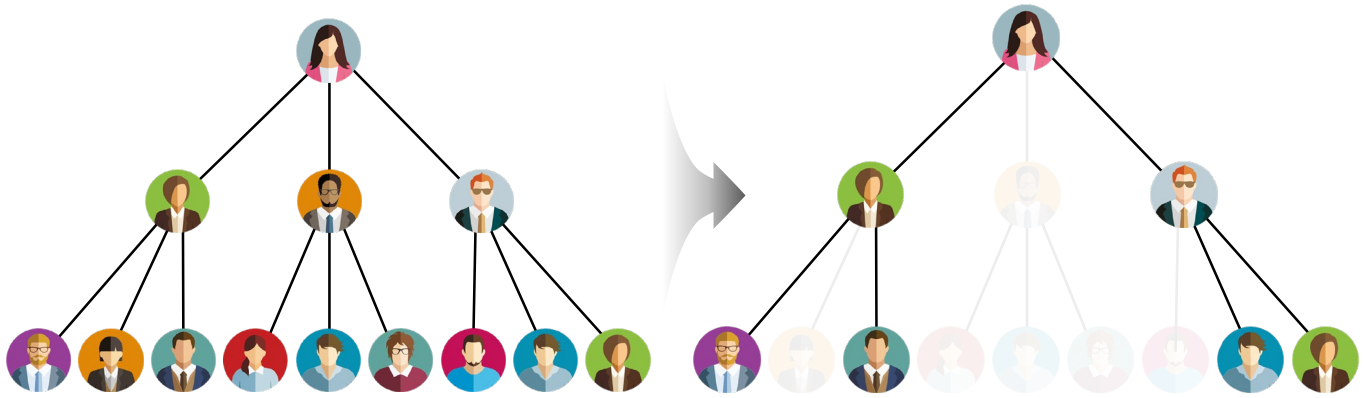
≈



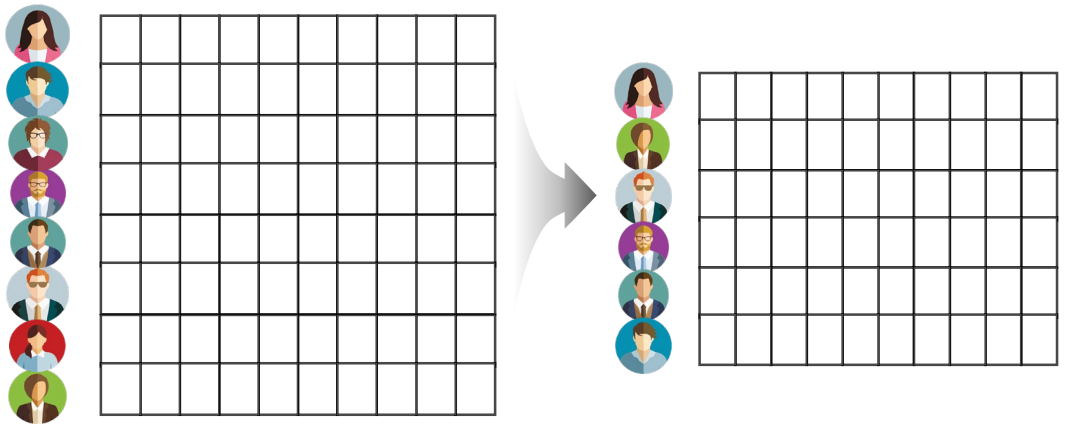
GNN Preprocessing

Batch preprocessing: to prepare *small graph*

Graph structure sampling



Embedding sampling



1. Background

2. Motivation and Design Considerations

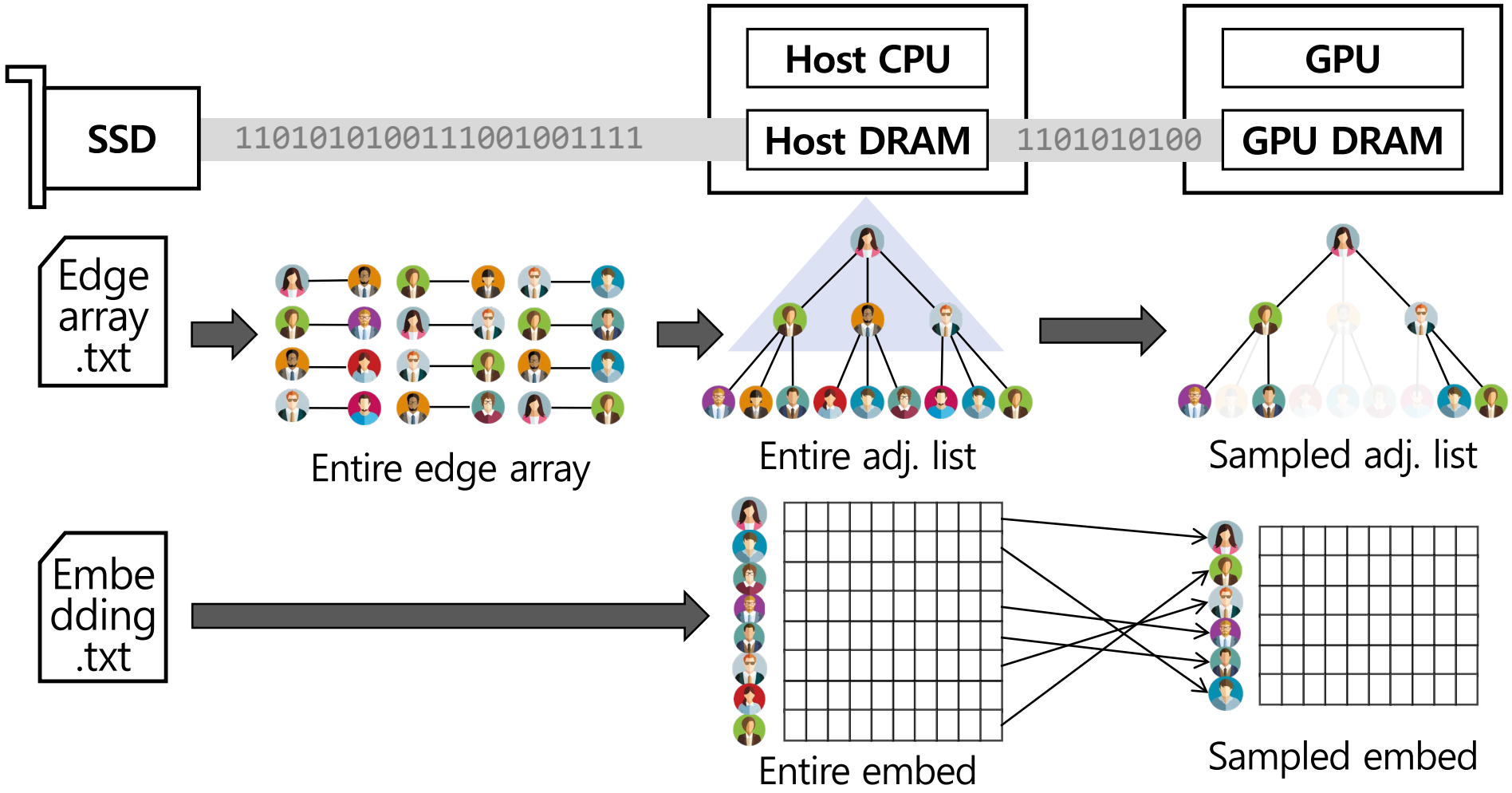
3. Overview of HolisticGNN Framework

4. Details of HolisticGNN Components

5. Evaluation

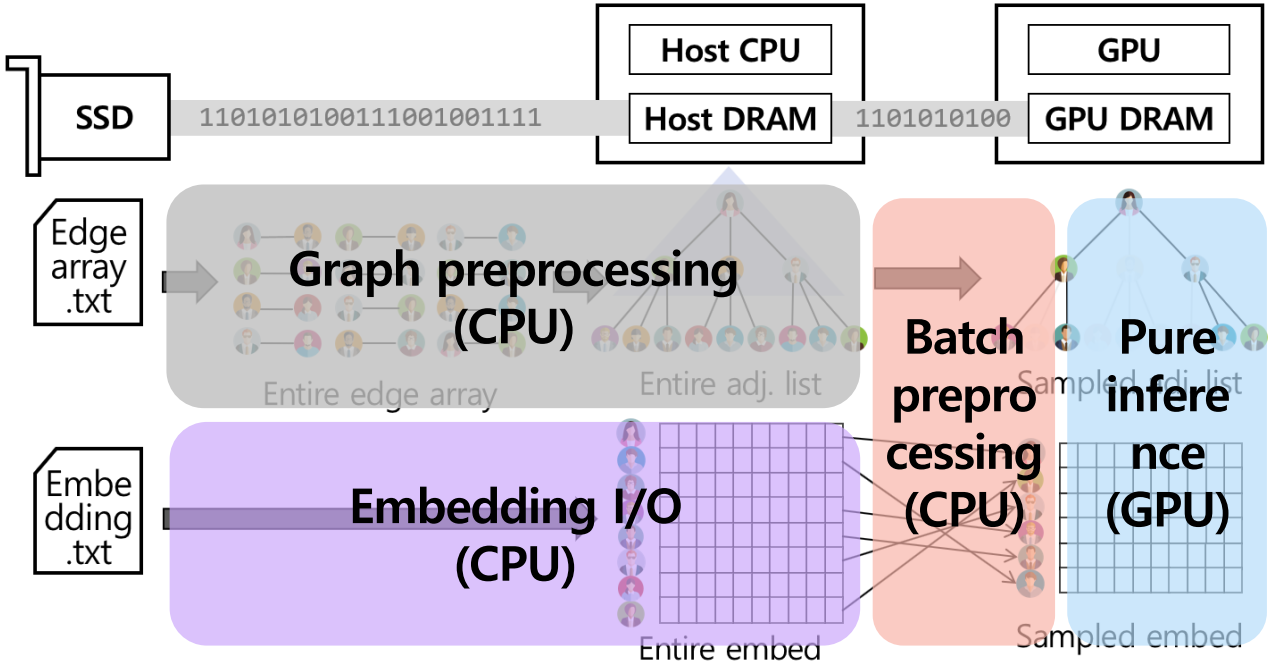
End-to-End GNN Inference

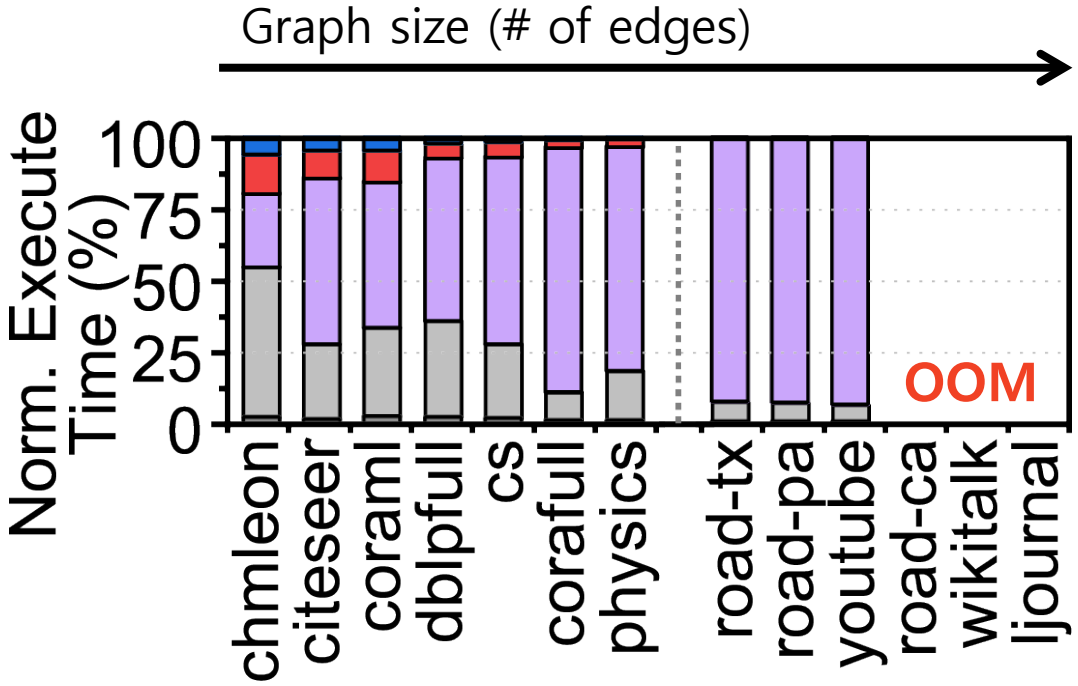
Visualization



End-to-End GNN Inference

Execution time analysis

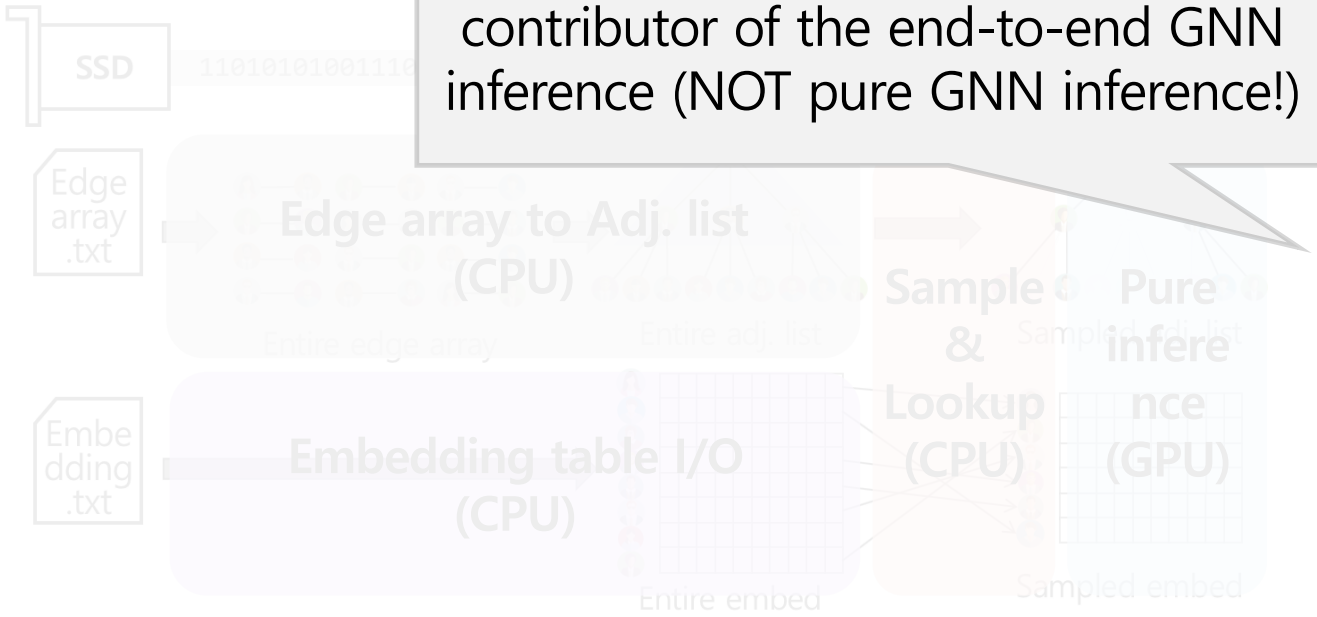


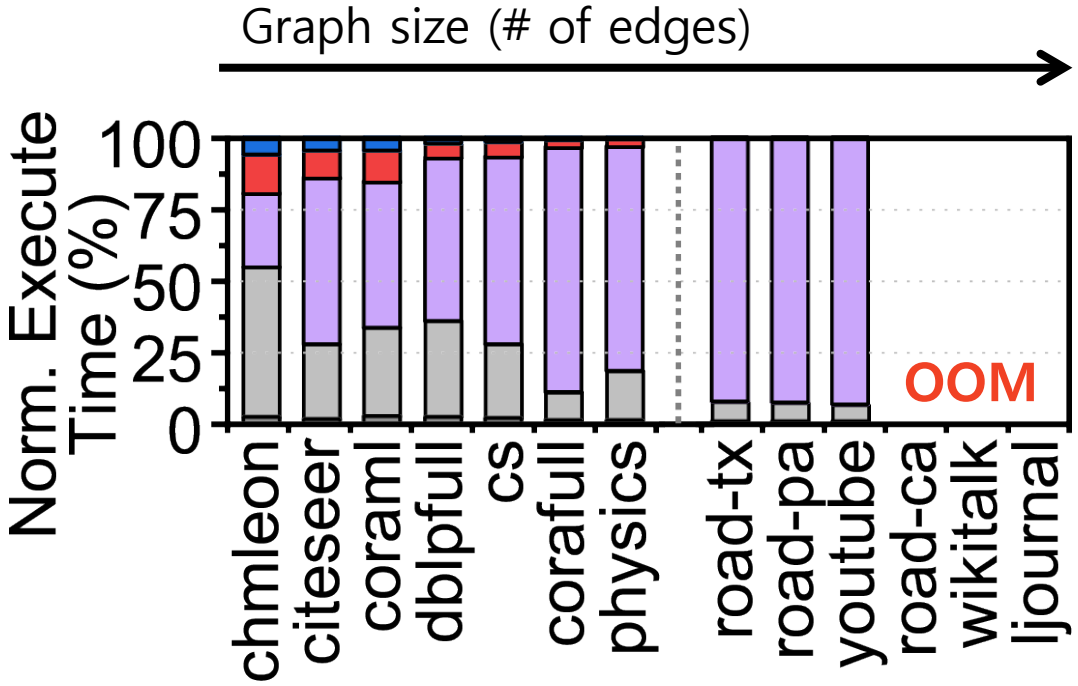


End-to-End GNN Inference

Execution time analysis

