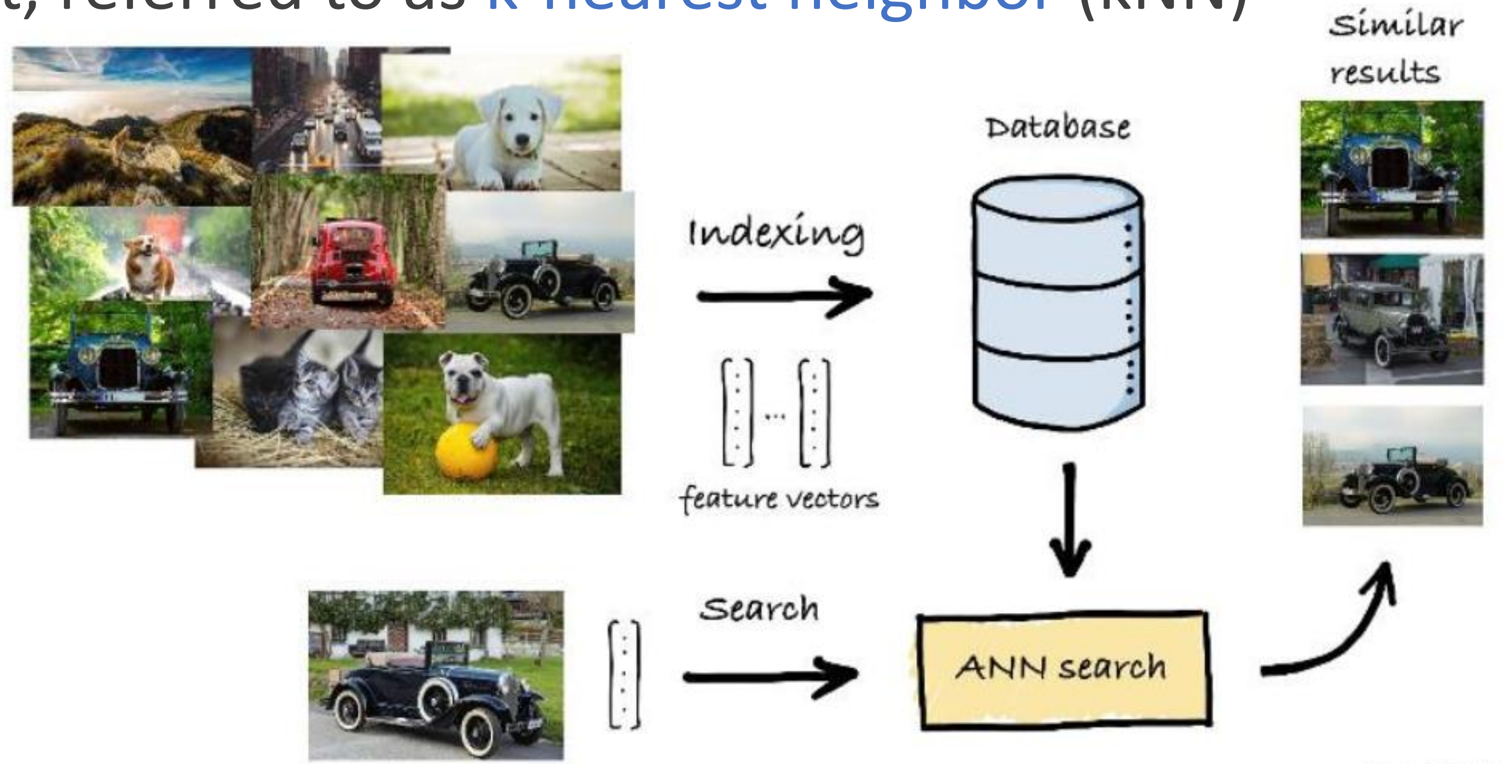
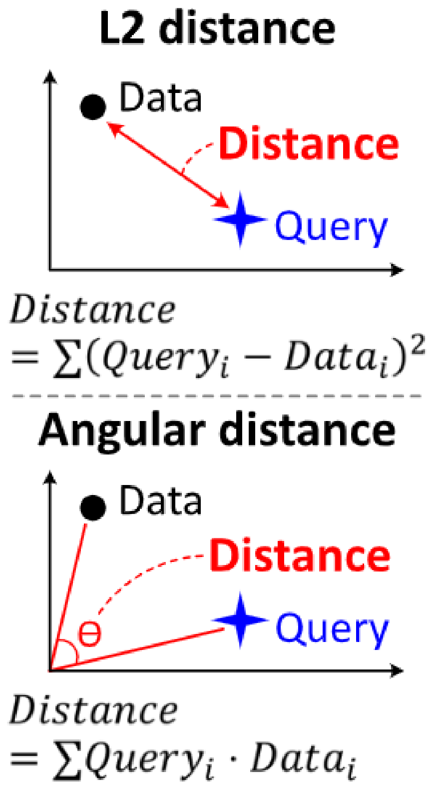
CXL-ANNS: Software-Hardware Collaborative Memory Disaggregation and Computation for Billion-Scale Approximate Nearest Neighbor Search

ATC' 23

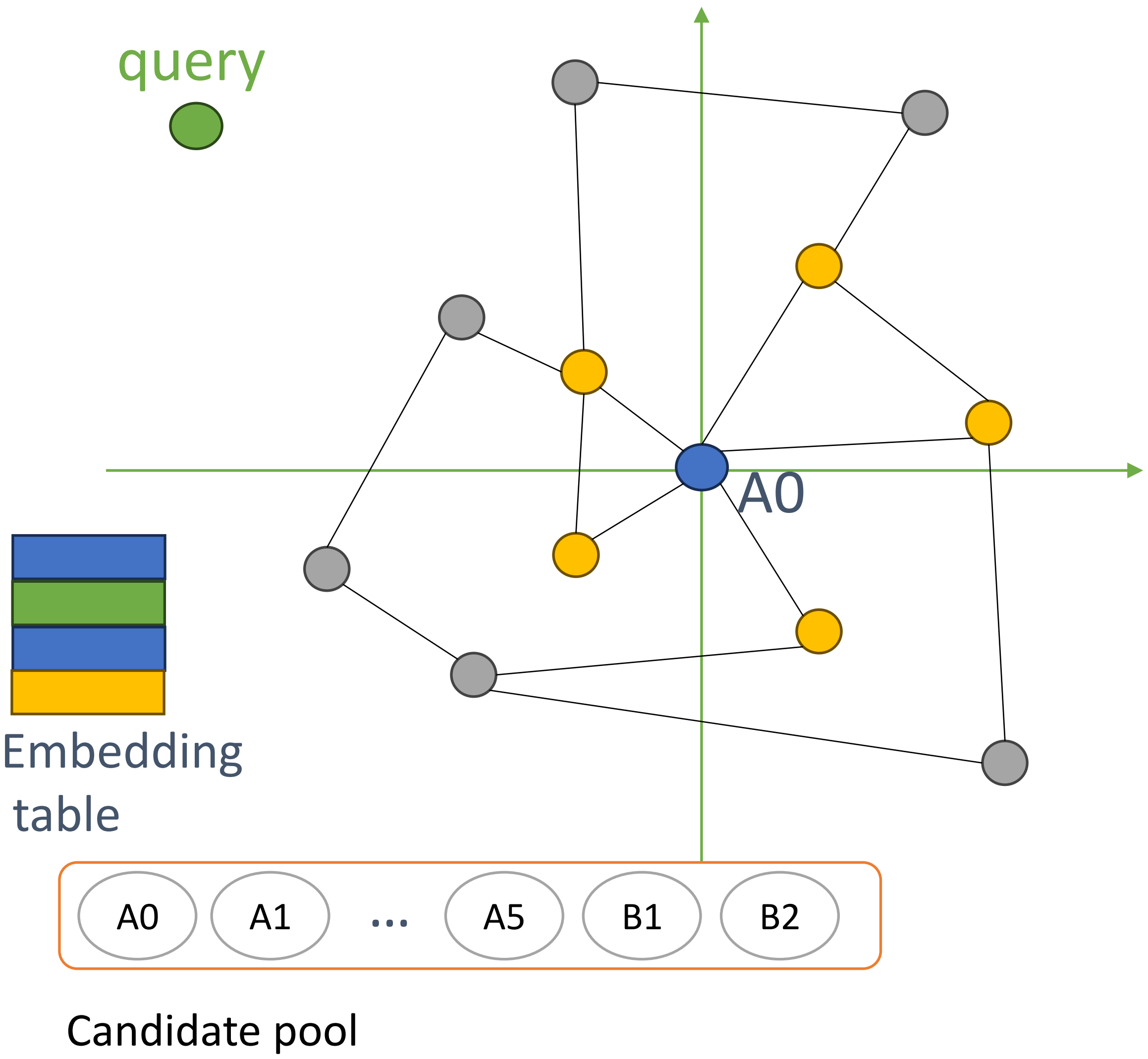
Background: ANNS

- Compares the similarity across different objects using their distance
- Retrieves a given number of objects, similar to the query object, referred to as k-nearest neighbor (kNN)





Background: Graph based ANNS



1. Start the search by visiting the entry node

   The entry node is fixed to 0 centroid to minimize the number of nodes to visit.

