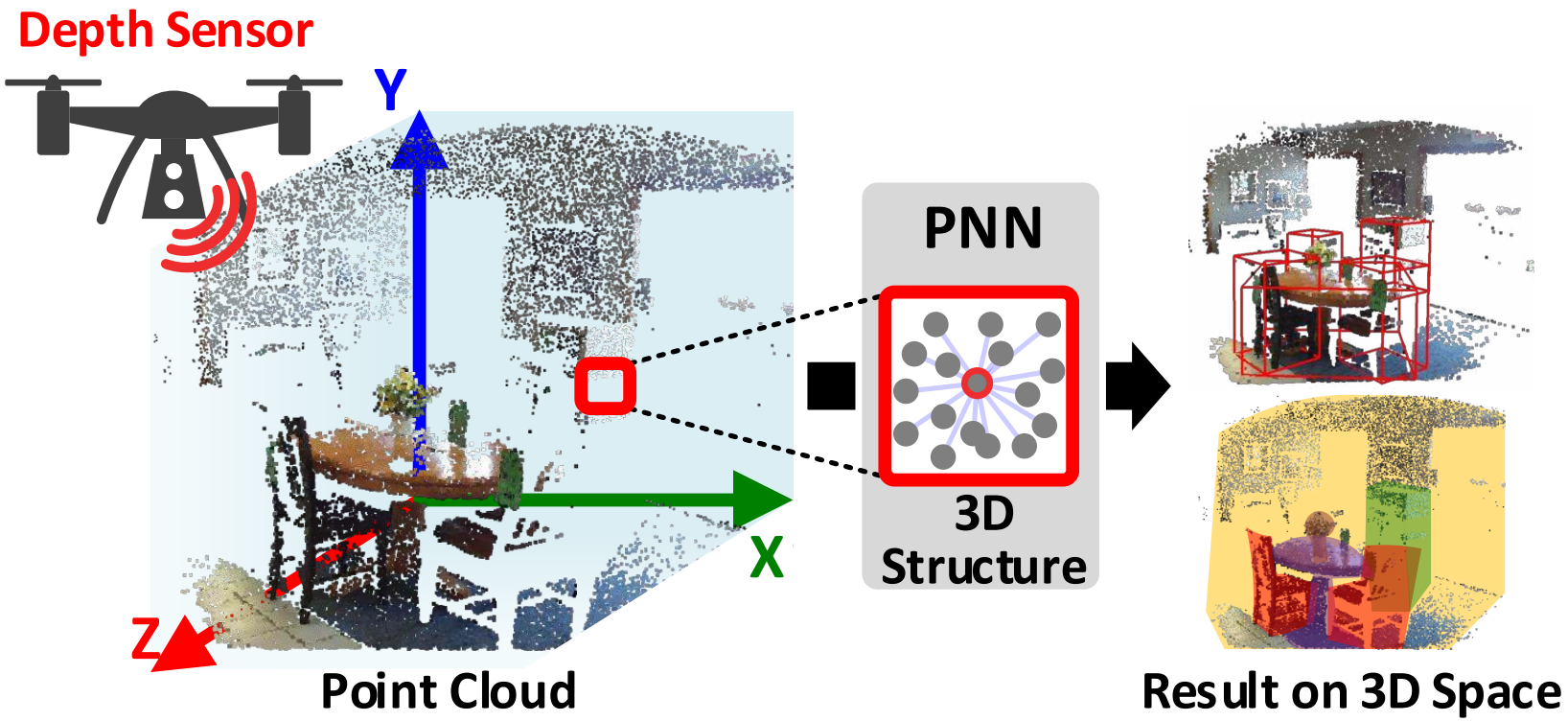
PNNPU: A Fast and Efficient 3D Point Cloud-based Neural Network Processor with Block-based Point Processing for Regular DRAM Access

Sangjin Kim, Juhyoung Lee,
Dongseok Im and Hoi-Jun Yoo

Semiconductor System Lab.
School of EE, KAIST

PNN* for Intelligent 3D Vision

- **Intelligent 3D Vision on Mobile Devices**
  - Accurate & Robust Perception with 3D Structural Information
  - Mobile 3D Sensor Already Commercialized

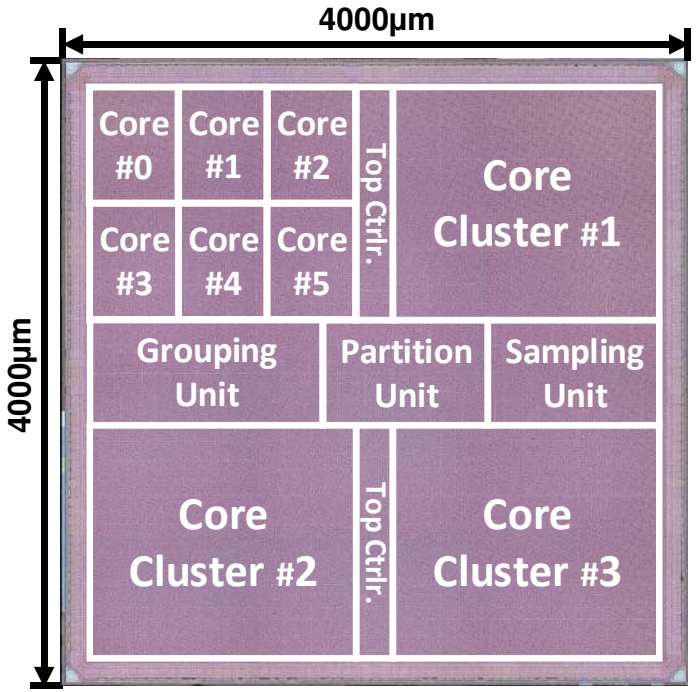


*PNN = Point Cloud-based Neural Network

PNNPU: A Fast & Efficient PNN Processor

- **Block-based Point Managing & Processing**

Block-based Point Managing	**Page-based Point MMU**	**Linked List-based Page Table**
Block-based Point Processing	**Hierarchical-FPS**	**Block-skipping BQ**
Point Feature Extraction	**Skipping-based Max-pooling Prediction**	

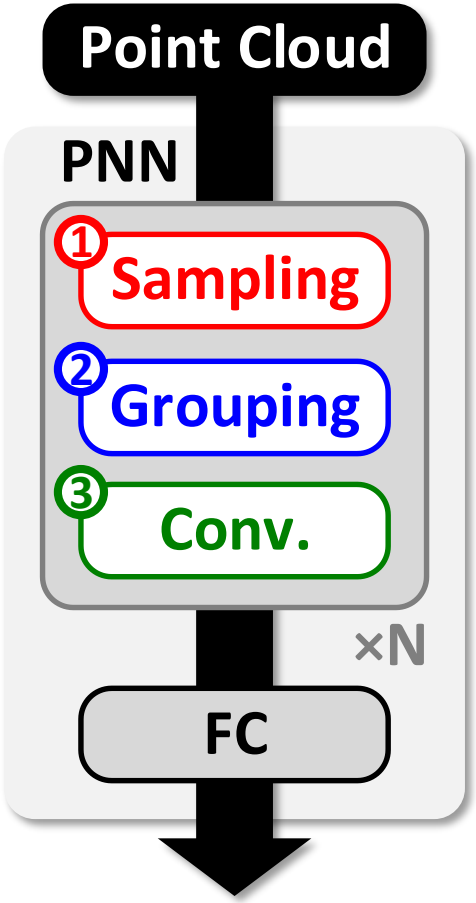


– **In-efficient Point Processing & Irregular Ext. DRAM Access**
➔ ***Block-based Method*** **w/ HW architecture & SW algorithm**

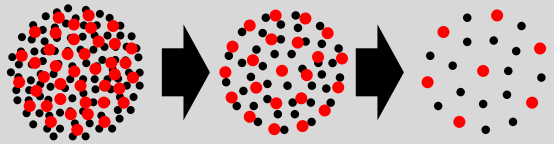
– **For Efficient Point Feature Extraction ➔ Max-pooling Prediction**

