山东大学 SHANDONG UNIVERSITY

THE OHIO STATE UNIVERSITY

SIG METRICS

A Case Study for Ray Tracing Cores: Performance Insights with Breadth-First Search and Triangle Counting in Graphs

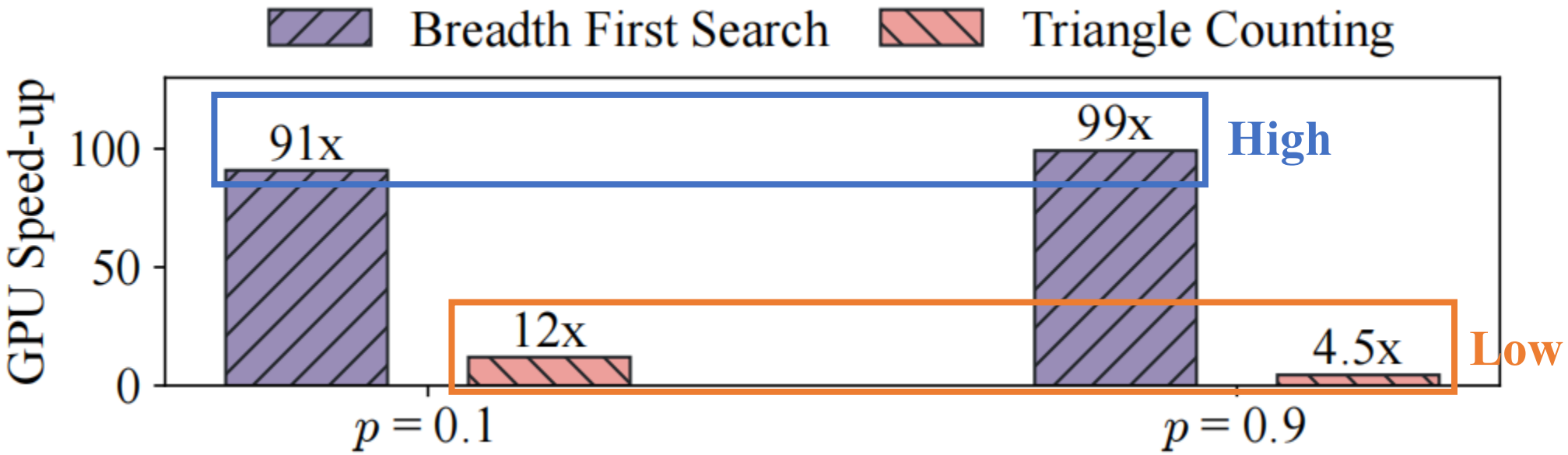
Zhixiong Xiao*, Mengbai Xiao*, Yuan Yuan*, Dongxiao Yu*, Rubao Lee†, Xiaodong Zhang‡

*: Shandong University
†: Freelance
‡: The Ohio State University

June 12, 2025
Stony Brook, New York, USA

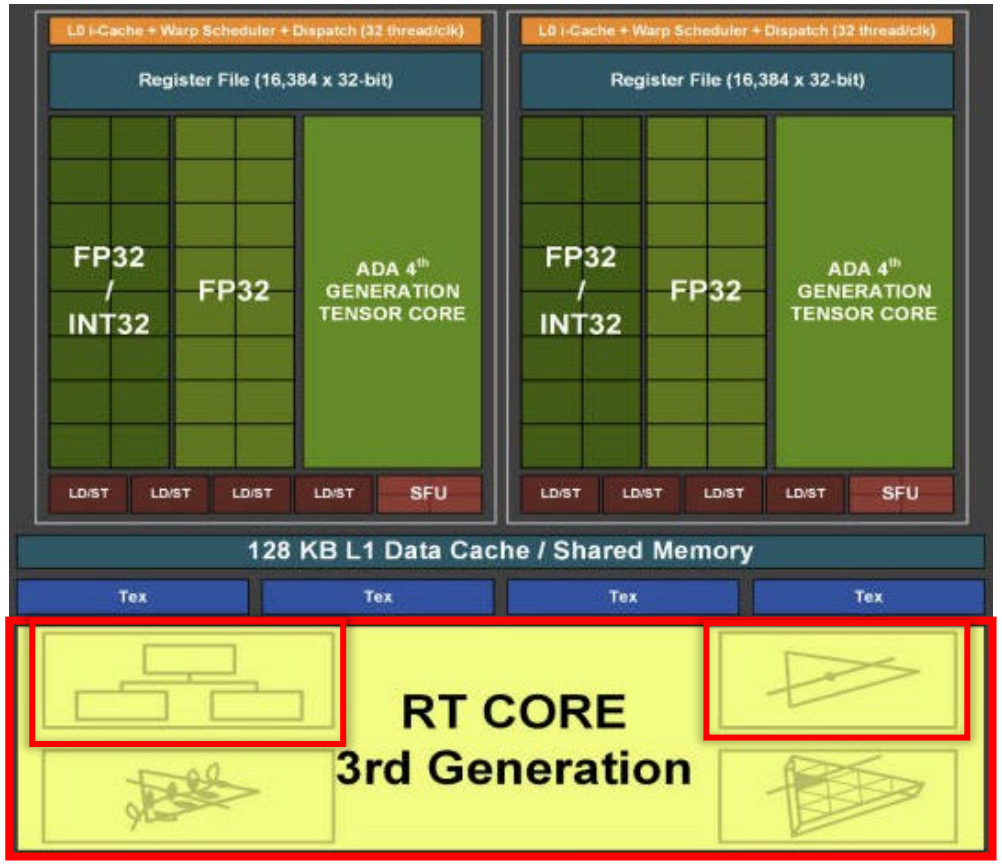
GPU Performance Gains Vary across Graph Algorithms

- Data-parallel graph algorithms (e.g., BFS) fit SIMT execution model well
- Control flow-intensive graph algorithms (e.g., TC) hardly obtain performance gains