Oops.. **Graph preprocessing** and **embedding I/O** is dominant contributor of the end-to-end GNN inference (NOT pure GNN inference!)





Design Questions

Then, what does GNN acceleration look like?

Graph preprocessing (CPU) → **Store graph <u>directly</u> as a neighbor-oriented format (But also, update-efficient)**

Embedding I/O (CPU) → **Process end-to-end GNN inference <u>near storage</u>**

1. Background

2. Motivation and Design Considerations

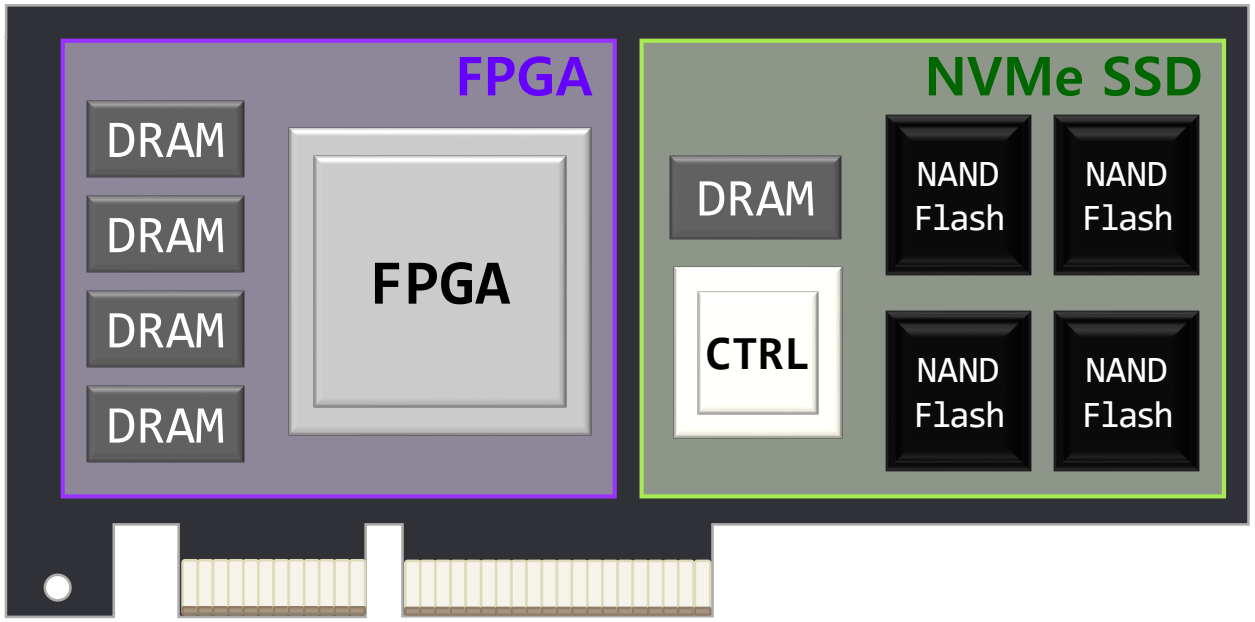
3. Overview of HolisticGNN Framework

4. Details of HolisticGNN Components

5. Evaluation

HolisticGNN

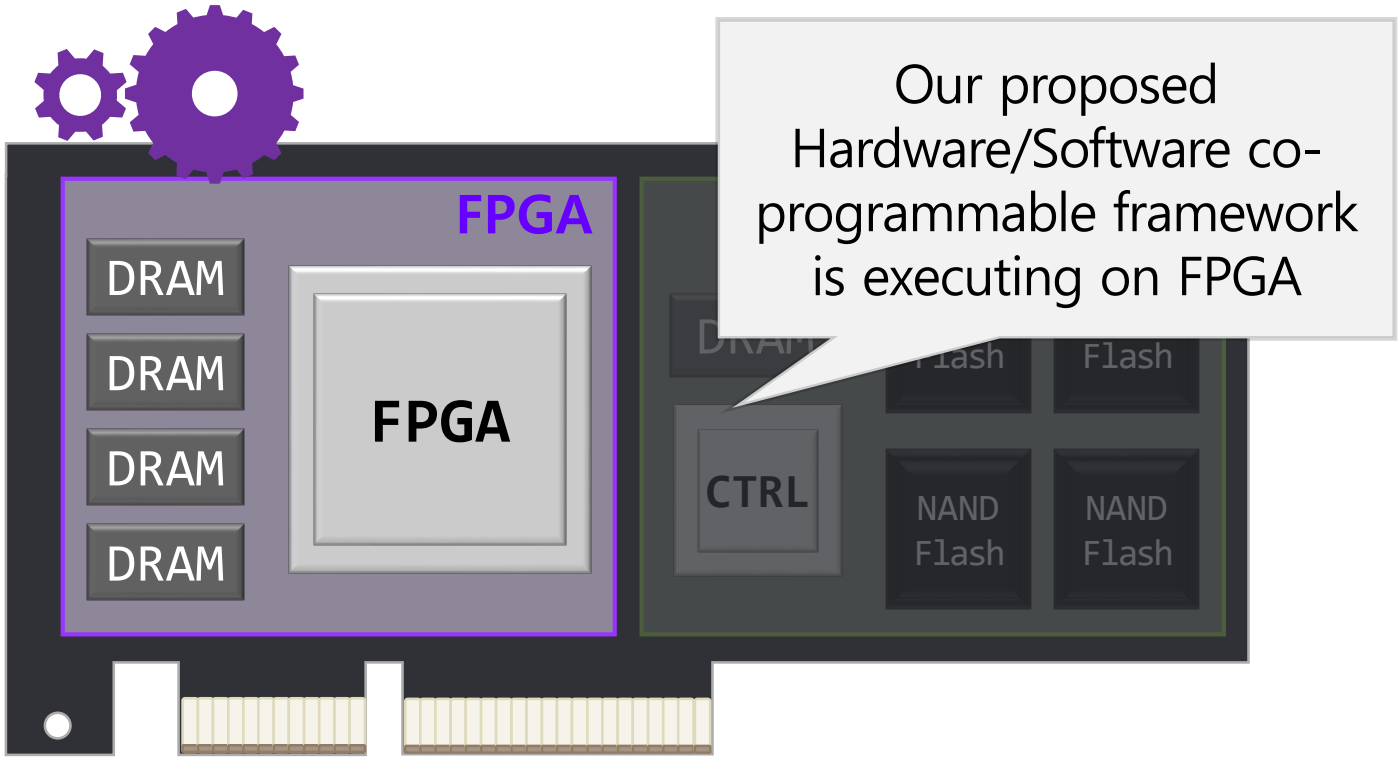
Adopts the concept of computational SSD (CSSD)