2. For each neighbor nodes, calculate the distance between the node and the query.

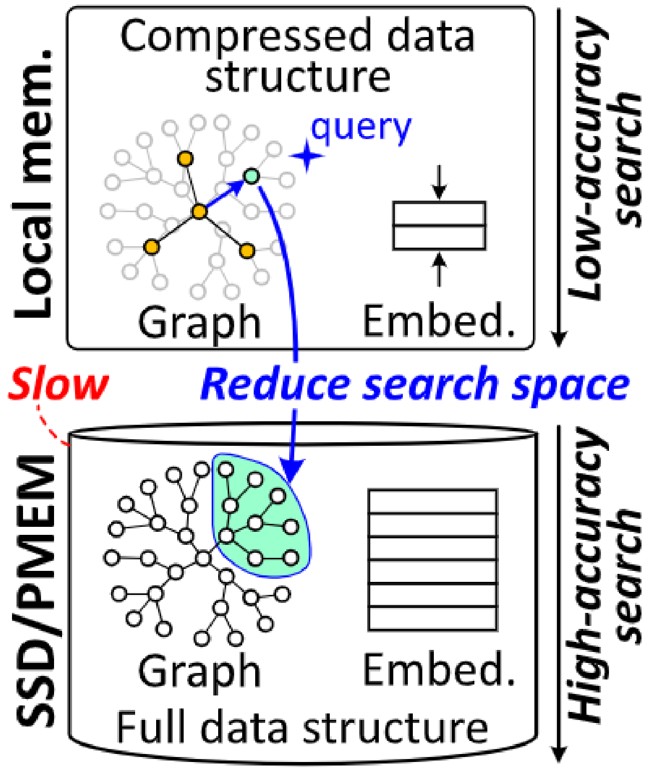
3. Determine the nearest, unvisited neighbor and move to that node

4. Repeat2-3.

   If the traverse keeps getting farther from the query, terminate.

Challange

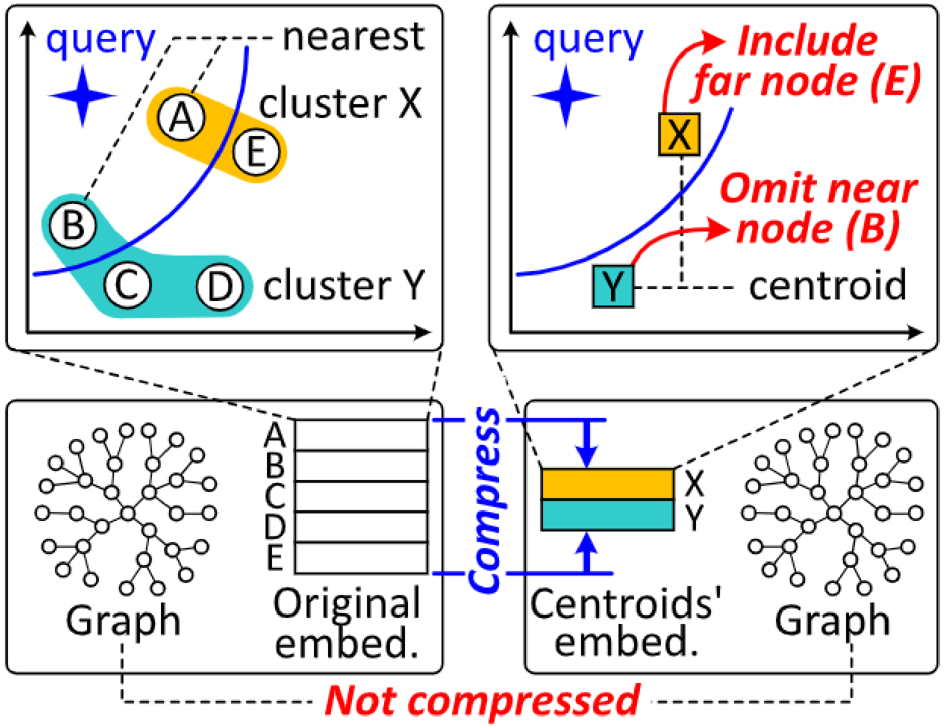
√ **Store dataset in large capacity SSDs**
× **Inevitable slow storage access**
× **Low search performance**