PNN Operation

- 3 Stages: Sampling ➔ Grouping ➔ Convolution Layer



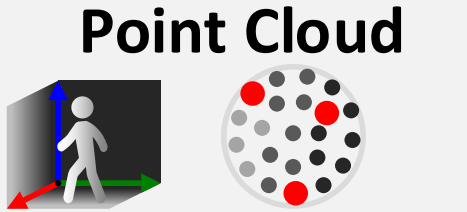
① *Farthest Point Sample*



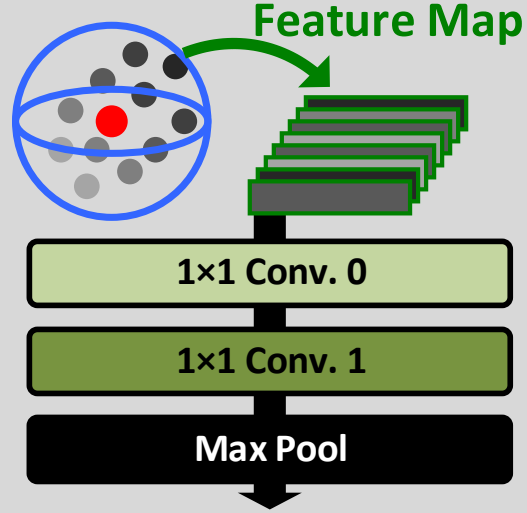
Select Points Most Evenly

② *Ball Query*

Organize 3D Structure



③ Convolution

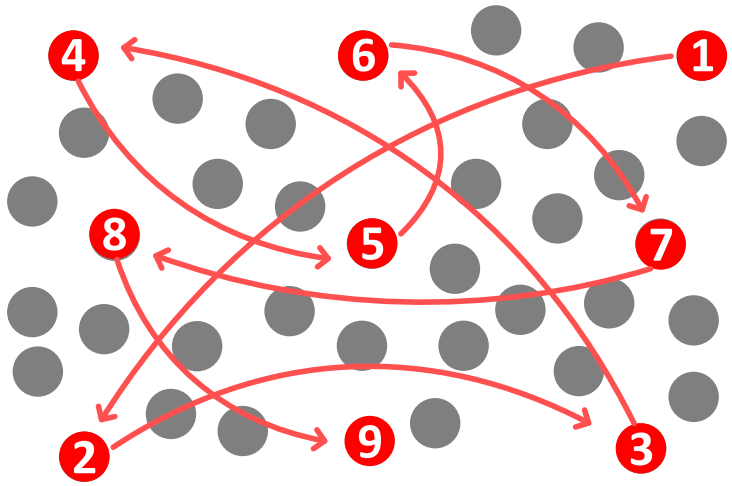


Understand 3D Structure

PNN Challenge: 1) Point Processing

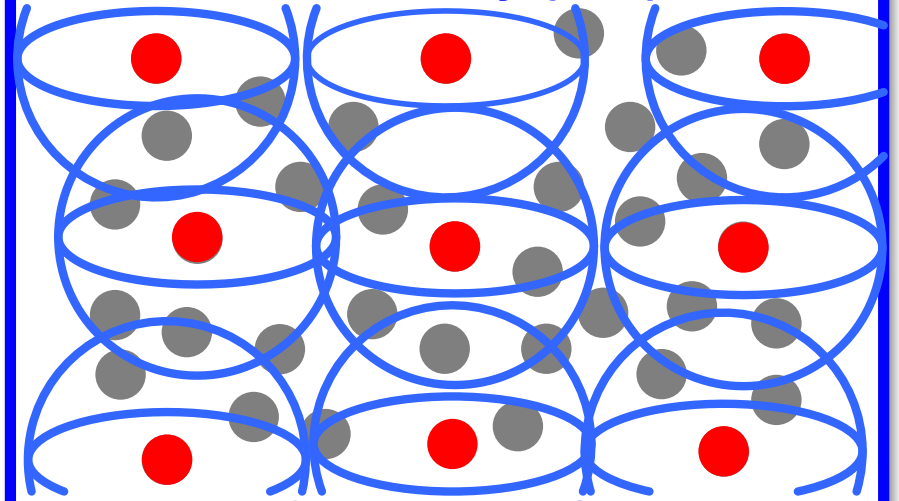
- **Complex & Inefficient Point Processing**
  - **Computation & Memory-access ∝ O(n2)**

Farthest Point Sampling (FPS)



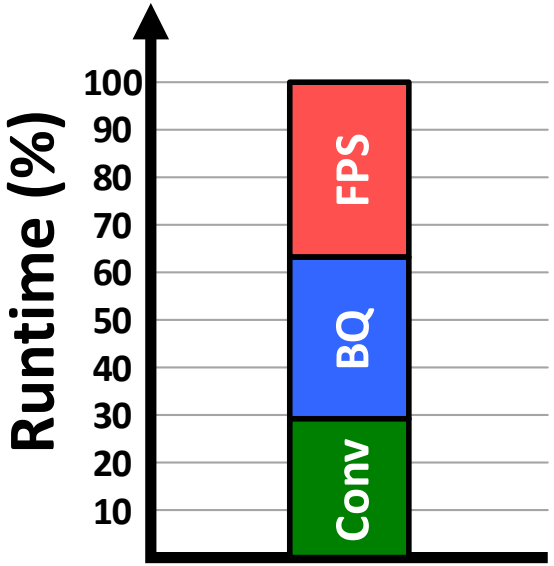
Iteratively Find Farthest Point
➔ *$O(n^2)$ Distance Compare*

Ball Query (BQ)



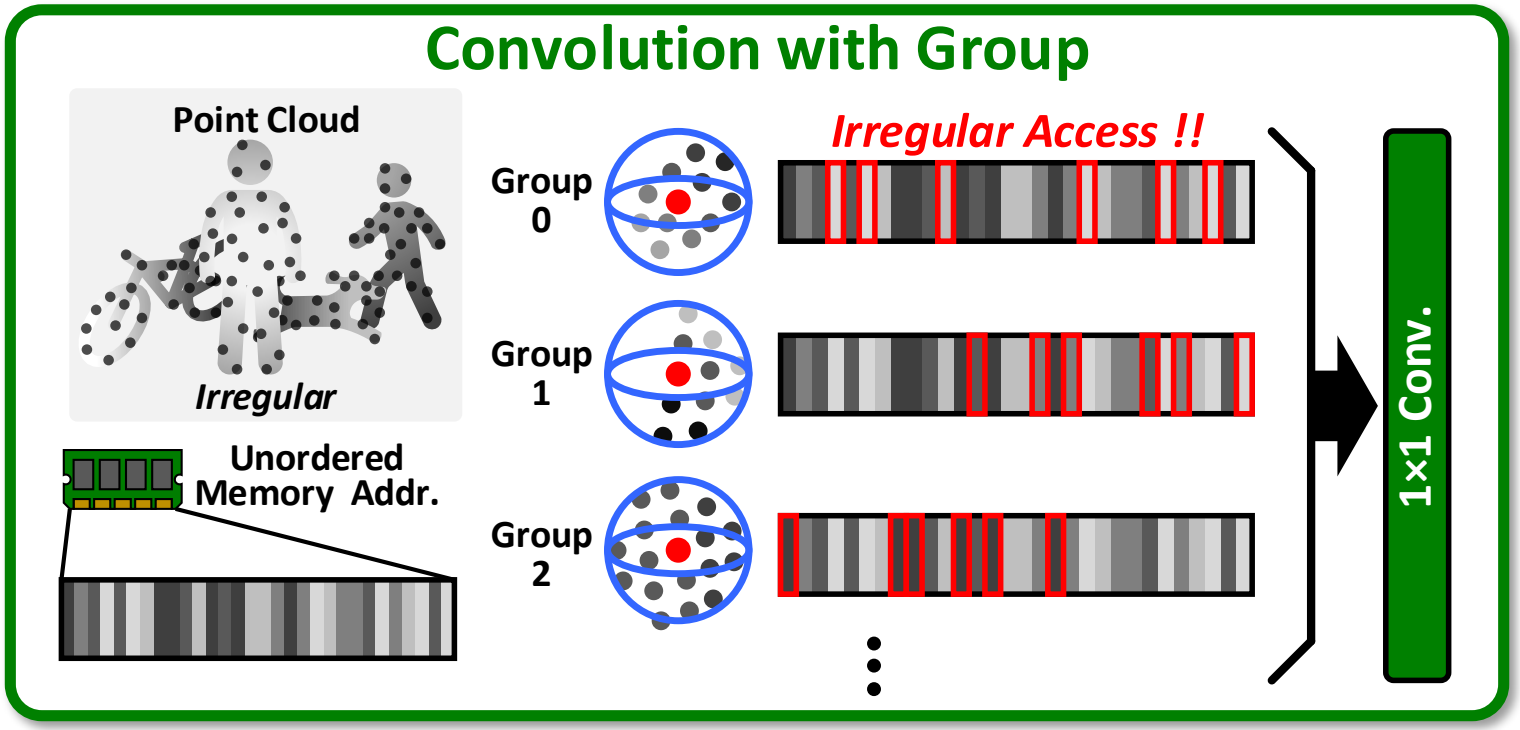
Find Point within R
➔ *$O(n^2)$ Distance Compare*

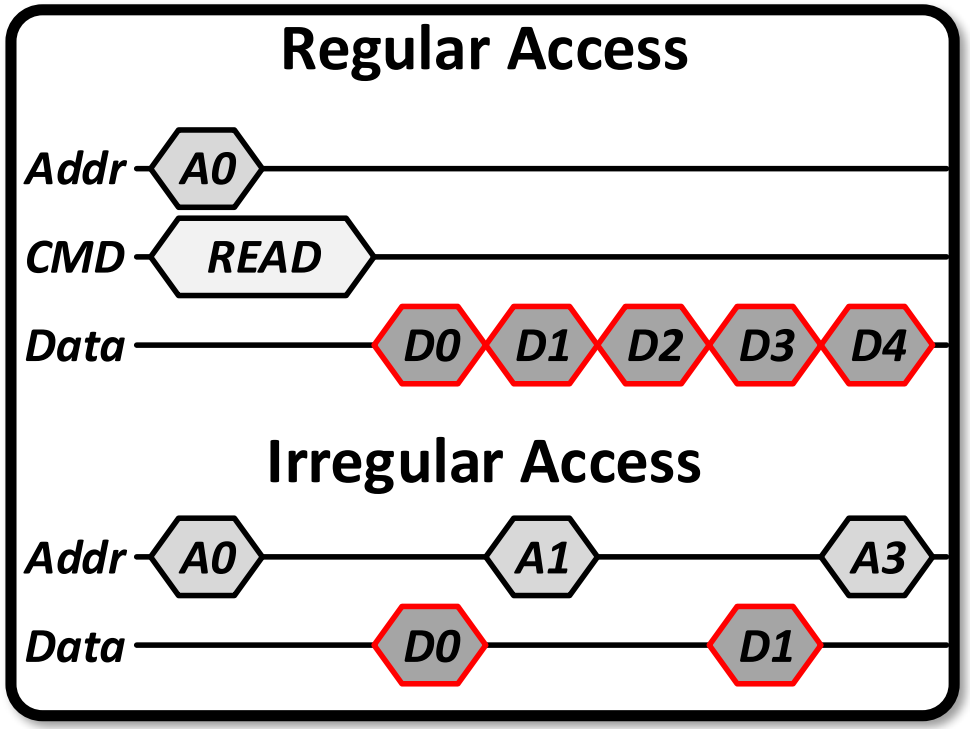
Latency on GPU



PNN Challenge: 2) Irregular DRAM Access

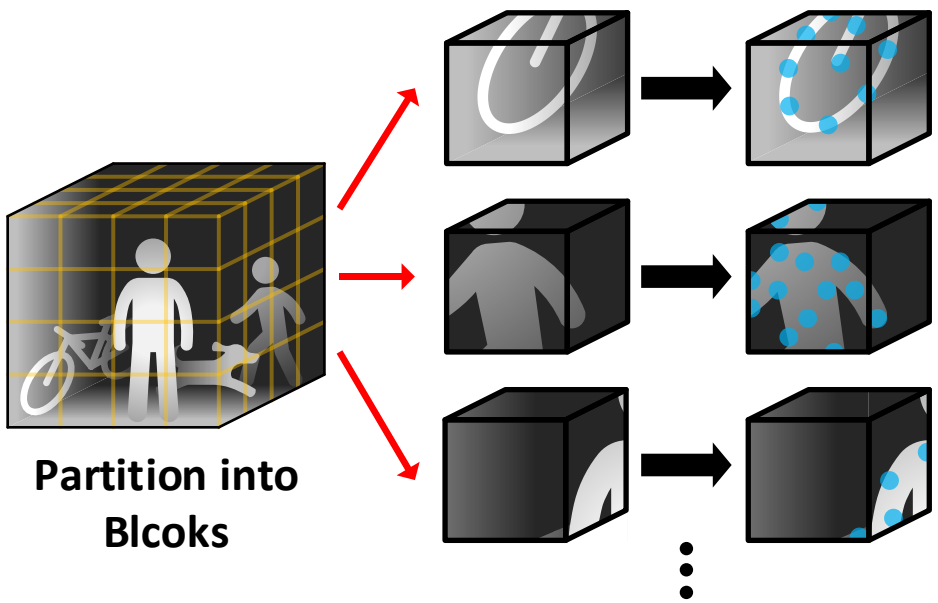
- **Irregular Access during Convolution after Point Processing**
  - Gather Points in Group ➔ **Irregular & Frequent DRAM Access**
  - Incapable of DRAM Buffer Hit & Low Bandwidth Utilization