CSSD decouples the compute unit from the storage resources unlike conventional ISP (In-Storage Processing)

HolisticGNN

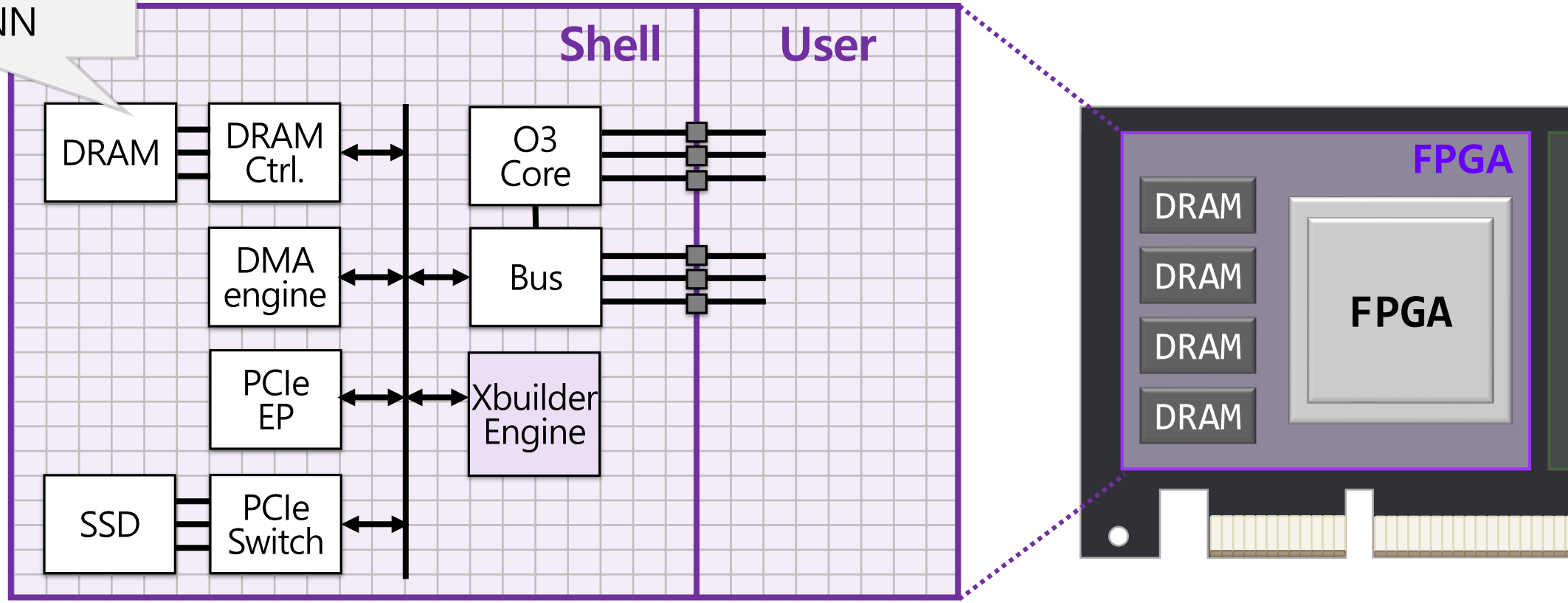
"Hardware/Software Co-Programmable Framework" for CSSDs



HolisticGNN

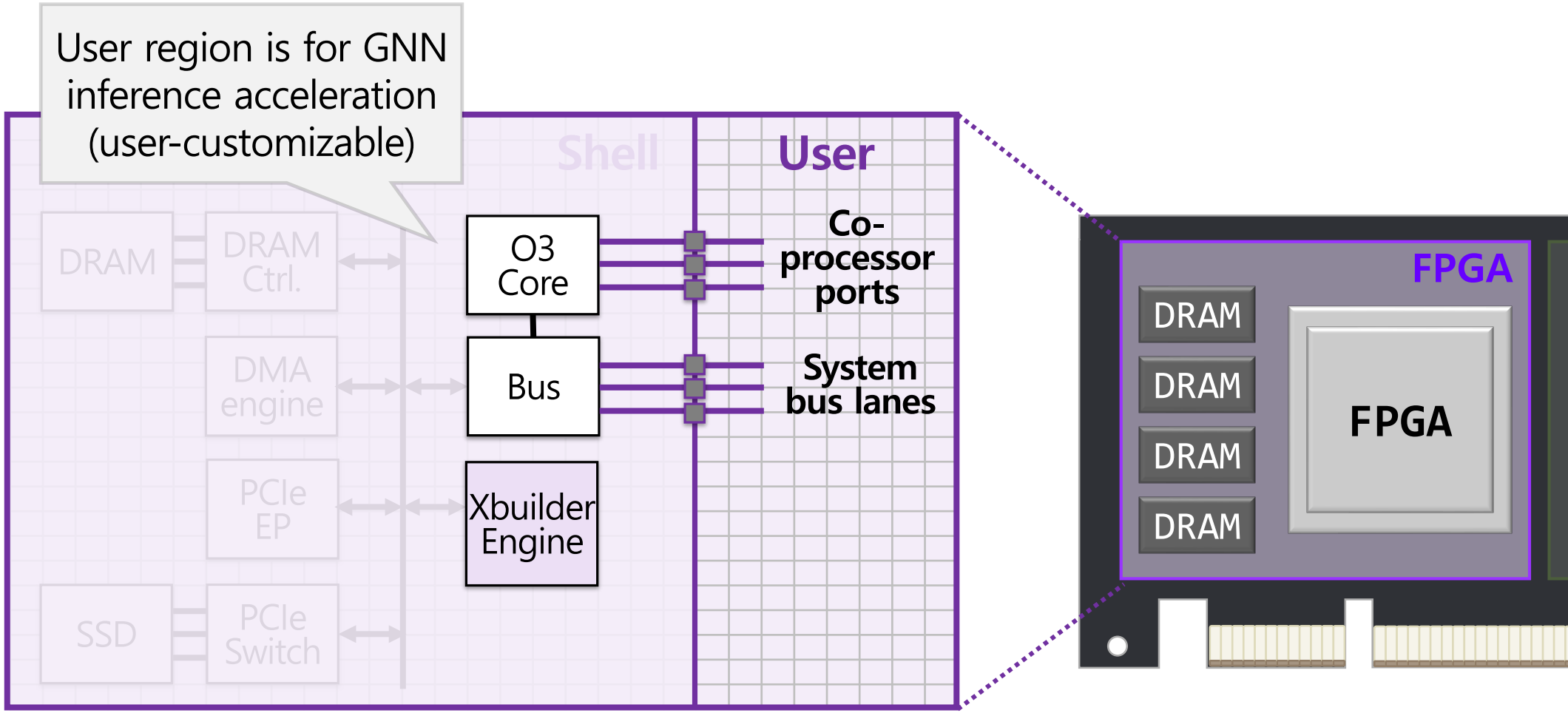
"Hardware/Software Co-Programmable Framework" for CSSDs

Shell region is for essential HW logics of HolisticGNN



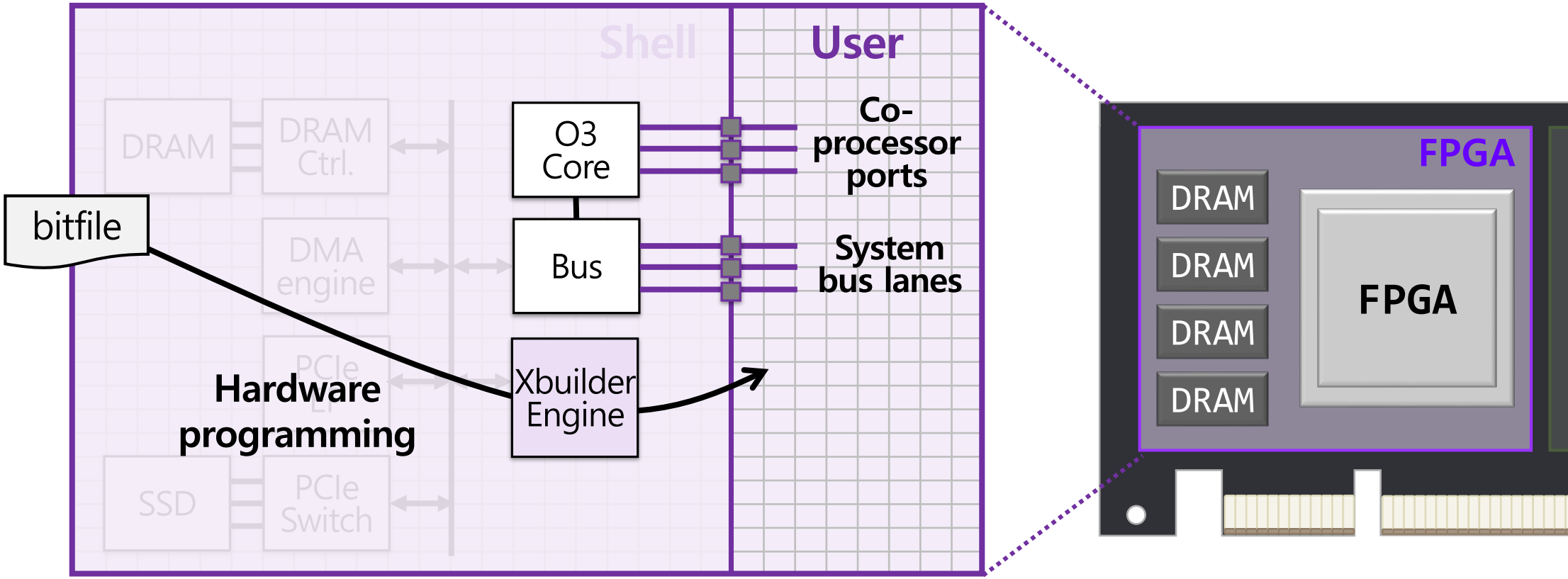
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



