NVMe + CPU + GPU = Memory Efficient Analytics

HetCache: Synergising NVMe Storage and GPU acceleration for Memory-Efficient Analytics

Hamish Nicholson, Aunn Raza, Periklis Chrysogelos, Anastasia Ailamaki

DATA-INTENSIVE APPLICATIONS AND SYSTEMS

EPFL

The Broken Pillars of Fast Analytics

~~Memory is cheap~~

- Memory is (relatively) expensive

~~Cache hits are key to performance~~

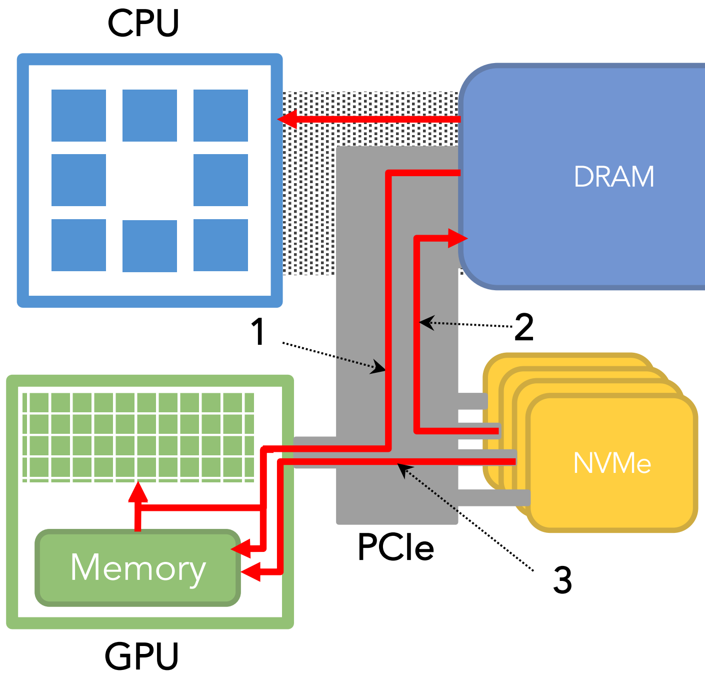
- NVMe array bandwidth competitive with memory bandwidth

~~NUMA is insignificant compared to persistent storage access~~

- Increasing accelerator heterogeneity

Storage must be workload & hardware aware

Heterogeneous Hierarchies have Multiple Transfer Paths



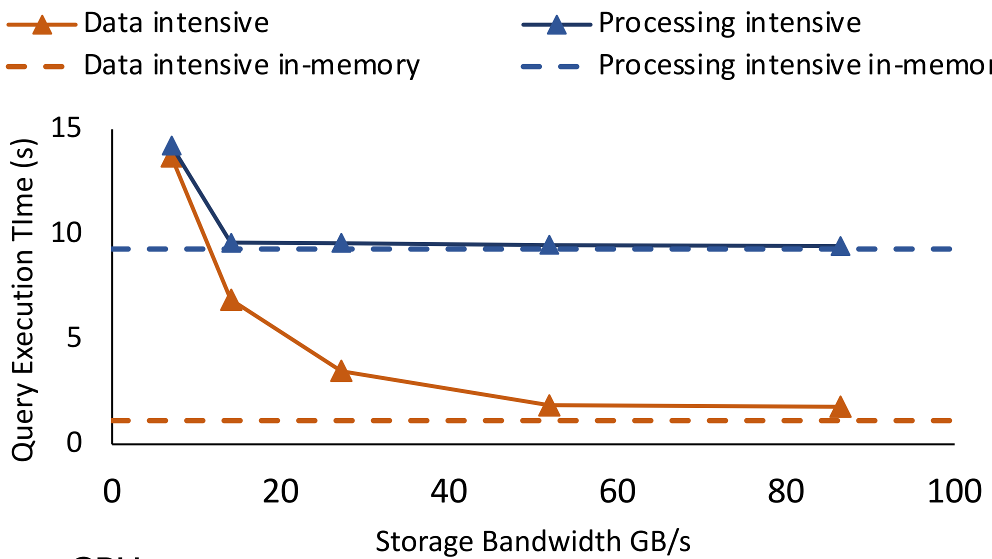
1. DRAM to GPU (32 GB/s)
   - Eagerly transfer pages to GPU-memory
   - Byte-addressable access by GPU

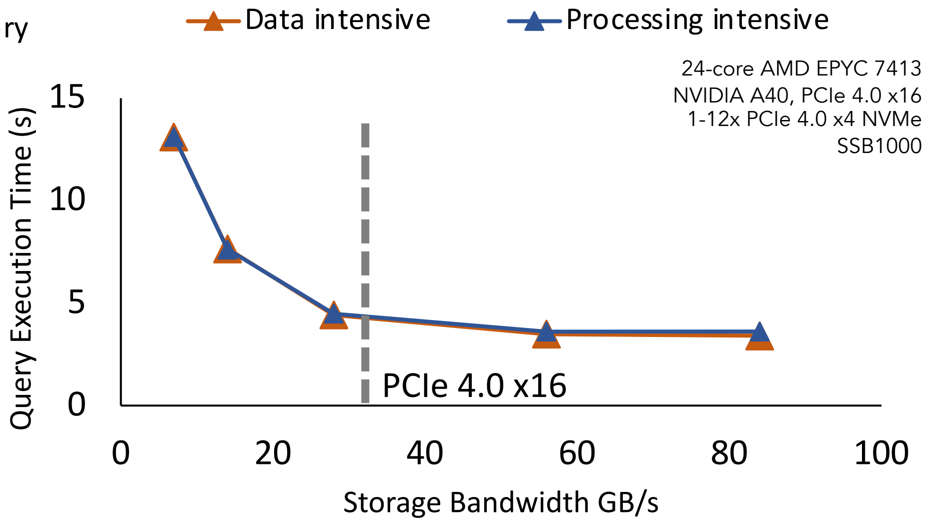
2. NVMe to DRAM (86 GB/s, block)

3. NVMe to GPU-memory to GPU (32GB/s, block)

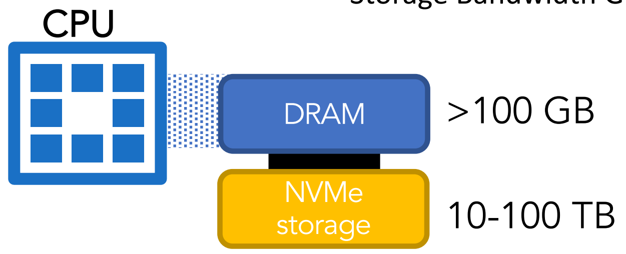
Data routing requires optimizing for path BW & granularity