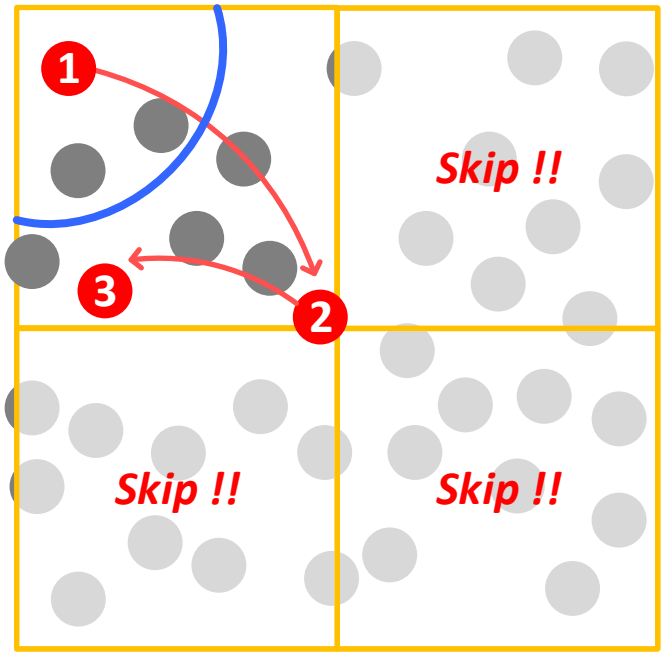


Proposed Block-based Point Processing

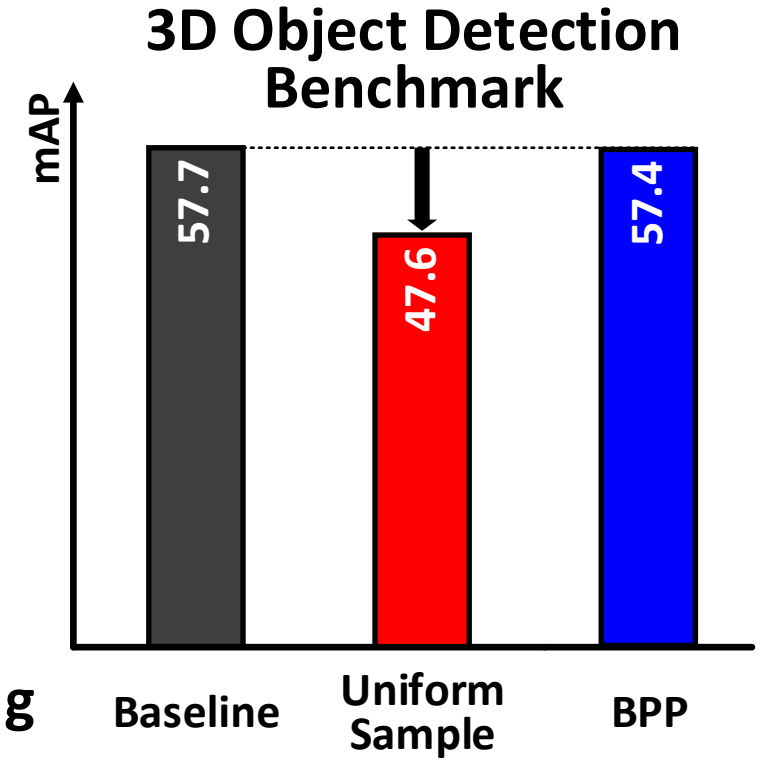
- **Dividing into Multiple Blocks + Block-wise Processing**
  - Reduced Searching Space for Sampling & Grouping
  - Maintaining Accuracy of PNN



Block-wise Even Samples



Block-based Point Processing



Overall Architecture of PNNPU

1. Partitioning Core

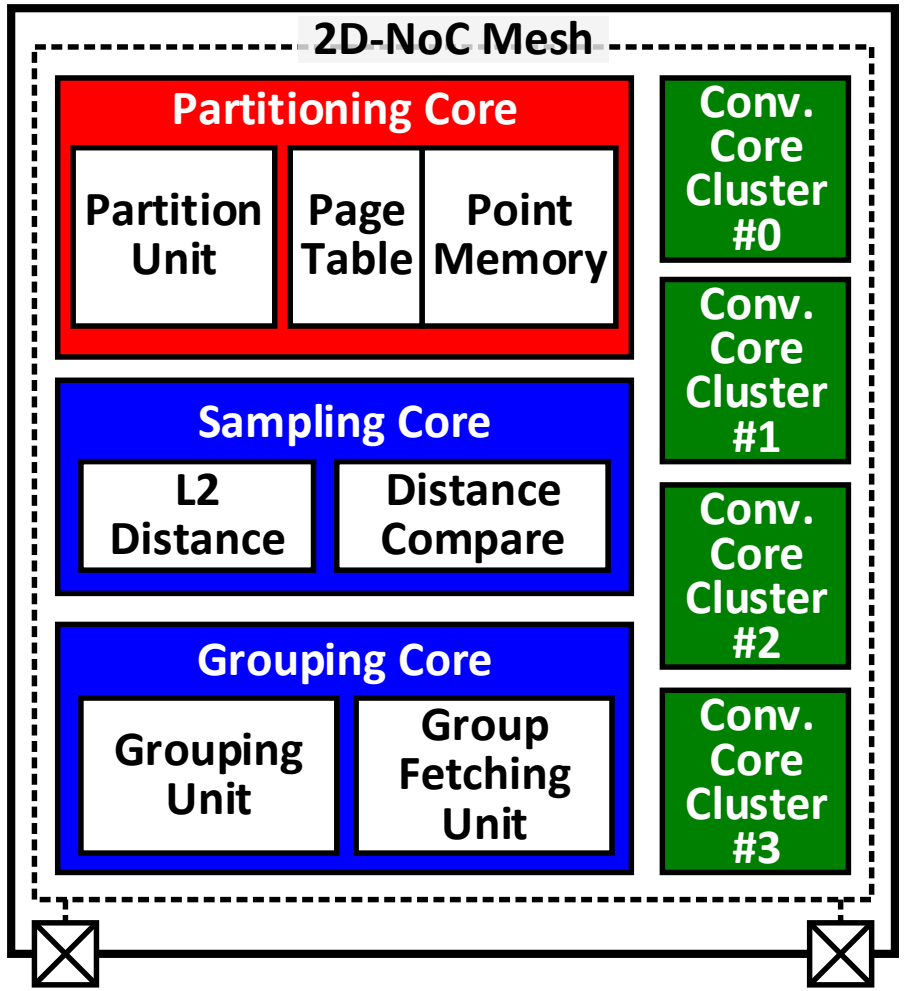
- *Page-based Point Memory Management Unit*
- *Linked List-based Page Table*