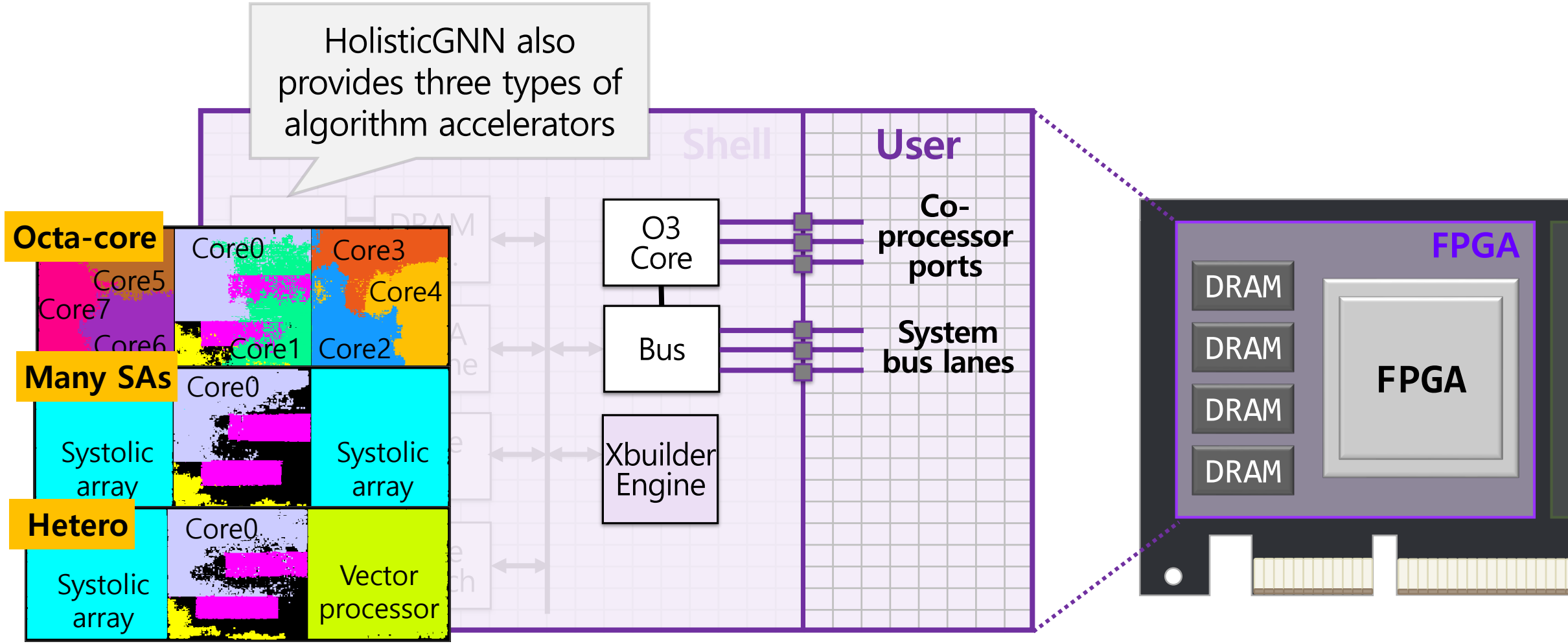
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



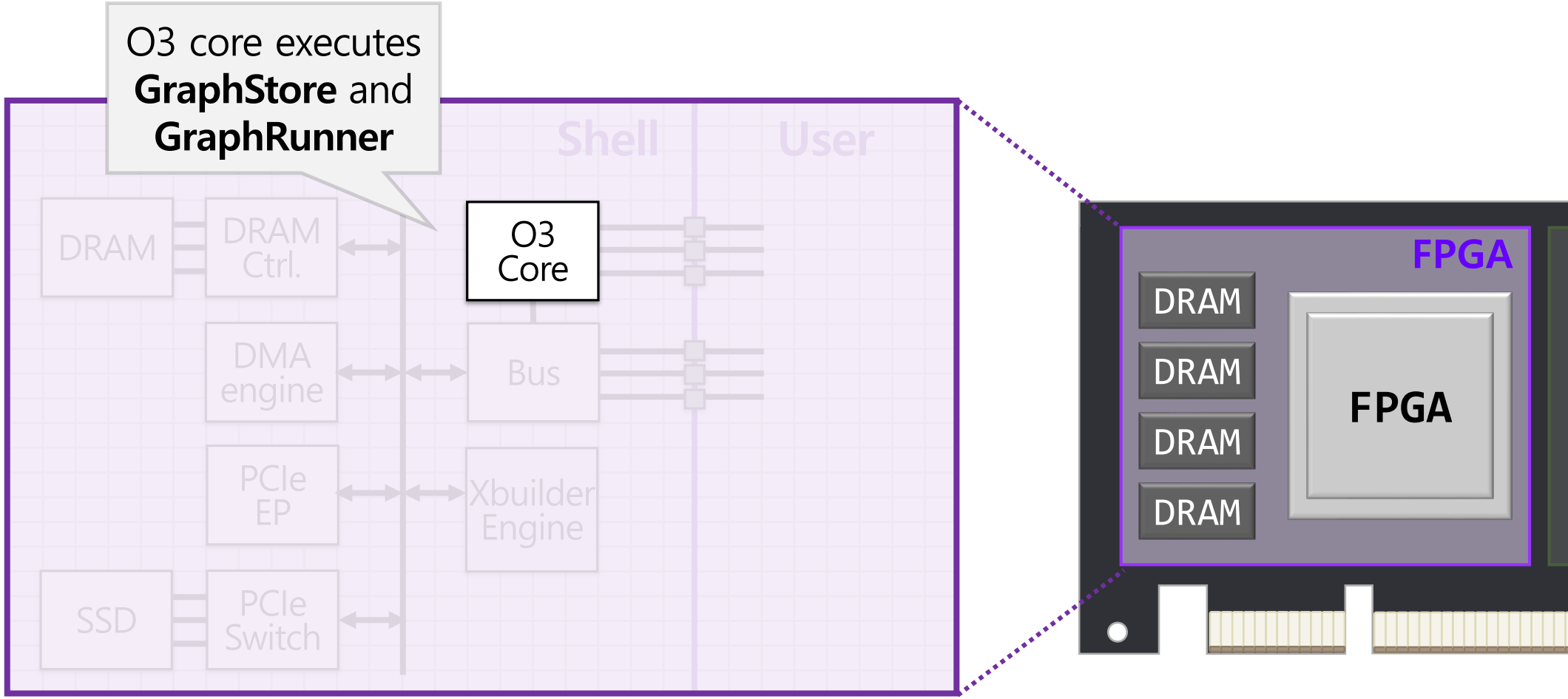
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



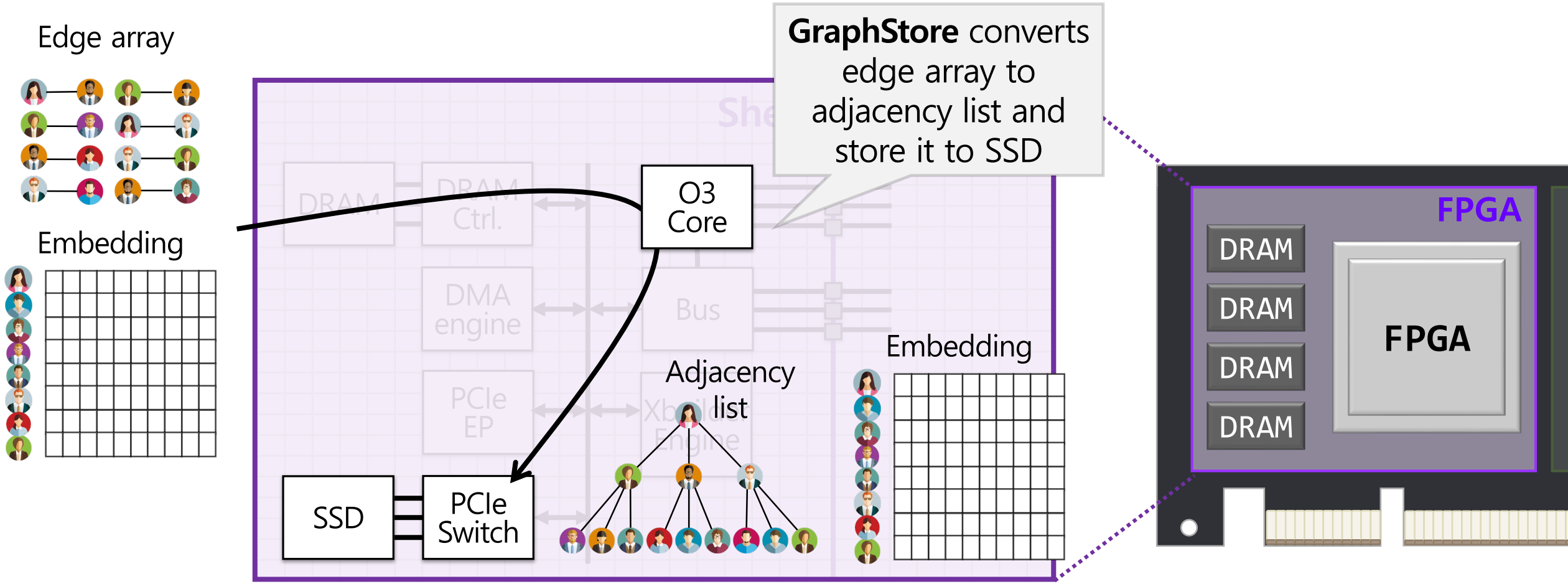
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



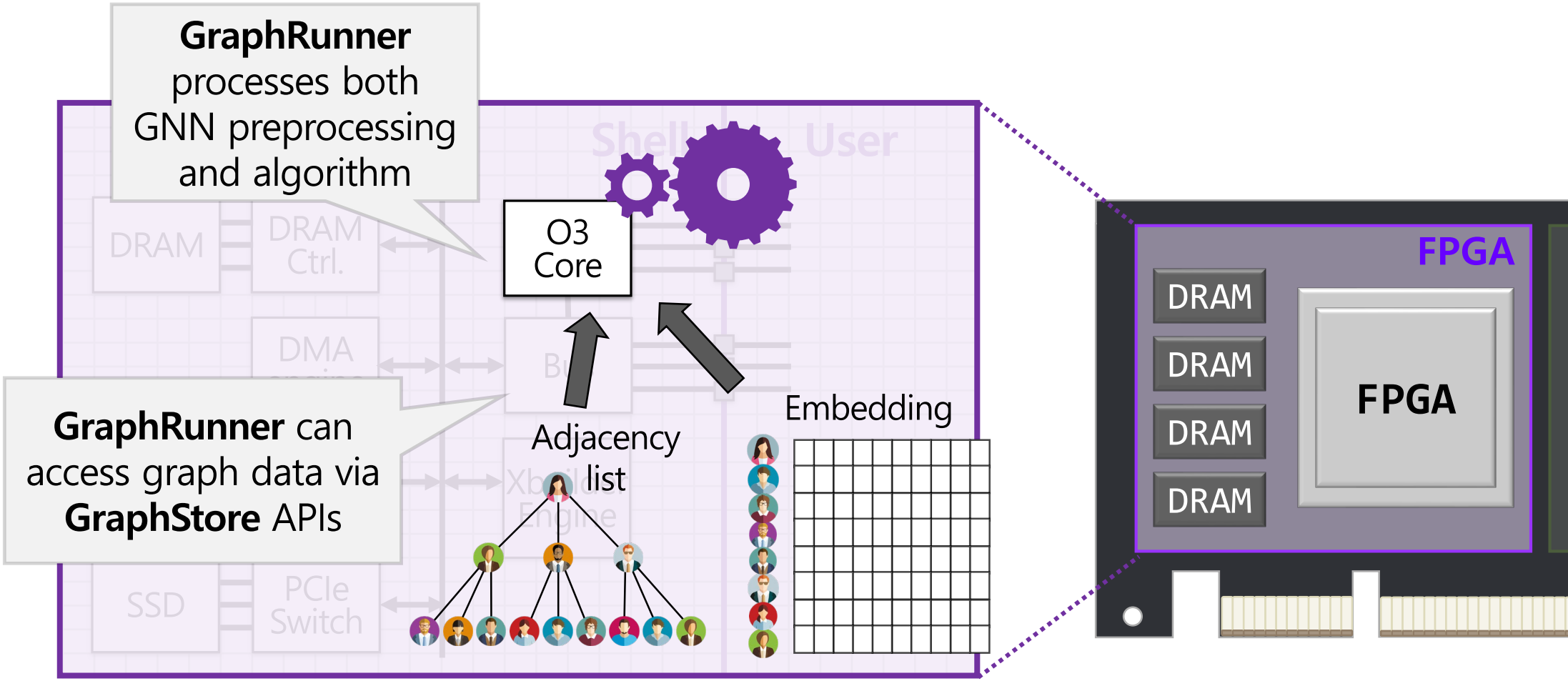
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



