Large-scale Graph Neural Network Services through Computational SSD and In-Storage Processing Architectures

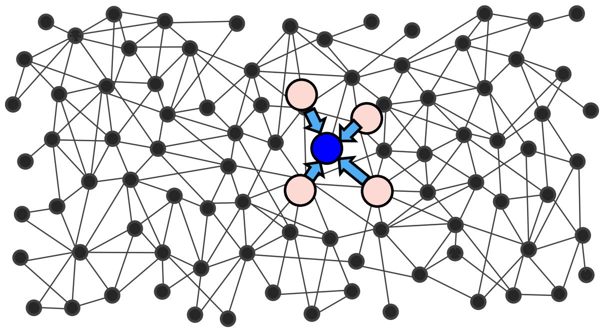
Miryeong Kwon, Donghyun Gouk, Sangwon Lee, Myoungsoo Jung

Computer Architecture and Memory systems Laboratory

First Step

High-level summary of talk

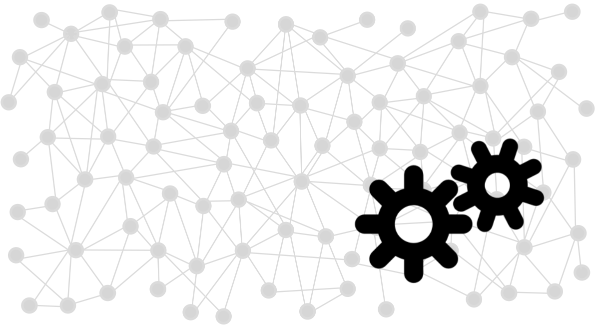
GNN have shown great success



High accuracy

Well accelerated

GNN preprocessing is missed out on

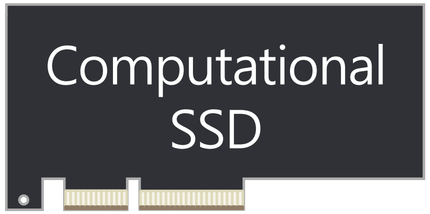


Current GNN works are only focusing on GNN algorithms

Now, we need "HolisticGNN"

GNN preprocessing + GNN algorithm

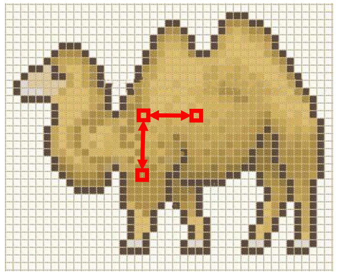
By leveraging



Graph Neural Networks (GNN)

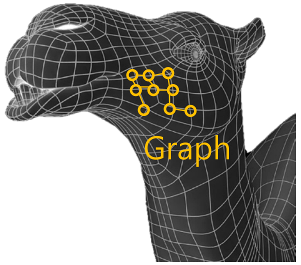
Why is it emerging?

Conventional CNN Model



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

Emerging GNN Model



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

How can GNN algorithm learn the relationship?

Response of CNN model

"Women near the sofa"

Query image

Characteristic: "pain"

Response of GNN model

"pain"

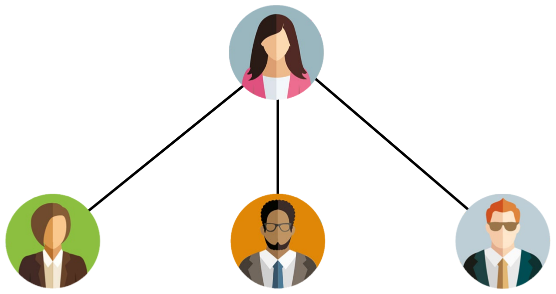
Image source: Personalized Image Retrieval with Sparse Graph Representation Learning (KDD'20)

Graph Neural Networks (GNN)

GNN algorithm

What do we have to do before GNN algorithm execution?

Input

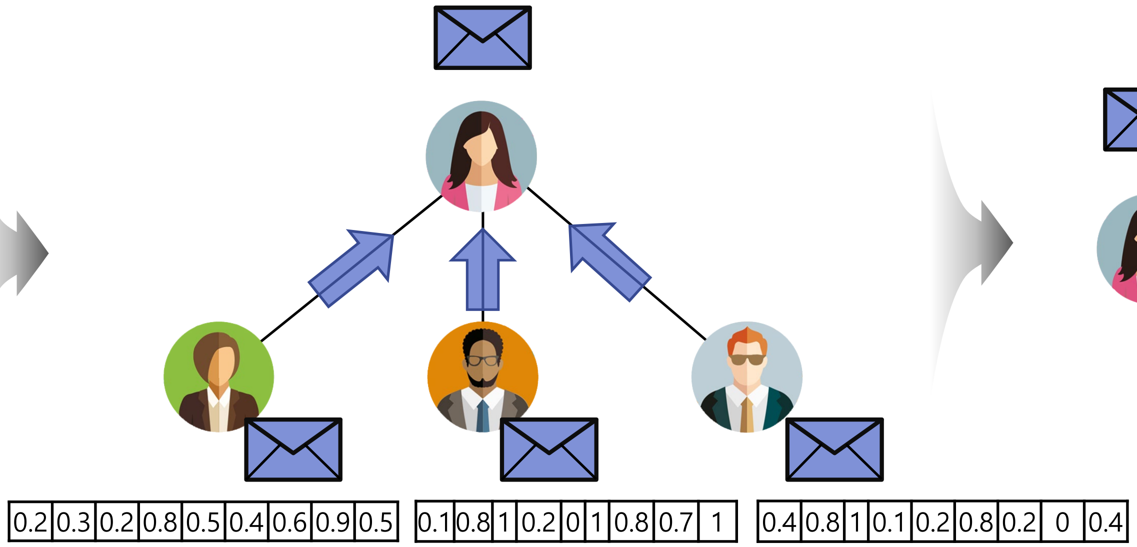


Graph structure

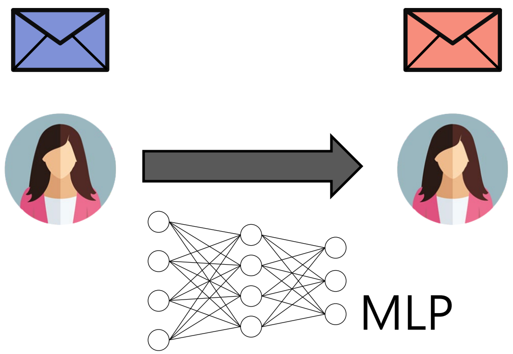
	0.1	0.8	1	0.2	0	1	0.8	0.7	1
	0	1	0.1	1	0.8	0.1	1	0.2	0
	0.4	0.8	1	0.1	0.2	0.8	0.2	0	0.4
	0.2	0.3	0.2	0.8	0.5	0.4	0.6	0.9	0.5

Node embedding

#1: Aggregation



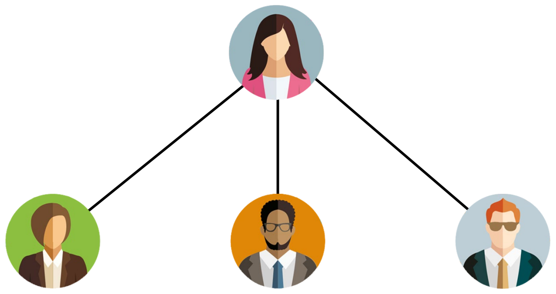
#2: Transformation



Graph Neural Networks (GNN)

GNN algorithm

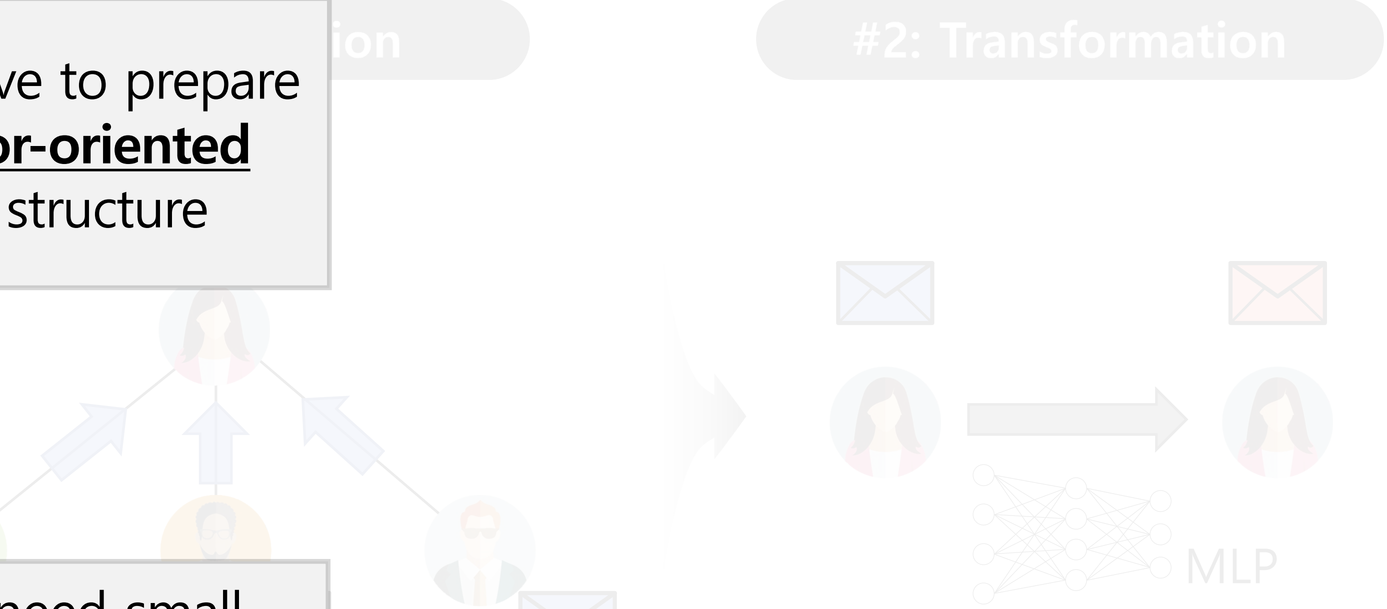
Input



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 **<u>neighbor-oriented</u>** graph structure