Compression-based approach

Hierarchical approach

Challange



√ **Reduces the memory consumption**

× **Low search accuracy**

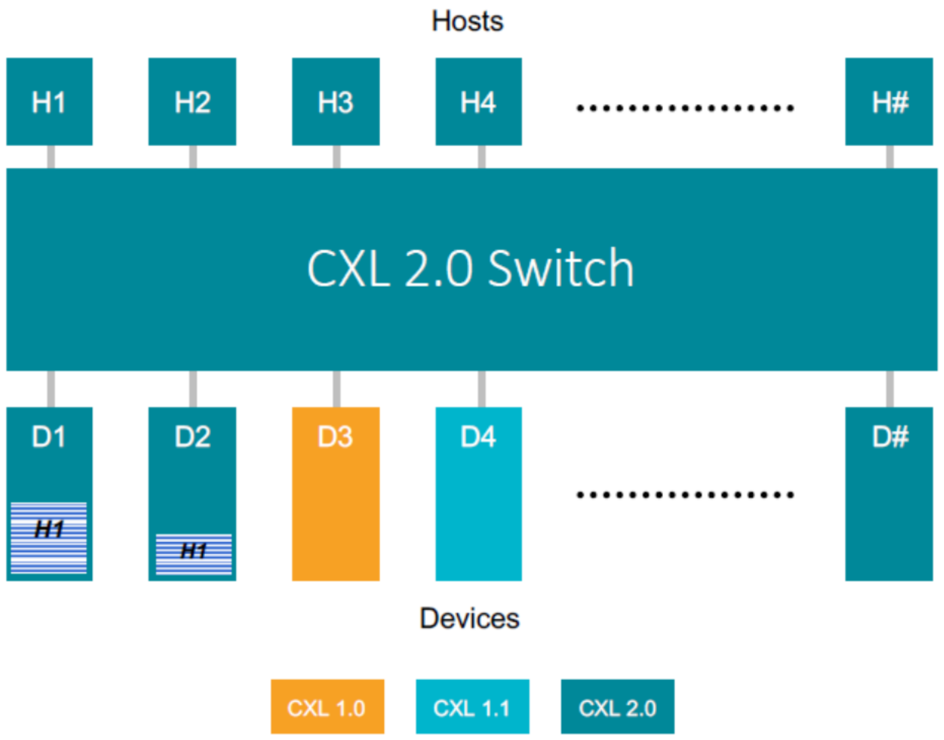
× **Errors in distance calculation**

Compression-based approach

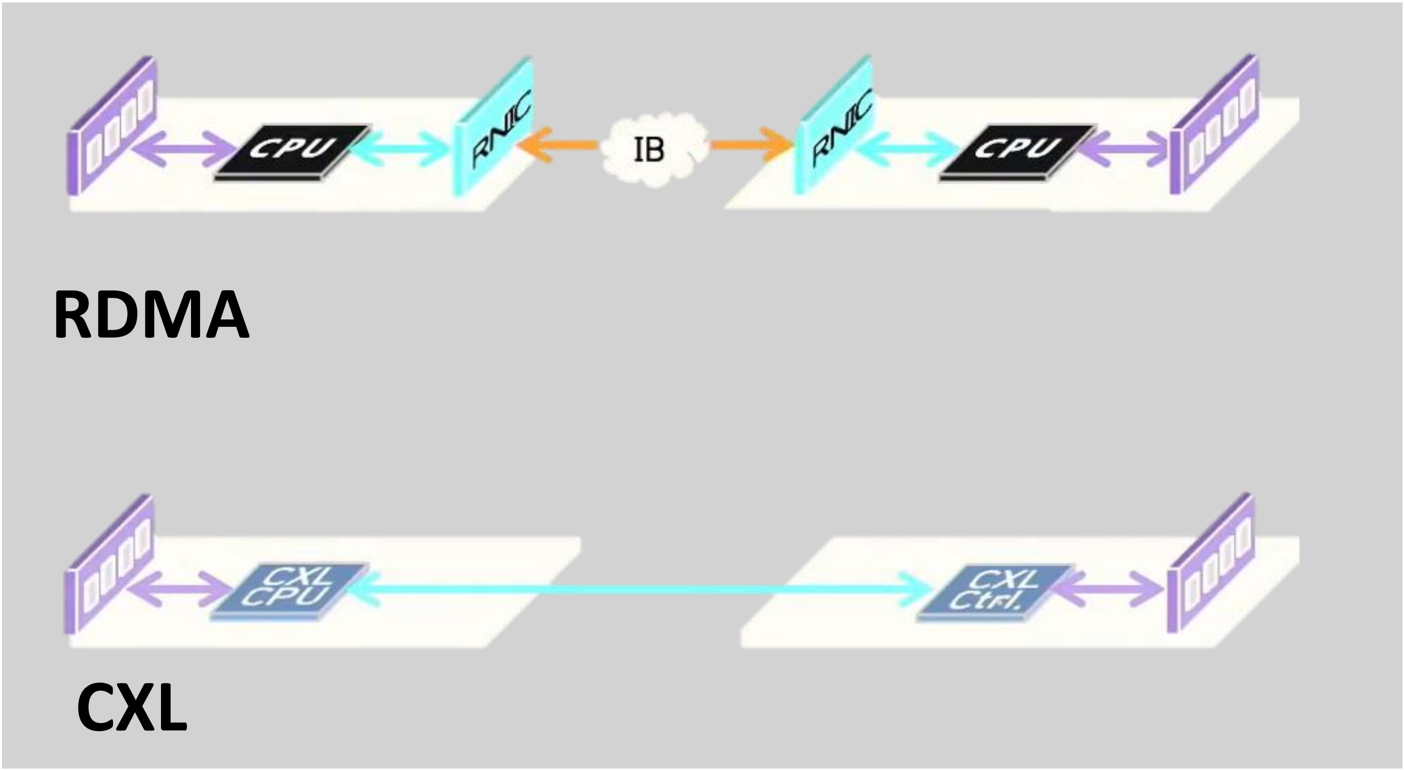
Hierarchical approach

Background: CXL

CXL based Memory disaggregation

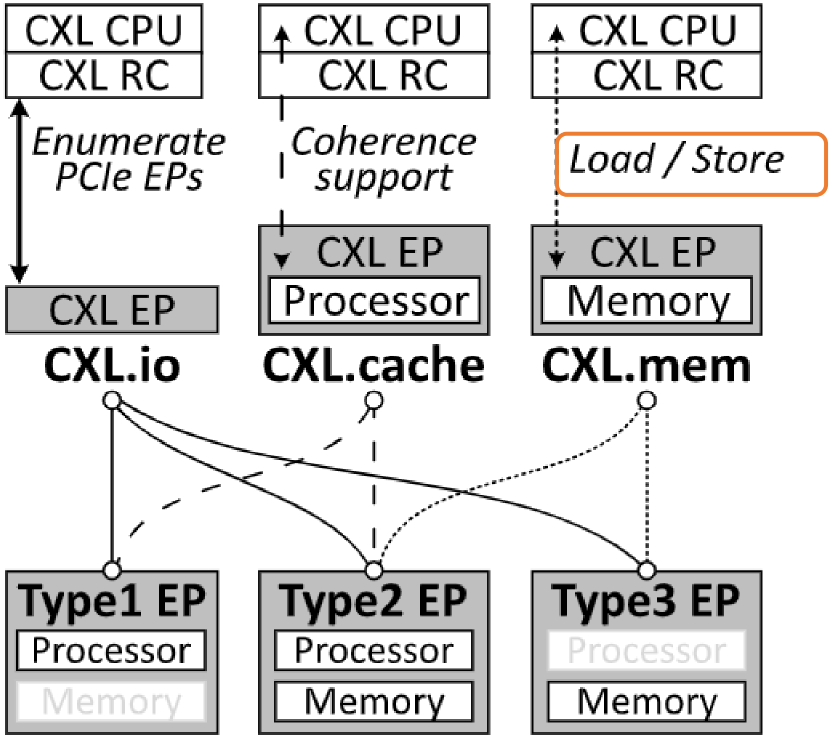


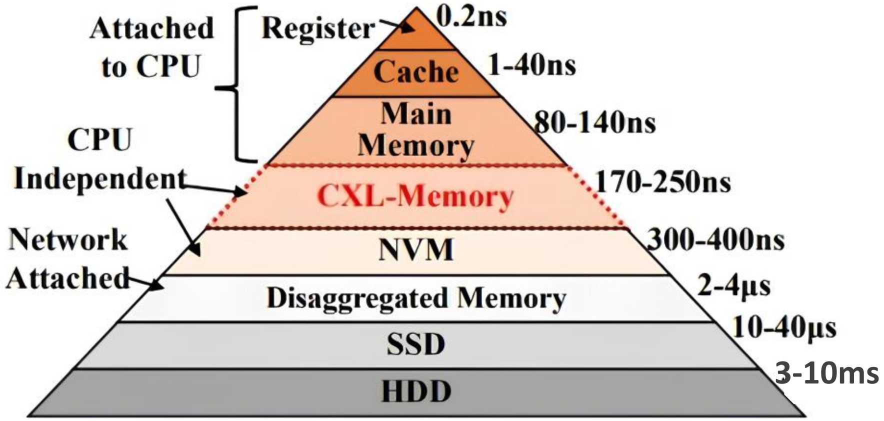
Compared with RDMA



Background: CXL

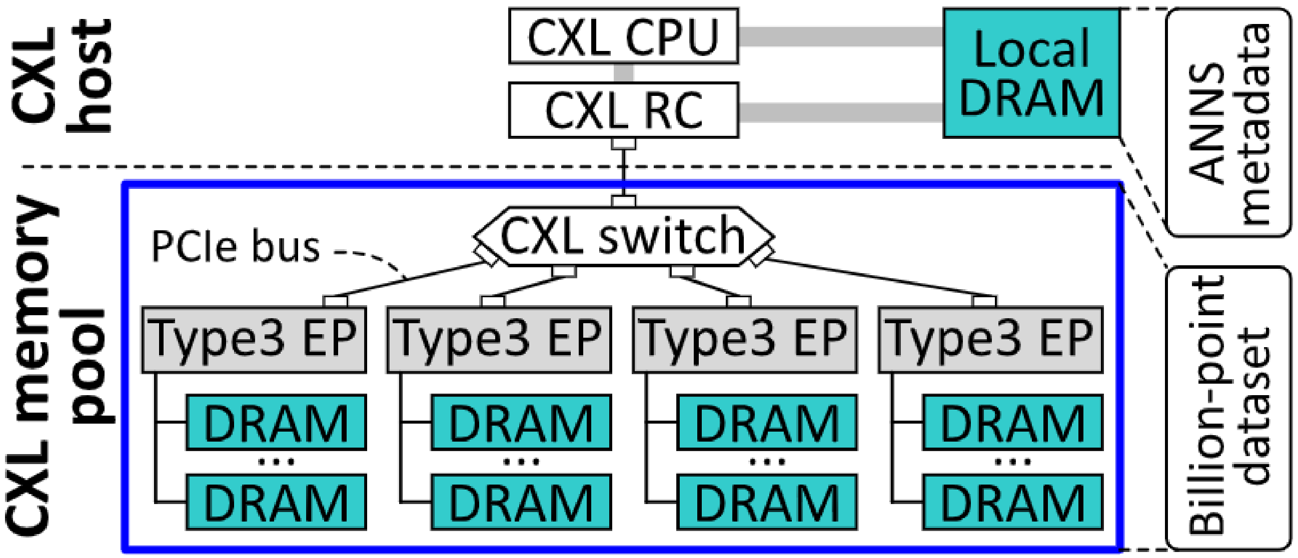
- Tpye of CXL endpoint devices (EP)