1. Background

2. Motivation and Design Considerations

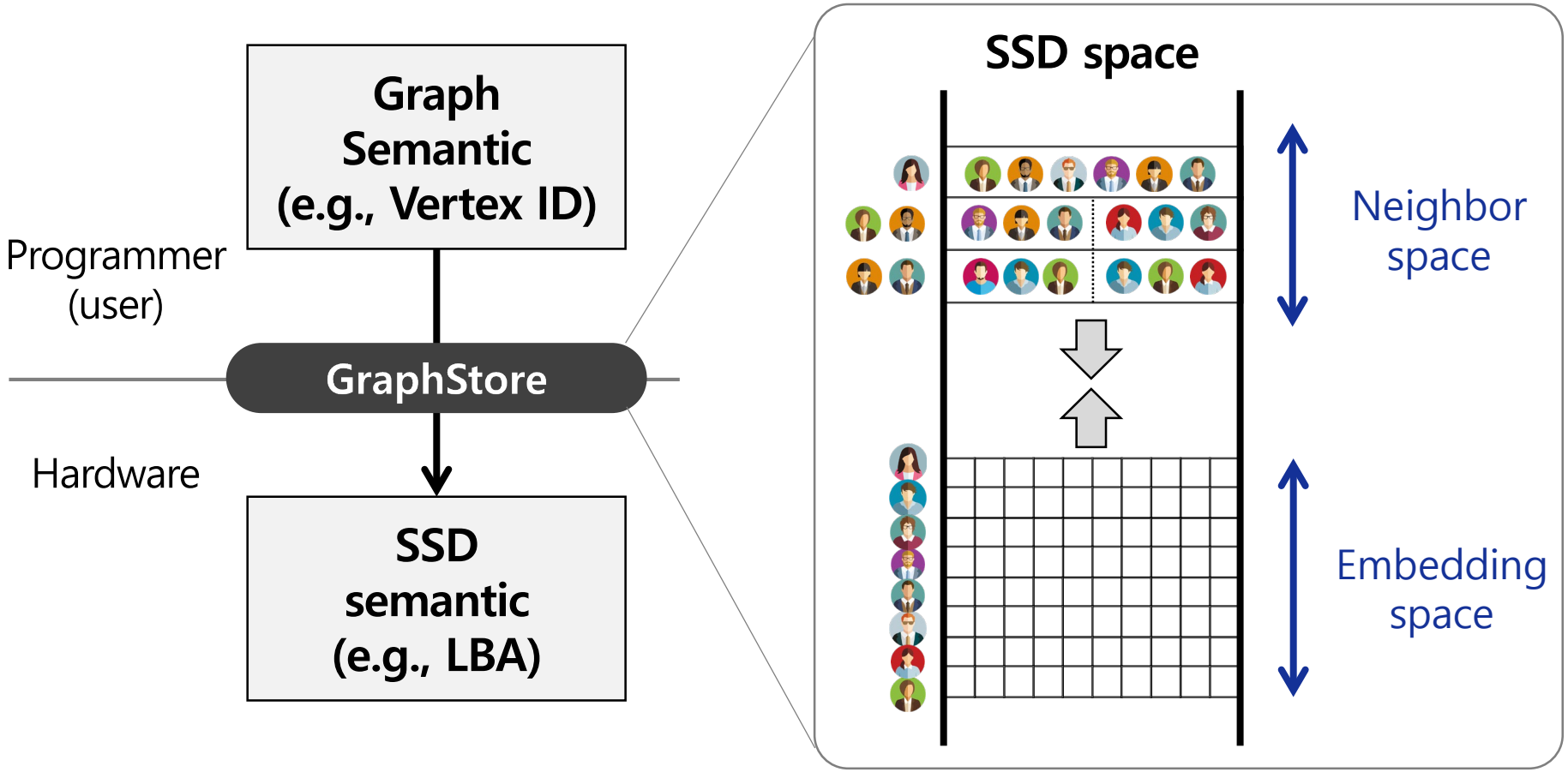
3. Overview of HolisticGNN Framework

4. Details of HolisticGNN Components

5. Evaluation

GraphStore

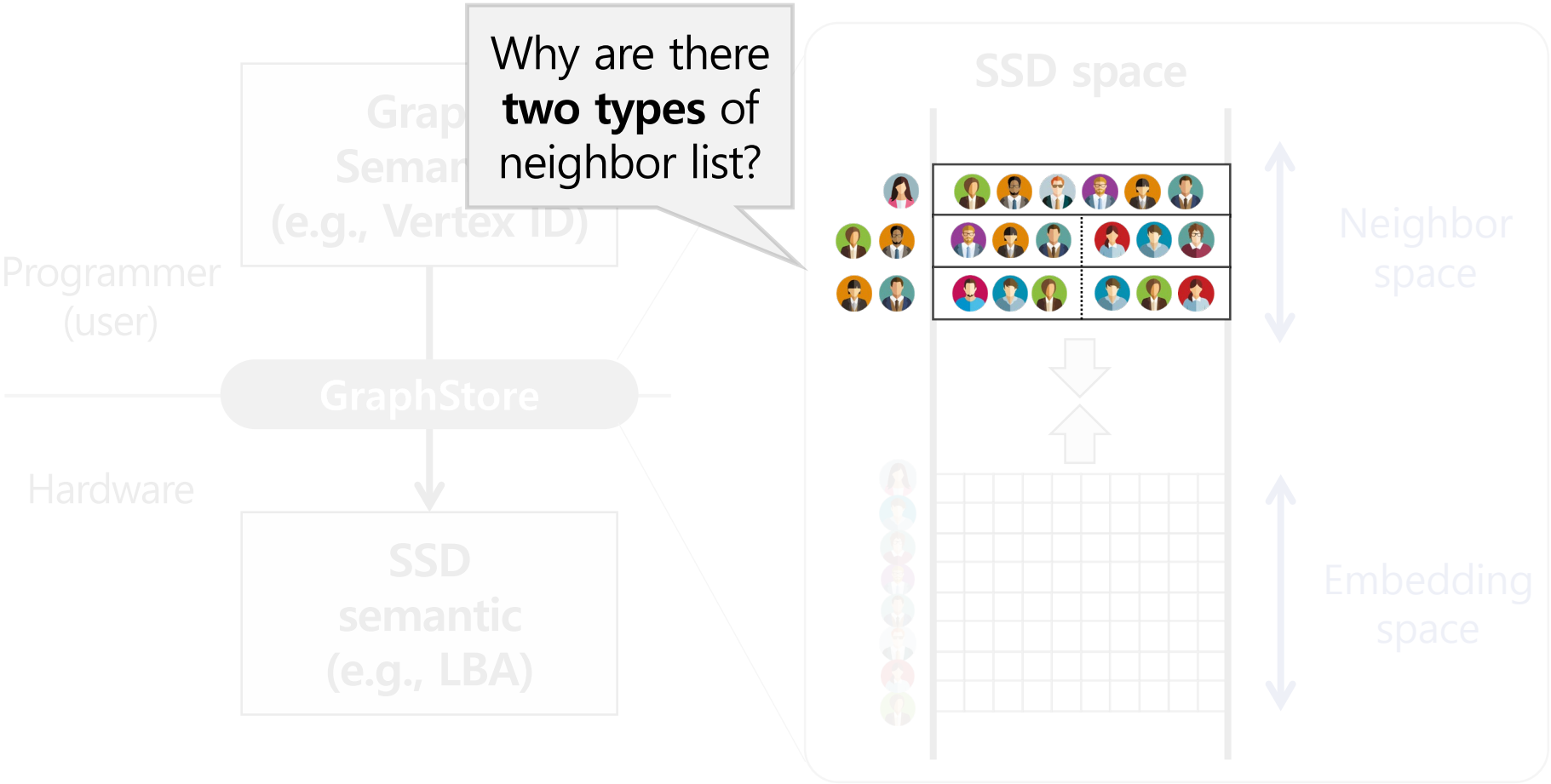
Graph-centric archiving system



GraphStore separates SSD space into two for making sure update-efficient

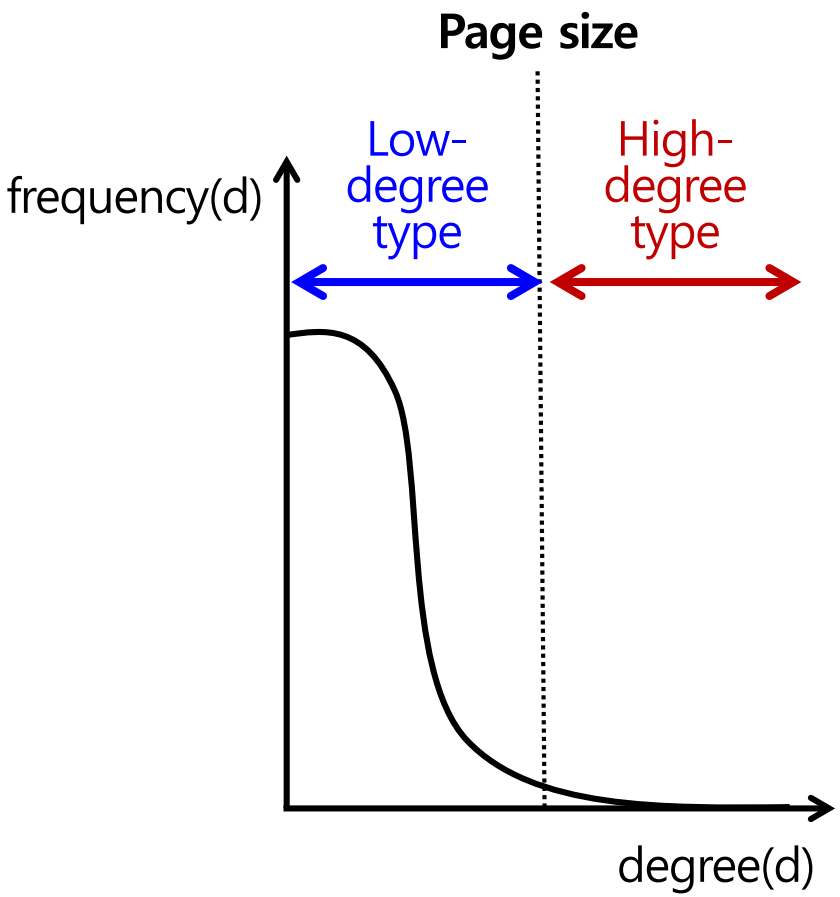
GraphStore

Graph-centric archiving system



GraphStore

Graph-centric archiving system

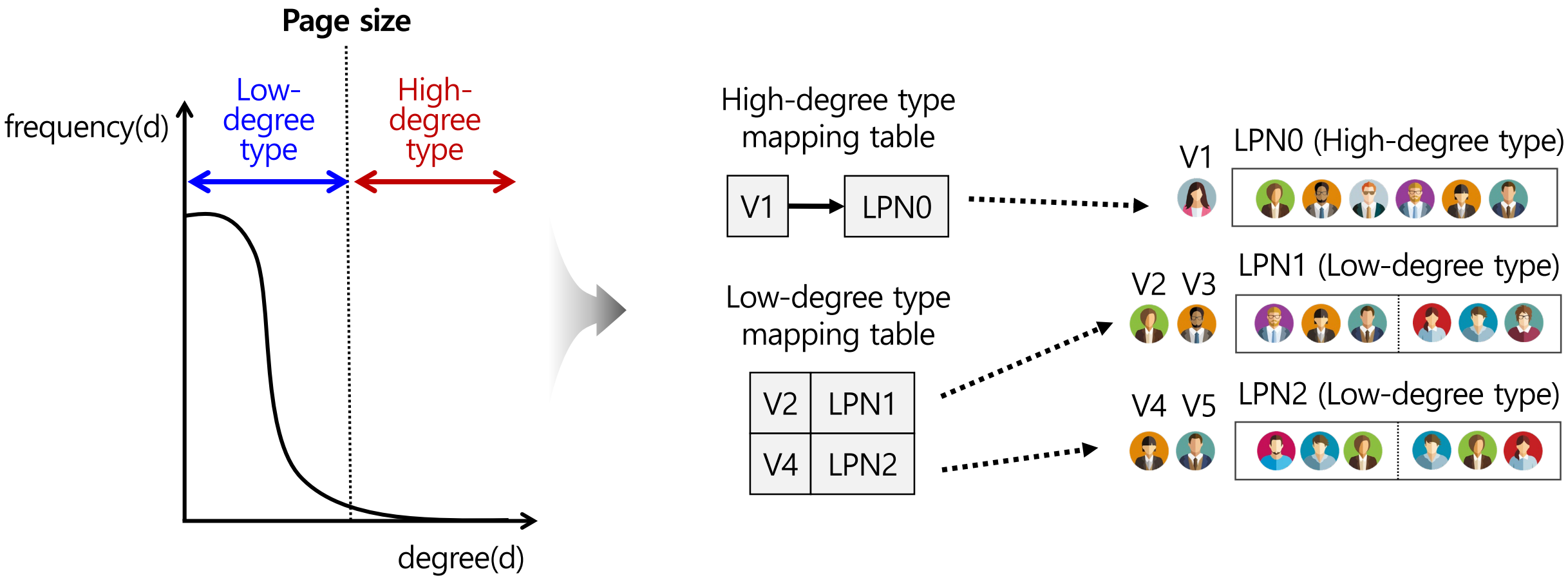


Insight comes from **power-law distribution** of degree (# of neighbors)