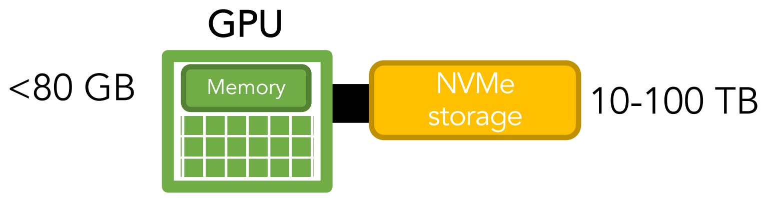
NVMe BW Saturates CPU Throughput





24-core AMD EPYC 7413
NVIDIA A40, PCIe 4.0 x16
1-12x PCIe 4.0 x4 NVMe
SSB1000





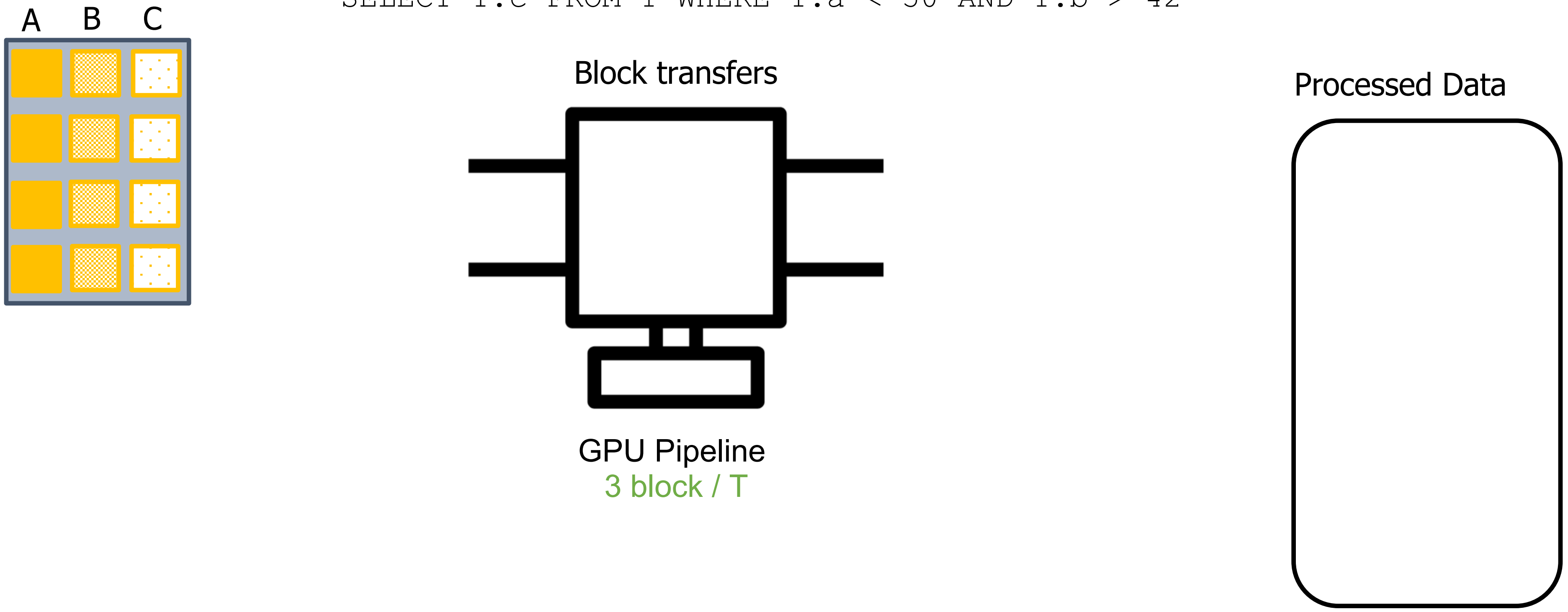
GPU needs caching to mitigate interconnect bottleneck

Block Storage Wastes Interconnect BW

NVMe Storage

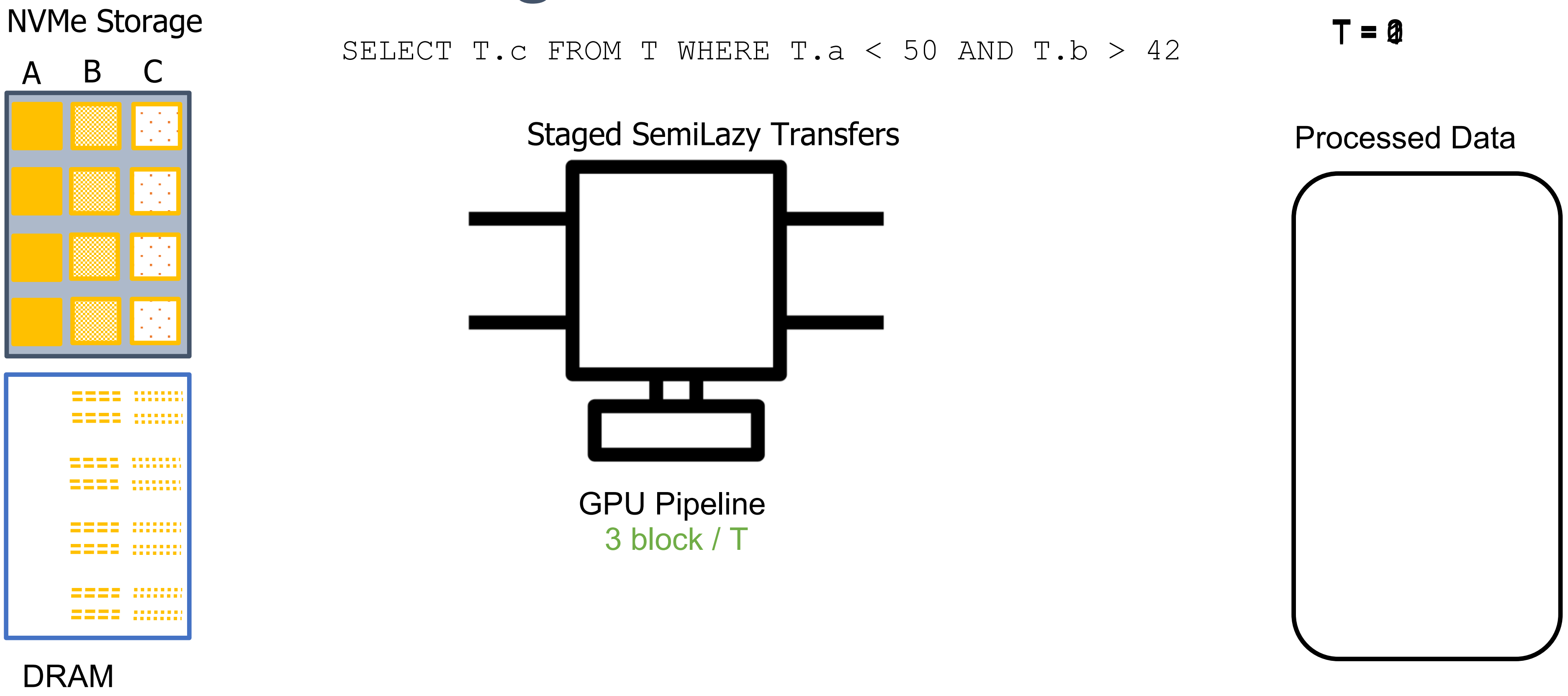
```
SELECT T.c FROM T WHERE T.a < 50 AND T.b > 42
```

T = 0



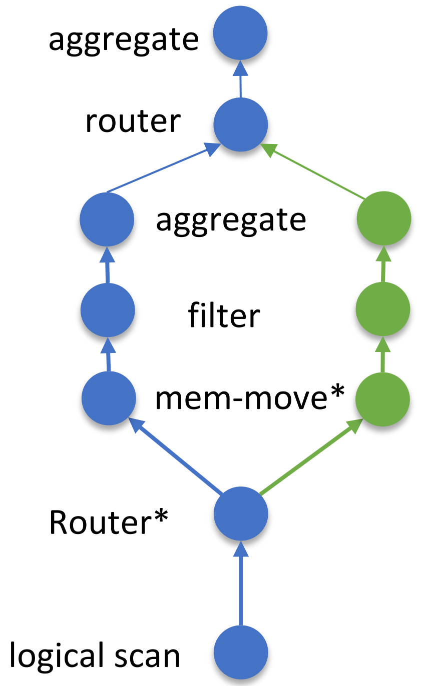
Block Storage Wastes Interconnect BW

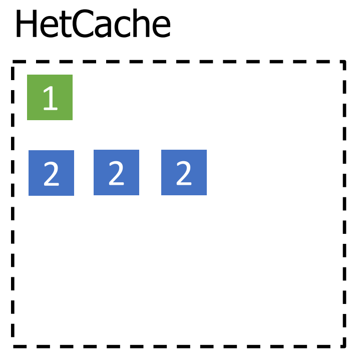
```
SELECT T.c FROM T WHERE T.a < 50 AND T.b > 42
```