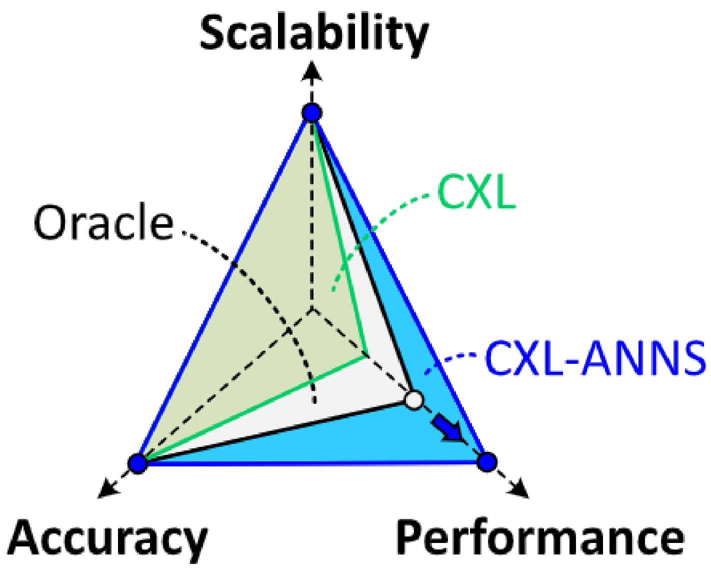
Baseline: CXL-augmented ANNS

- Directly have billion-point datasets in a scalable memory pool, disaggre-gated using CXL.
  - ANNS metadata in the local DRAM.
  - Locate all the billion-point graphs and corresponding vectors to EPs

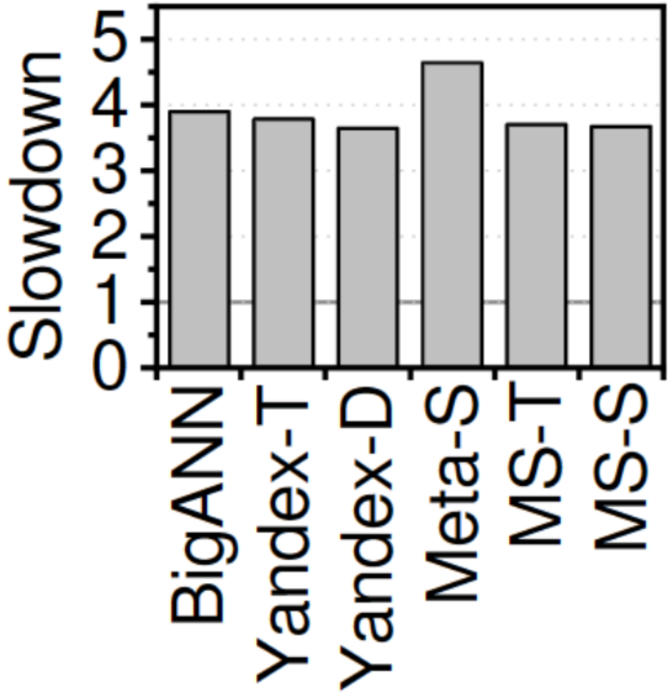


Problem

- CXL-ANNS exhibits 3.9× slower search latency than the oracle
  - Accessing DRAM in EPs is still slower than dir
  - Graph traversa frequent visits
  - Distance comp due to the high



(b) CXL-based approaches.



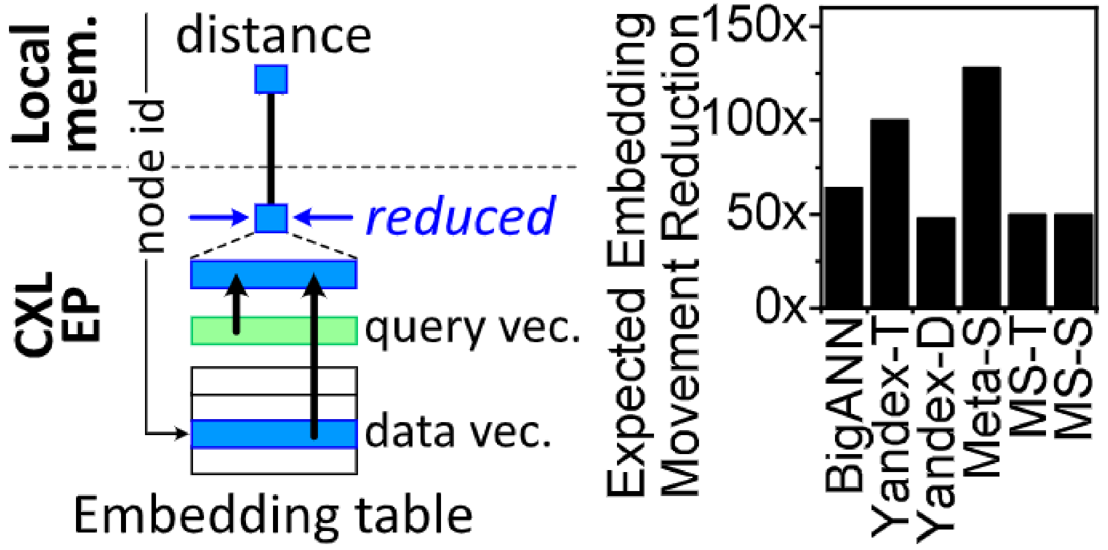
Main idea

- Cache frequently accessed nodes and vectors in local DRAM
- Distance calculation using EP-side computing resources