Introduction of Ray Tracing (RT) Cores on GPU

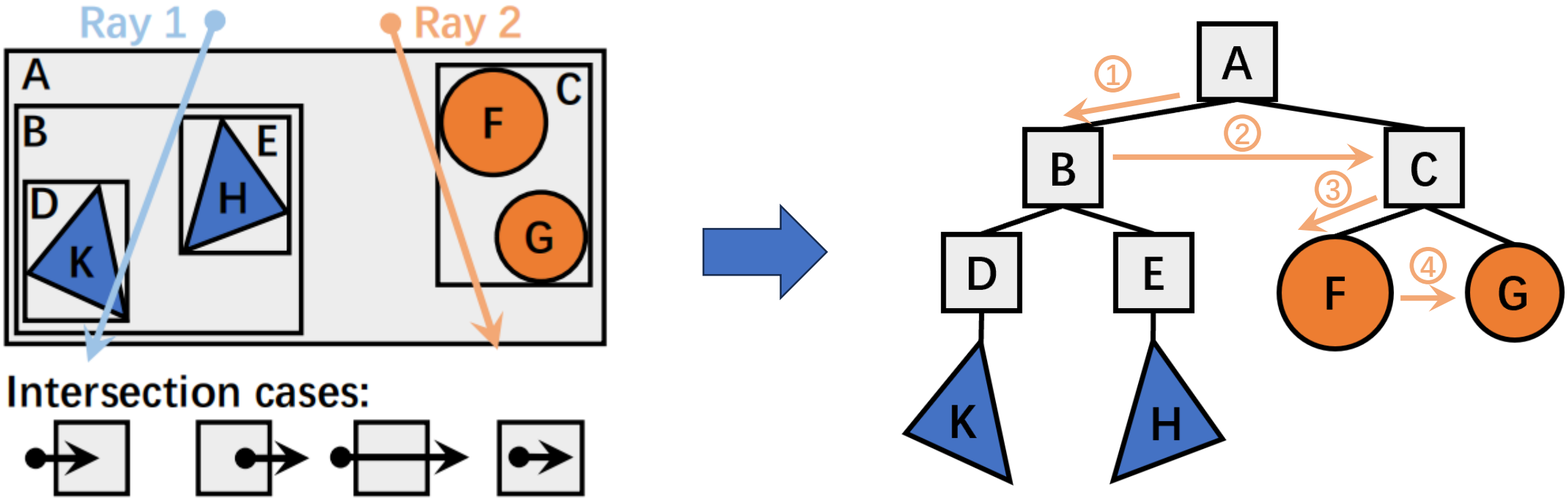
- A specialized hardware for ray tracing pipeline
- Hardware-accelerated bounding volume hierarchy (BVH) traversal
- Hardware-accelerated ray-triangle intersection
- Potential for control flow-intensive workloads:
  - RTNN
  - RTScan



RTNN: Accelerating Neighbor Search Using Hardware Ray Tracing, PPoPP'22
RTScan: Efficient Scan with Ray Tracing Cores, VLDB'24

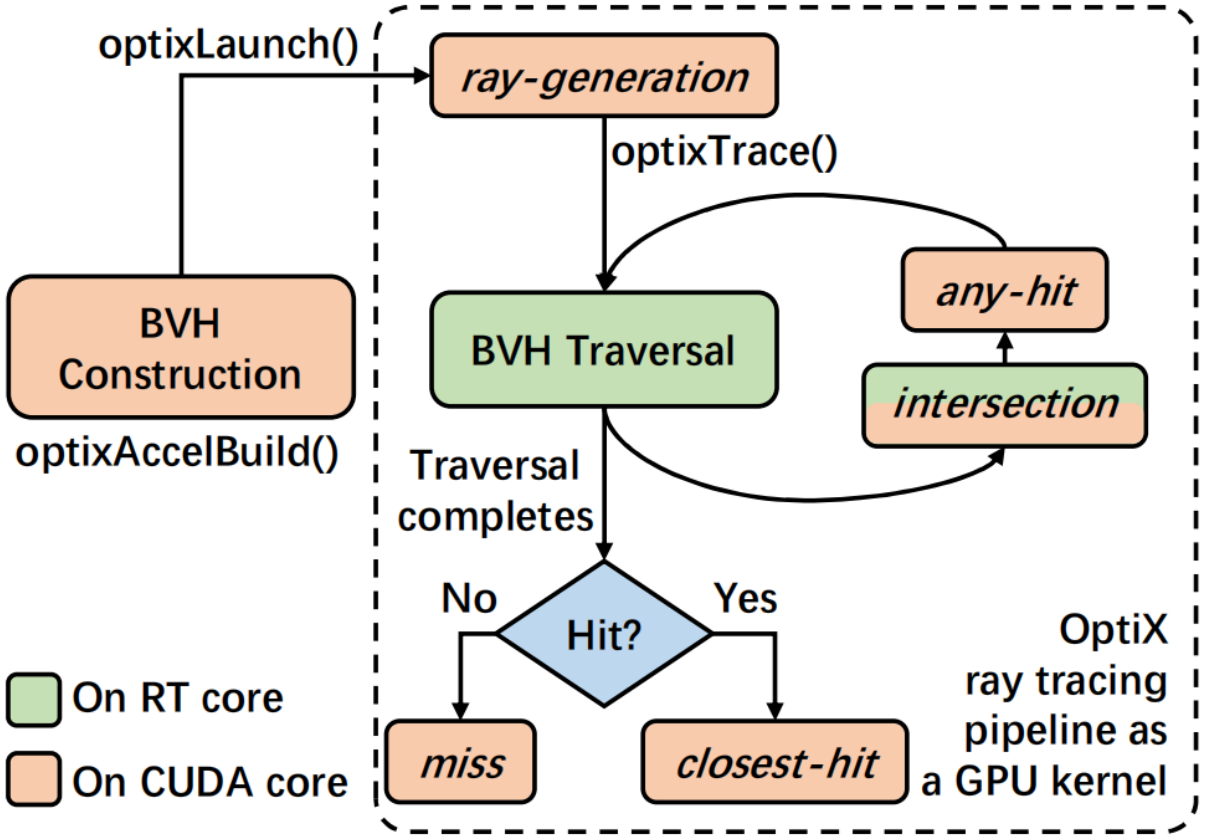
BVH Construction and Traversal

- BVH is a tree index where primitives are recursively wrapped by axis-aligned bounding boxes (AABBs).

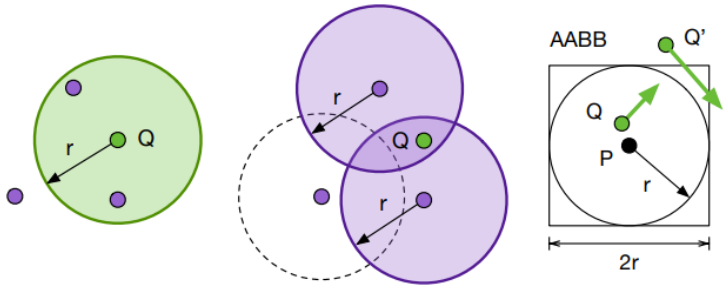


Ray Tracing Pipeline

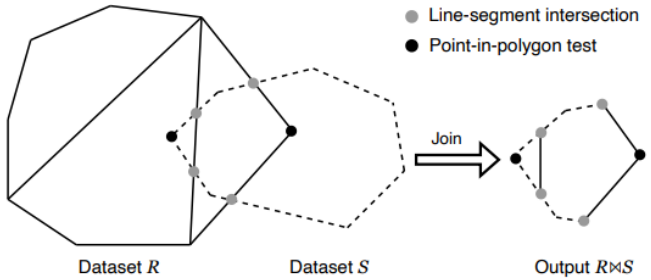
- Utilizing RT cores with Nvidia OptiX APIs.