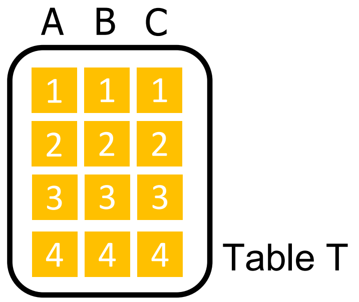


Byte addressability minimizes overfetching

Transfer Path Depends on Workload and Hardware





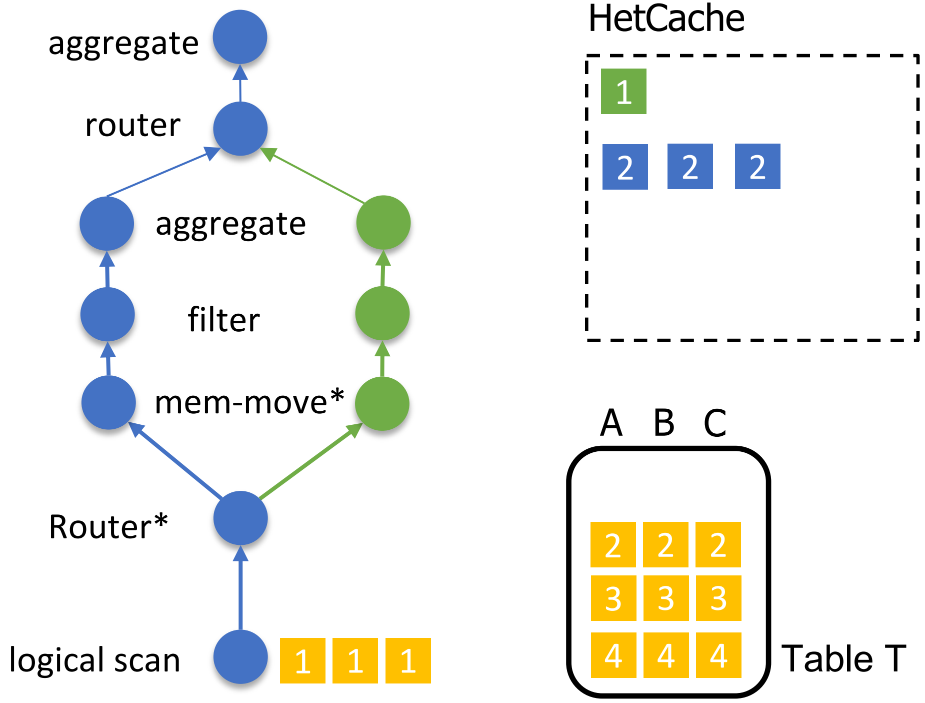


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs

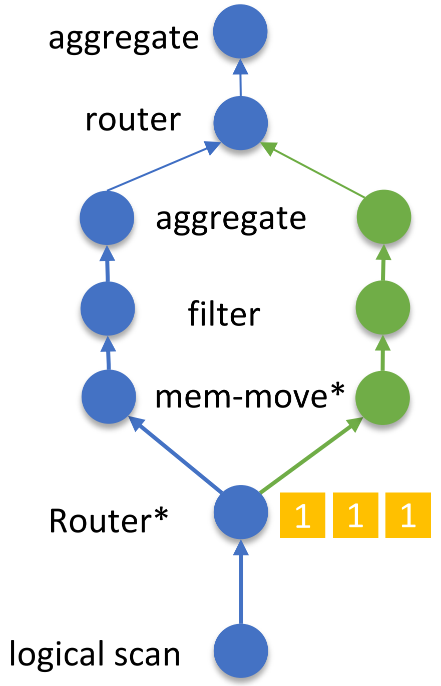


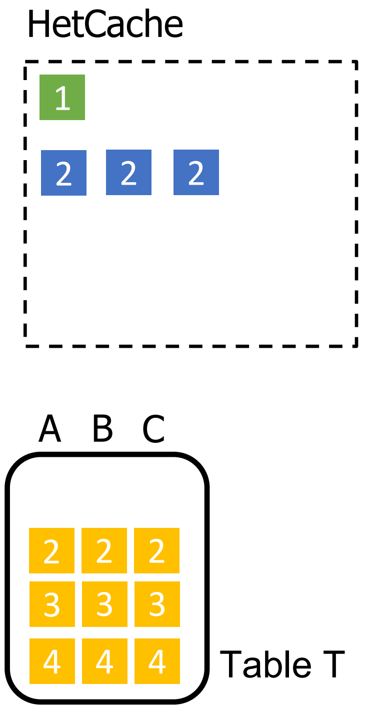
*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents



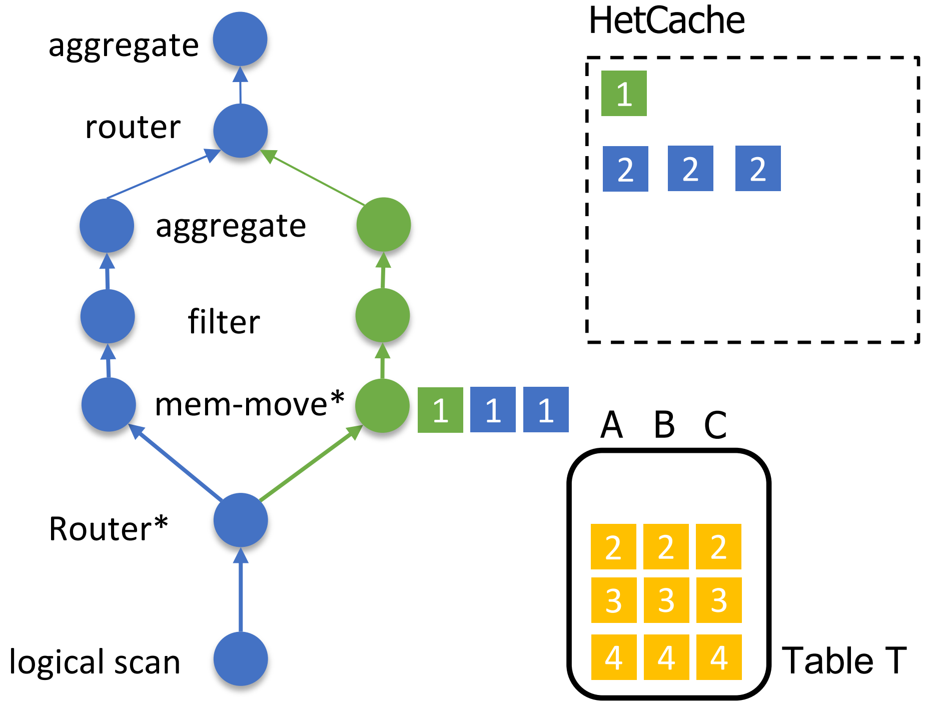


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
    - Locations of in-memory copies
    - Preferred location to cache, if any