2. Sampling & Grouping Core

- *Hierarchical Farthest Point Sampling*
- *Block-skipping Ball Query*

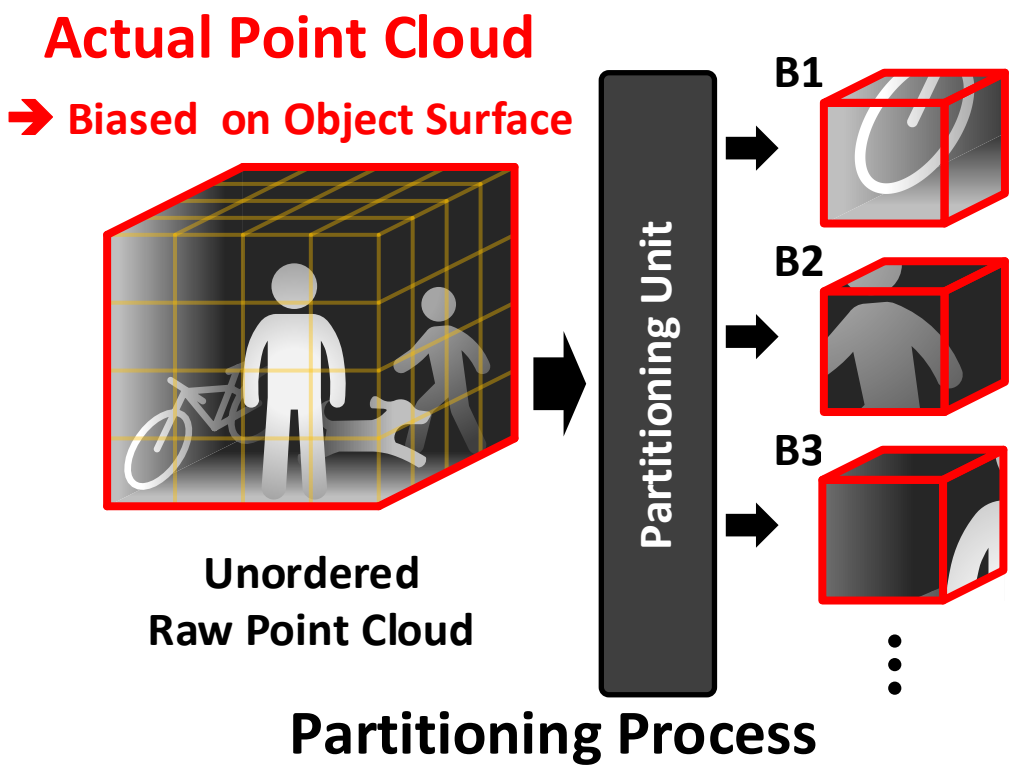
3. Convolution Core

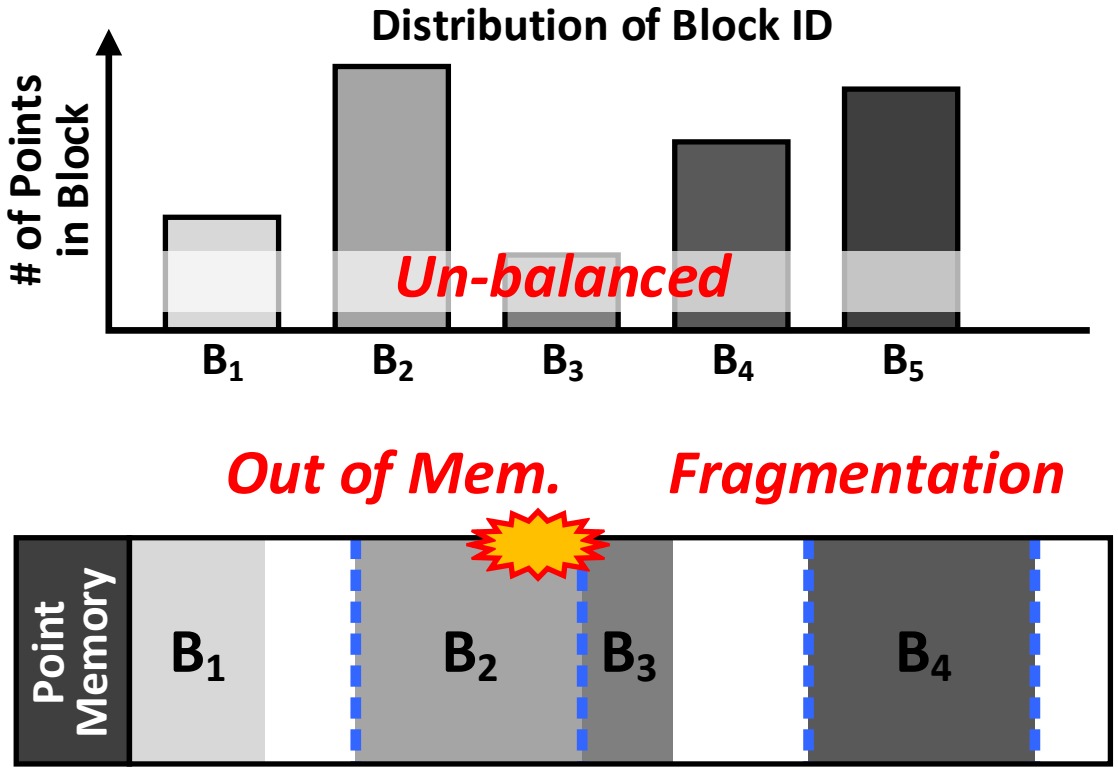
- *Skipping-based Max-pooling Prediction*



Streaming Point Cloud Partitioning

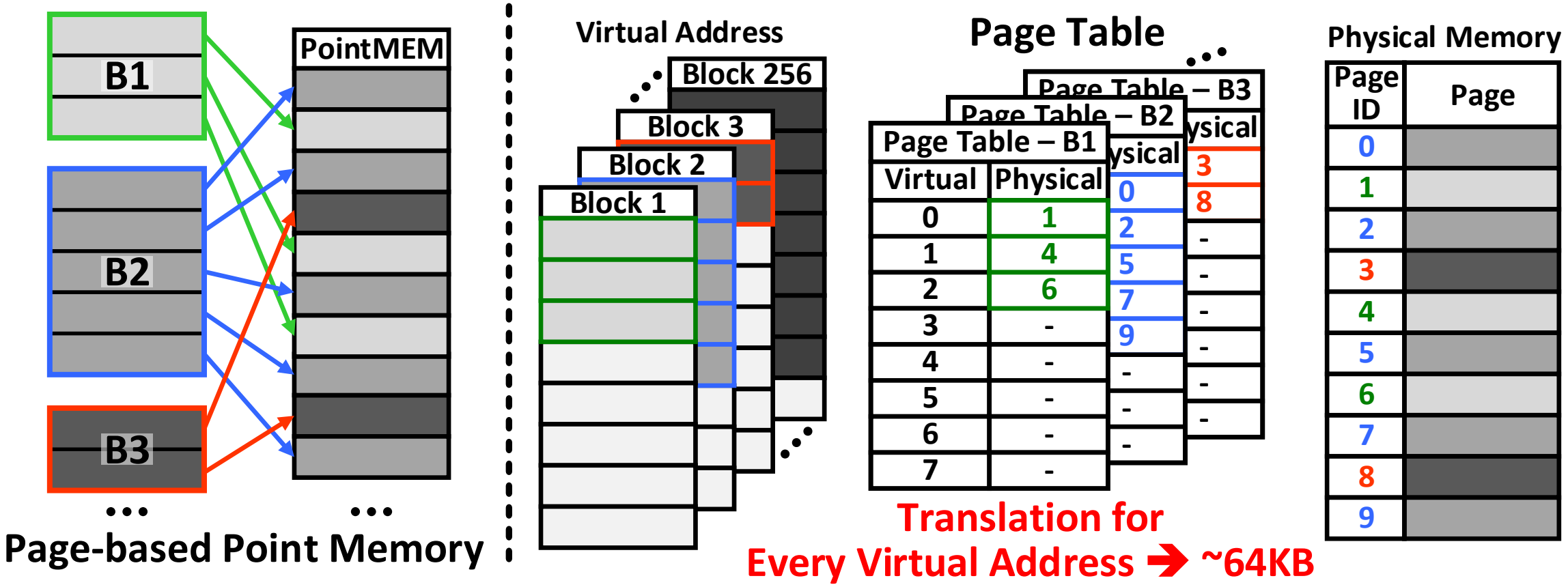
- **Dividing Point Cloud into Blocks during Fetching from External**
  - Static pre-allocation of memory for blocks ➔ *fragmentation* ☹





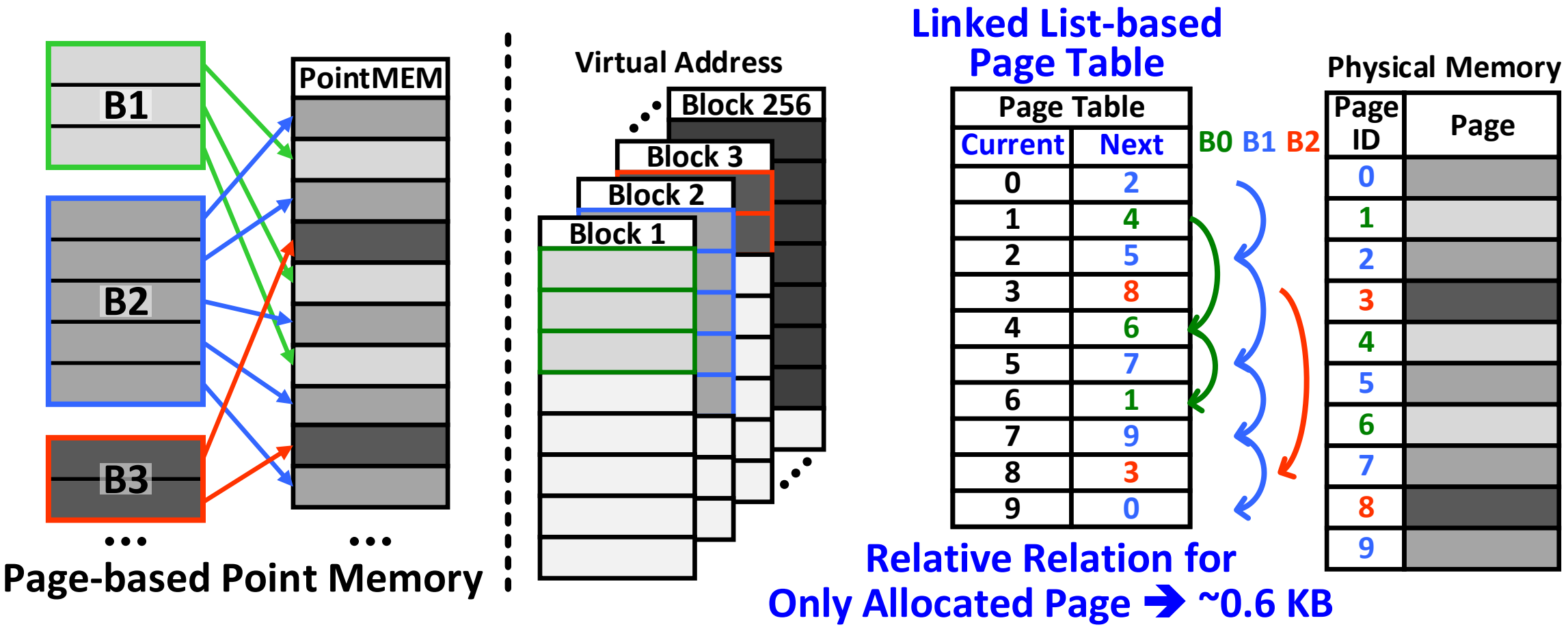
Page-based Point Memory Management

- **Managing Point Memory w/ Paging**
  - Wide virtual address (~4M) incur large page table (~64KB) ☹