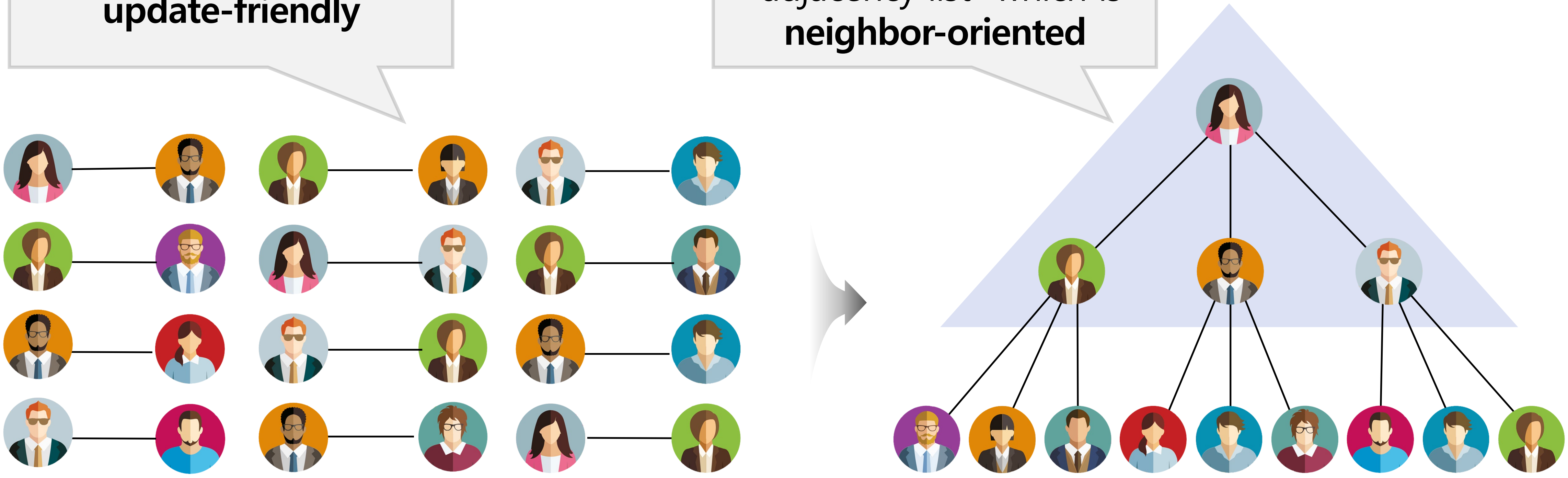
❷ We need small input data which **<u>can be loaded</u>** into accelerator memory

GNN Preprocessing

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

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

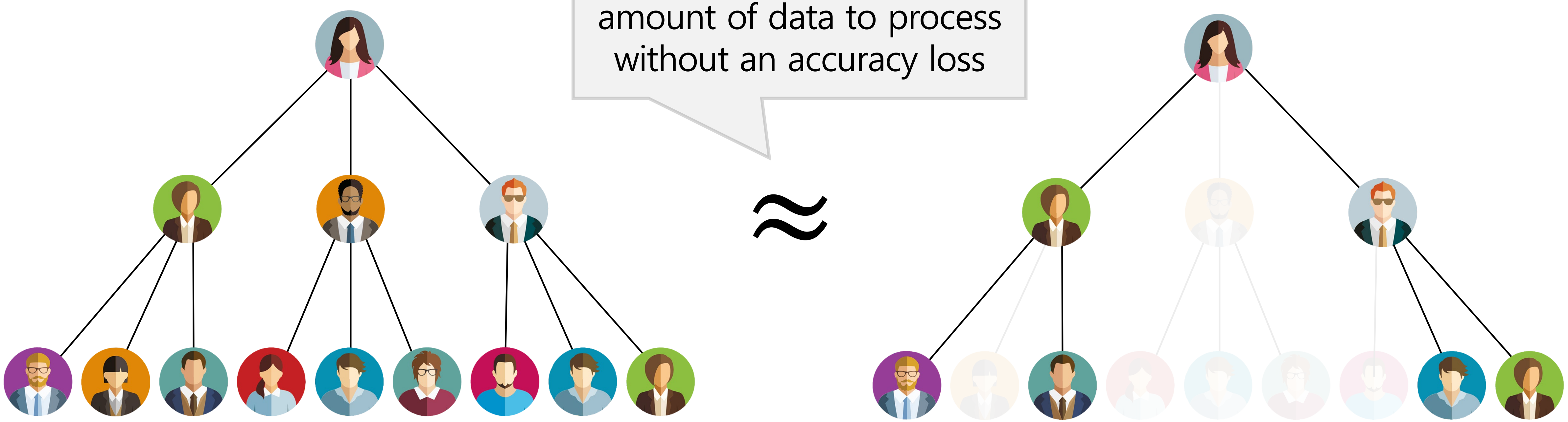
Graph preprocessing converts edge array to “adjacency list” which is **neighbor-oriented**



GNN Preprocessing

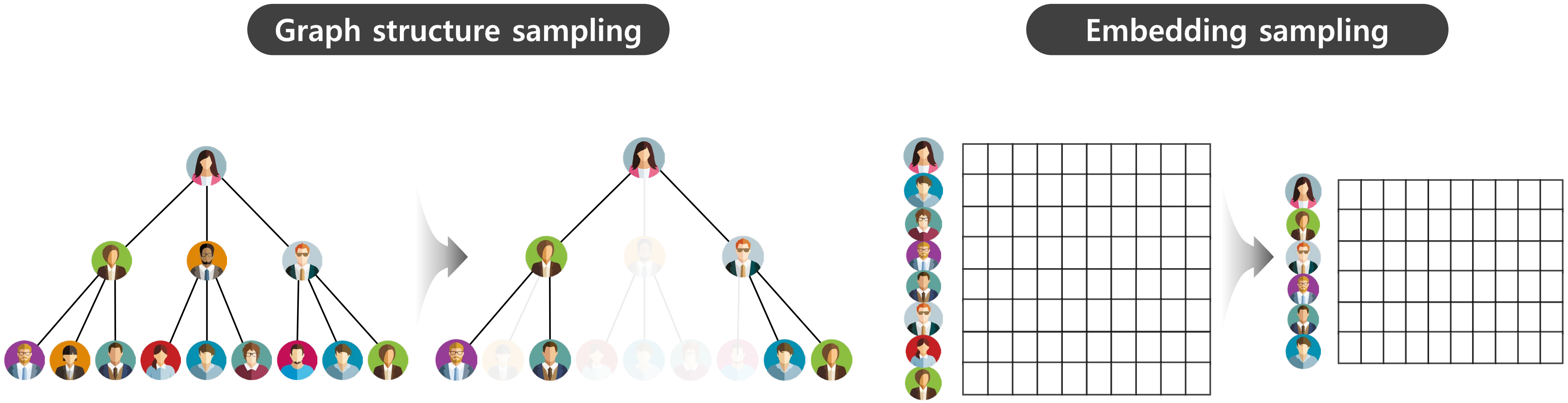
Batch preprocessing: to prepare *small graph*