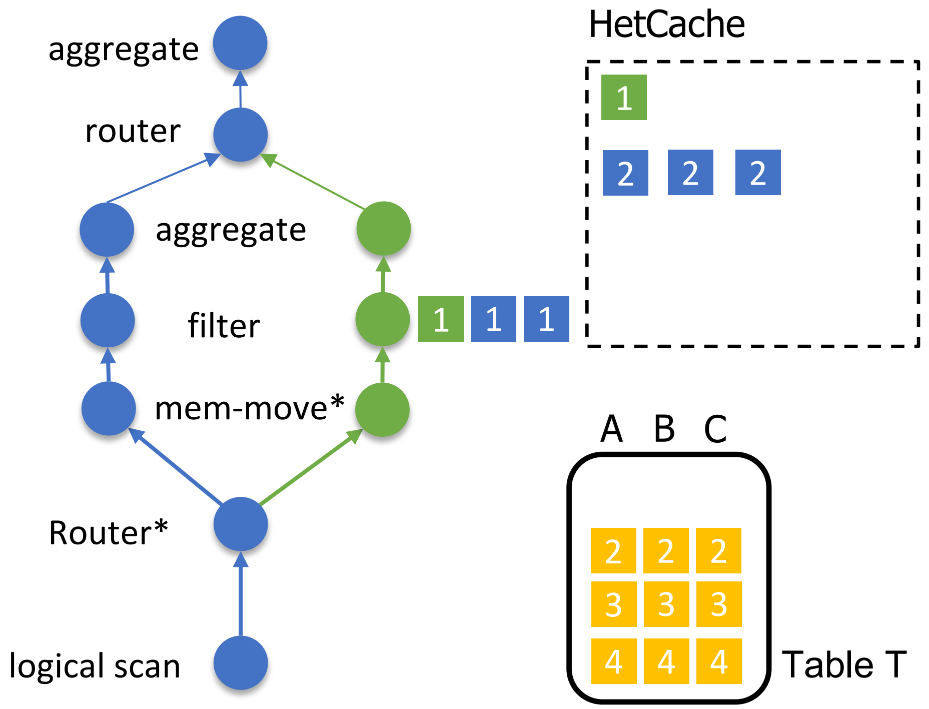


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

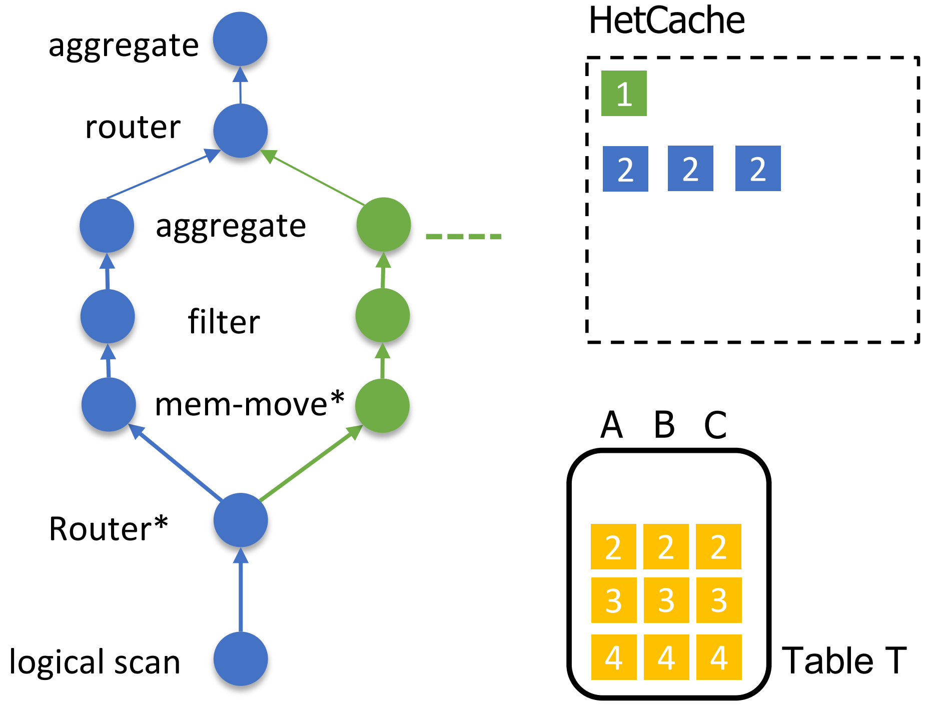


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

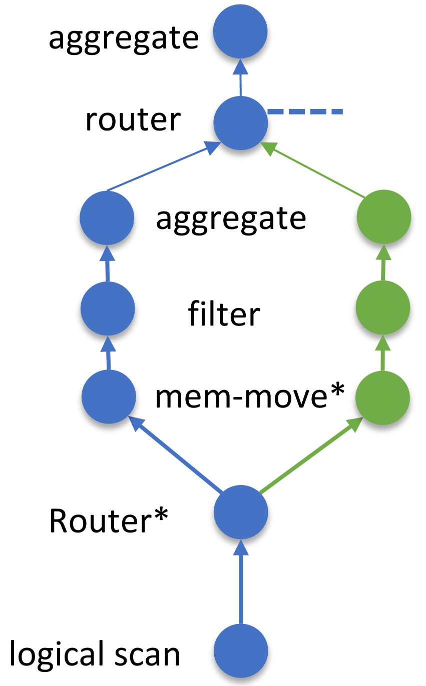


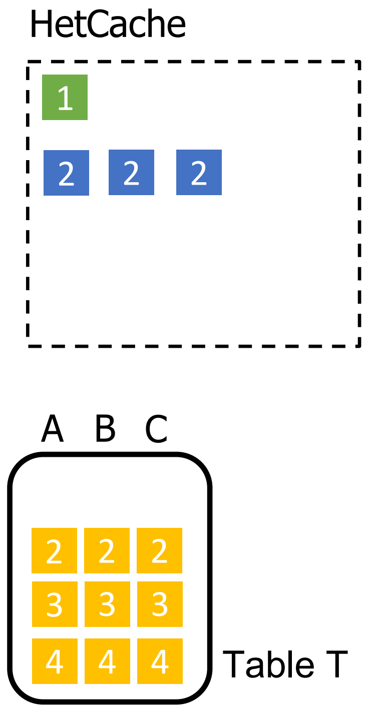
*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any