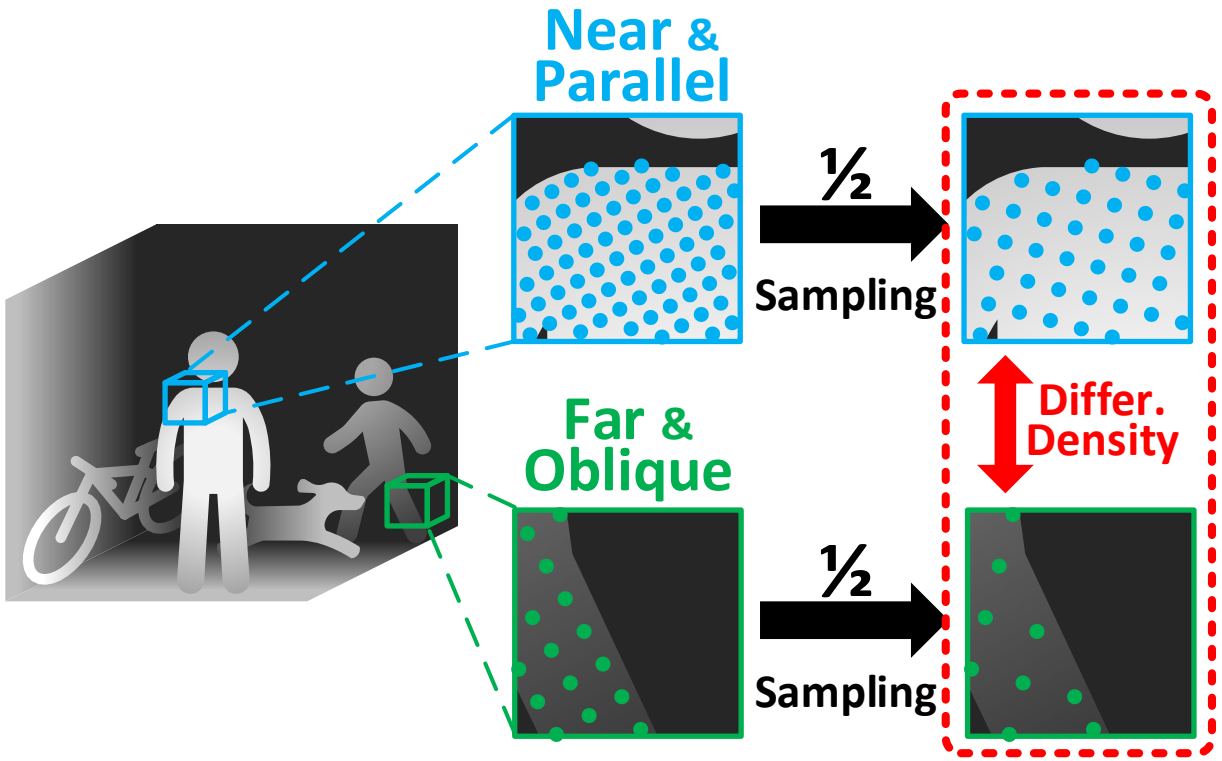
Linked List-based Page Table (LLPT)

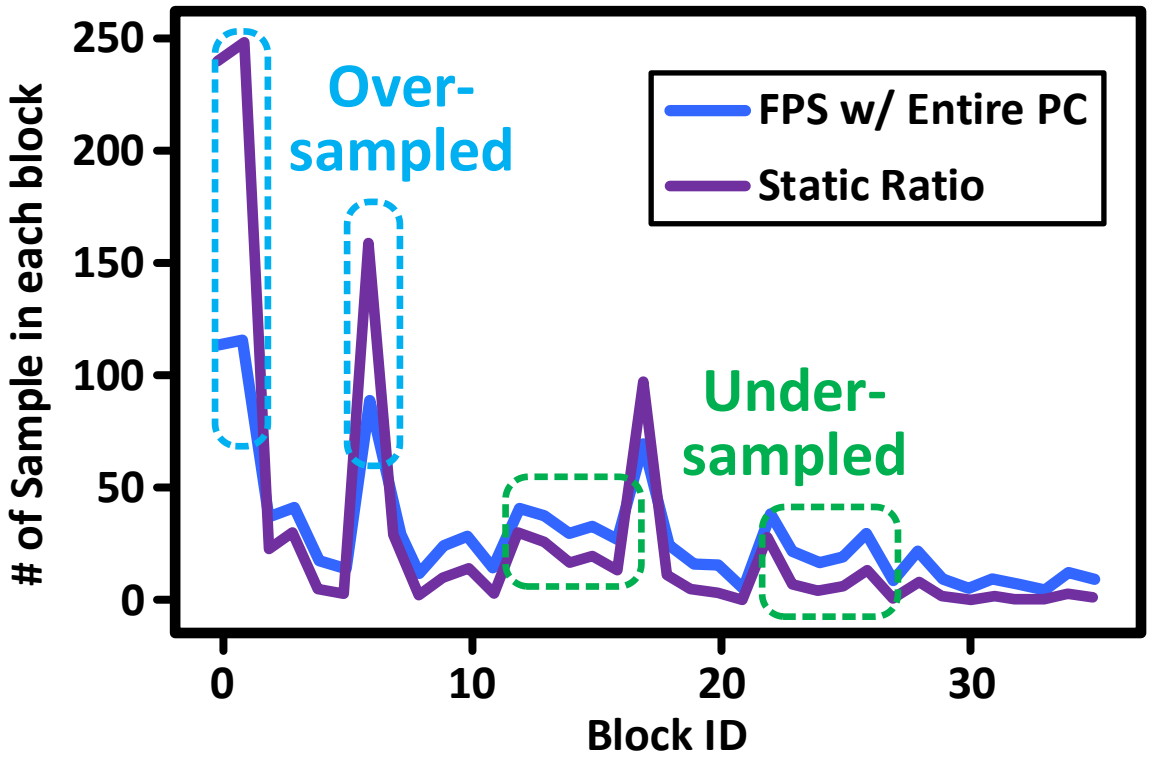
- **Page Table w/ Linked List**
  - Store only first page of each block & next page of each page



Problem of Simple Block-wise Sampling

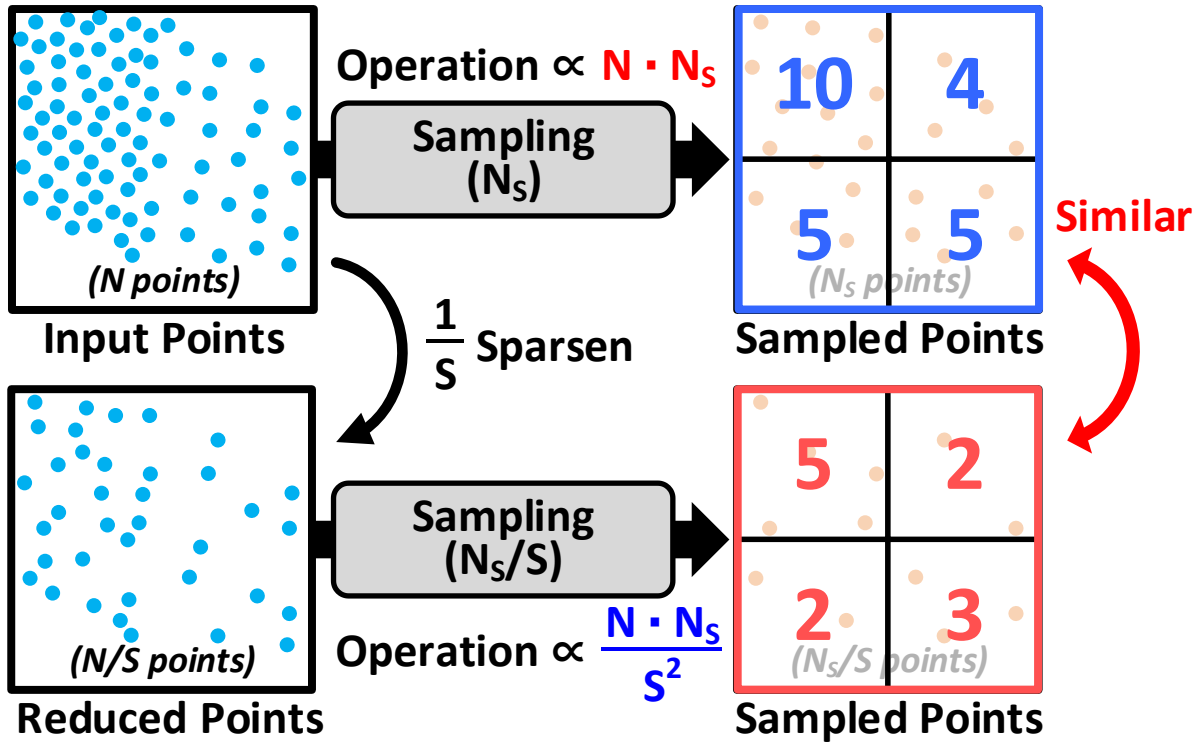
- **Varying Point Density in Each Block**
  - Block-wise FPS: Ignoring Density Difference among Each Blocks