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



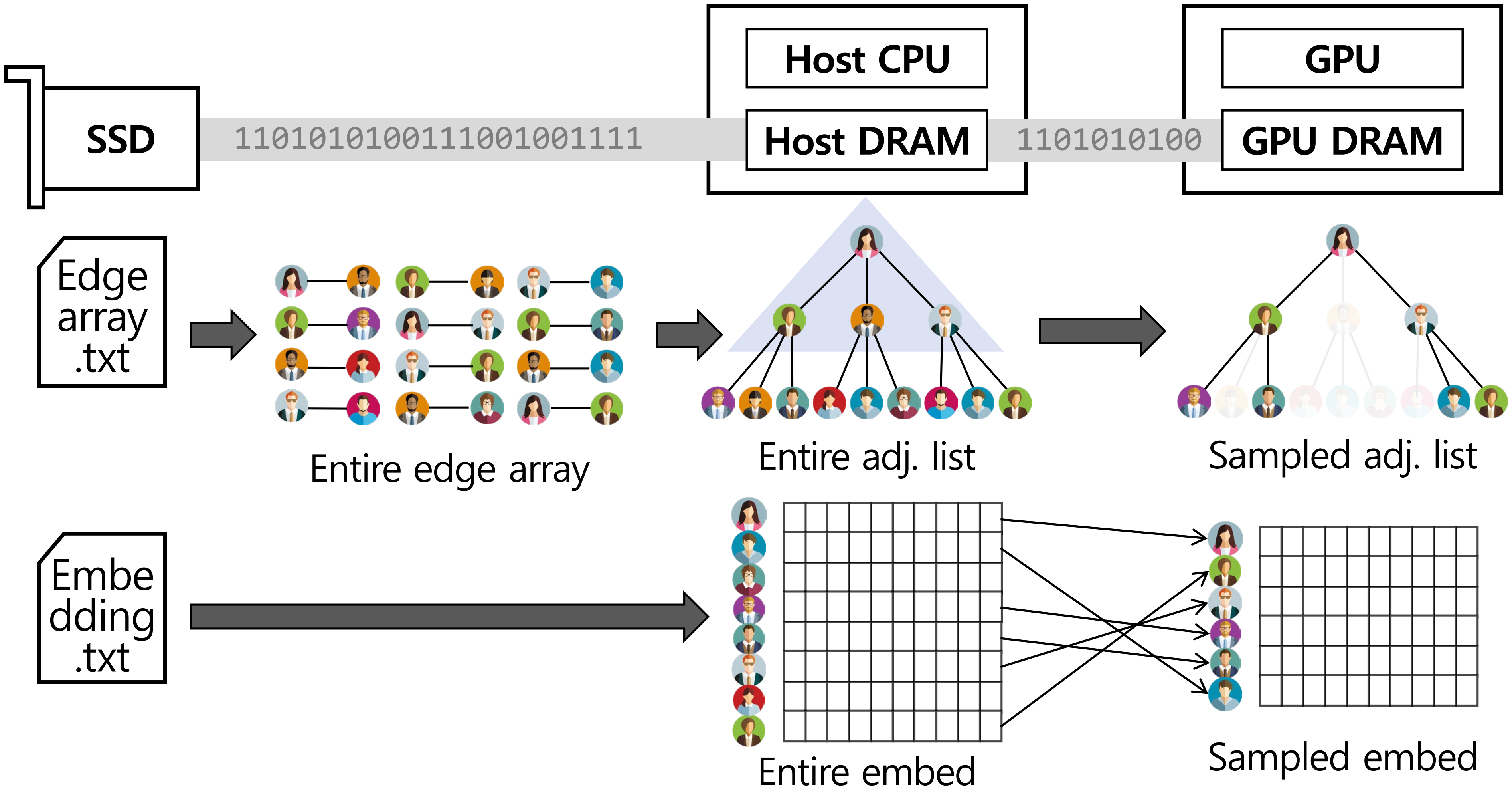
GNN Preprocessing

Batch preprocessing: to prepare *small graph*



End-to-End GNN Inference

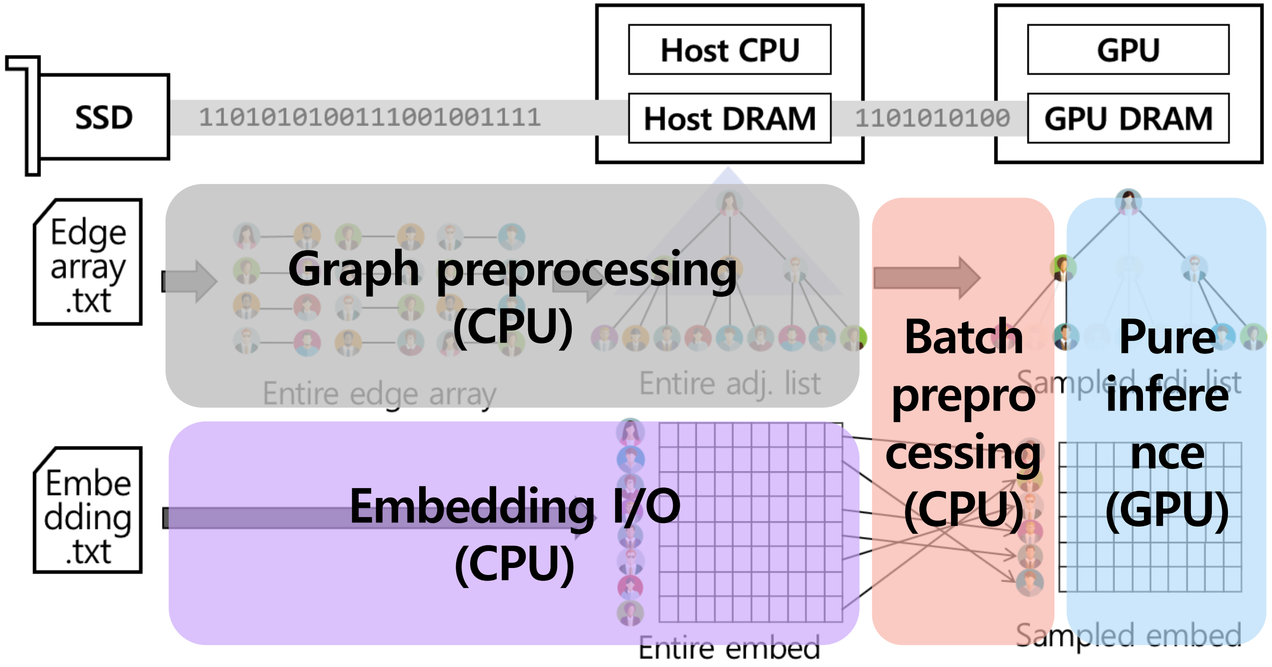
Visualization

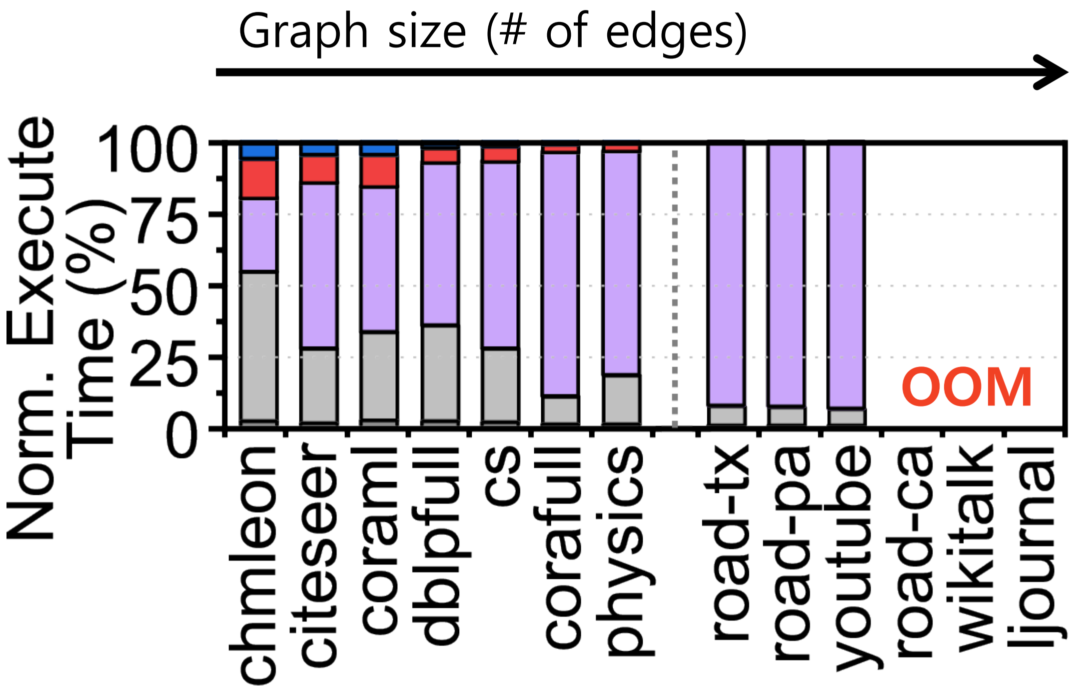


End-to-End GNN Inference

Execution time analysis

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





Design Questions

Then, what does GNN acceleration look like?