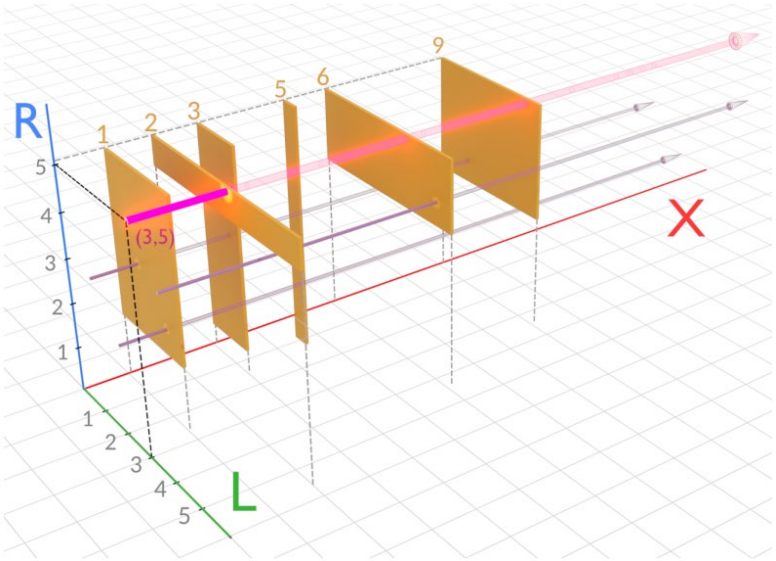
General Computing on RT Cores



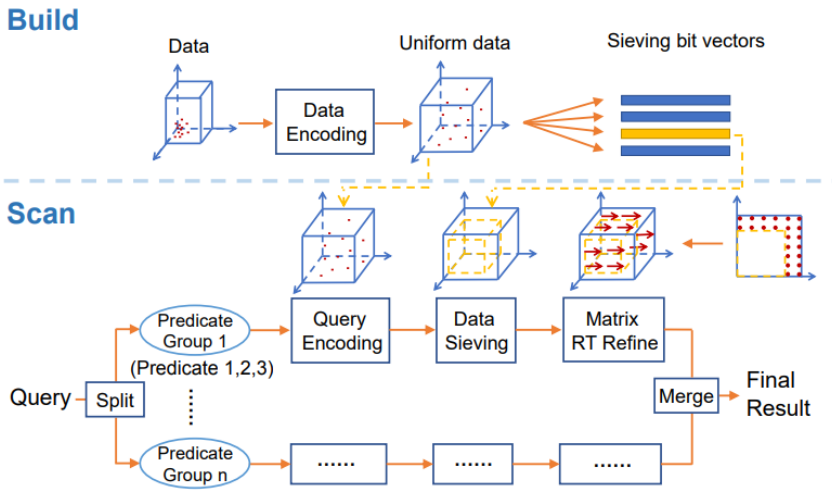
Accelerating Low-Dimensional Neighbor Search



Accelerating Spatial Join



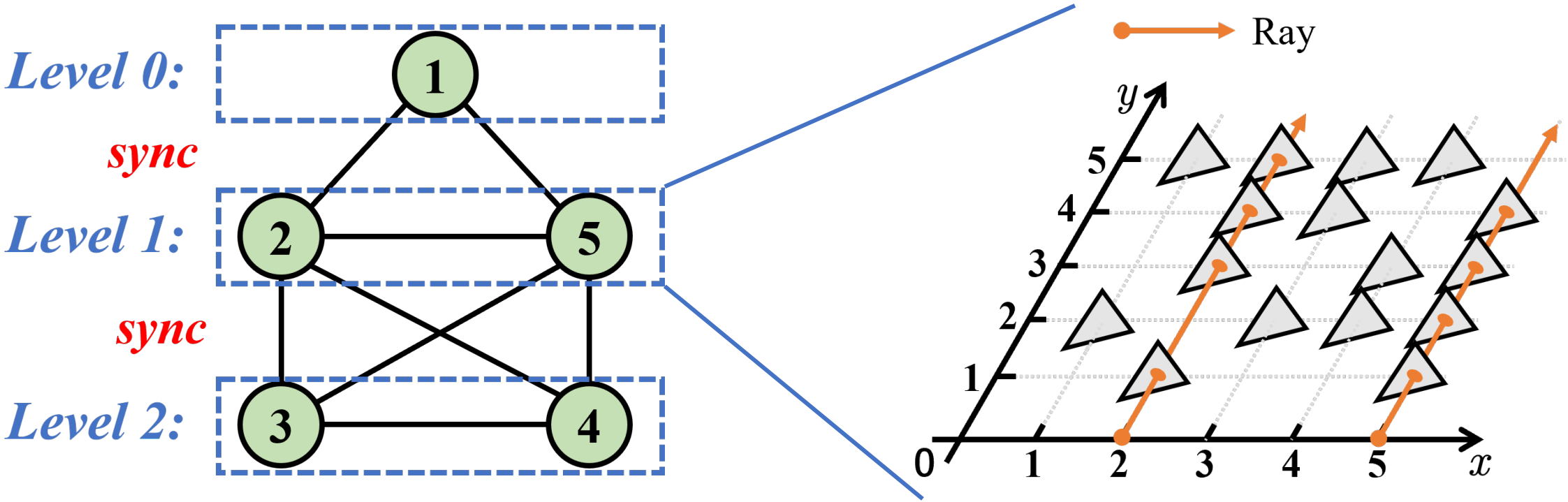
Accelerating Range Minimum Queries



Accelerating Scans in Databases

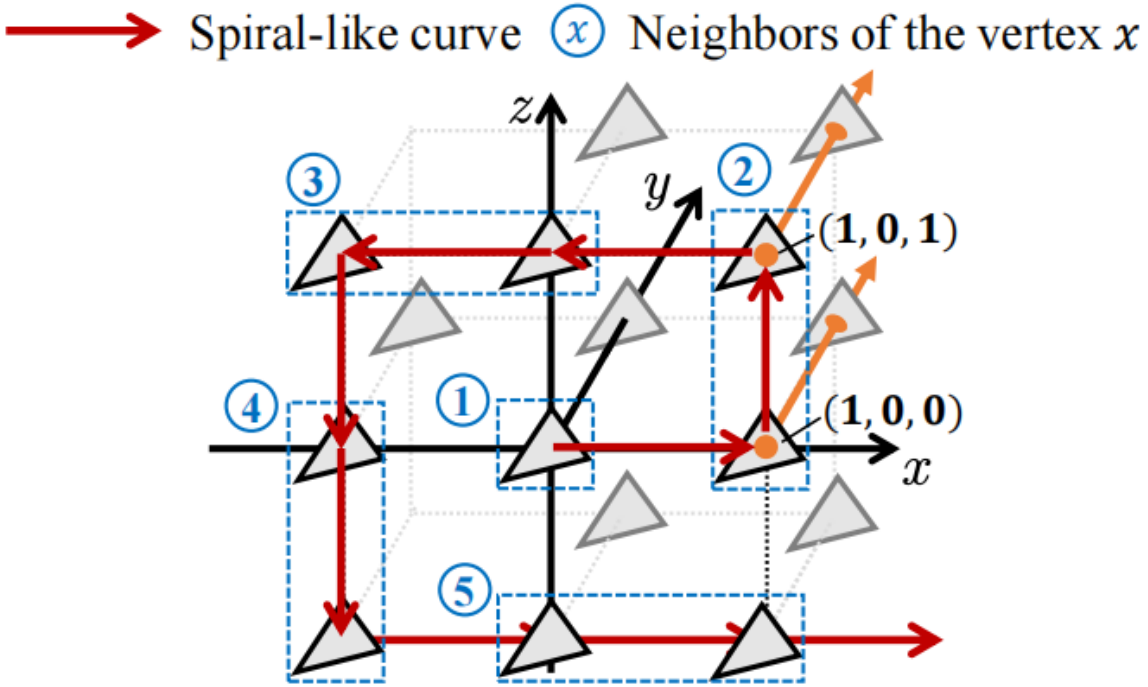
RT-based Breadth-First Search

- Adopting a parallel strategy based on hop-by-hop traversal
- Arranging primitives in a form of adjacency matrix
- Replacing expansion on CUDA cores with hardware-accelerated BVH