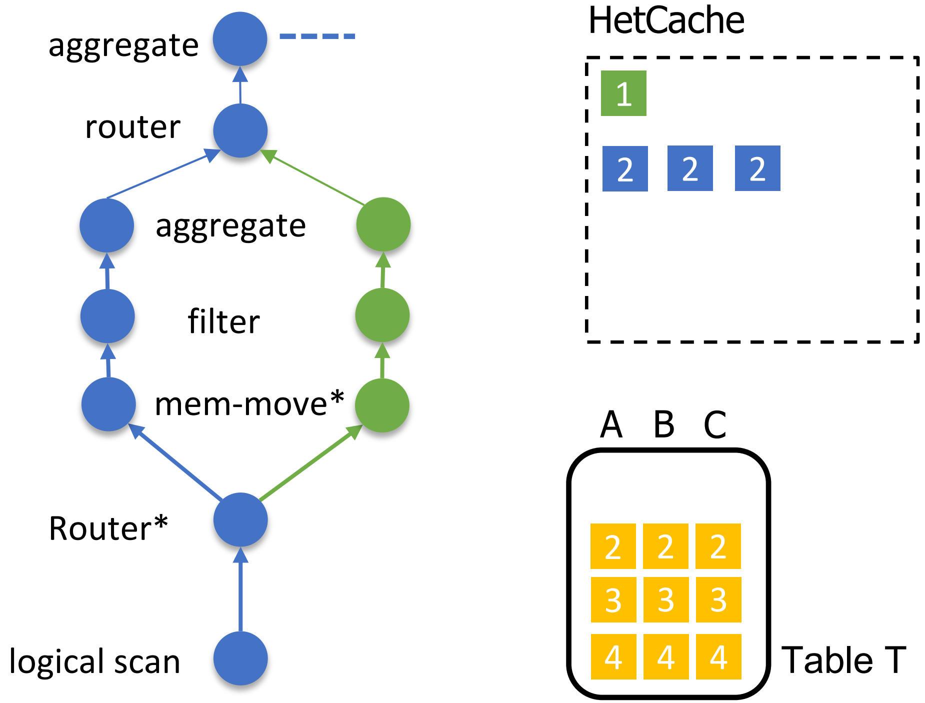


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

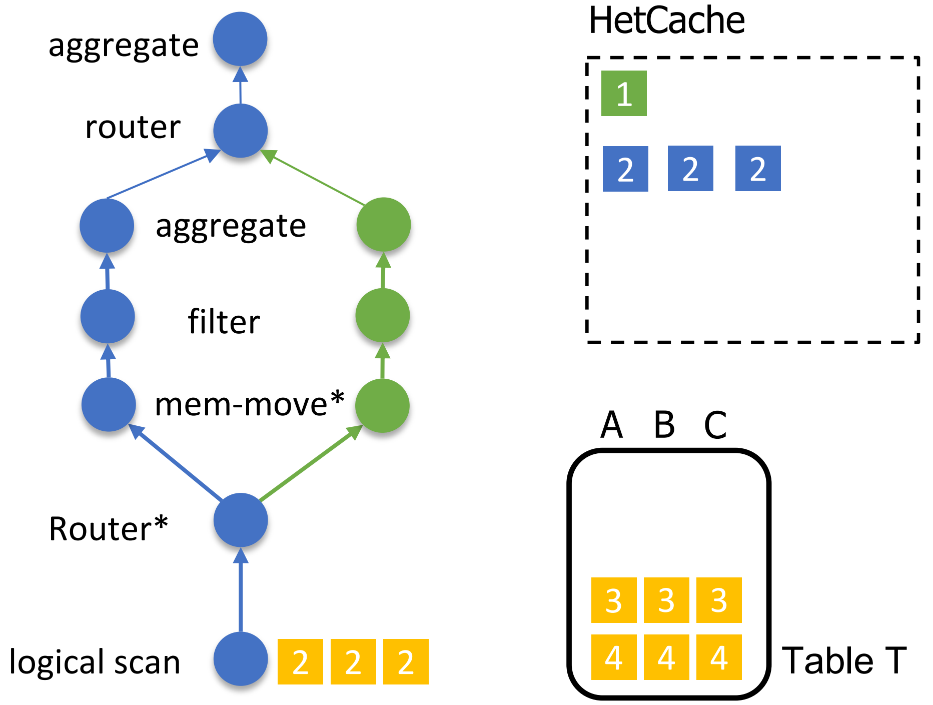


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

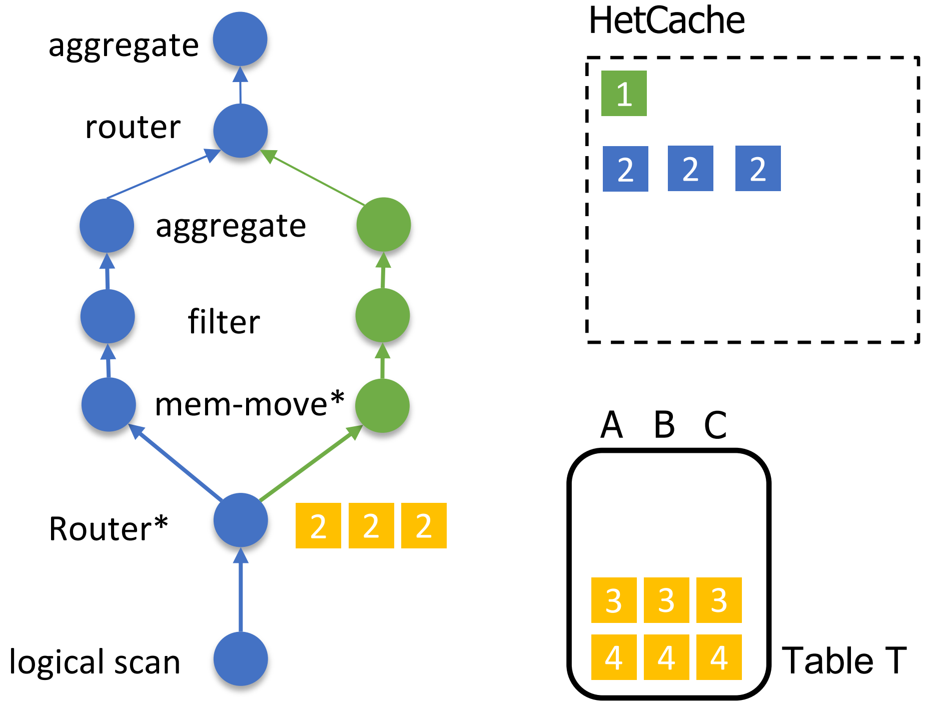


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

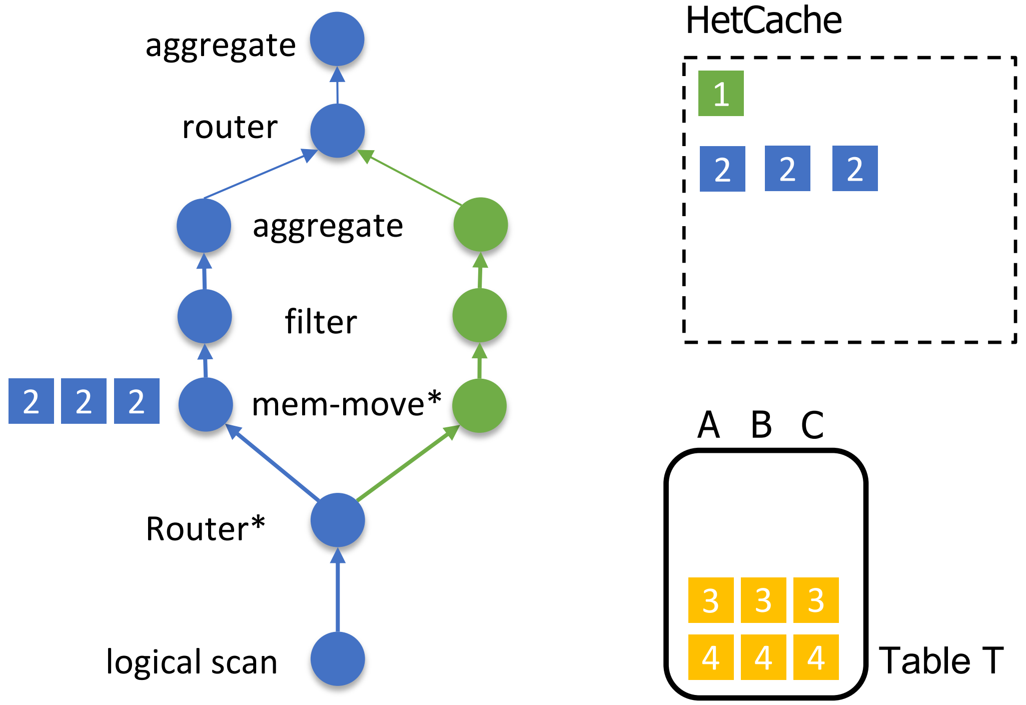


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