※ Nature characteristic of graph

GraphStore

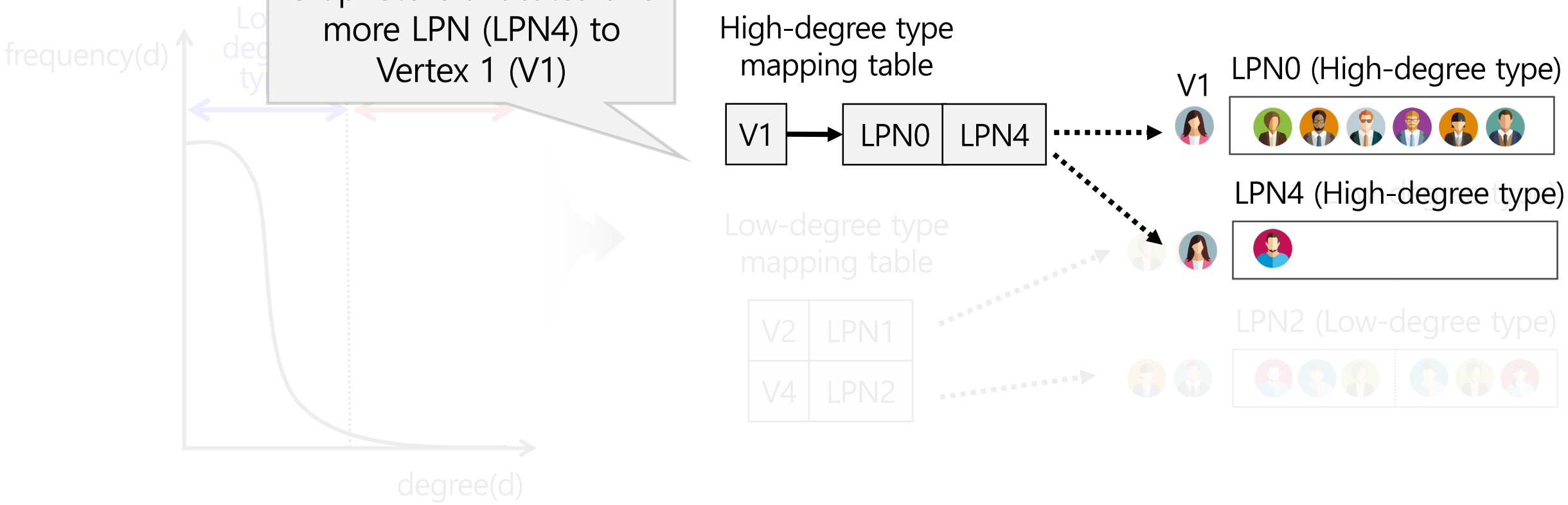
Graph-centric archiving system



GraphStore

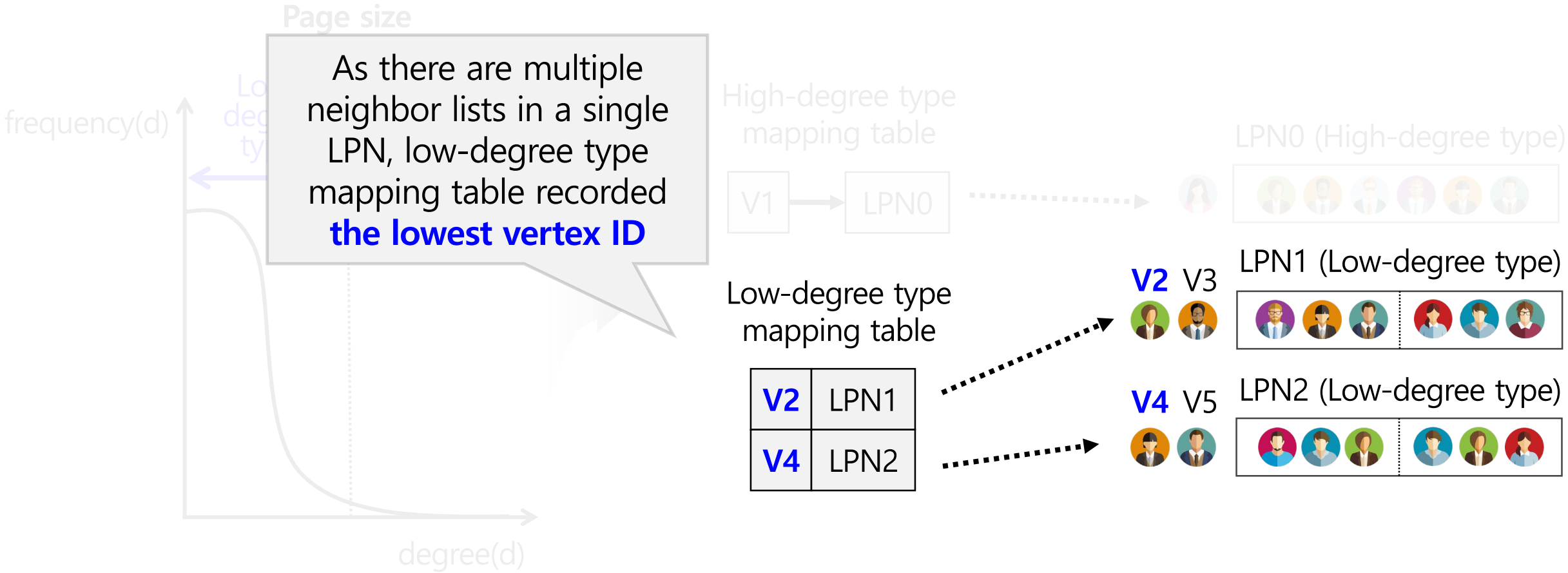
Graph-centric archiving system

If new neighbor is added, GraphStore allocates one more LPN (LPN4) to Vertex 1 (V1)



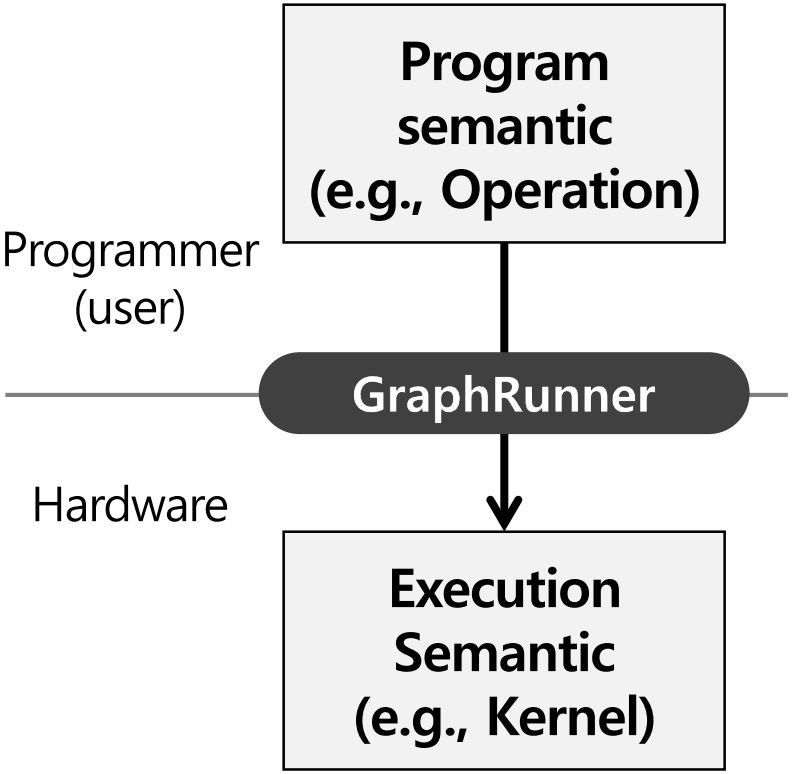
GraphStore

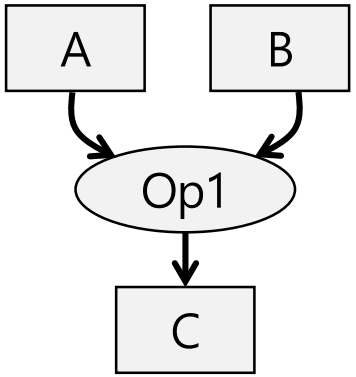
Graph-centric archiving system



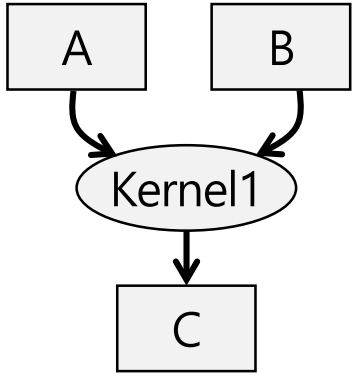
GraphRunner

Programmable inference model





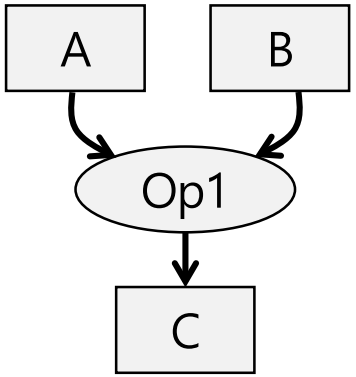
Programmer programs the GNN inference sequence as a **DFG** (Dataflow graph)

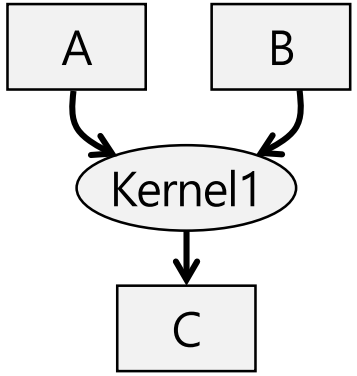


GraphRunner **converts** operation to kernel and executes the DFG

GraphRunner

Programmable inference model





BTW, why are there **gap** between program semantic and execution semantic?

GraphRunner

Programmable inference model

Because there are many devices which can process the same operation!