    Reducing data vector transfers

- Optimise query schedule

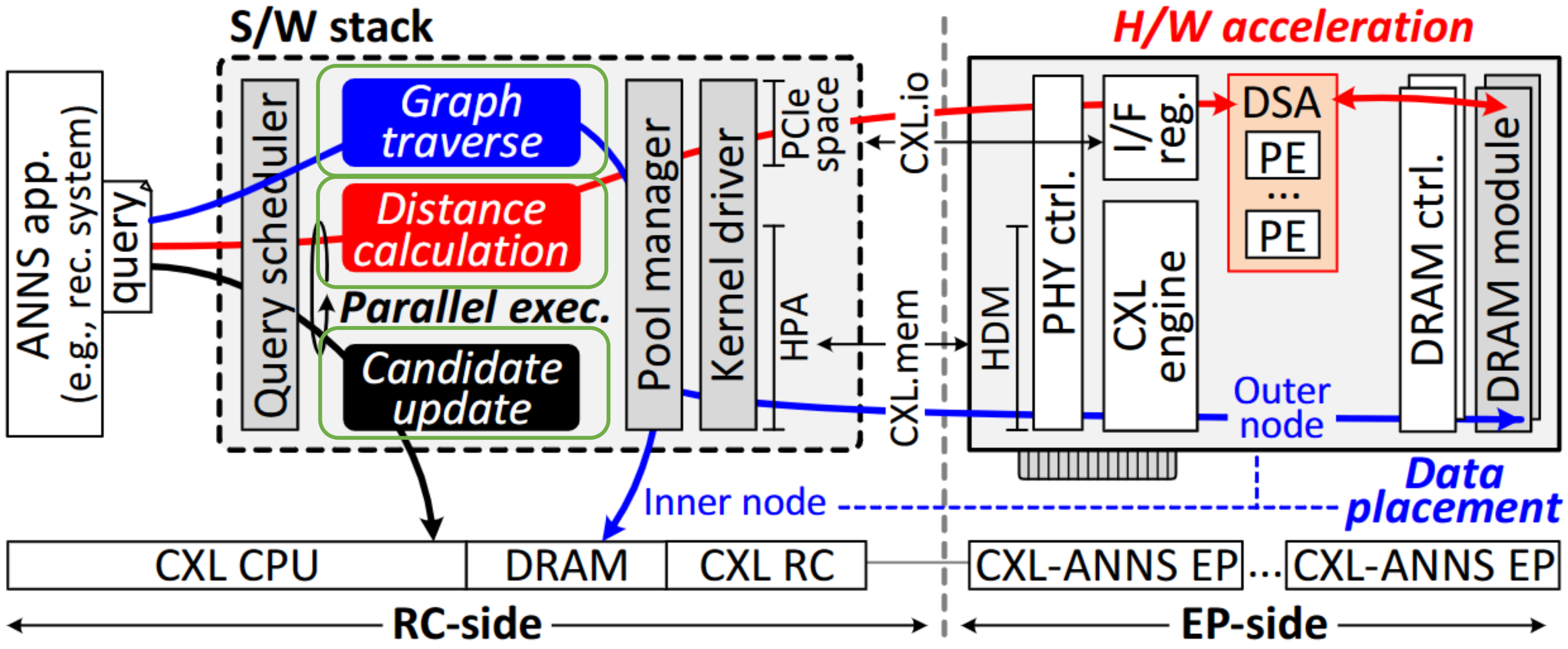
    parallelism, granularity



Data reducing

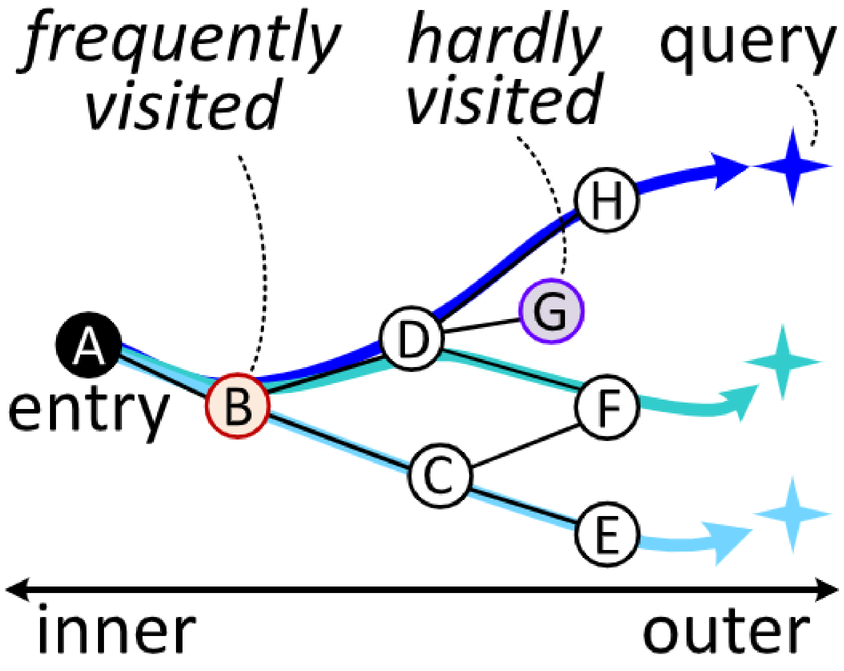
Architecture

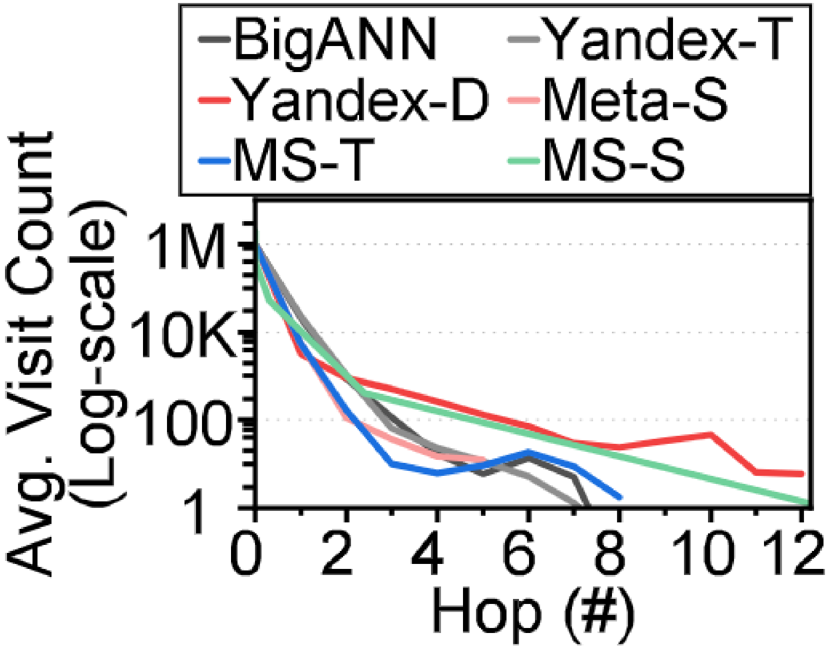
- **RC-side: handle query and manages the EPs**
- **EP-side: distance calculation**



Design1: Local Caching for Graph

- The graphs starts their traverses from a unique, single entry-node
- The graph traverse of ANNS visits the nodes closer to the entry-node much more frequently

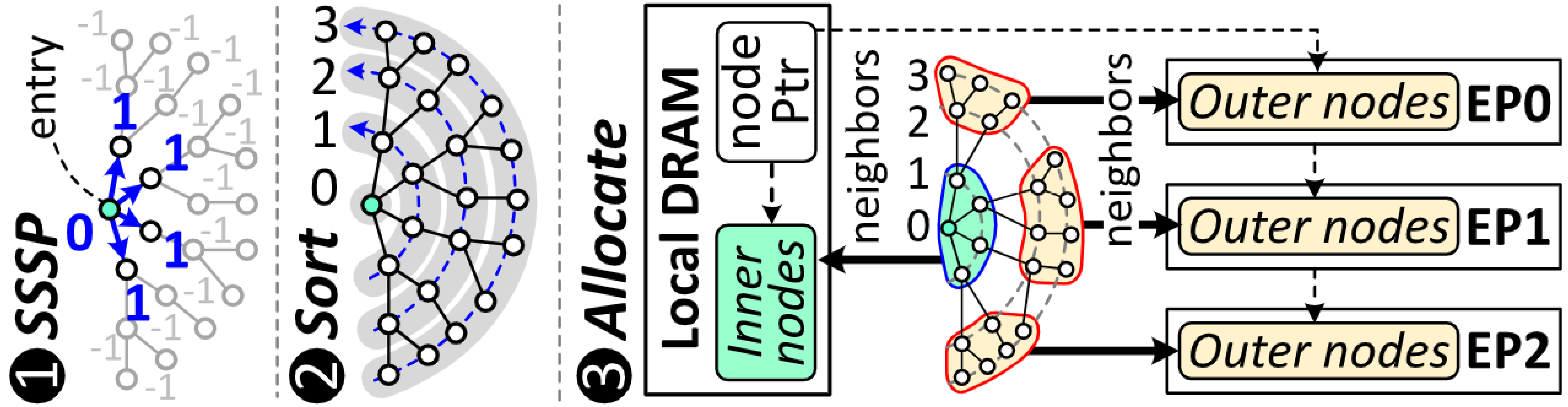




Node's visit count

Design1: Local Caching for Graph

- Caches the nodes, expected to be most frequently accessed, in local DRAM
- Considers how many edge exist from the fixed entry-node to each node for its relationship-aware graph cache

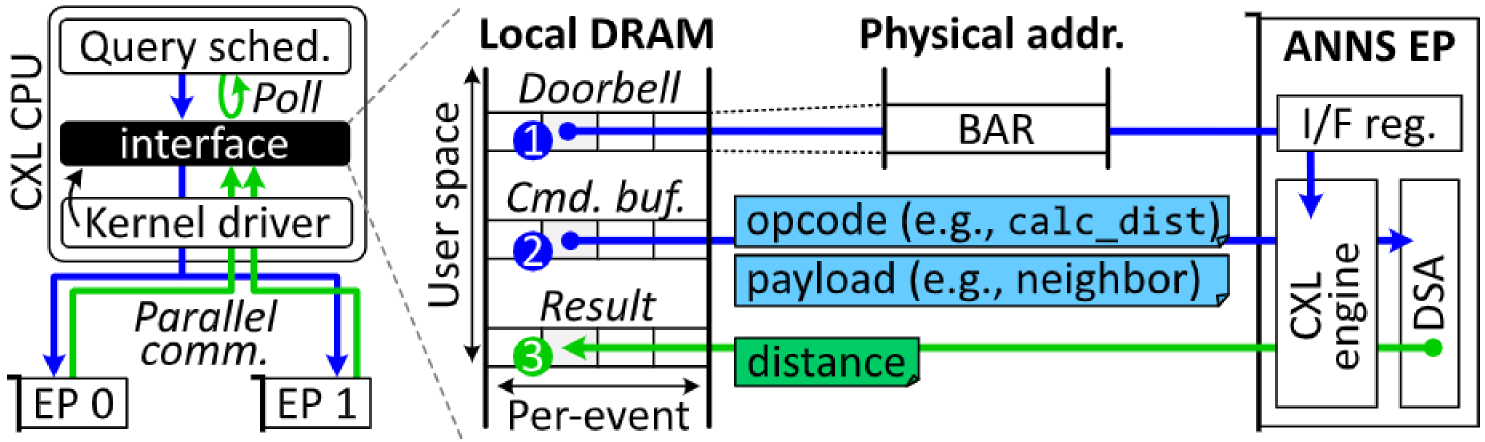


Data placement

Design2: Accelerating Distance Calculate

- Doorbell: Notify EPs to caluate distance
- Cmd buf: EP's CXL engine pulls the opt type and neighbor list