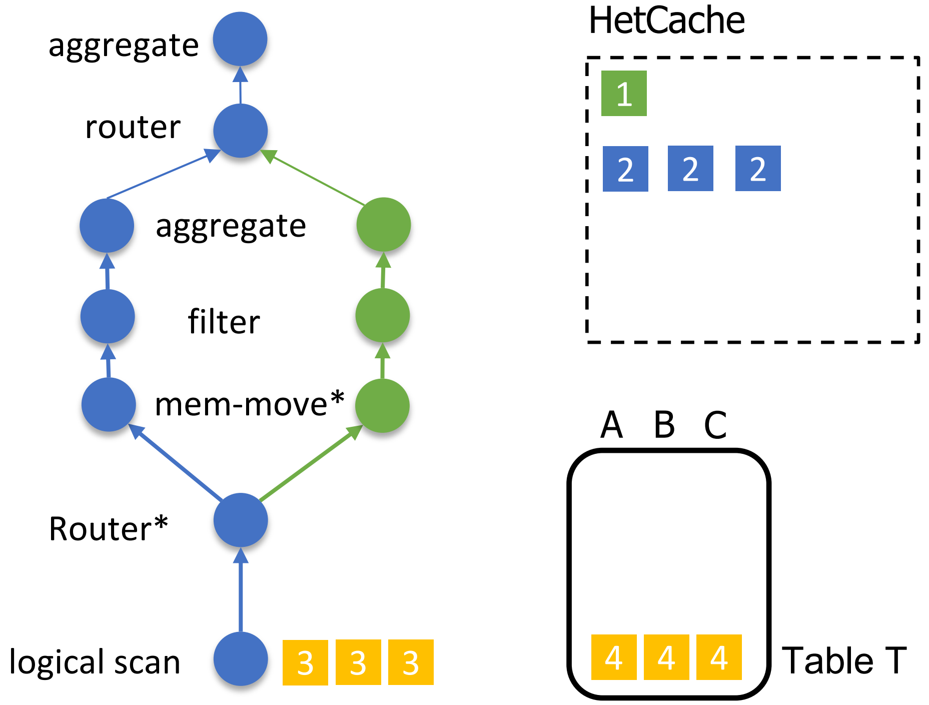


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

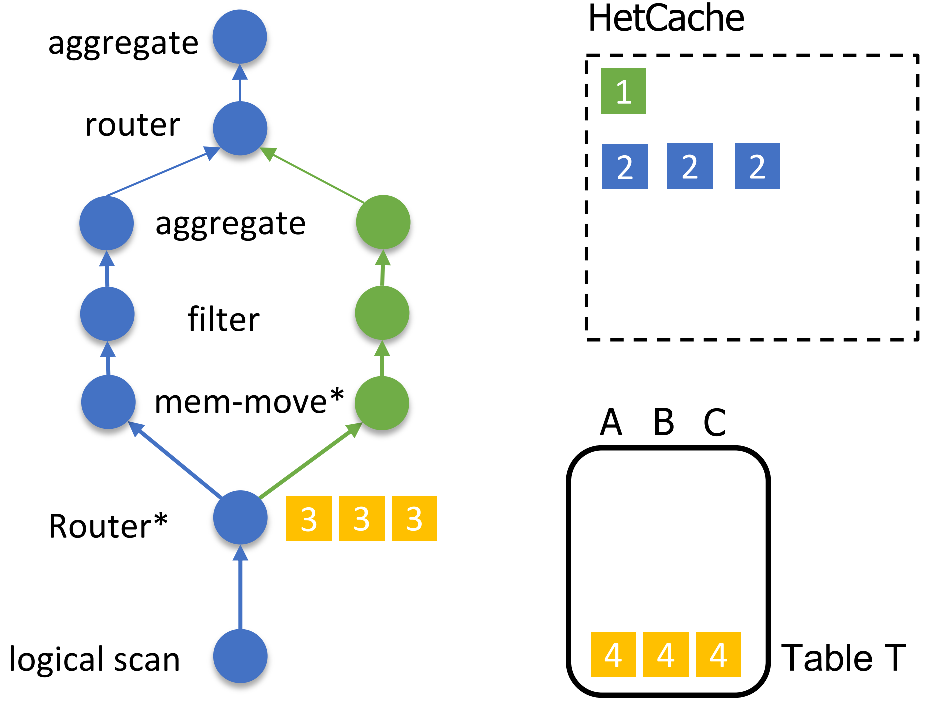


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

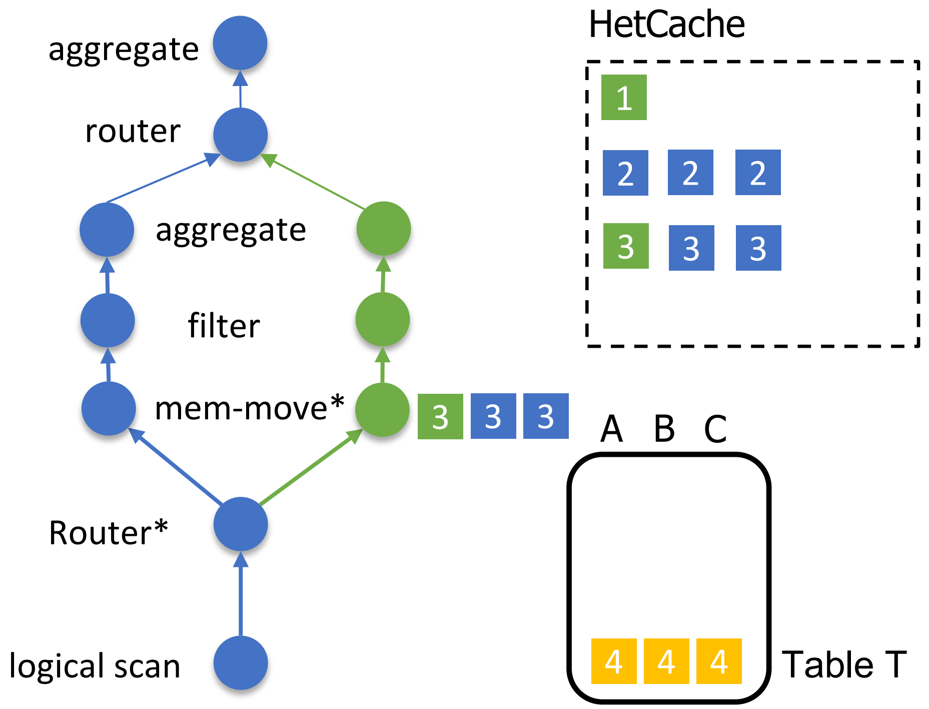


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

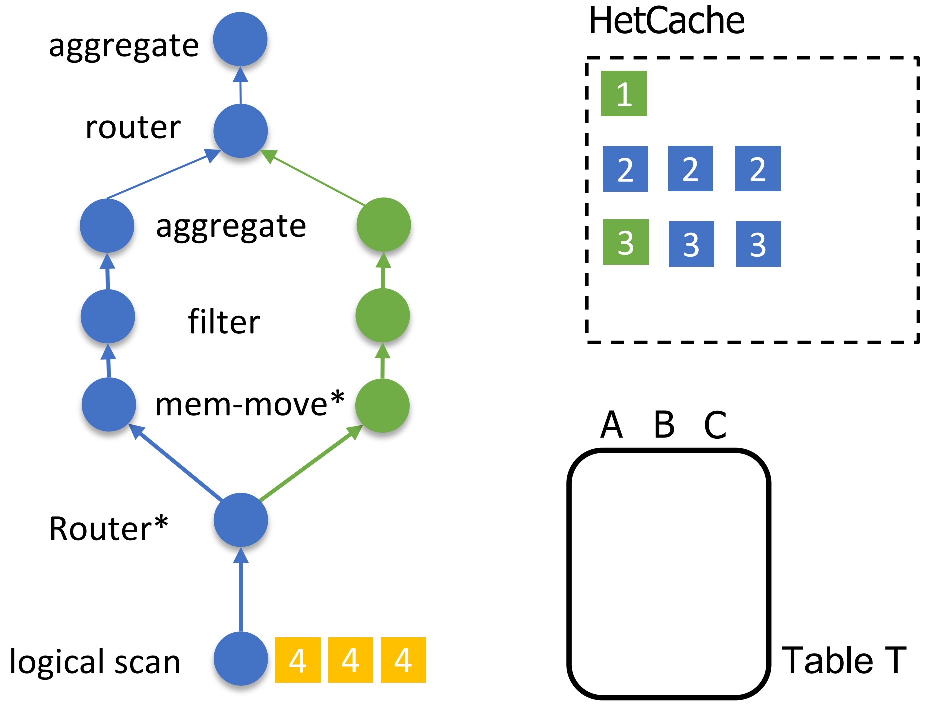


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

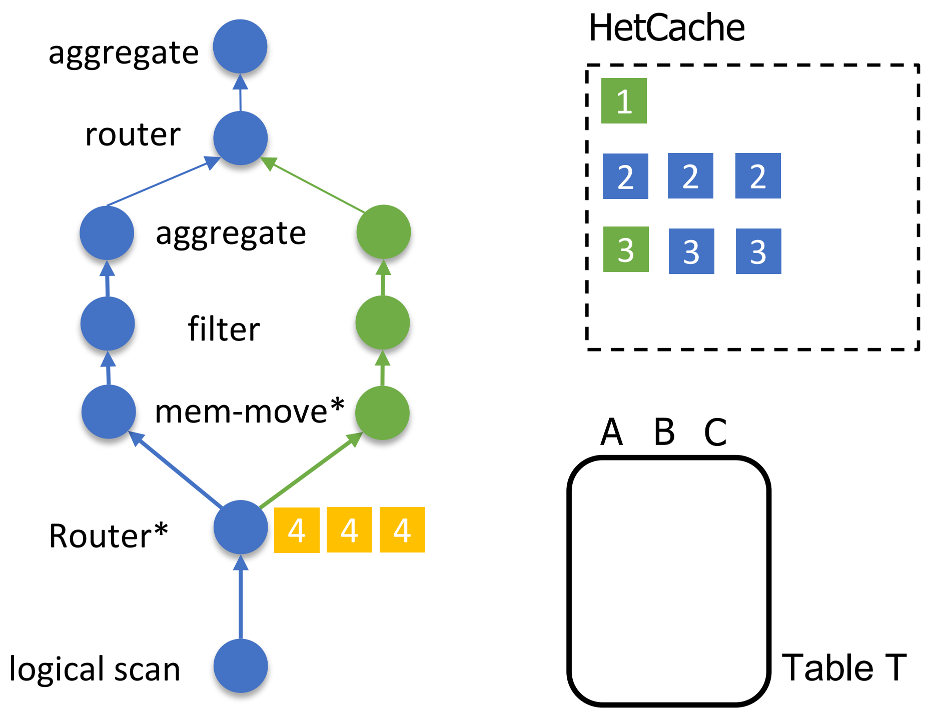