Graph preprocessing
(CPU)

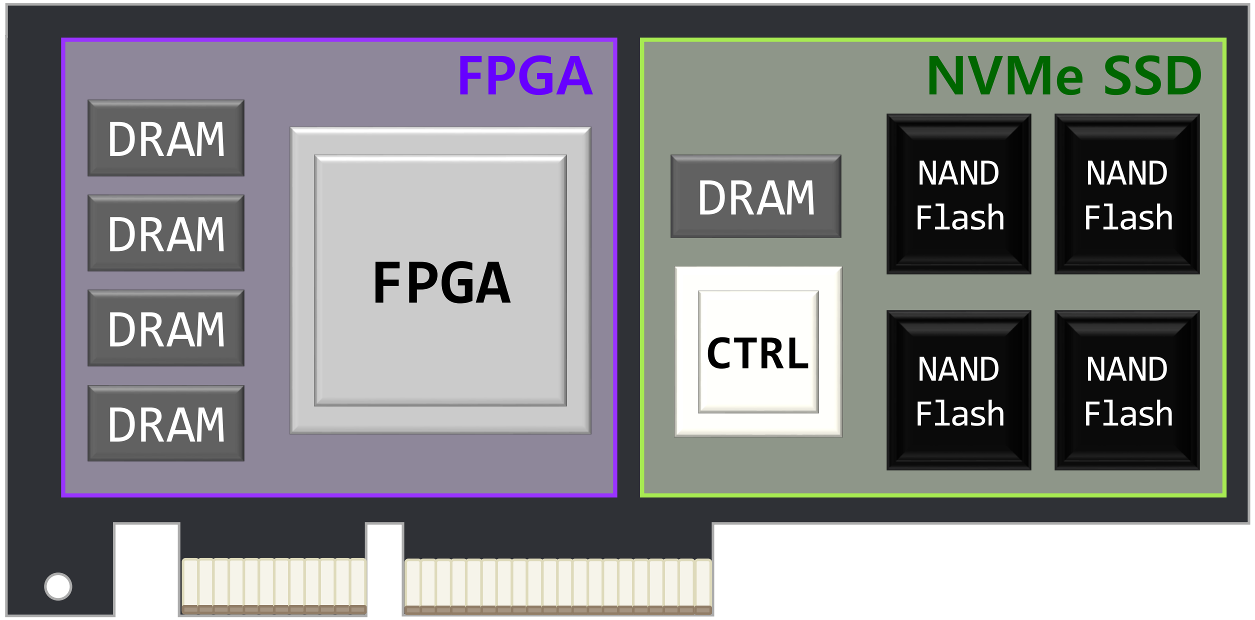
Store graph directly as a
neighbor-oriented format
(But also, update-efficient)

Embedding I/O
(CPU)

Process end-to-end GNN
inference near storage

HolisticGNN

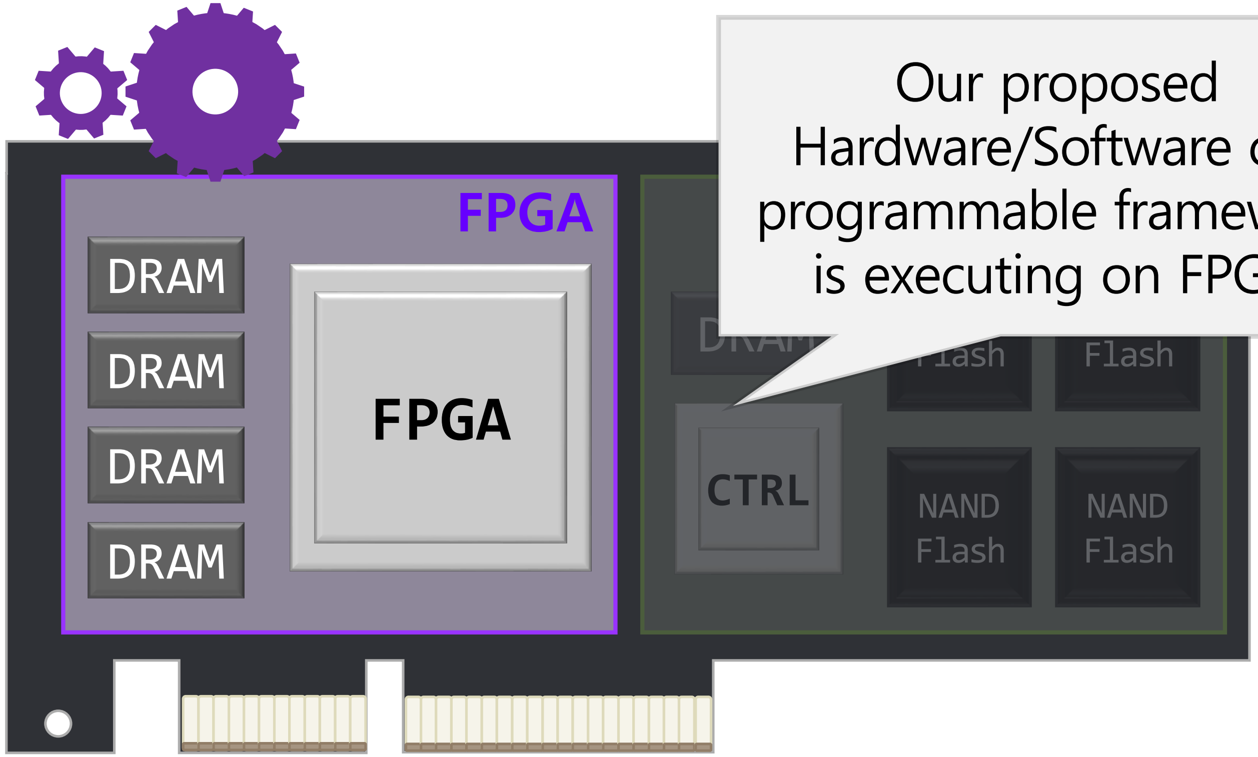
Adopts the concept of computational SSD (CSSD)