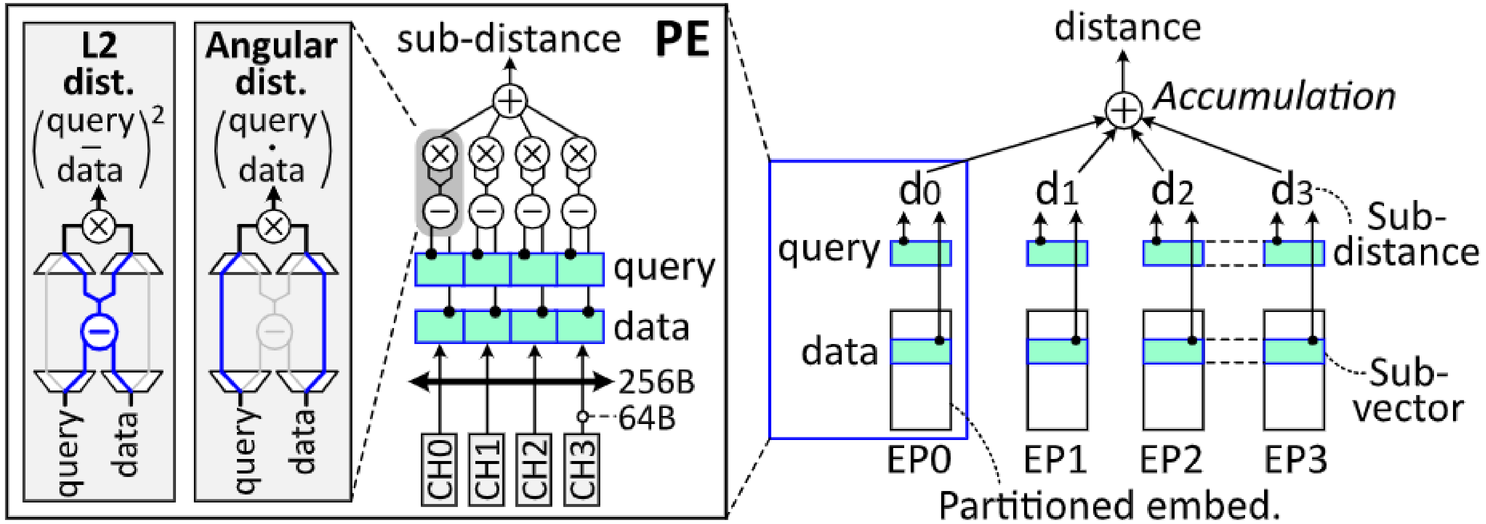
  CXL engine also pushes results of distance calculation to the local DRAM



Interface

Design2: Accelerating Distance Calculate

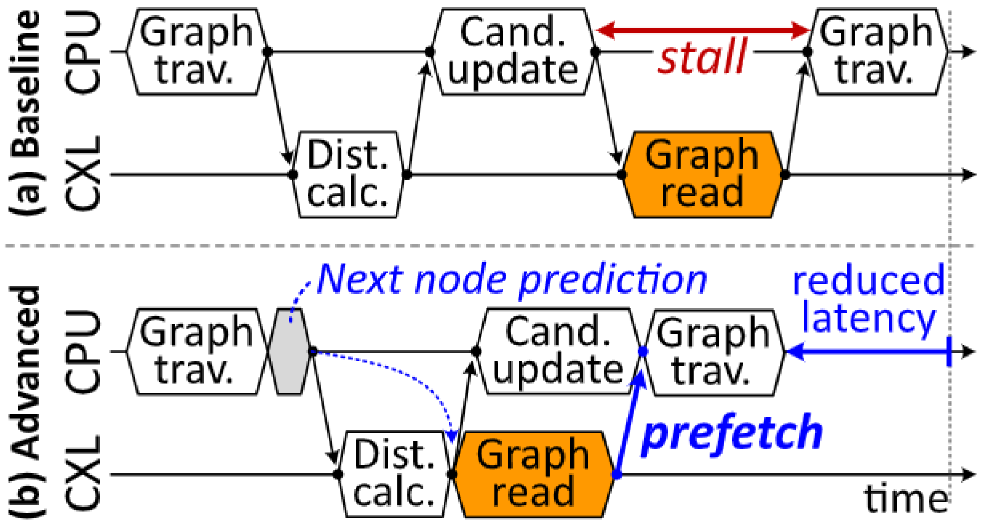
- Sharding: shards the embedding table and stores in the different EPs.
- processing element (PE)
  - multiplier and subtractor for element-wise operations.
  - reads data from all four different DIMM channels in parallel.

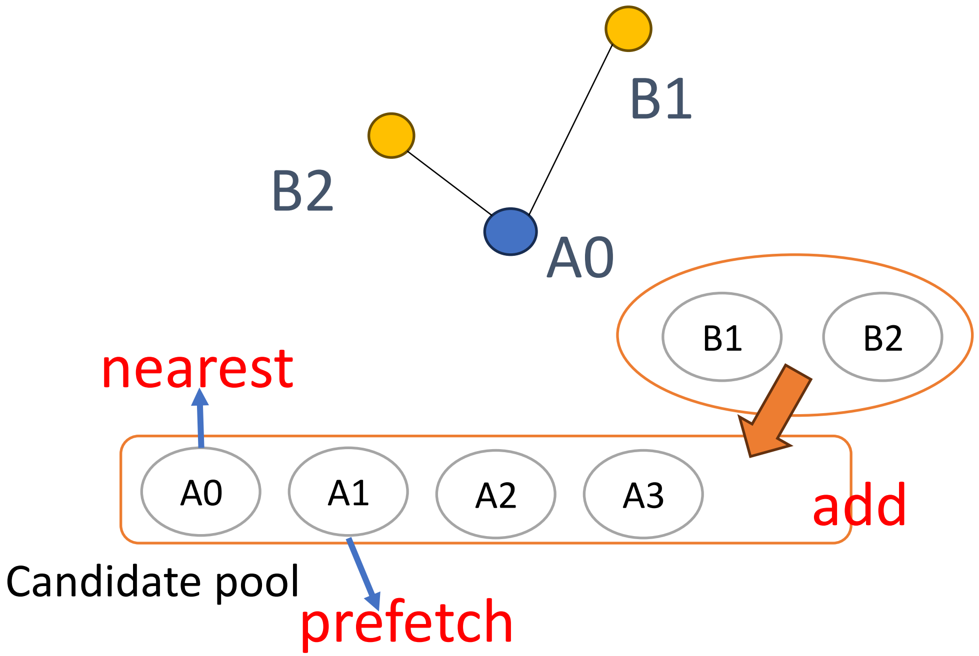


(a) PE architecture.　　　(b) Sharding.