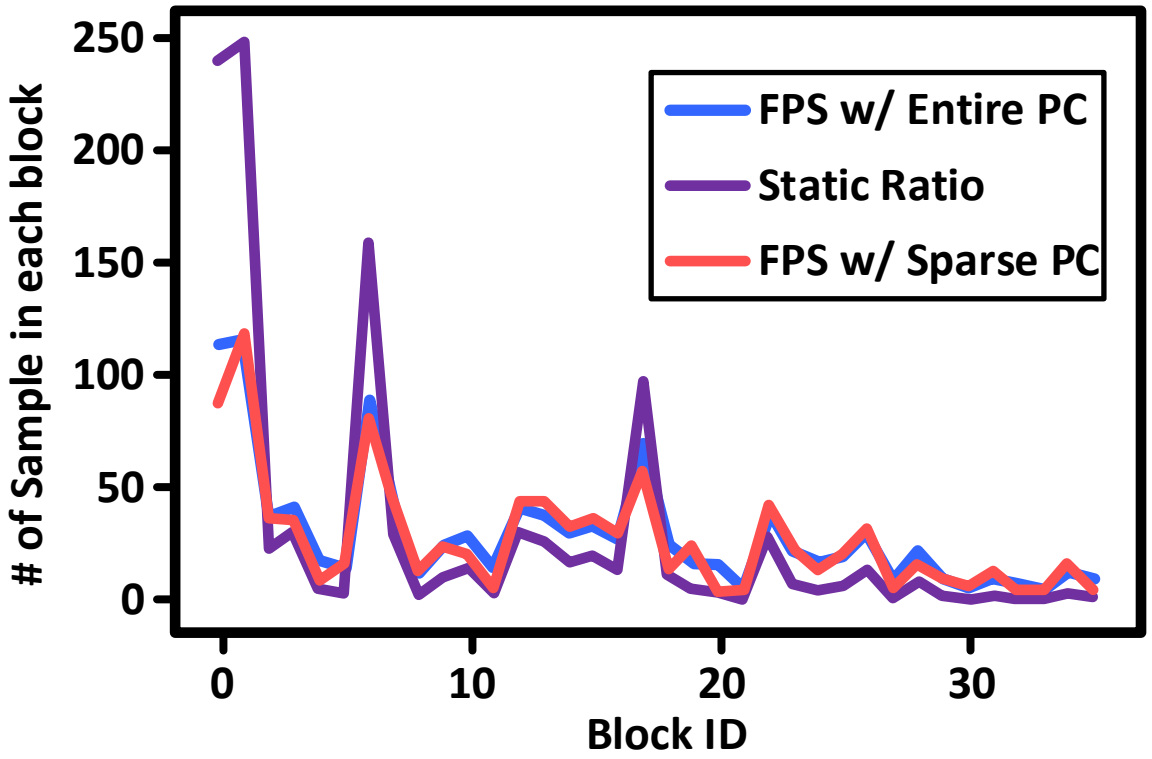




Lightweight Block Density Estimation

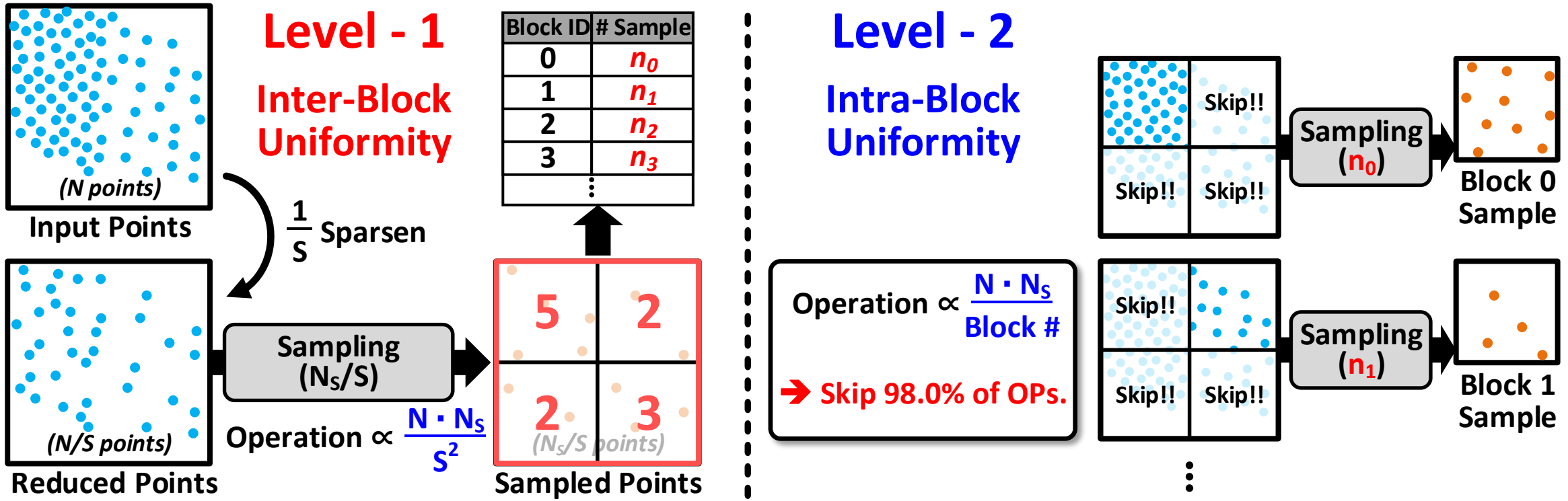
- **Sampling on Reduced Point Cloud**
  - Representing Density of Each Block





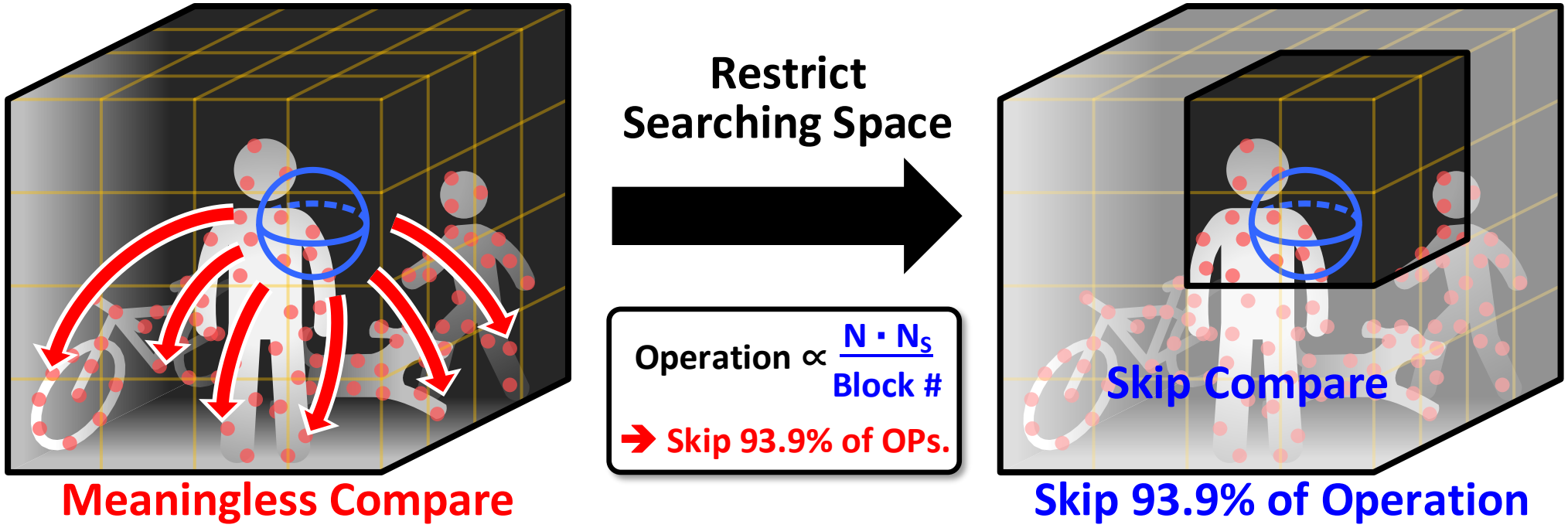
Hierarchical Block-wise FPS (HFPS)

- **2-Level Farthest Point Sampling**
  - Level-1: Determine # of Samples in Each Blocks ➔ **Inter-block Uniformity**
  - Level-2: Sampling Evenly inside Blocks ➔ **Intra-block Uniformity**



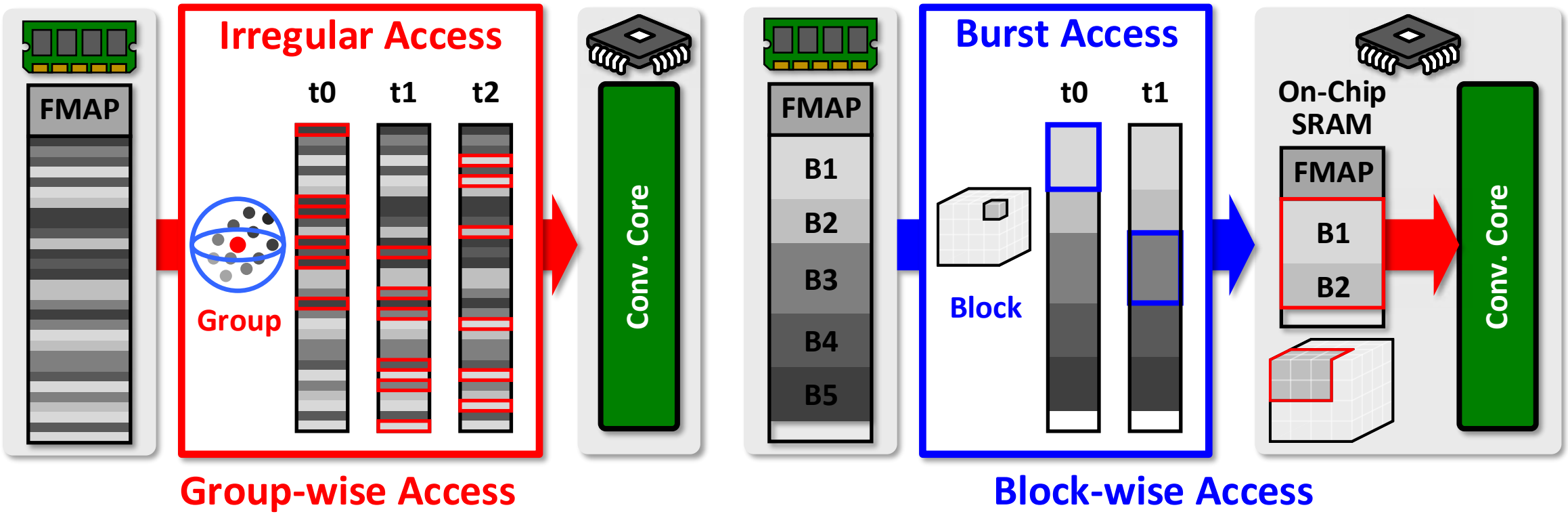
Block-skipping Ball Query (BSBQ)

- **Ball Query with Block-based Point Processing**
  - Meaningless Distance Compare with Points in Distant Block
  - Ball Query with Only Points in Nearby Blocks



Block-skipping Ball Query (BSBQ)

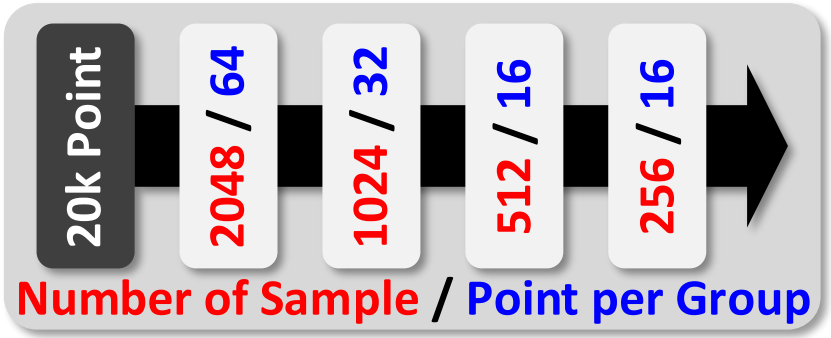
- **Ball Query with Block-based Point Processing**
  - Irregular Access for Loading a Group
  - Caching Only Neighboring Block & Loading Feature Map Block-wise