Further Optimizations

- Deficiencies of the naive method
  - Inaccuracy
  - High BVH traversal cost
  - Workload imbalance
- Optimizations:
  - Spiral-like curve layout of primitive queues
  - Compacting placement of primitives
  - Splitting long rays

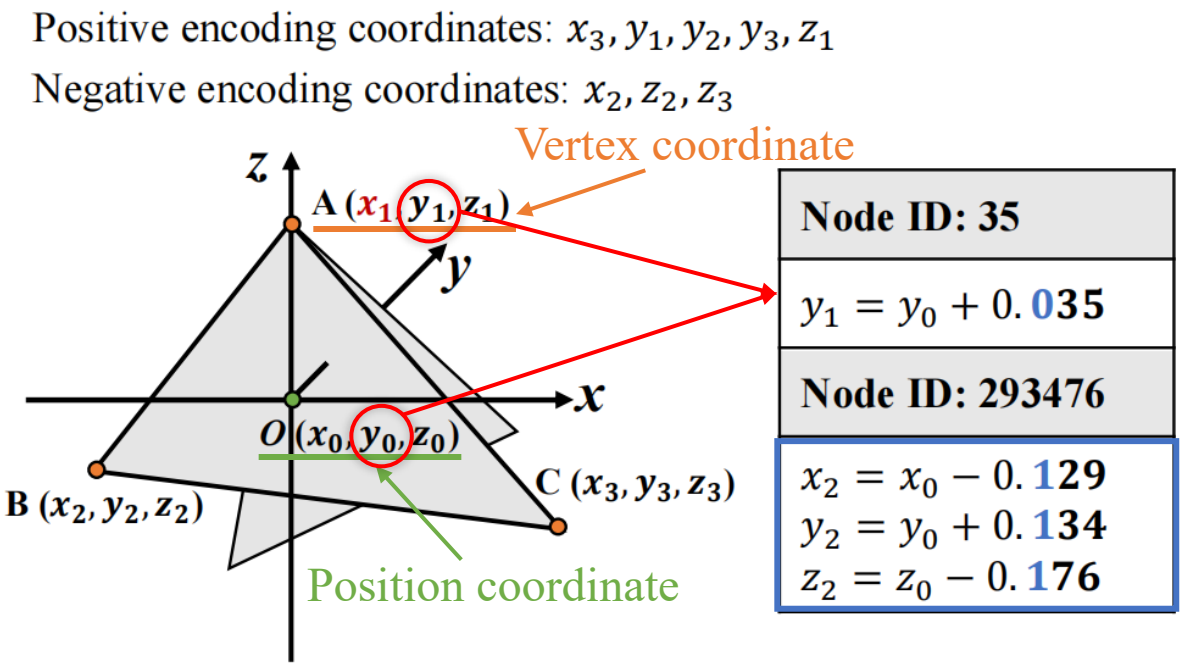


Vertex Encoding

- Representing a neighbor with a triangle primitive is memory-inefficient

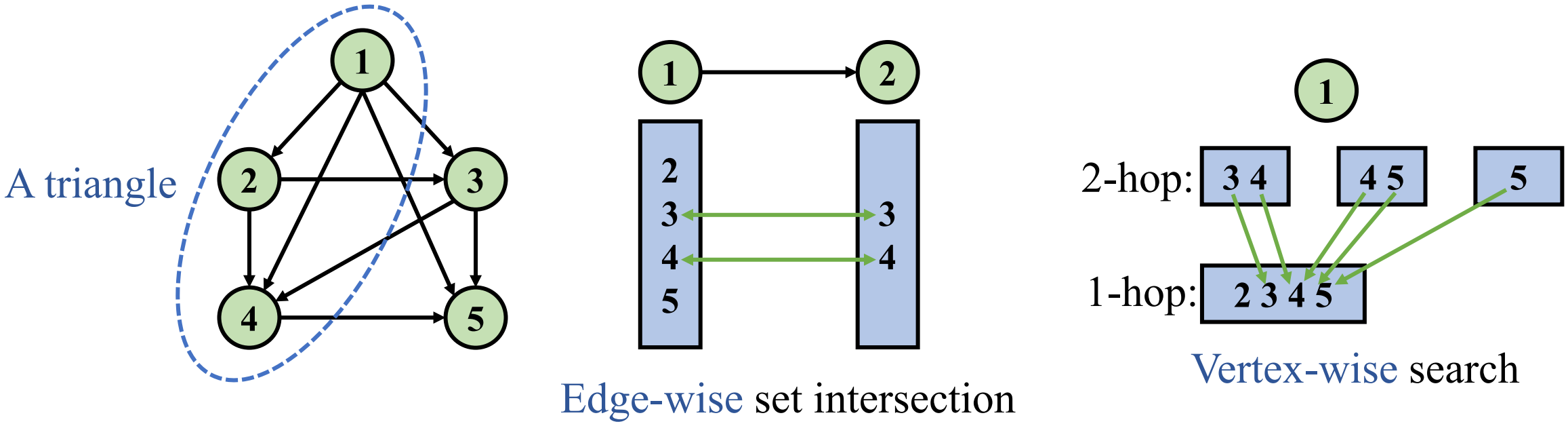
- Encoding IDs into triangle coordinates
  - Difference between vertex coordinates and position coordinates
  - Splitting a large ID into multiple parts

- Reducing memory footprint and I/O



RT-based Triangle Counting

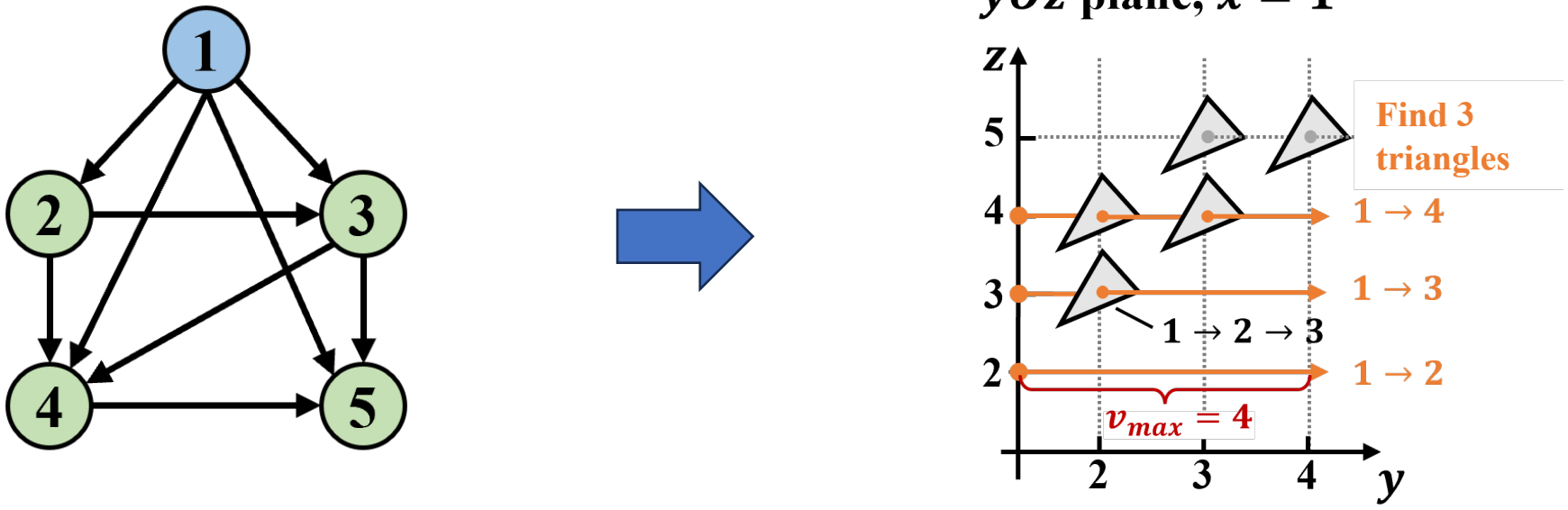
- Triangle in graph is a clique of three vertices
- Common strategies:
  - Edge-wise set intersection
  - Vertex-wise search