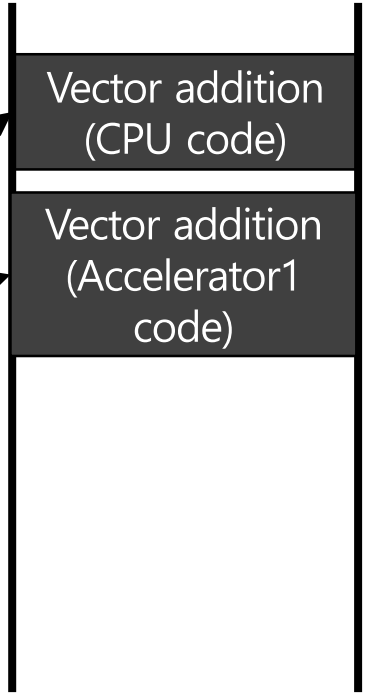
Device table

Device Name	Priority
"CPU"	50
"Accelerator1"	150
…	…

Operation table

Operation Name	Kernel
"Vector addition"	<"CPU", ptr> <"Accelerator1", ptr>
…	…

Memory space



GraphRunner

Programmable inference model

Users can add their own accelerator and also accelerator's kernel ☺

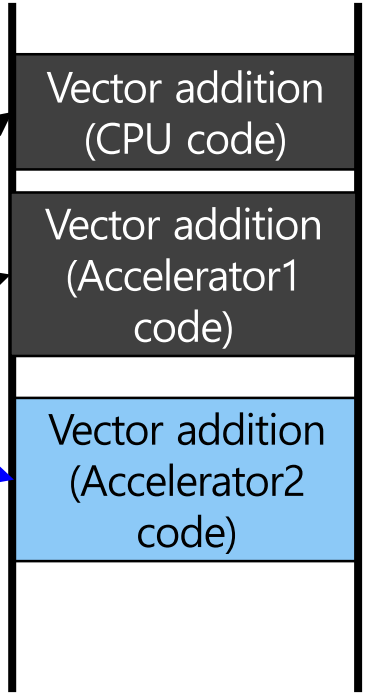
Device table

Device Name	Priority
"CPU"	50
"Accelerator1"	150
…	…

Operation table

Operation Name	Kernel
"Vector addition"	<"CPU", ptr> <"Accelerator1", ptr>
…	…

Memory space



1. Background

2. Motivation and Design Considerations

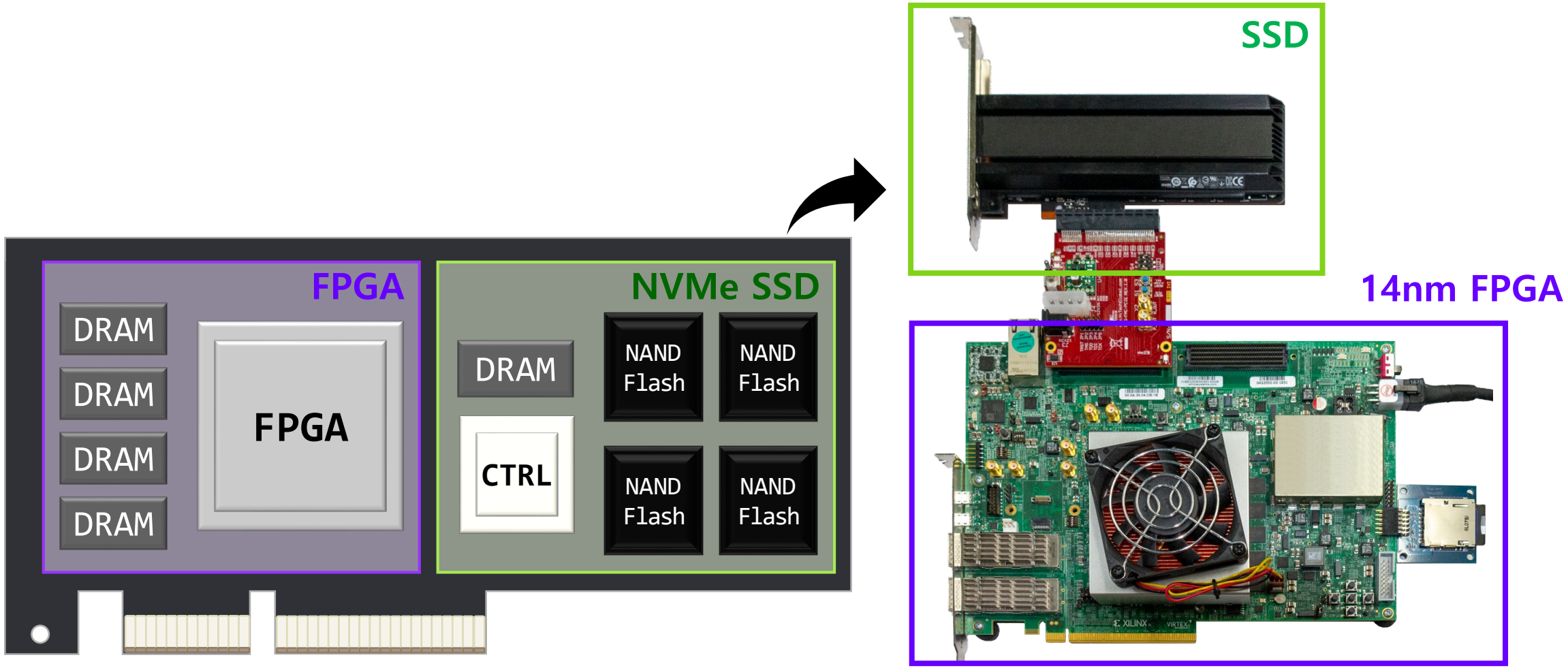
3. Overview of HolisticGNN Framework

4. Details of HolisticGNN Components

5. Evaluation

Experimental Setup

HolisticGNN prototype



Experimental Setup

Graph dataset

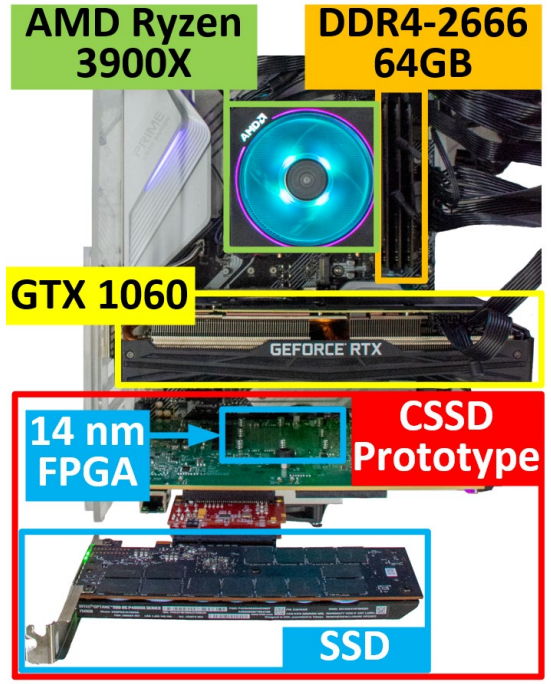
Small (<1M Edge)

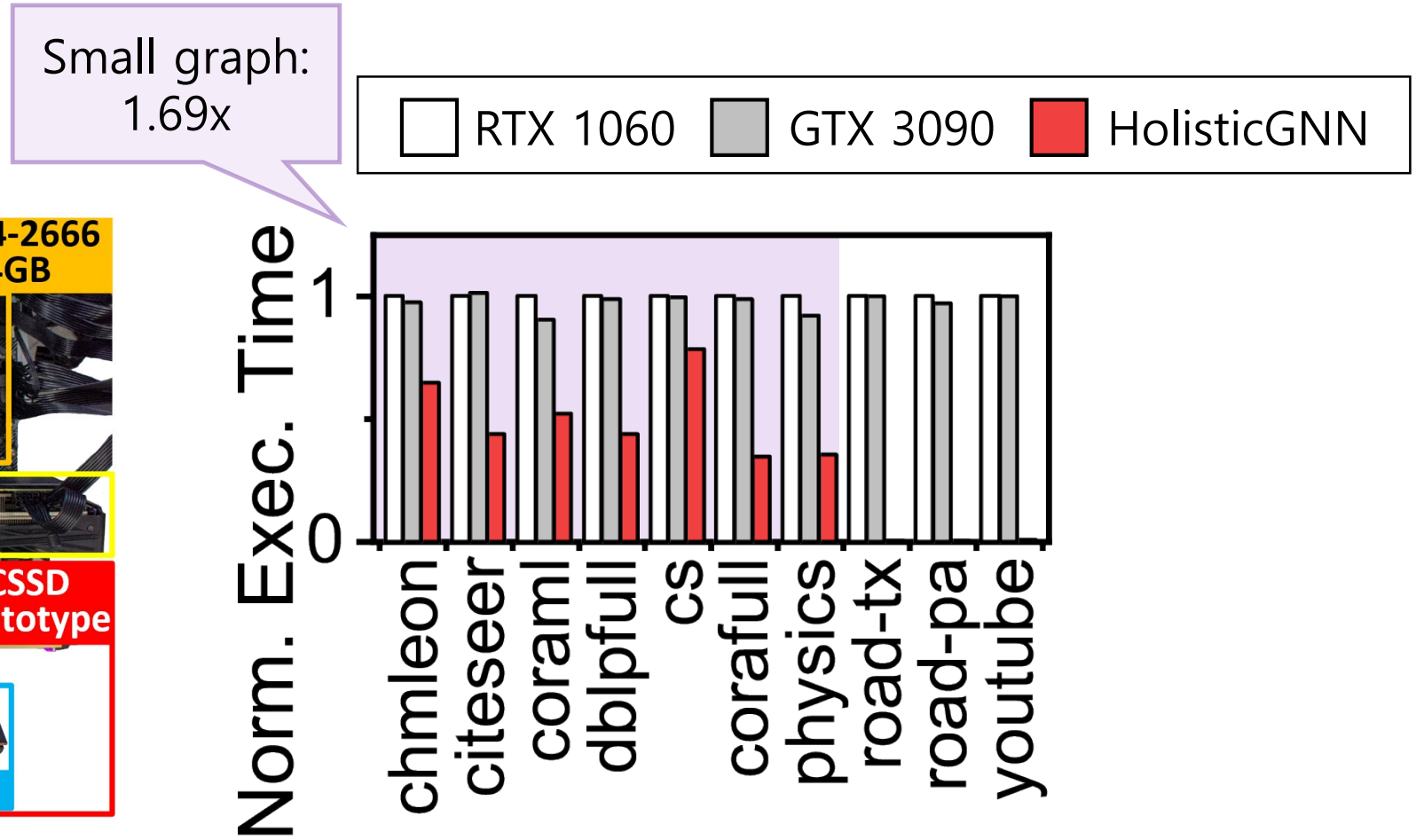
Large (>3M Edge)

	Vertices	Edges	Feature length
chmleon	2.3K	65K	2326
citeseer	2.1K	9K	3704
cormal	3.0K	19K	2880
dblpfull	17.7K	123K	1639
cs	18.3K	182K	6805
corafull	19.8K	147K	8710
physics	34.5K	530K	8415
road-tx	1.39M	3.84M	4353
road-pa	1.09M	3.08M	4353
youtube	1.16M	2.99M	4353
road-ca	1.97M	5.53M	4353
wikitalk	2.39M	5.02M	4353
ljournal	4.85M	68.99M	4353