Design3: Optimise query scheduling

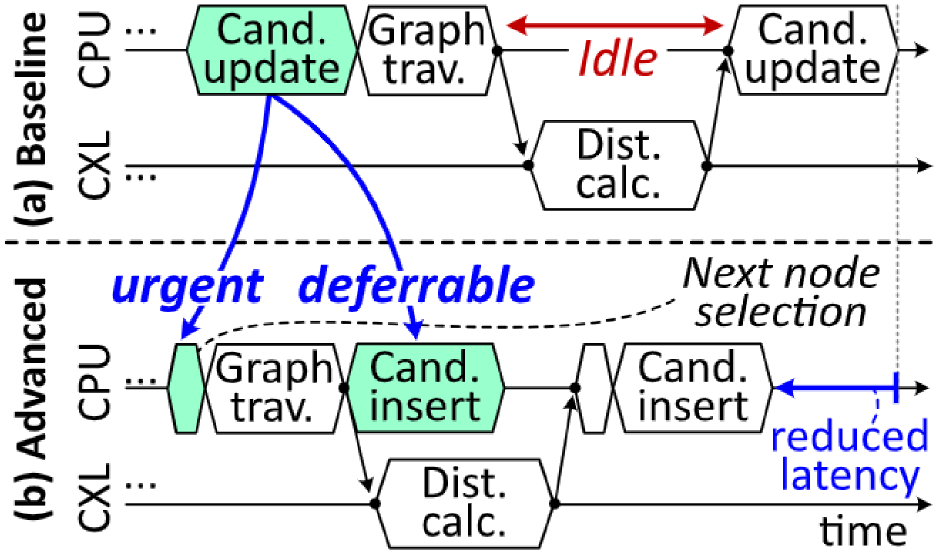
- It is required to go through the CXL memory pool to get nodes, which does not sit in the inner most edge hops.
  - prefetches the graph information earlier than the actual traverse subtask needs
  - speculates the nodes to visit and brings their neighbor information by referring to the candidate array



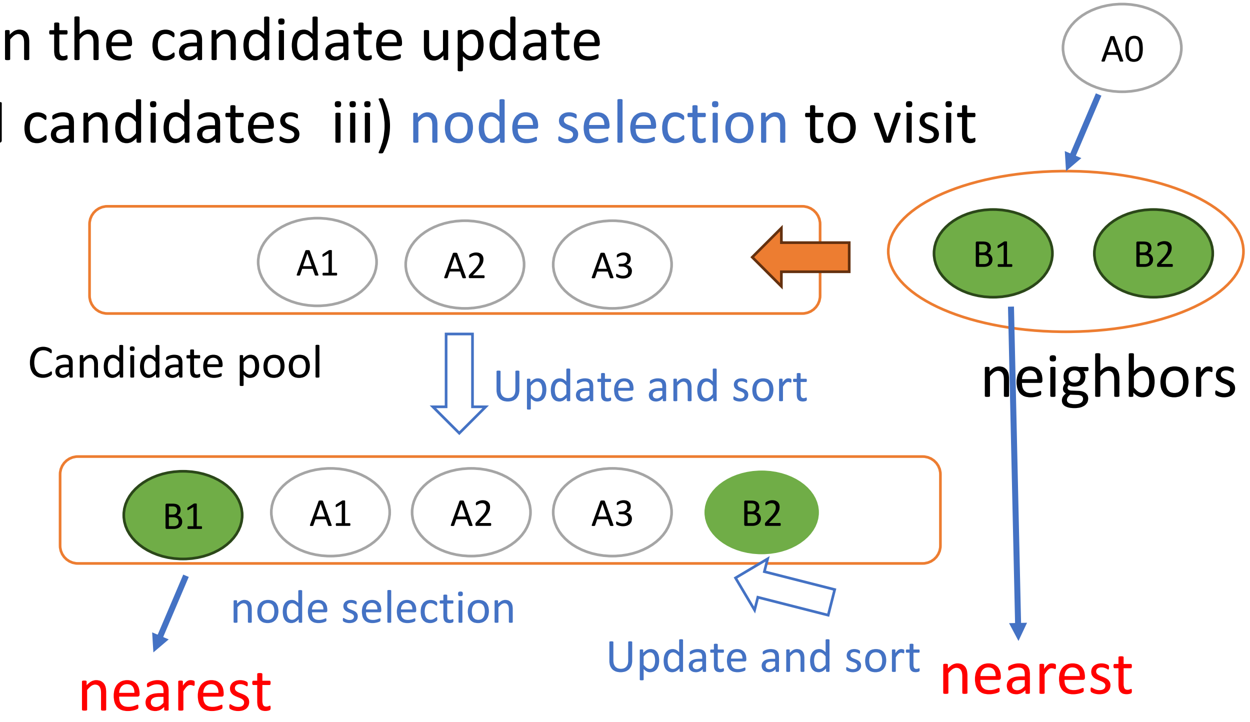


Design3: Optimise query scheduling

- Computing kNN search in different places makes the RC-side ANNS subtasks pending.
  - relaxes the execution dependency on the candidate update
  - i) update candidates ii) sorting kNN candidates iii) node selection to visit



Query scheduling



Resource utilization

Evaluation

CPU	40 O3 cores, ARM v8, 3.6GHz L1/L2 $: 64KiB/2MiB per core
Local memory	128GiB, DDR4-3200
CXL memory pool	1 CXL switch 256GiB/device, DDR4-3200
Storage	4× Intel Optane 900P 480 GB
CXL-ANNS	1 GHz, 10 ANNS PE/device, 2 distance calc. unit/PE

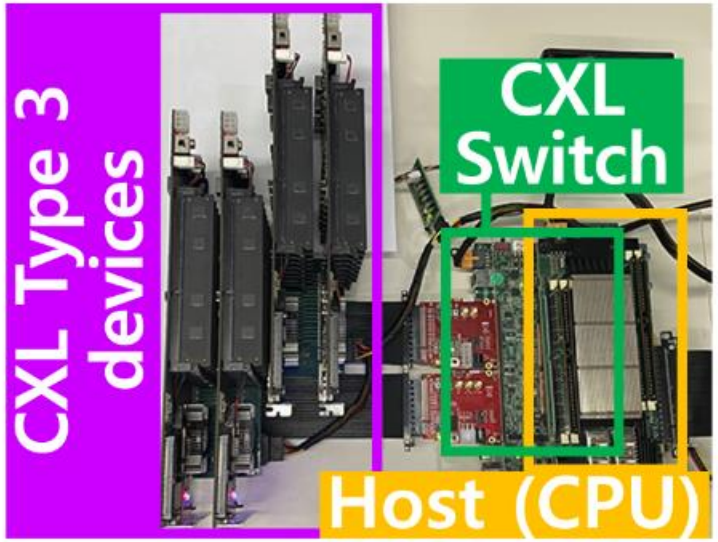


Figure 21: Prototype.

- Comparisons
  - compression approach [TPAMI'10]
  - hierarchical approach [NeurIPS'19][NeurIPS'20]

Evaluation

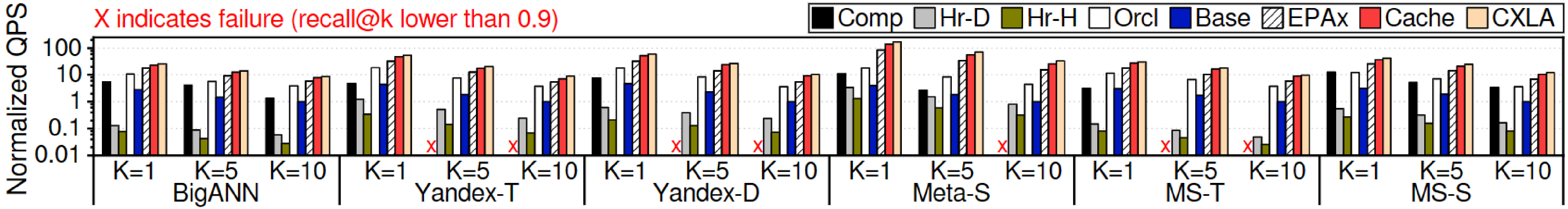


Figure 22: Throughput (queries per second).