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any

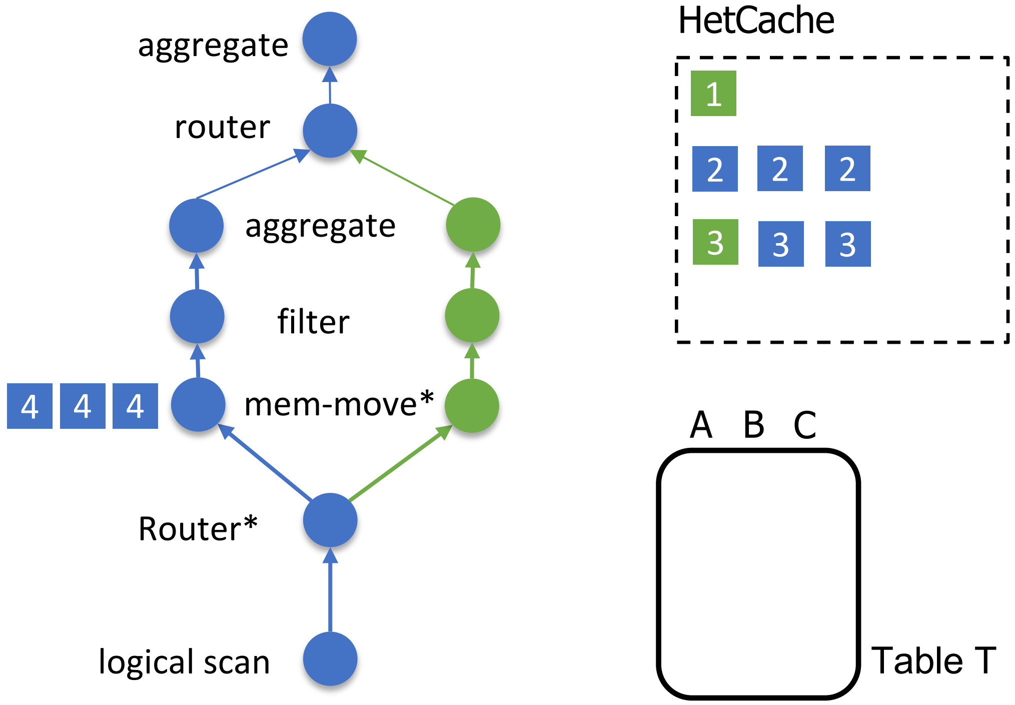


*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

Transfer Path Depends on Workload and Hardware

1. Logical scan emits page IDs
2. Route based on cache contents
3. mem-move consults HetCache on:
   - Locations of in-memory copies
   - Preferred location to cache, if any