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

HolisticGNN

“Hardware/Software Co-Programmable Framework” for CSSDs

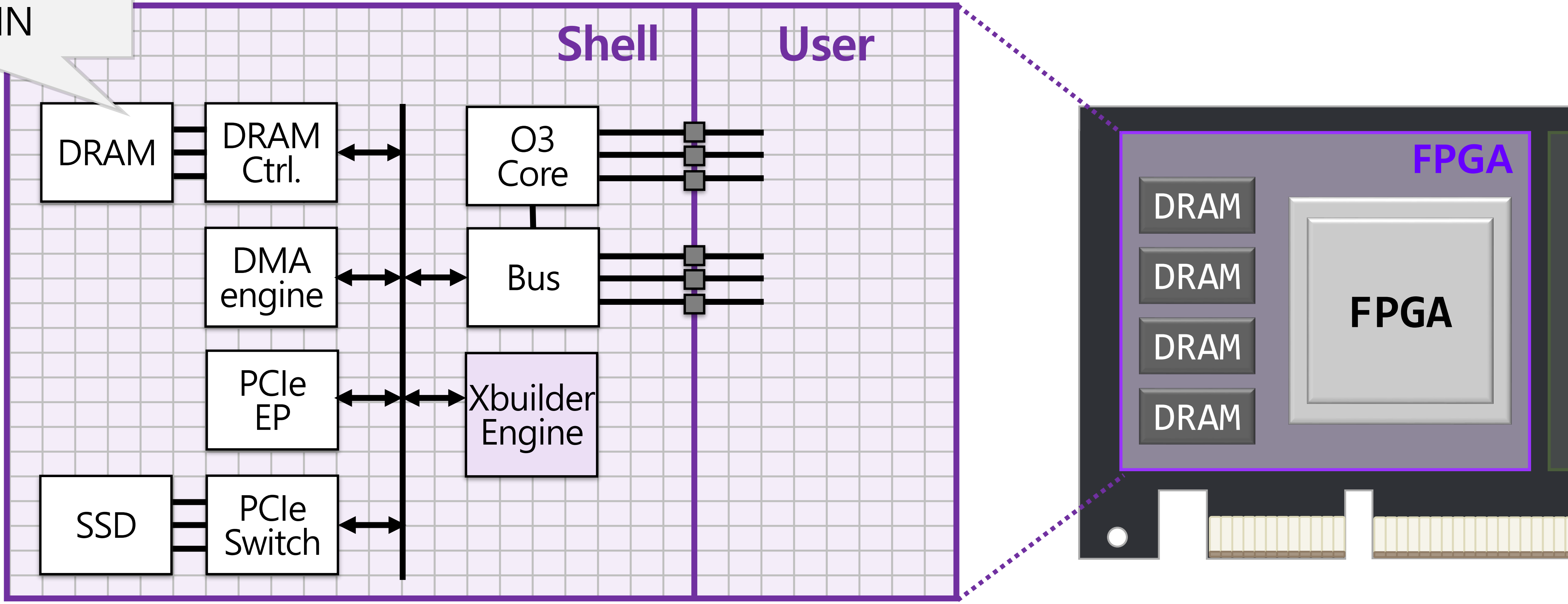


Our proposed Hardware/Software co-programmable framework is executing on FPGA

HolisticGNN

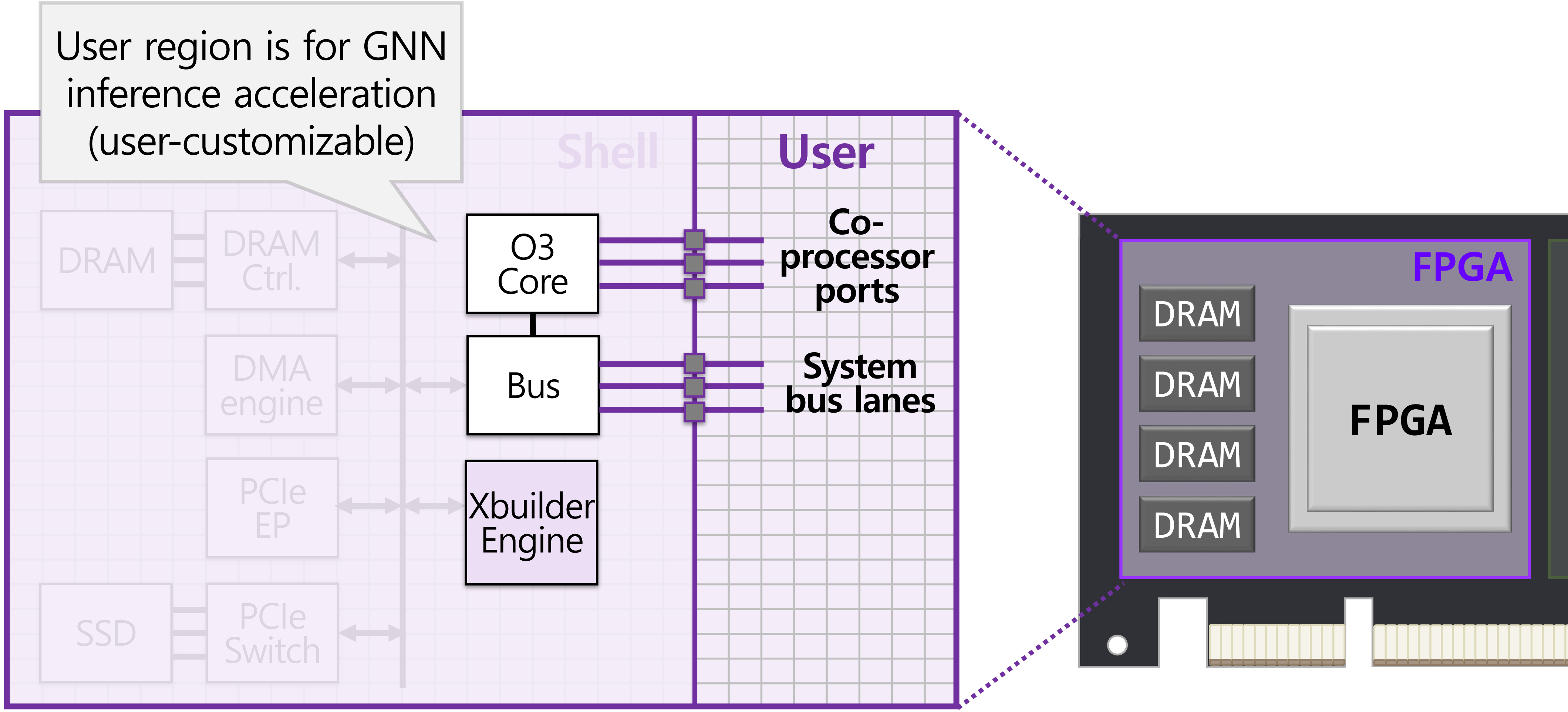
“Hardware/Software Co-Programmable Framework” for CSSDs

Shell region is for essential HW logics of HolisticGNN



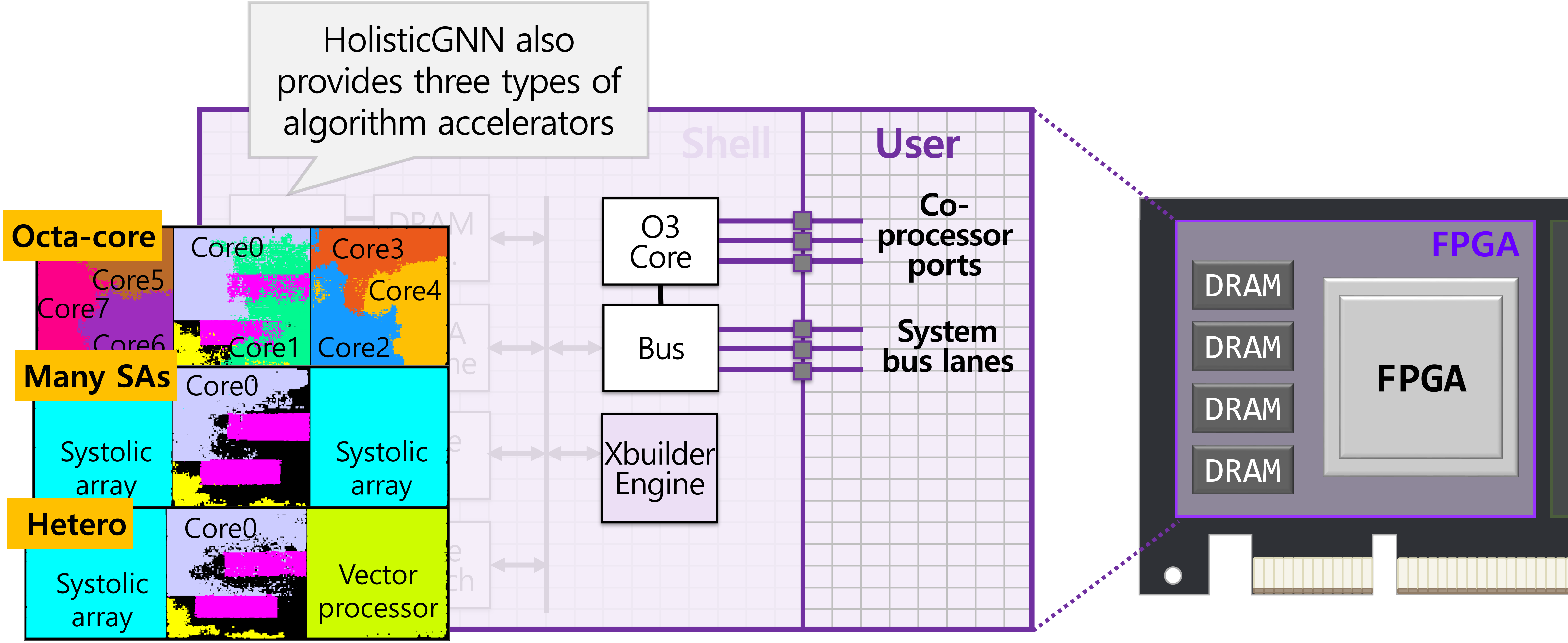
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



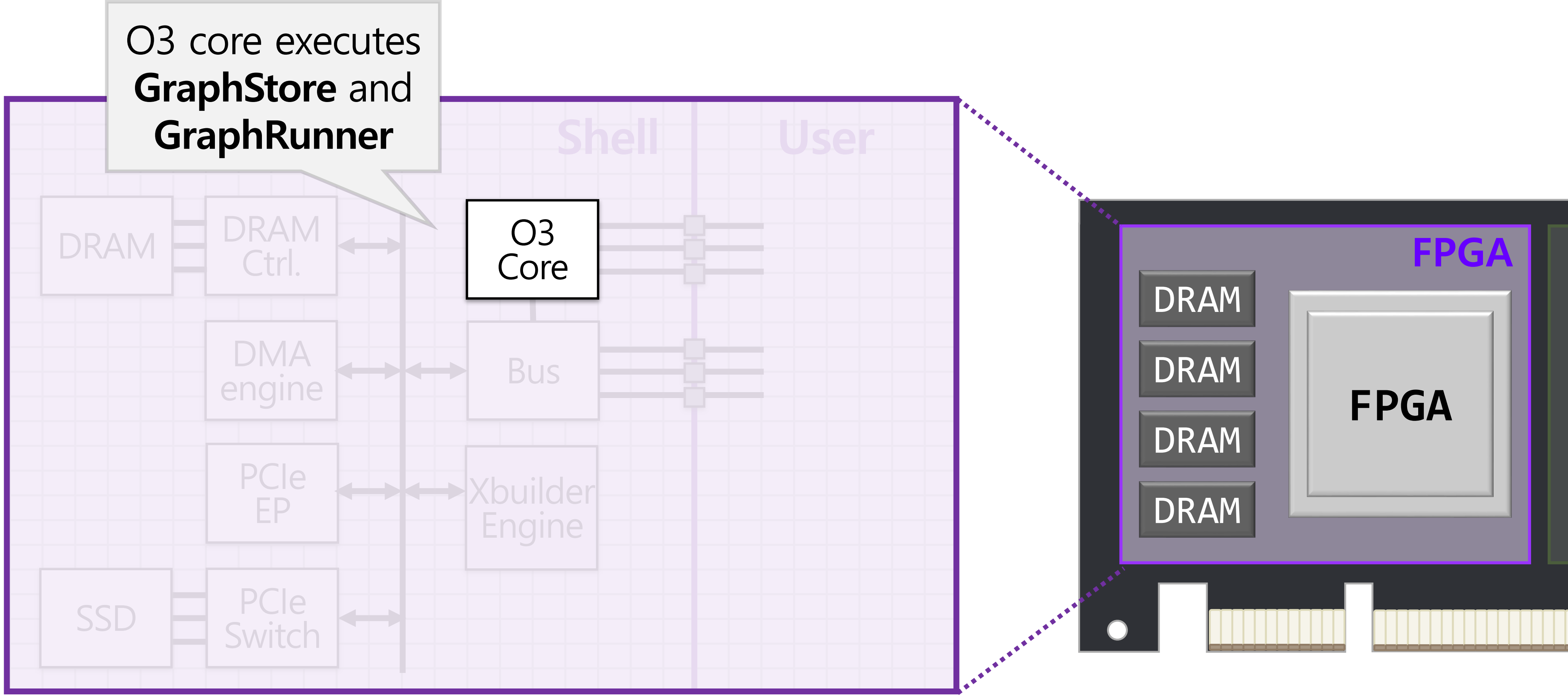
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