The 1-among-2 Method

- Finding 1-hop neighbors among 2-hop neighbors
  - Rays represent 1-hop neighbors, while primitives represent 2-hop neighbors
- The number of rays: $|E|$, the number of primitives: $\sum_{u \in V} \sum_{v \in N(u)} \text{degree}(v)$

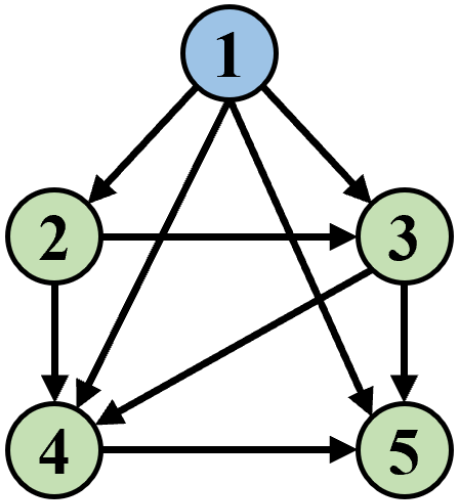
Take vertex 1 as an example:

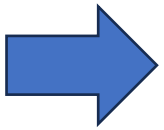


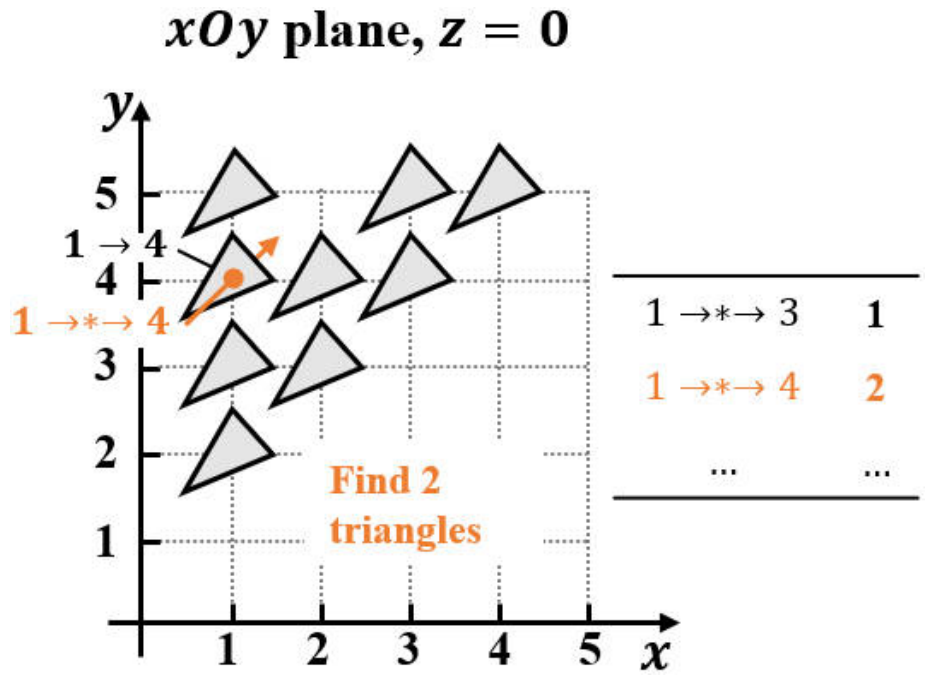
The 2-among-1 Method

- Checking if an edge exists between a vertex and its 2-hop neighbor
  - Rays represent 2-hop relationships, while primitives represent 1-hop relationships
- The number of rays: $\sum_{u \in V} \sum_{v \in N(u)} \text{degree}(v)$, the number of primitives: $|E|$

Take vertex 1 as an example:





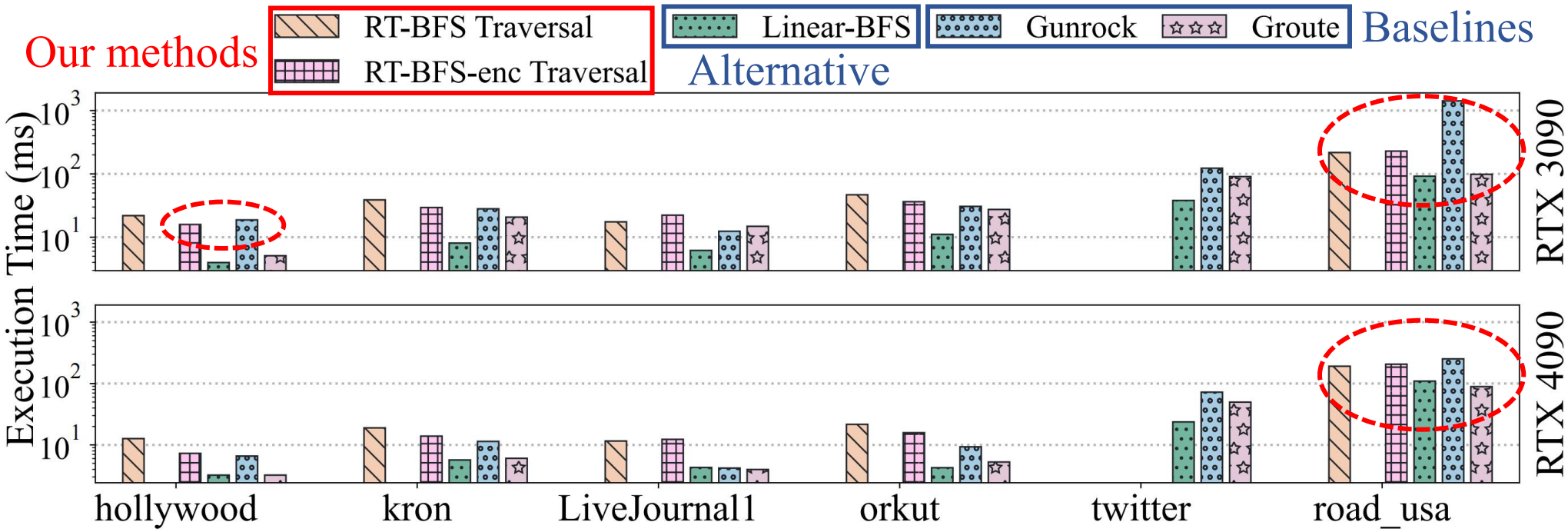


Experimental Setup

- We conduct experiments on two machines:
  - RTX 3090: 82 SMs, 10,496 CUDA cores, 24 GB memory, and 82 the 2nd-generation RT cores
  - RTX 4090: 128 SMs, 16,384 CUDA cores, 24 GB memory, and 128 the 3rd-generation RT cores
- Datasets consist of real-world graphs
- We use execution time as the primary performance metric and additionally report executed instructions, memory footprint, and power