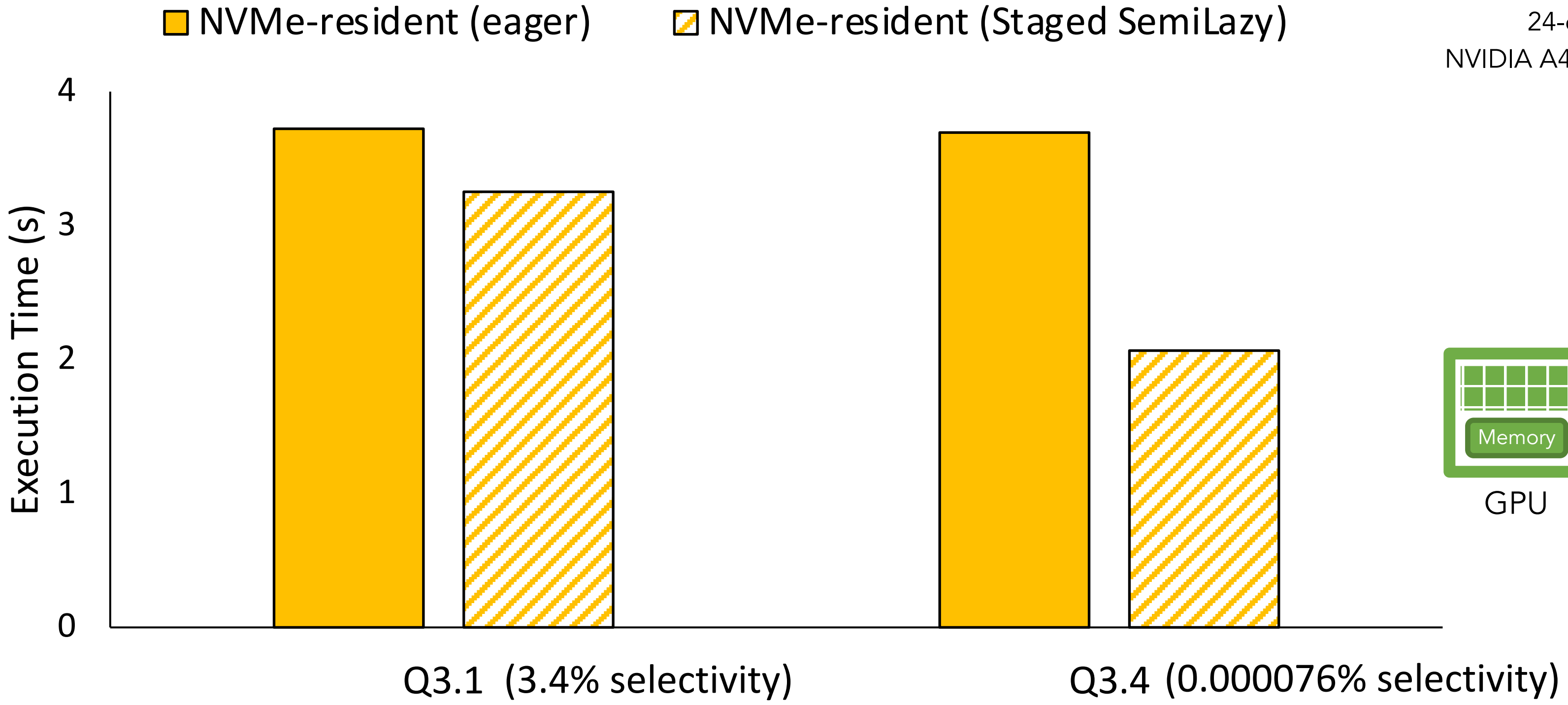
*Chrysogelos et. al [VLDB 2019]

Delay data transfers until first access

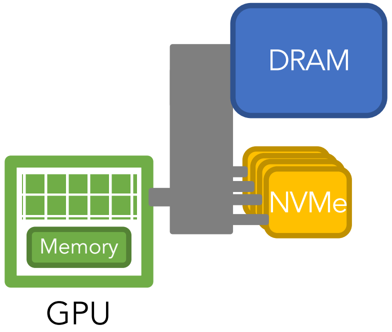
Experimental Setup

- Hardware
  - 24-core AMD EPYC 7413
  - NVIDIA A40, PCIe 4.0 x16
  - 12x PCIe 4.0 x4 NVMe, 7GB/s each
- Software
  - Proteus: Hybrid CPU-GPU analytical engine
- Benchmark
  - Star Schema Benchmark. (SF 1000)
  - ~96GB(Q1.x – 3.x) working set per query

Combining Transfer Paths for GPU

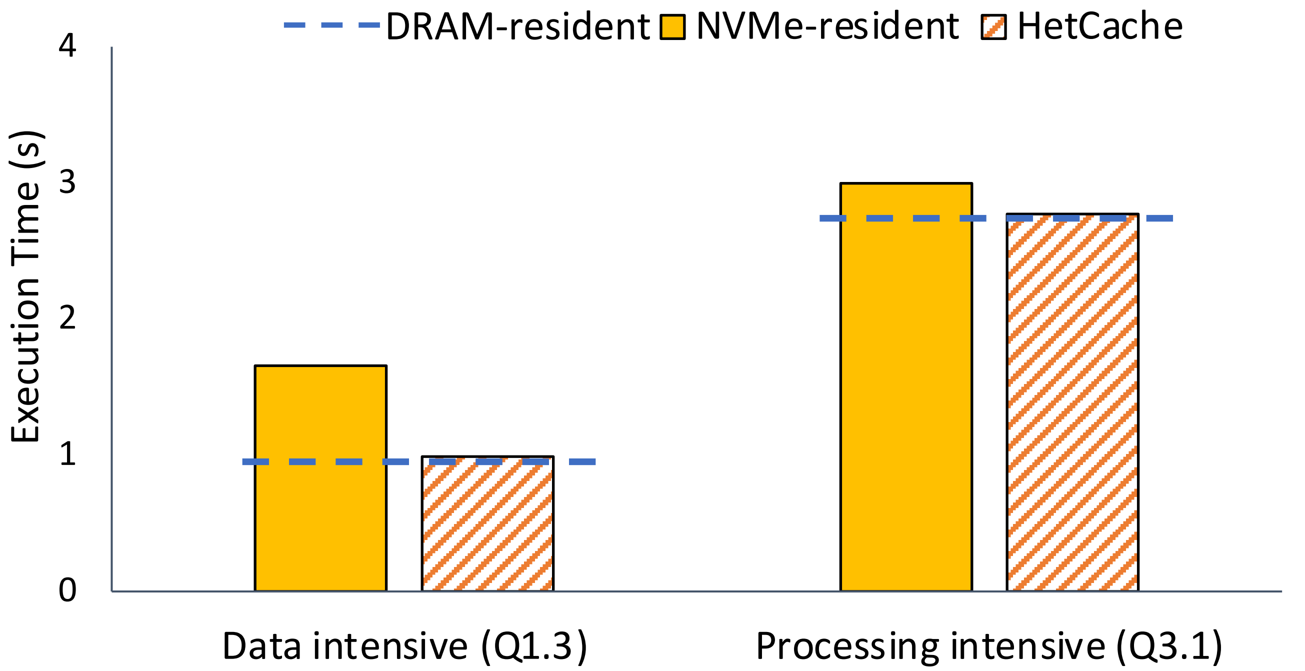


24-core AMD EPYC 7413
NVIDIA A40 48GB, PCIe 4.0 x16
8x (NVMe, PCIe 4 x4)
SSB SF=1000



Enabling sub-page accesses via DRAM staging => up to 45% faster

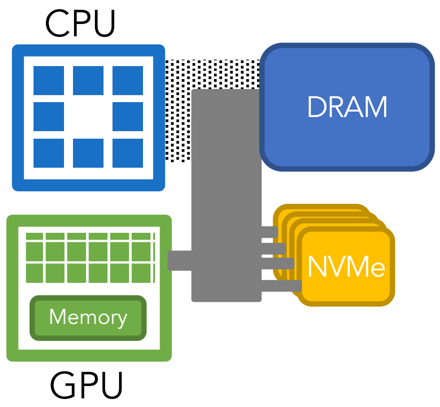
Memory Efficient CPU-GPU Execution



10 GB GPU memory
76 GB CPU memory

10GB GPU memory
0 GB CPU memory

24-core AMD EPYC 7413
NVIDIA A40 48GB, PCIe 4.0 x16
8x (NVMe, PCIe 4 x4)
96GB working set



25% - 100% less DRAM used for inputs

Storage BW is approaching DRAM BW

- (Near) in-memory performance on larger-than-DRAM datasets
  - Granularity and processing throughput-aware data placement

- Efficient interconnect use for NVMe-GPU transfers
  - Stage selectively accessed data in DRAM for GPUs

- Storage systems must be hardware & workload aware

Thank You!