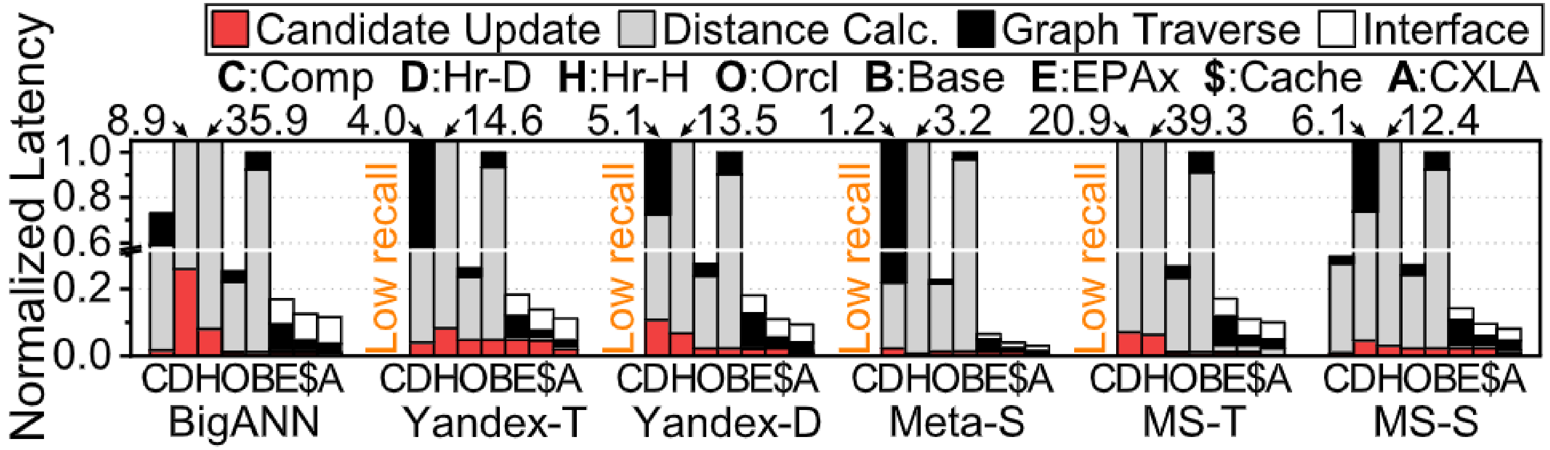


Figure 24: Single query latency ($k = 10$).

Evaluation

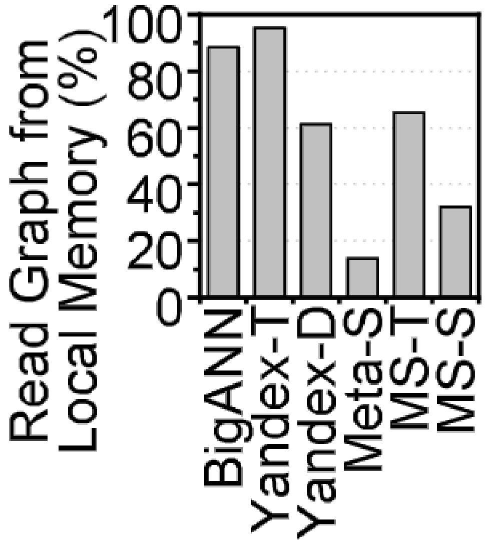


Figure 26: Local caching.

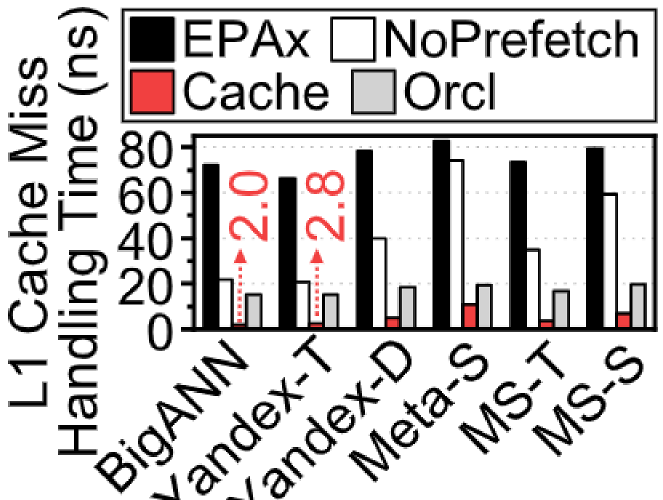
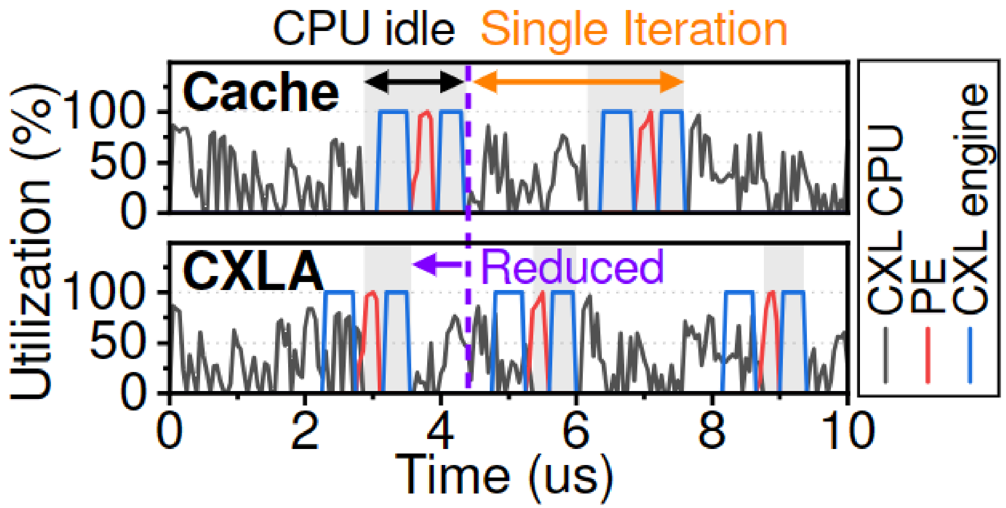
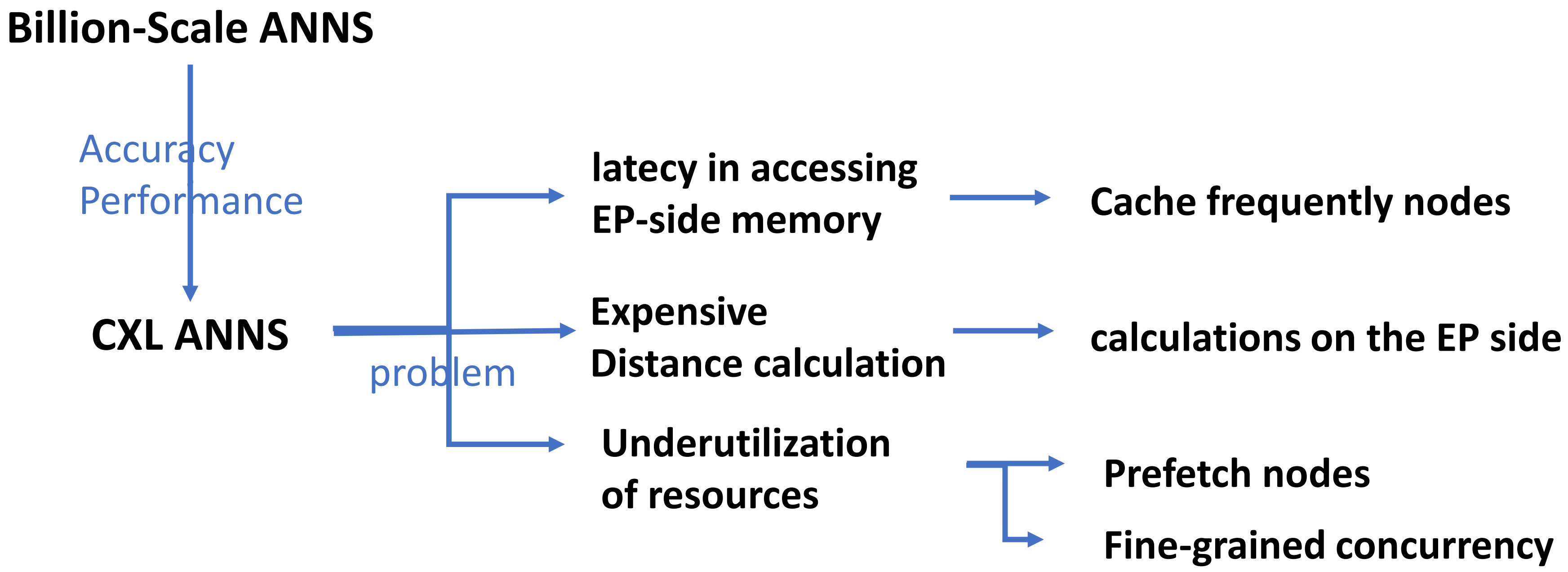


Figure 27: Cache miss handling time.



- Cache improves EPAx’s graph traversal time by3.3×
- CXLA reduces the idle time by 1.3×

Summary



Billion-Scale ANNS
Accuracy
Performance
CXL ANNS
problem
latecy in accessing
EP-side memory
Expensive
Distance calculation
Underutilization
of resources
Cache frequently nodes
calculations on the EP side
Prefetch nodes
Fine-grained concurrency