Inferior Performance of RT-based BFS Against Baselines

- RT cores can hardly accelerate BFS even with encoding technique
- 3.94× slower than Linear-BFS, a CUDA variant of the same workflow without RT cores



Understanding the Inferior Performance of RT-based BFS

- RT-BFS issues 4.03× more memory instructions
- BVH traversal incurs higher memory access overhead than classic CUDA methods in sequential access workloads such as BFS

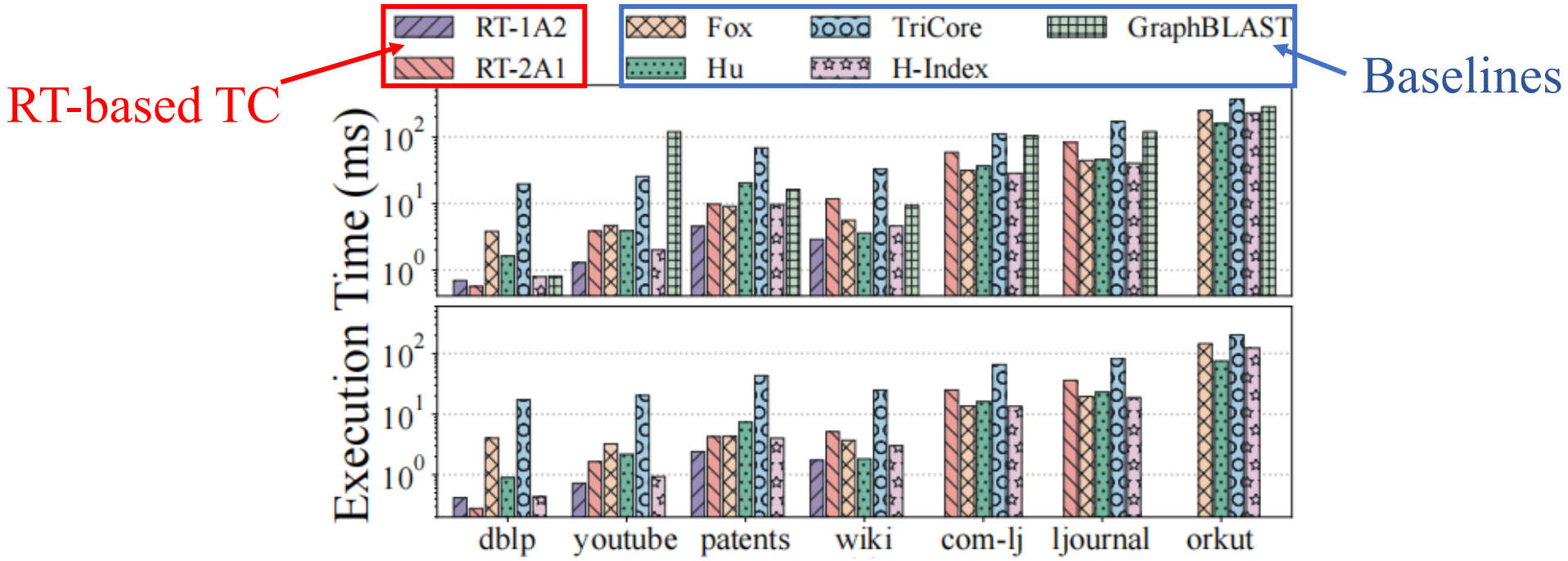
4.03× memory I/O instructions

less memory I/O instructions

Dataset	Load/Store Instructions		
	RT-BFS	RT-BFS-enc	Linear-BFS
hollywood	54.14M	20.55M	12.55M
kron	102.15M	39.84M	23.67M
LiveJournal1	46.03M	22.39M	11.71M
orkut	123.07M	48.05M	24.93M
twitter	N/A	N/A	77.88M
road_usa	98.21M	62.49M	36.87M

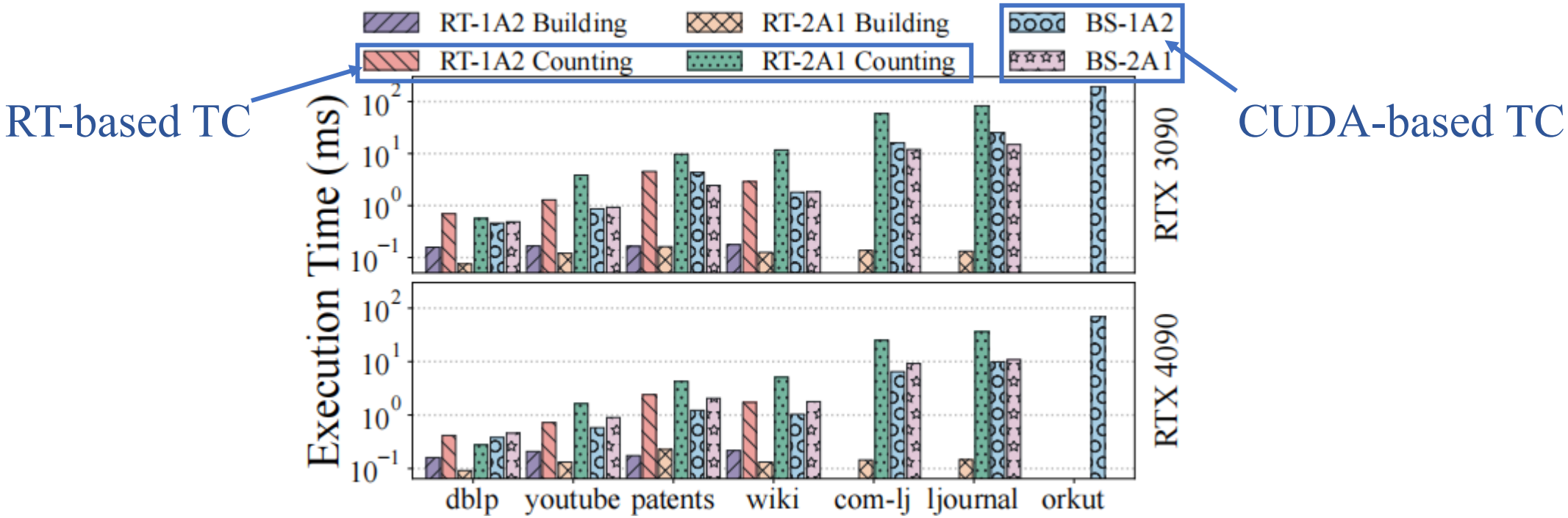
Improved Results of RT-based TC in Specific Scenarios