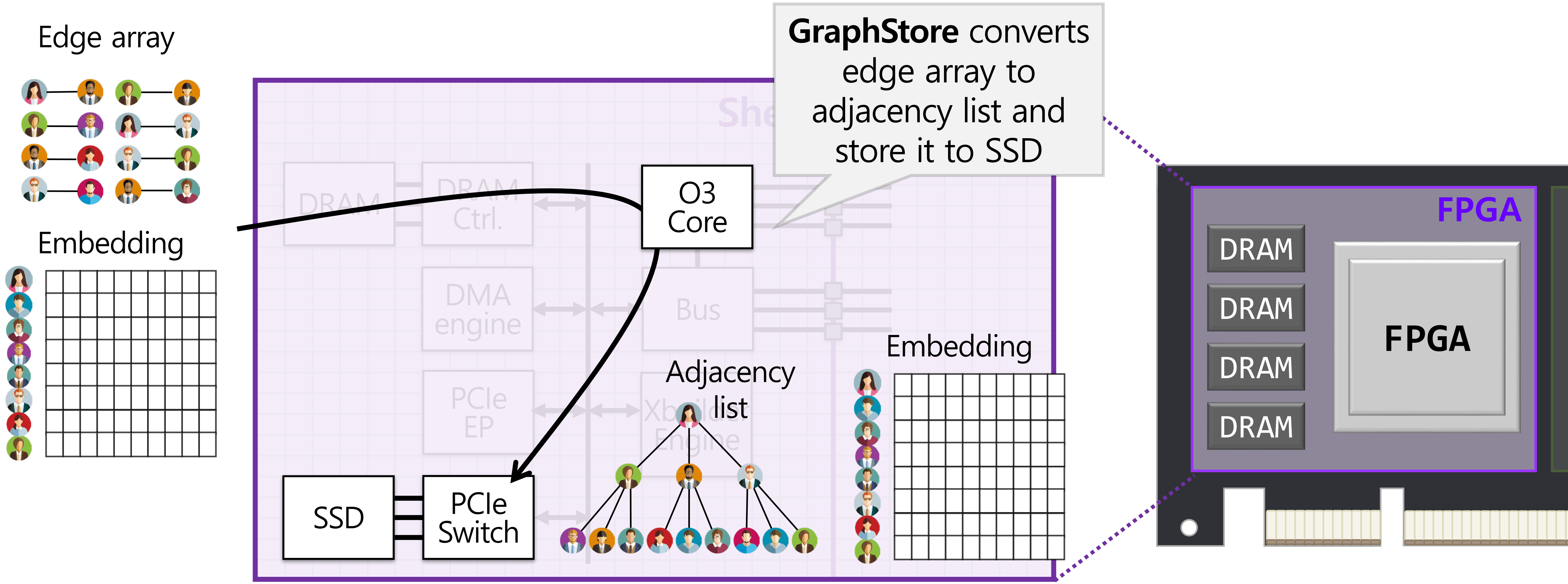
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