Evaluation Results

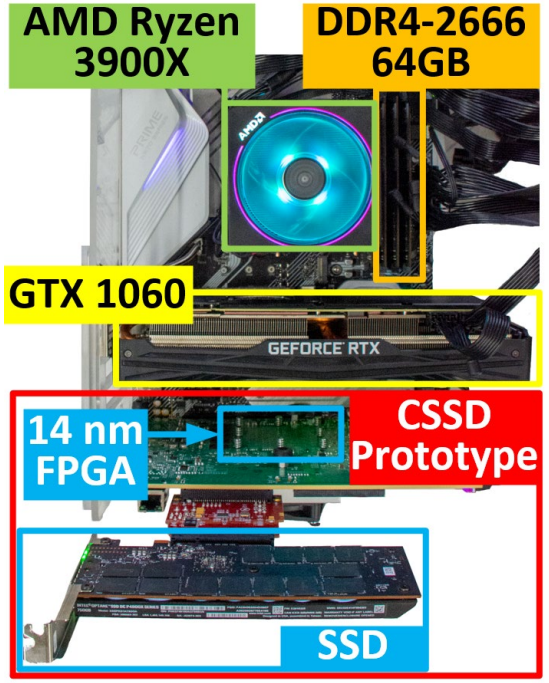
End-to-End latency comparison

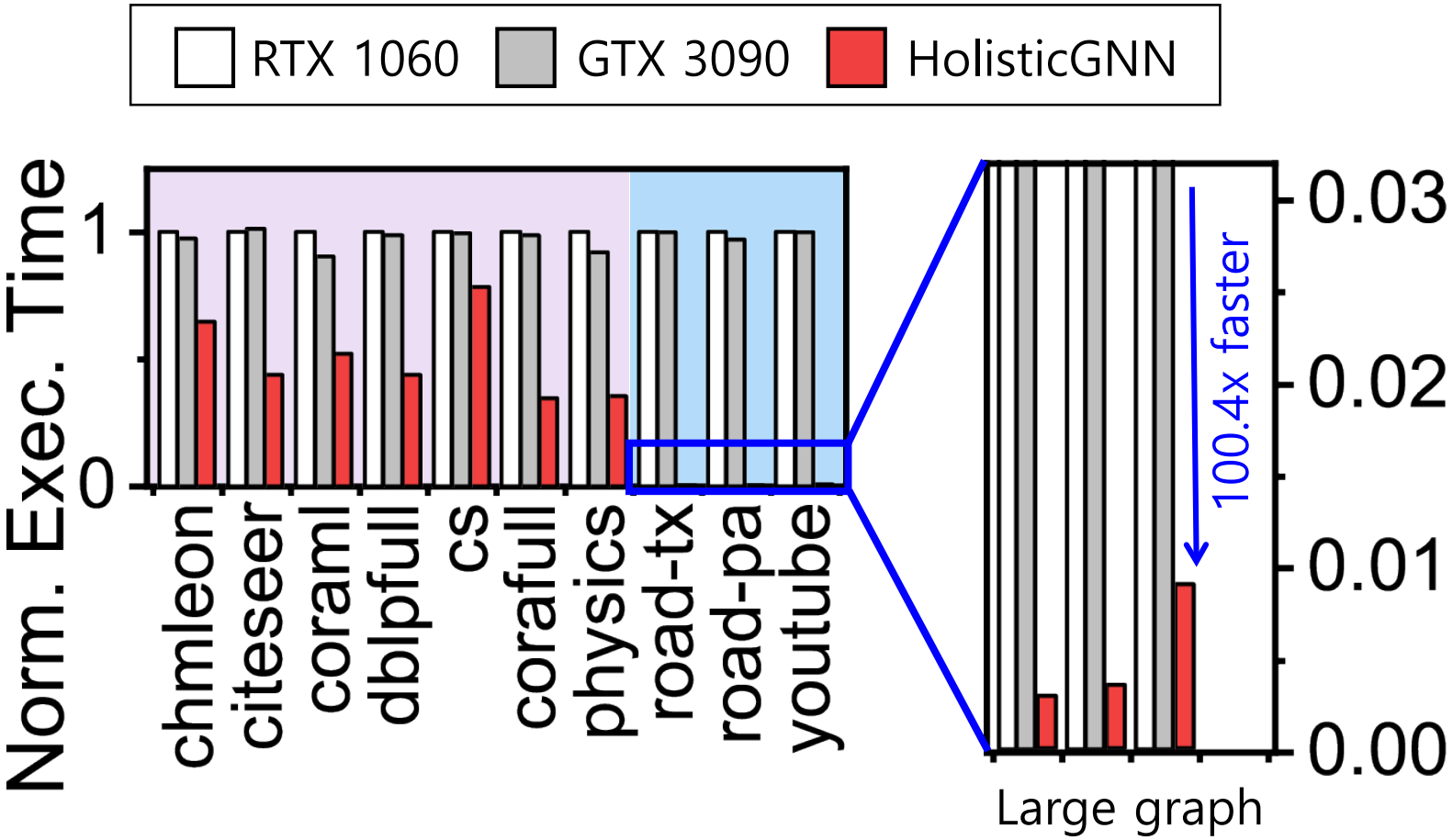




Evaluation Results

End-to-End latency comparison

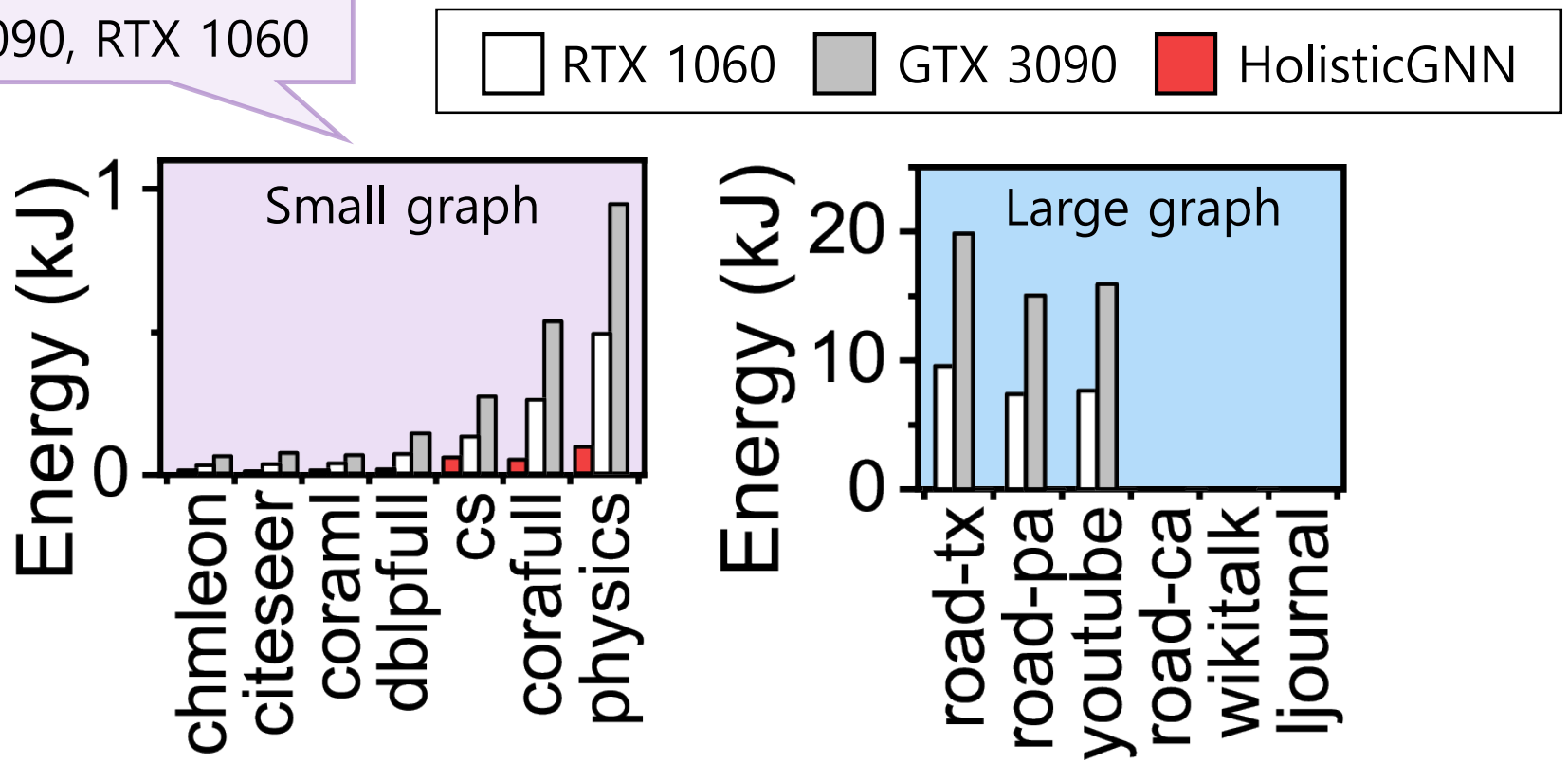




Evaluation Results

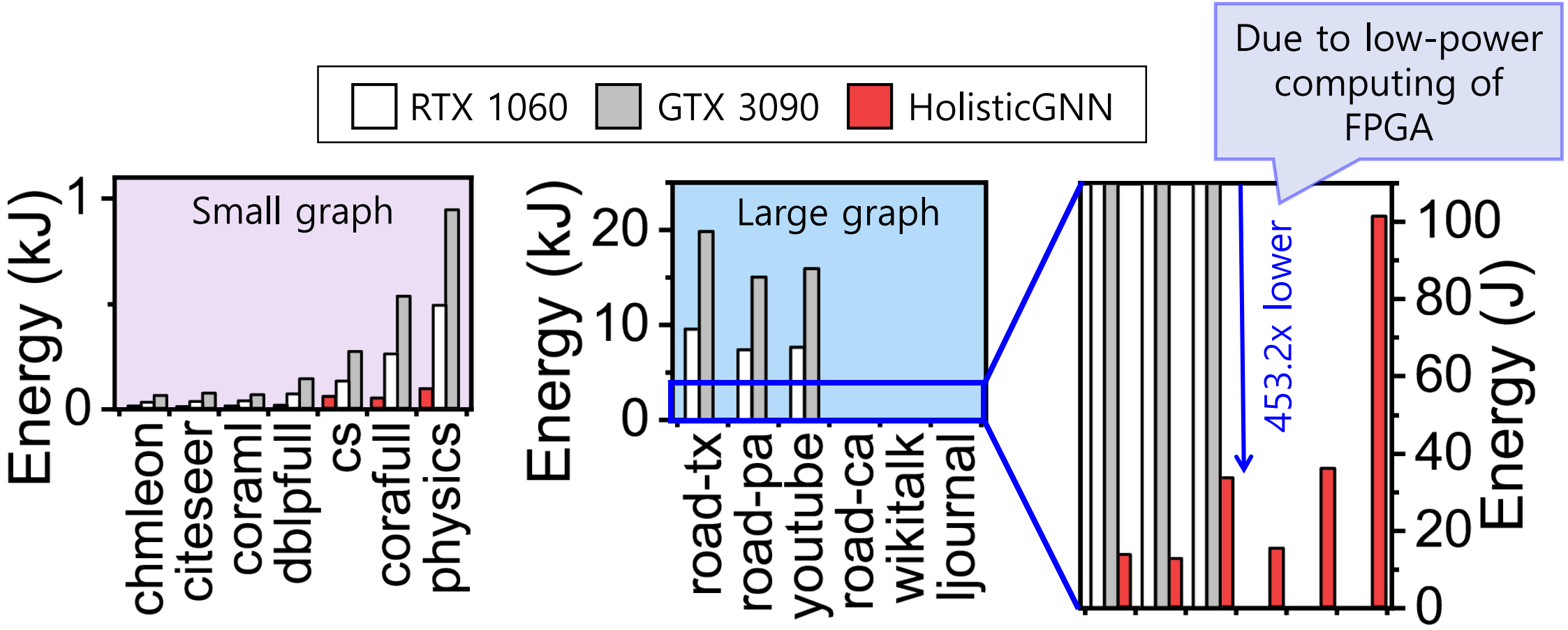
Energy Consumption

33.2x and 16.3x better than GTX 3090, RTX 1060



Evaluation Results

Energy Consumption



Demo

GNN execution in our HolisticGNN prototype

Conclusion

HolisticGNN is a “hardware/software co-programmable framework for computational SSDs”

1) Holistic solution for both GNN algorithm and preprocessing
2) Fast and energy-efficient near-storage inference infrastructure
3) Easy-to-use and user-customizable

Thank You

Contact: Miryeong Kwon (mkwon@camelab.org)

KAIST CAMEL