- RT-based TC has superior performance on some datasets
- RT-2A1 is slower due to high miss ratio of rays



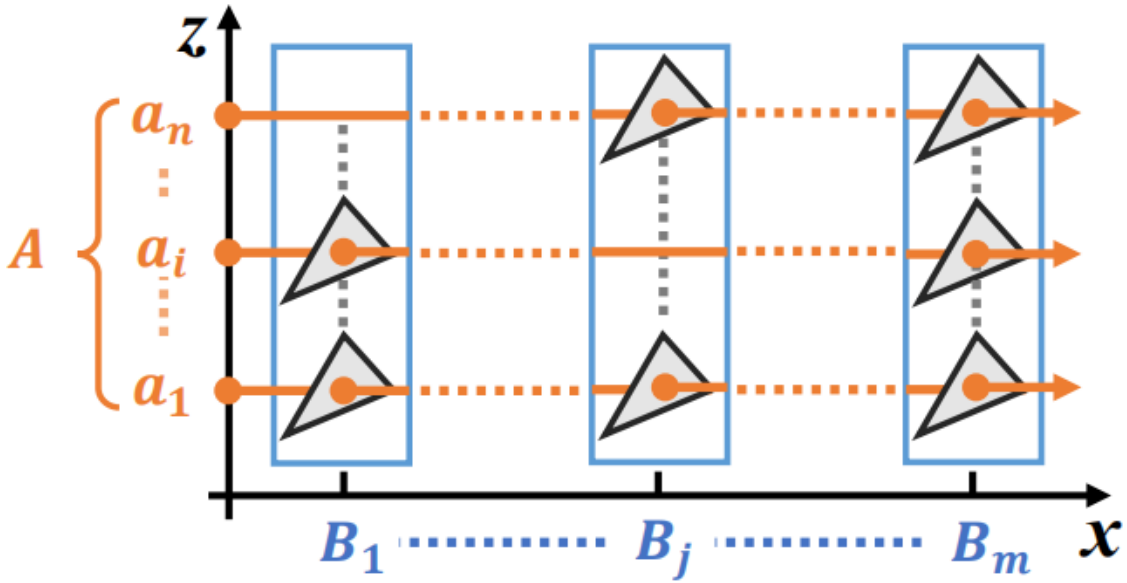
Understanding the Source of Performance Gains

- Replacing BVH traversal with CUDA-based binary search
- Superiority of RT-based TC results from a more load-balanced task scheduling
- BVH traversal on RT cores is less efficient for triangle counting



Evaluation of RT-based Set Intersection

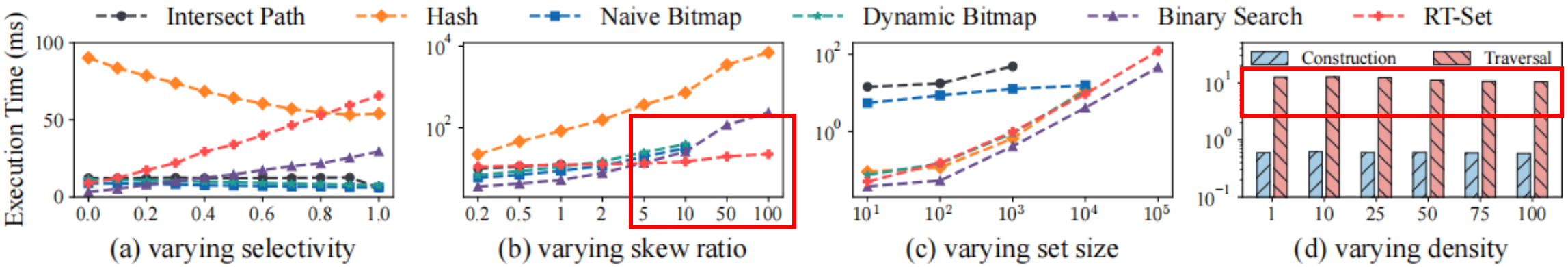
- Method description



- Intersection between two sets
- Intersection between one set and multiple sets

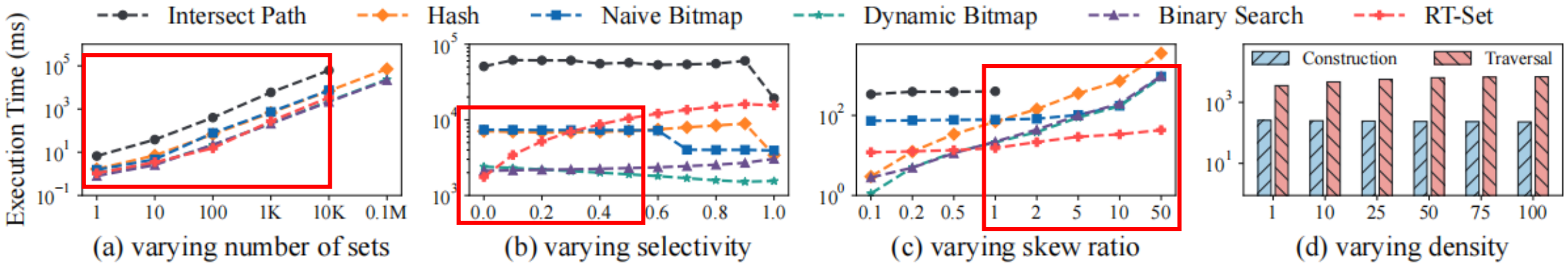
Pairwise Set Intersection

- RT-Set shows better performance when the skew ratio is high
- Higher density enhances RT-Set performance



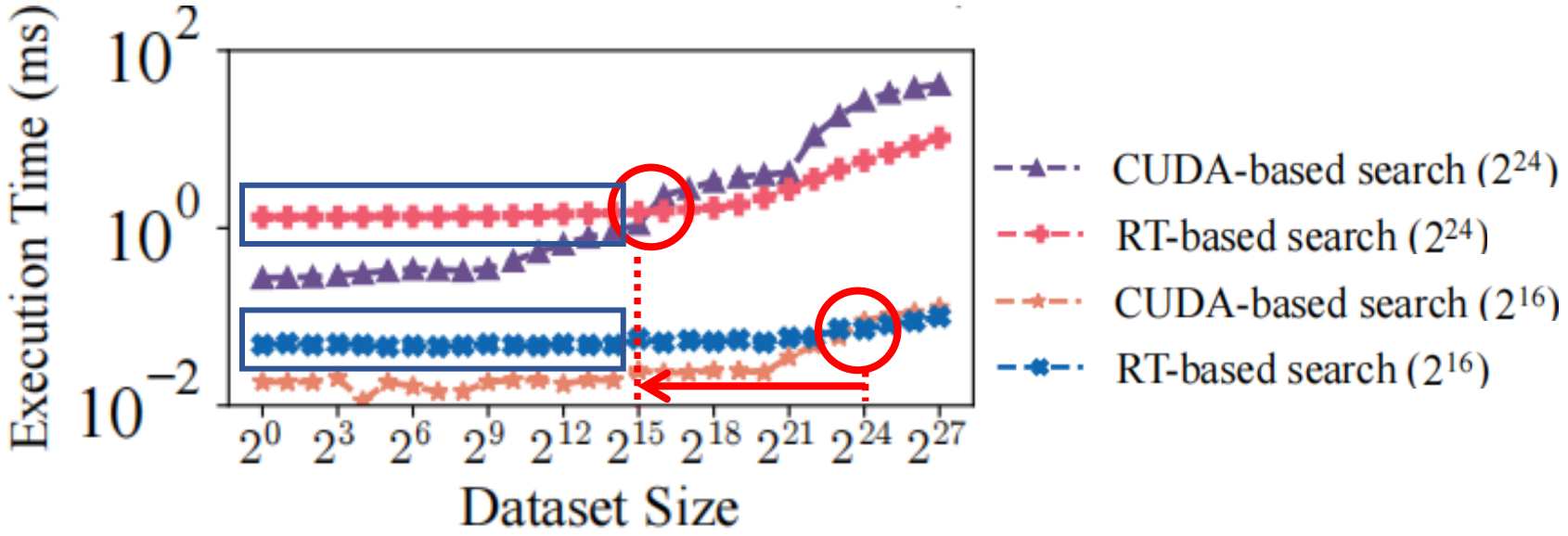
Intersection between One Set and Multiple Sets