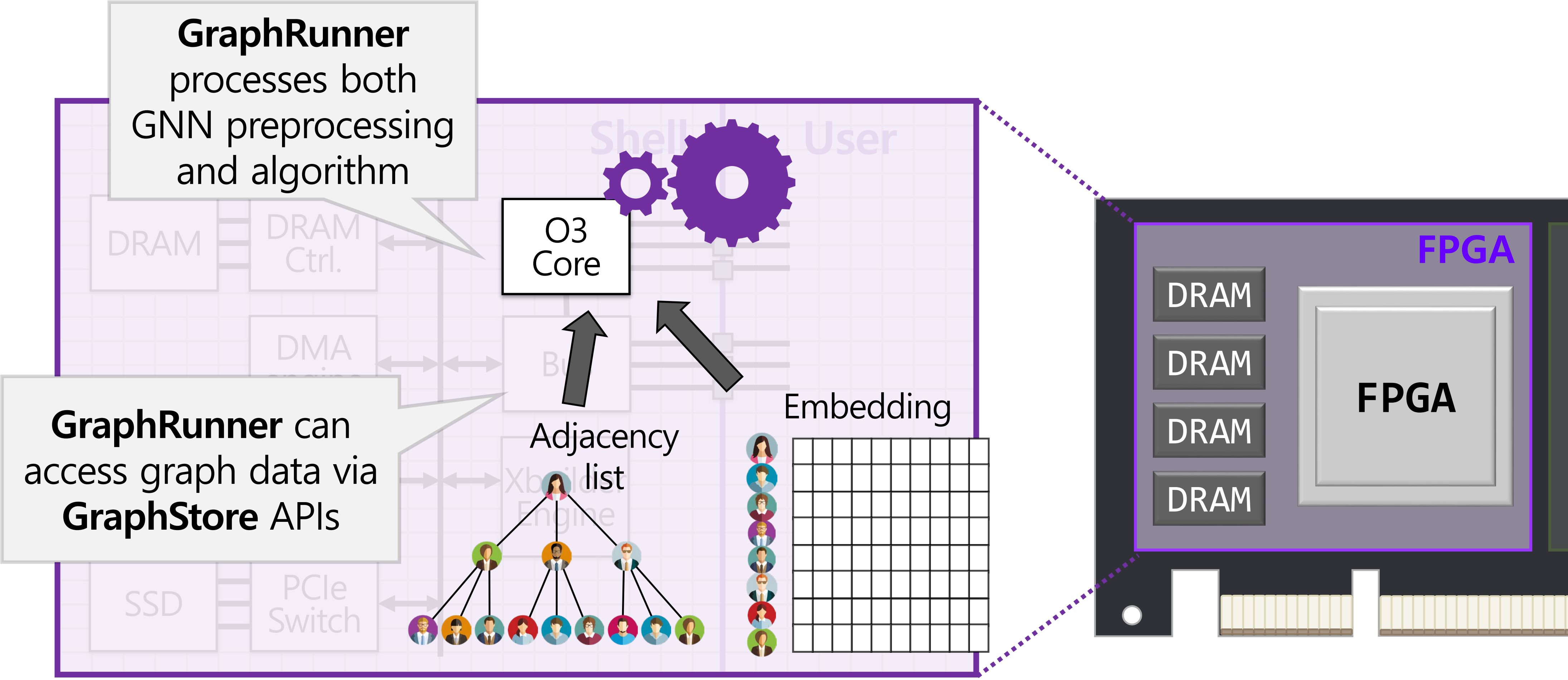
HolisticGNN

"Hardware/Software Co-Programmable Framework" for CSSDs



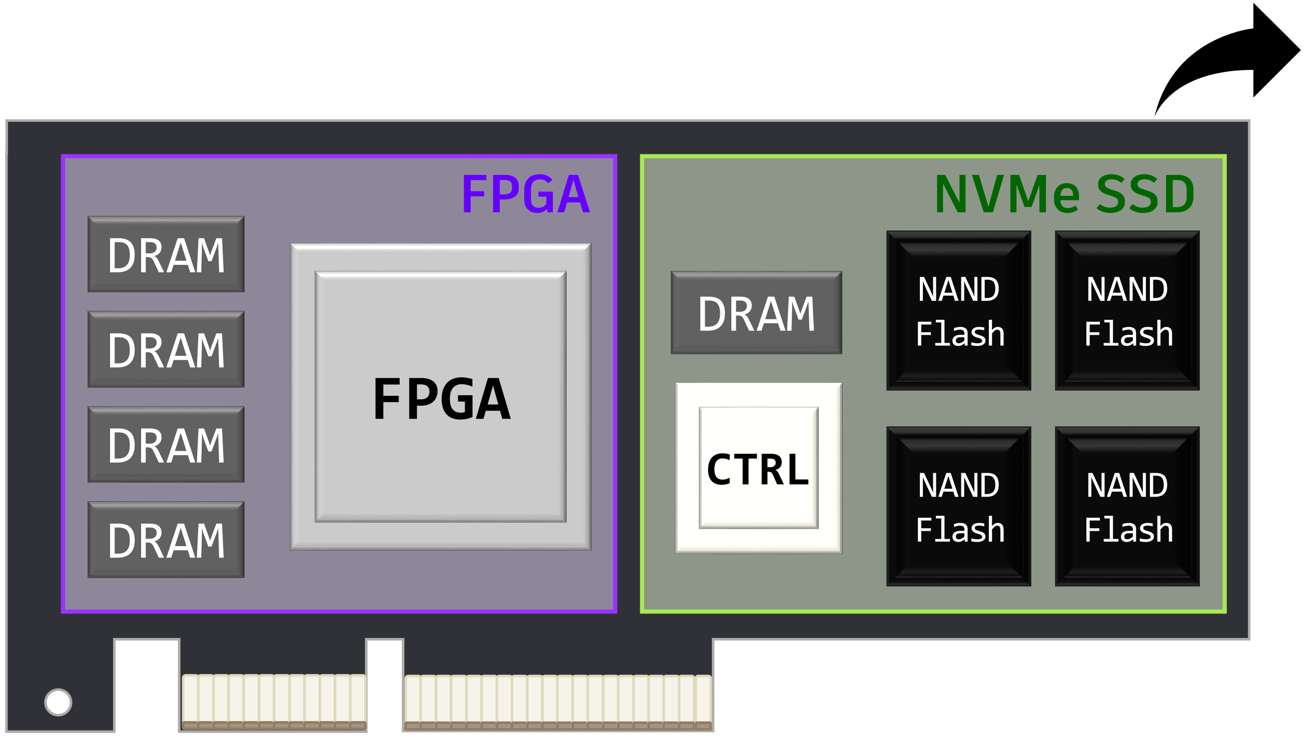
HolisticGNN

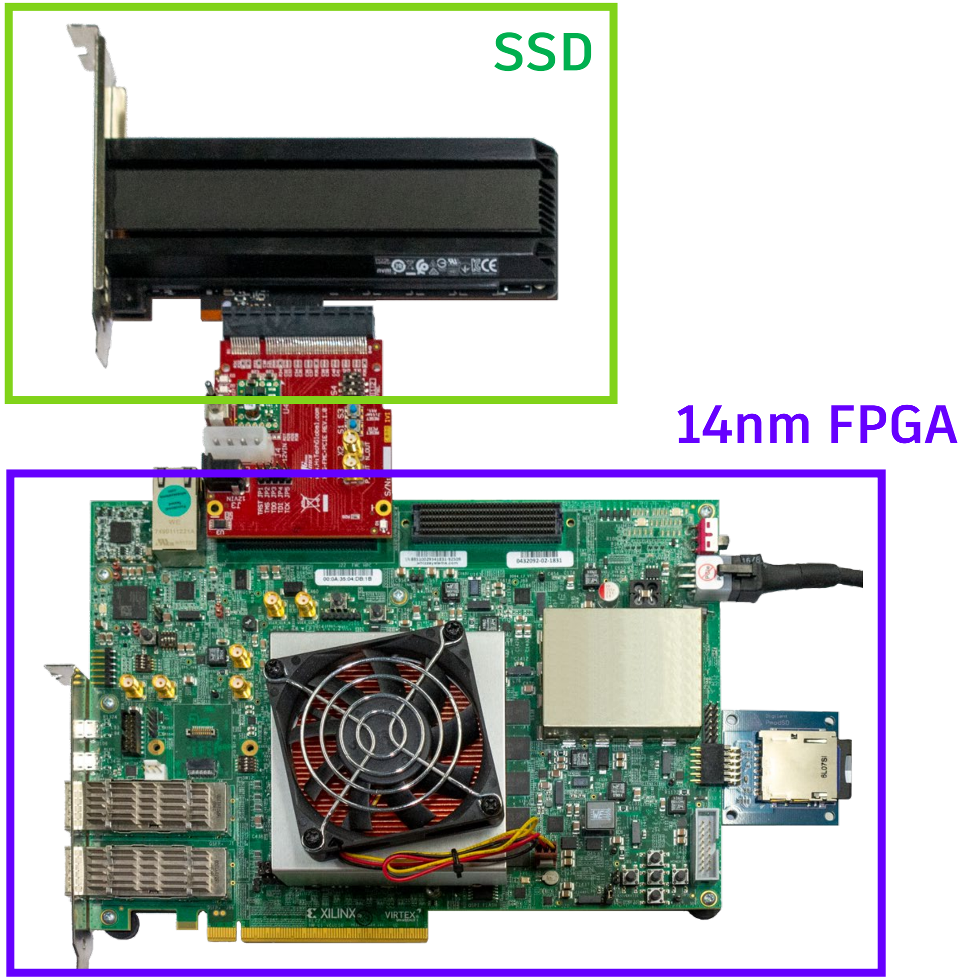
"Hardware/Software Co-Programmable Framework" for CSSDs