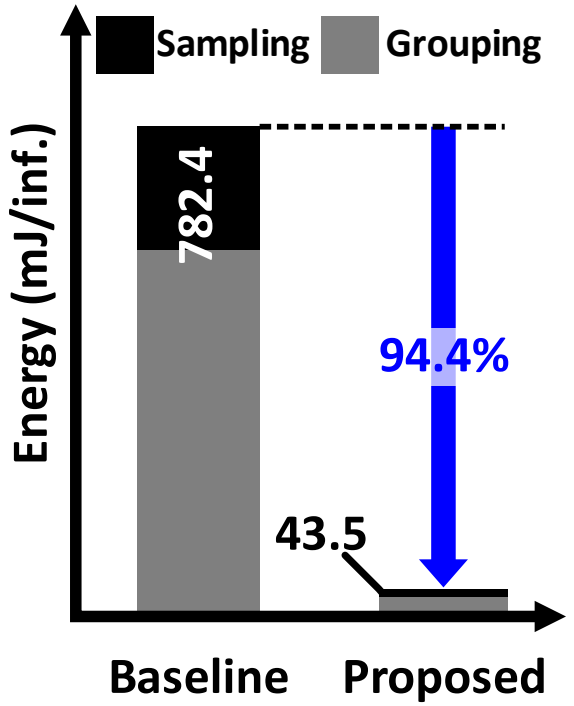
Improvement by HFPS & BSBQ

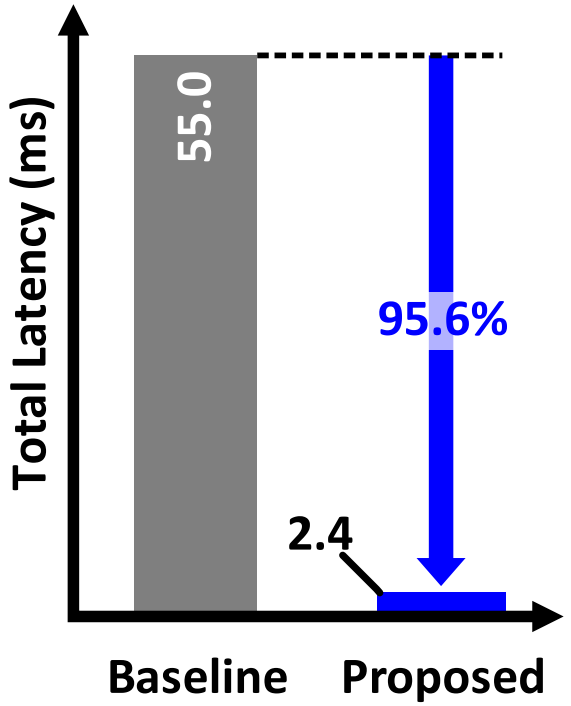
- **Reduce Latency & Energy by Circumscribing Searching Space**
  - Point Processing Latency: 95.6% ▼
  - Point Processing Energy: 94.4% ▼

HFPS & BSBQ Measurement



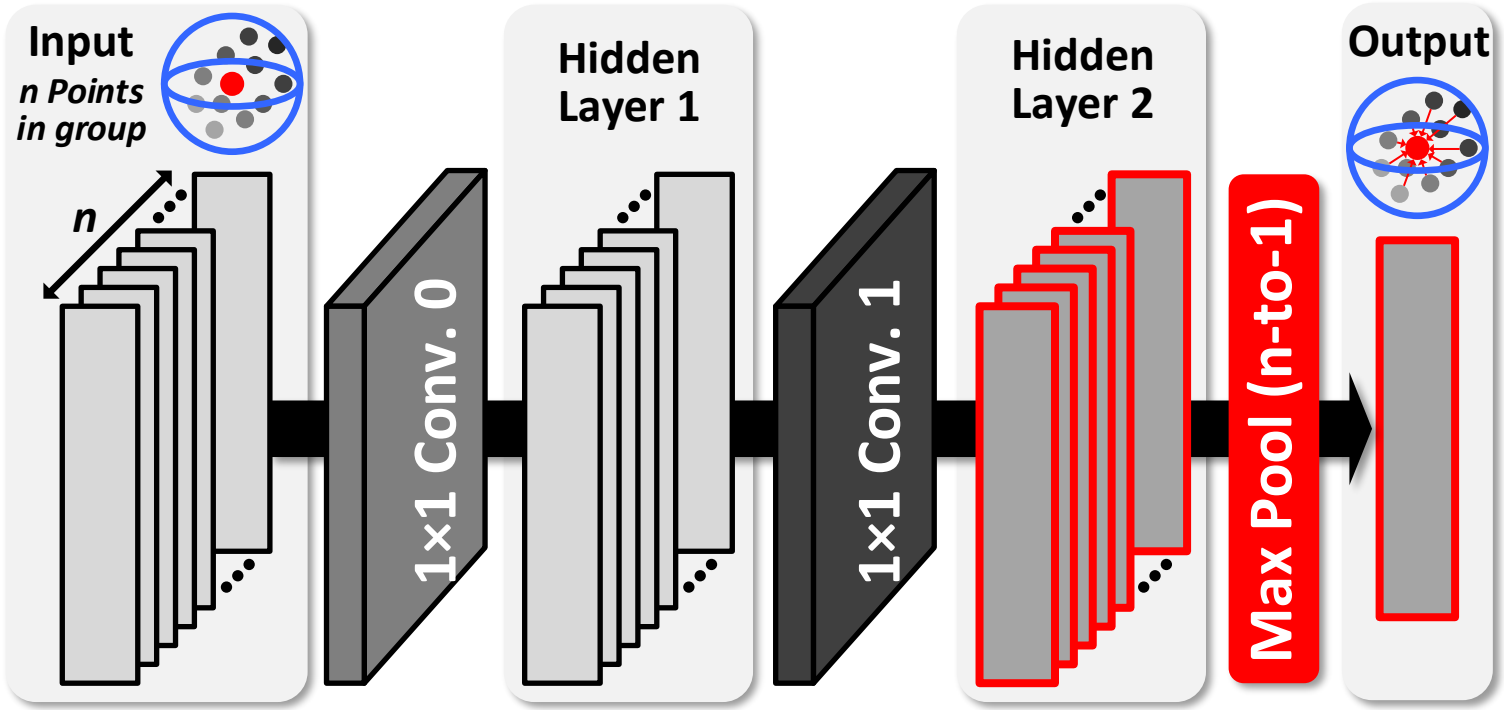
Task	3D Object Detction
# of Input Point	20k Points

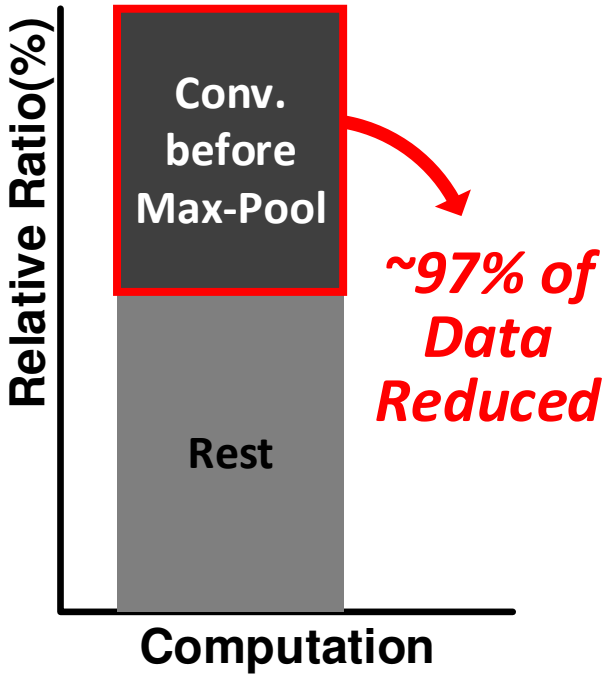




Convolution of PNN

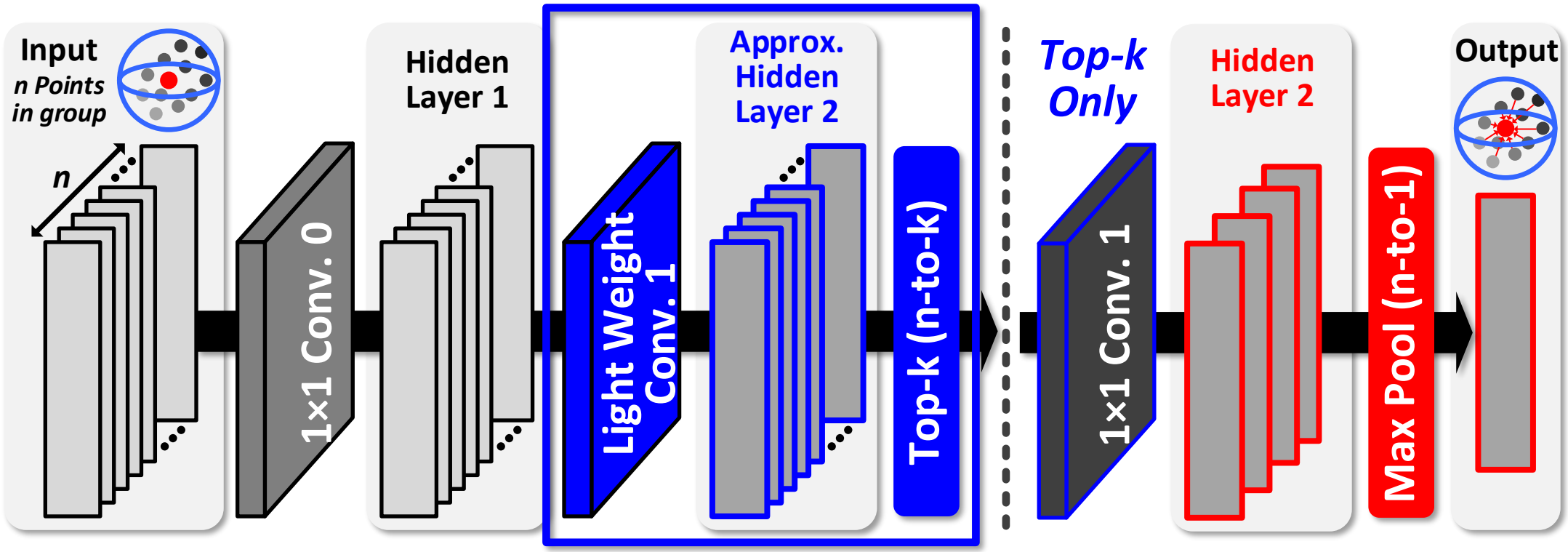
- **Large Max-pooling Layer (16~64-to-1) after Convolution**
  - Passing Only Max Value among Groups to Next Layer
  - Skipping Operation for Non-max Value by Max-pooling Prediction (MPP)