- RT-Set has competitive performance when the number of sets is increasing
- Low selectivity favors the indexing performance of the BVH
- The advantage when the skew ratio is high is further magnified



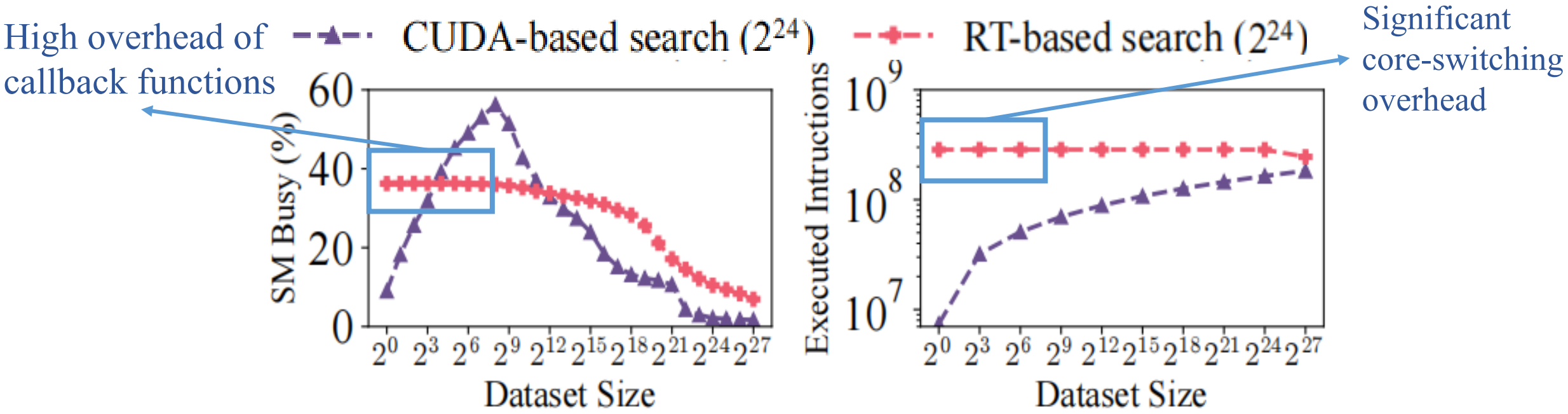
Analysis of Search Performance on RT Cores

- Comparison of RT-based search and CUDA-based search on one-dimensional sorted integers
  - Larger dataset and query size benefit RT-based search
  - RT-based search has non-trivial constant overhead independent of data size



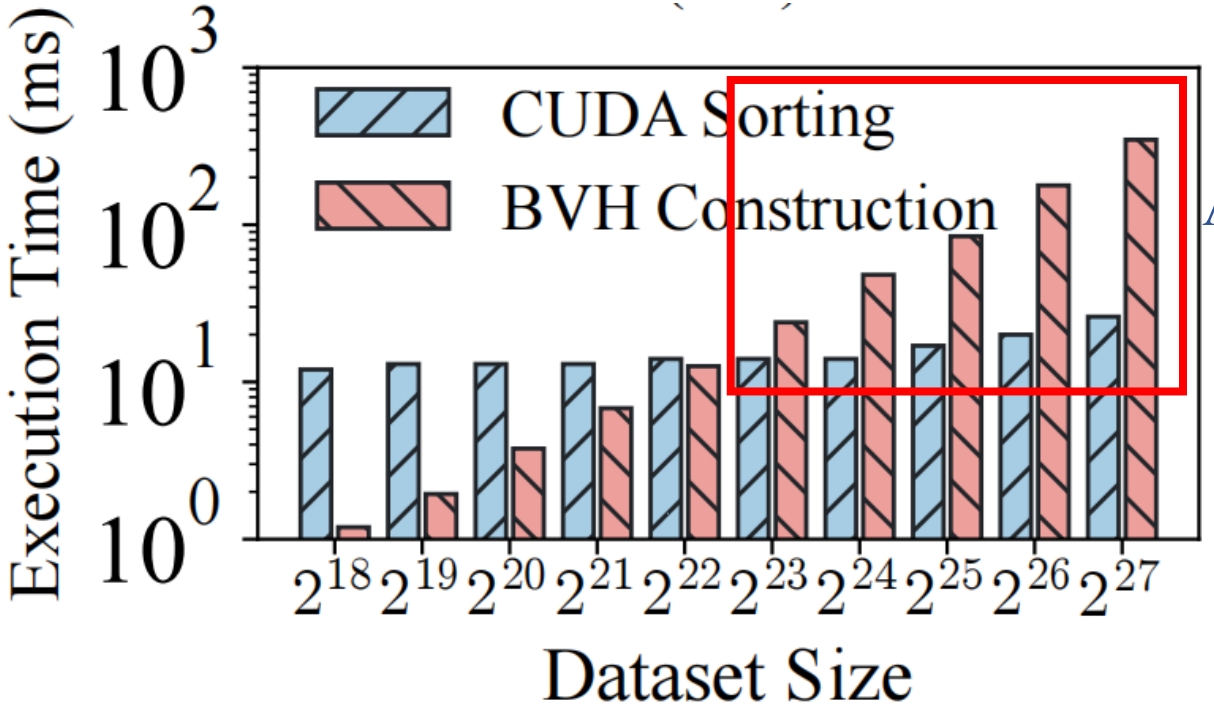
Kernel Profiling with Nsight Compute

- SM busy of RT-based search is more stable and callback overhead is high
- RT pipeline has notable core-switching overhead



BVH Construction Overhead Matters

- BVH Construction scales with dataset size while CUDA sorting remains stable



Advantage disappears

Conclusion

- RT-based methods offer little acceleration for sequential-access applications like BFS, and BVH is suboptimal for triangle counting
- For set intersection, RT cores gain advantages if the skew ratio between query and dataset is high
- The RT cores can more efficiently spot an element than CUDA cores, but they also incur a constant and non-trivial overhead from the OptiX execution pipeline
- The high BVH construction overhead and high memory footprint should be considered

Thank you!

Email: xiaozxiong@mail.sdu.edu.cn

The open-source code is available at: https://github.com/xiaozxiong/RT-Graph