Experimental Setup

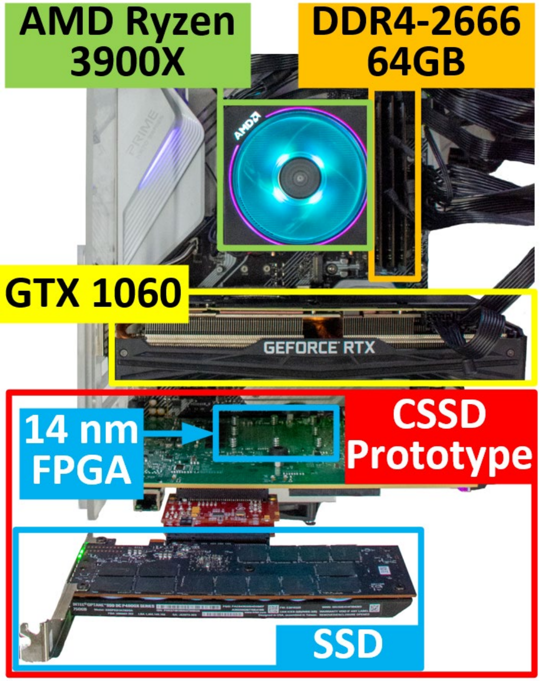
HolisticGNN prototype

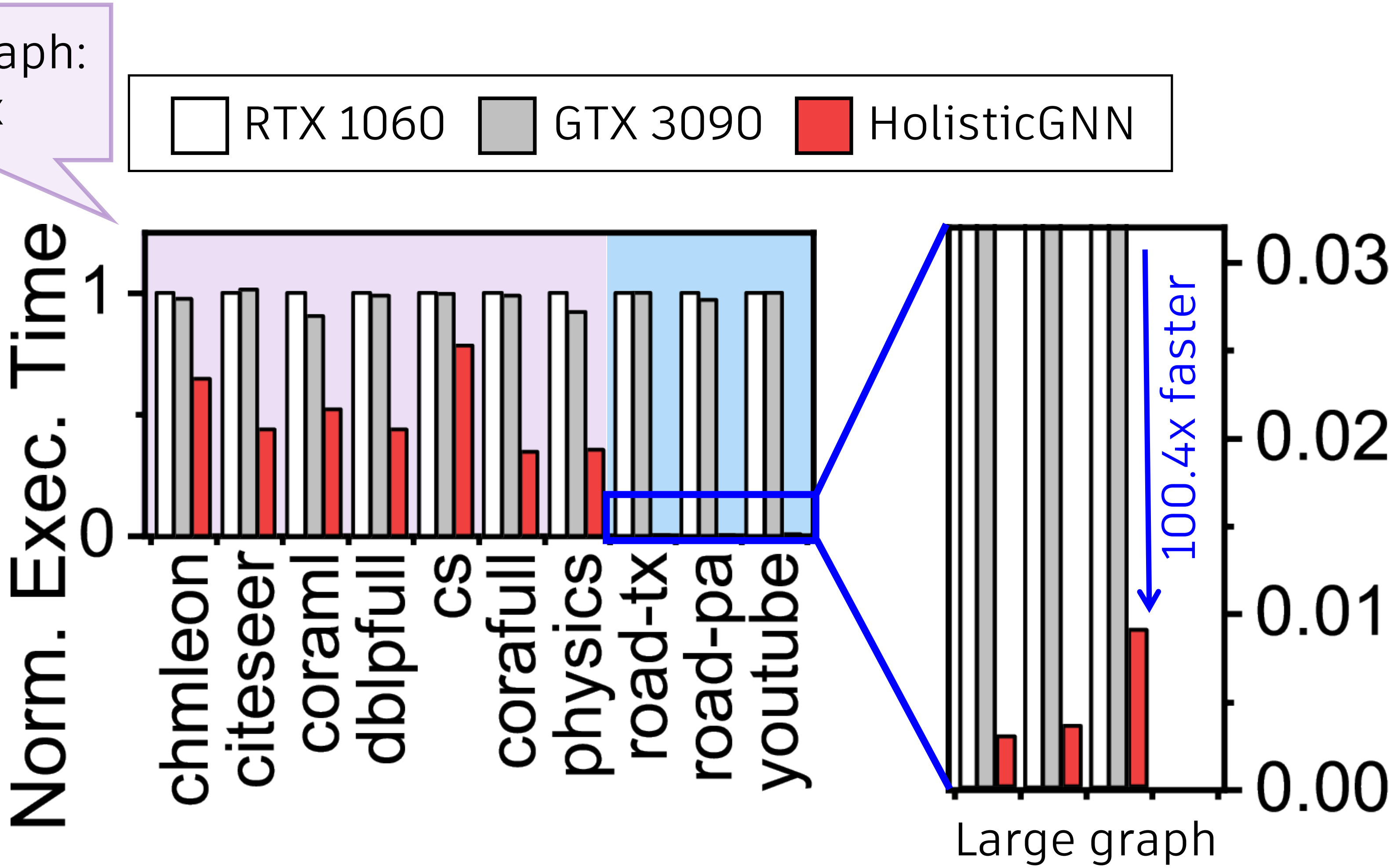




Evaluation Results

End-to-End latency comparison



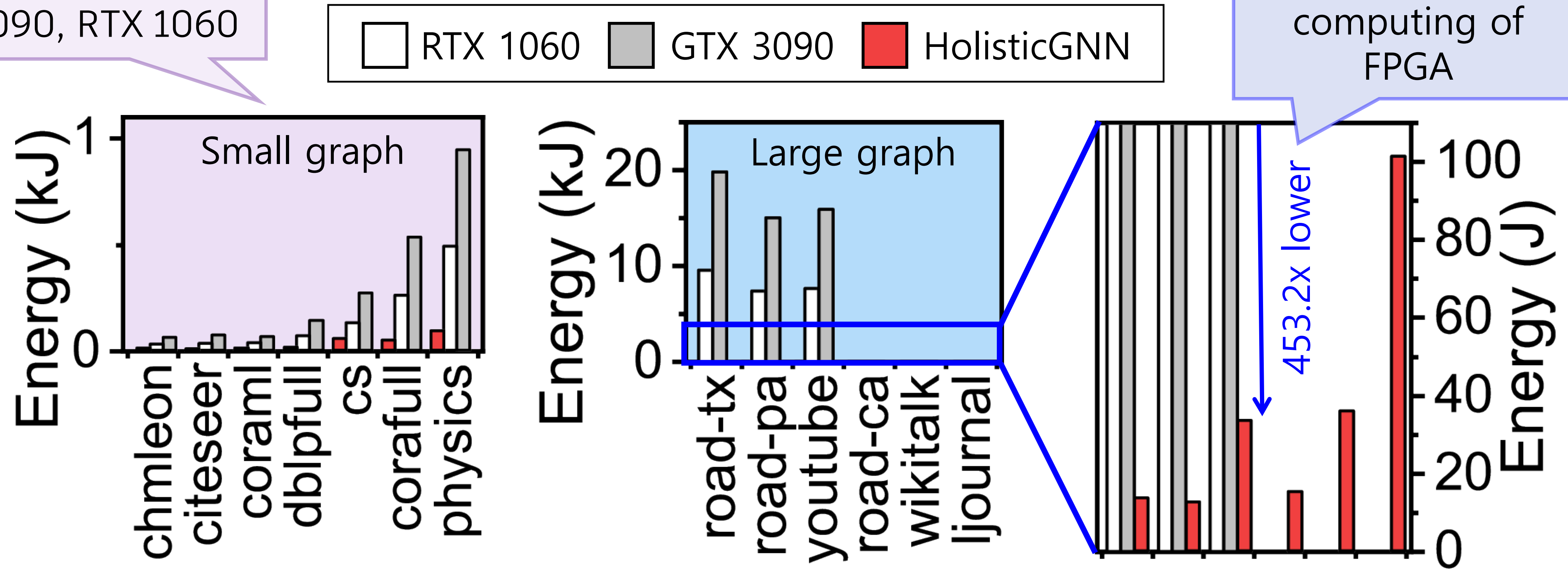


Evaluation Results

Energy Consumption

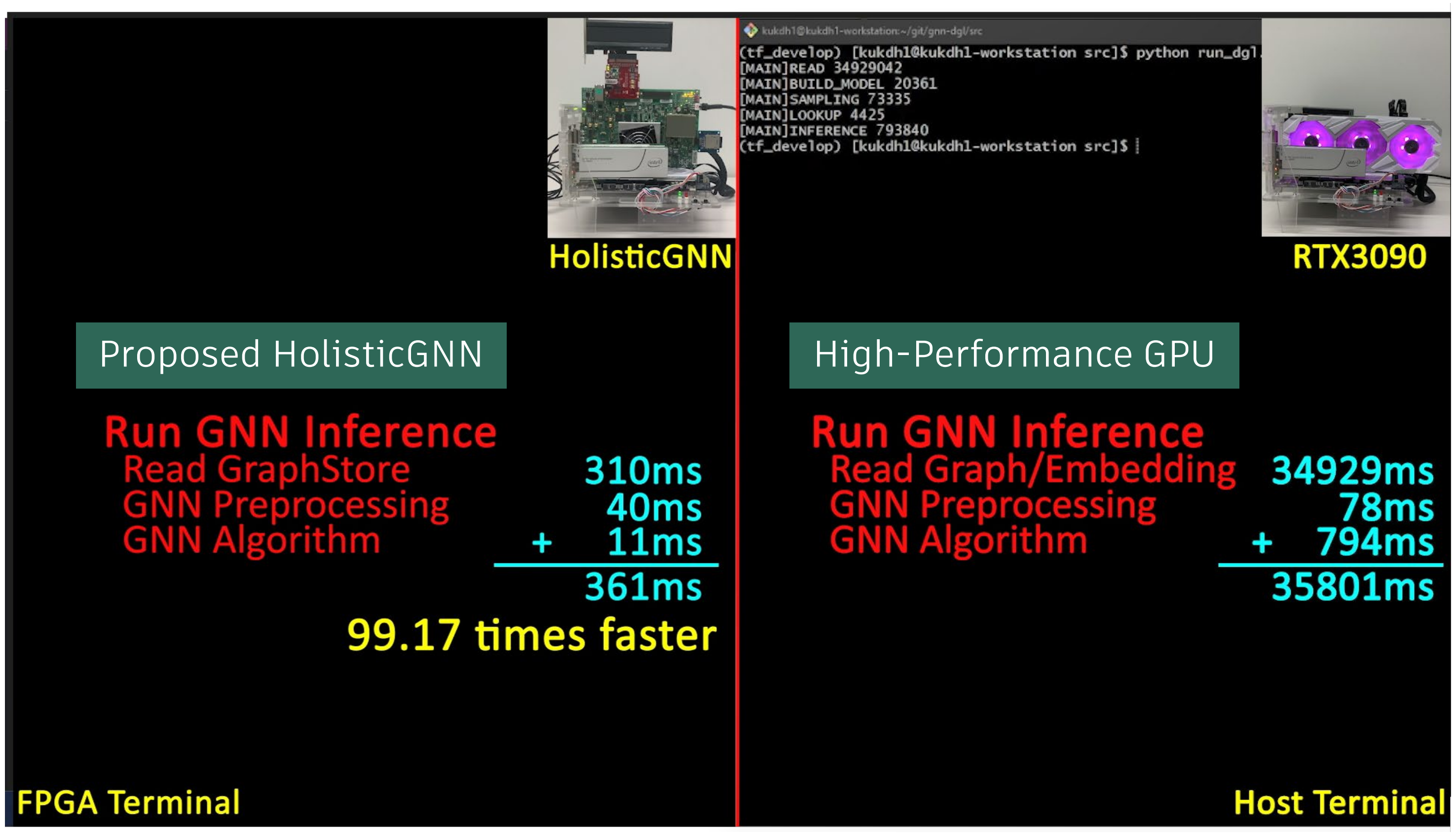
33.2x and 16.3x better than GTX 3090, RTX 1060

Due to low-power computing of FPGA



Demonstration

GNN execution in our HolisticGNN prototype



Demonstration Video Link: https://www.youtube.com/watch?v=b5fZBESH1TM

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)

Original publication: M. Kwon, D. Gouk, S. Lee, and M. Jung. USENIX FAST 2022. Hardware/Software Co-Programmable Framework for Computational SSDs to Accelerate Deep Learning Service on Large-Scale Graphs (https://www.usenix.org/system/files/fast22-kwon.pdf)

Acknowledgment: This research is supported by Samsung Research Funding & Incubation Center of Samsung Electronics (SRFC-IT2101-04). Myoungsoo Jung is the corresponding author.