Max-pooling Prediction

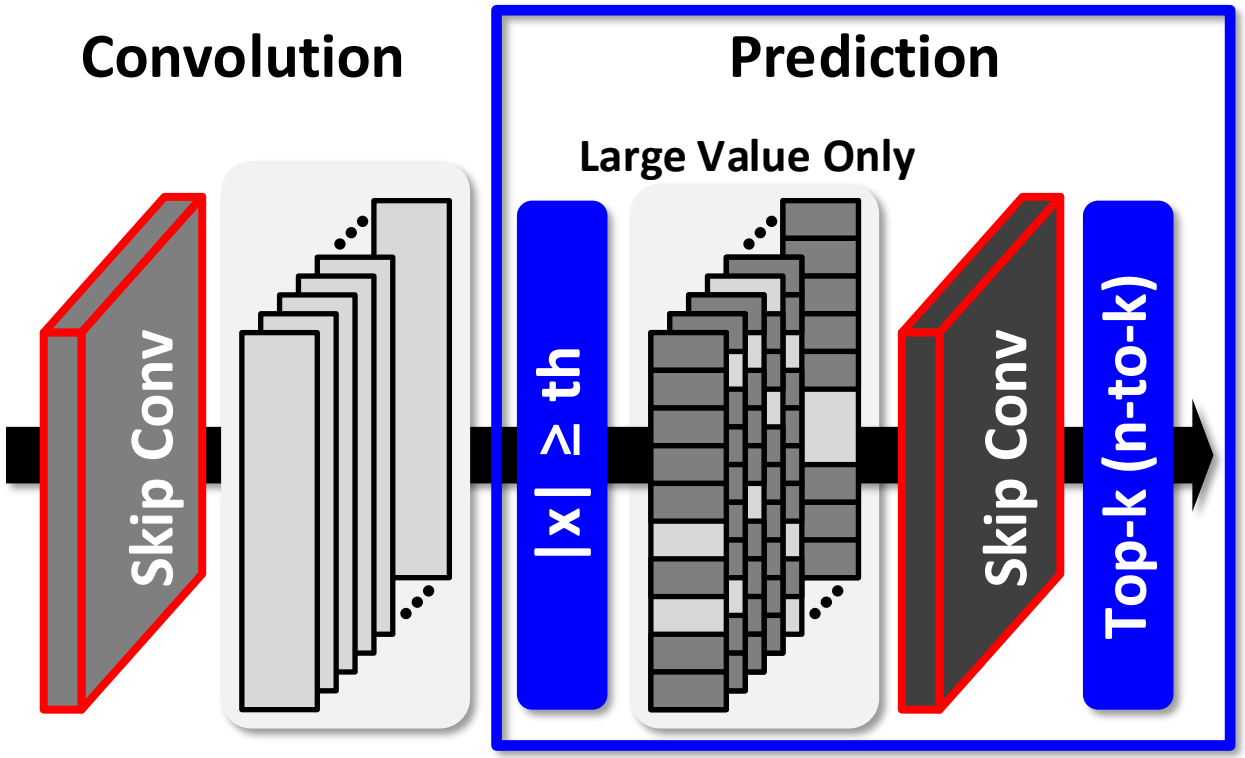
- **Large Max-pooling Layer (16~64-to-1) after Convolution**
  - Passing Only Max Value among Groups to Next Layer
  - Skipping Operation for Non-max Value by Max-pooling Prediction (MPP)

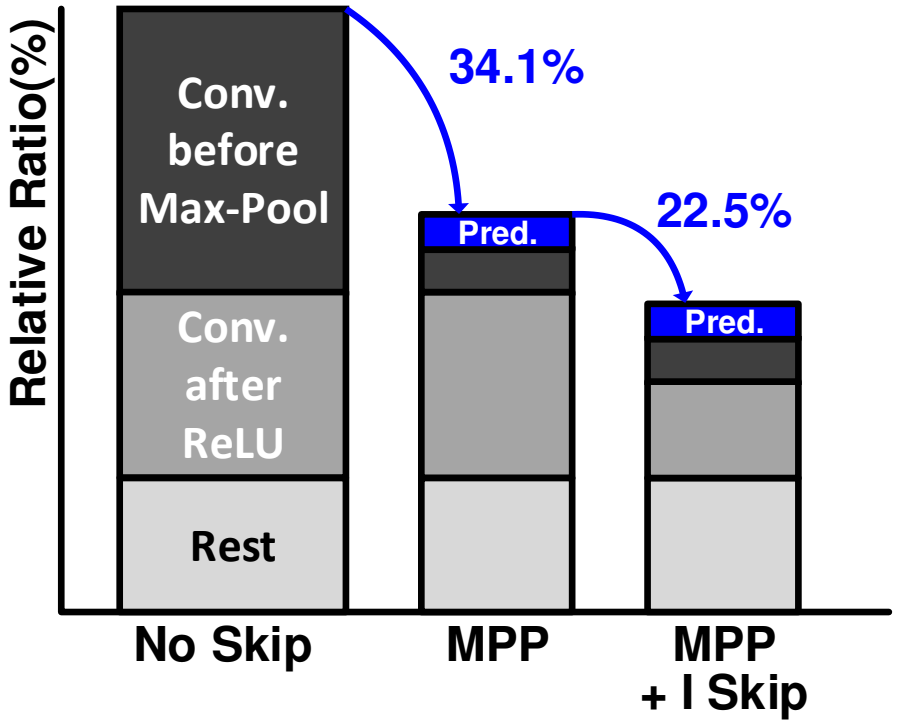


Skipping-based Max-pooling Prediction

- **Prediction with Large Value Input**
  - Unified Datapath for Prediction & Convolution using Input-skipping Conv. Core

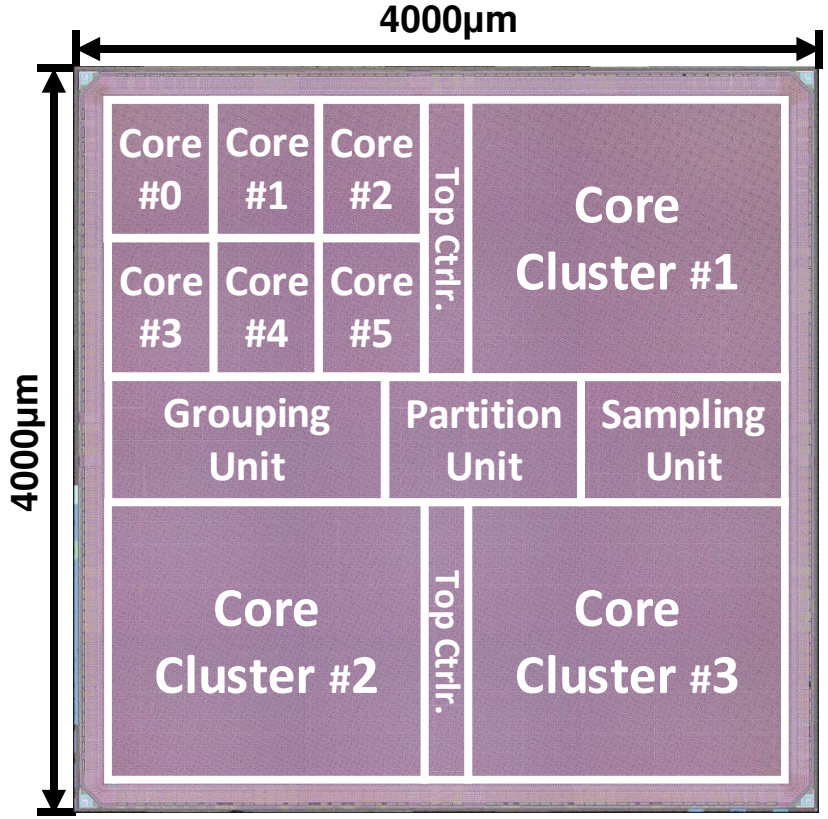
    ➔ **80.3% throughput ▲ & 69.0% higher efficiency ▲***





Chip Photography and Summary

- ×1.9 Faster PNN Inference than Previous SOTA
- ×2.1 Higher PNN Inference Efficiency than Previous SOTA



		Specification	
Technology		65nm Logic CMOS	
Die Area		4000μm × 4000μm	
SRAM		364 KB	
Frequency		200MHz @ 1.1V	50MHz @ 0.78V
Peak Power [mW]	SG	23.2	3.3
	CL	237.0[1] - 308.2[2]	27.2[1]-38.5[2]
Peak Performance [GOPS]	SG	23.2[3] - 604.3[4]	5.8[3] – 151.1[4]
	CL	614.4[1]- 2808.7[2]	153.6[1]-702.17[2]
Efficiency @ MAX. GOPS [TOPS/W]	SG	1.00[3] – 26.1[4]	1.74[3]-45.4[4]
	CL	2.59[1]-9.11[2]	5.65[1]-18.23[2]

1) 0% Input Sparsity, 2) 90% Input Sparsity, 3) w/o HB-FPS & BS-BQ, 4) w/ HB-FPS& BS-BQ

Conclusion

1. **Point Memory Management Unit**
   - Enable memory-efficient Partitioning ➔ on-chip footprint 98.4%▼
2. **Hierarchical-FPS & Block-skipping BQ**
   - Block-based point processing ➔ 94.4% energy & 95.6% latency ▼
3. **Skipping-based MPP**
   - Utilizing skipping core as predictor ➔ 80.3% higher throughput

PNNPU: A 11.9 TOPS/W & High-speed 3D PNN Processor with Block-based Point Processing for Regular DRAM Access

Thank You!

- **Questions? Feel Free to Contact Me!**
  - E-mail: sangjinkim@kaist.ac.kr
  - LinkedIn: https://www.linkedin.com/in/sangjin-kim-2b730421a/
  - Zoom Meeting: https://zoom.us/j/96308950362?pwd=UWh3TG4zNlZNOEU4ZkRMeVgyWlBCZz09 (Password: HC_PNNPU)