StreamBox-HBM

Stream Analytics on High Bandwidth Hybrid Memory

Hongyu Miao, *Purdue ECE;* Myeongjae Jeon, *UNIST*; Gennady Pekhimenko, *UToronto;*

Kathryn S. McKinley, *Google;* Felix Xiaozhu Lin, *Purdue ECE*

http://xsel.rocks/p/streambox

PURDUE UNIVERSITY

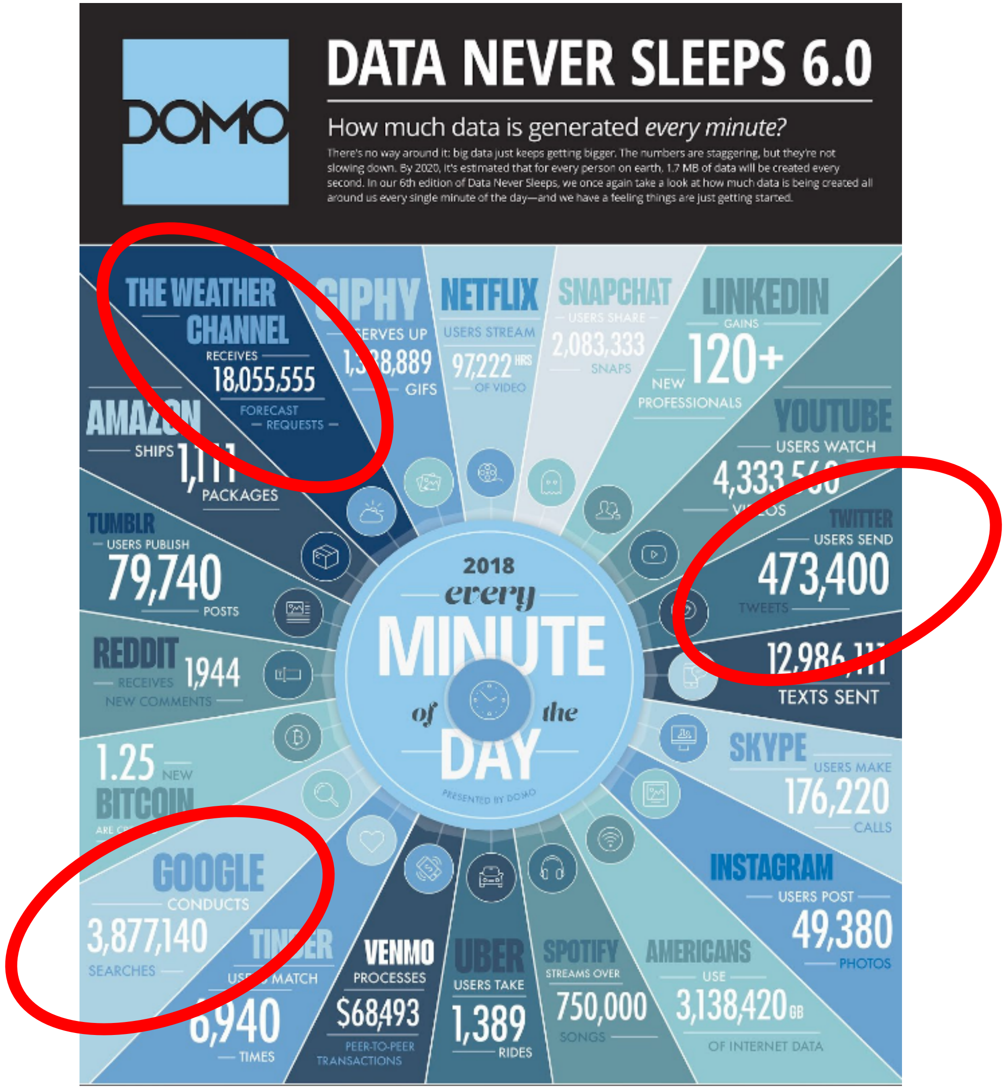
UNIST ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

UNIVERSITY OF TORONTO

Google

NSF

Timely processing of streaming data



There's no way around it: big data just keeps getting bigger. The numbers are staggering, but they're not slowing down. By 2020, it's estimated that for every person on earth, 1.7 MB of data will be created every second. In our 6th edition of Data Never Sleeps, we once again take a look at how much data is being created all around us every single minute of the day—and we have a feeling things are just getting started.

DOMO
DATA NEVER SLEEPS 6.0
How much data is generated *every minute?*
THE WEATHER CHANNEL RECEIVES 18,055,555 FORECAST REQUESTS
GIPHY SERVES UP 1,388,889 GIFS
NETFLIX USERS STREAM 97,222 HRS OF VIDEO
SNAPCHAT USERS SHARE 2,083,333 SNAPS
LINKEDIN GAINS 120+ NEW PROFESSIONALS
AMAZON SHIPS 1,111 PACKAGES
YOUTUBE USERS WATCH 4,333,560 VIDEOS
TUMBLR USERS PUBLISH 79,740 POSTS
TWITTER USERS SEND 473,400 TWEETS
12,986,111 TEXTS SENT
REDDIT RECEIVES 1944 NEW COMMENTS
2018 every MINUTE of the DAY
PRESENTED BY DOMO
1.25 NEW BITCOIN
SKYPE USERS MAKE 176,220 CALLS
INSTAGRAM USERS POST 49,380 PHOTOS
GOOGLE CONDUCTS 3,877,140 SEARCHES
TINDER USERS MATCH 6,940 TIMES
VENMO PROCESSES $68,493 PEER-TO-PEER TRANSACTIONS
UBER USERS TAKE 1,389 RIDES
SPOTIFY STREAMS OVER 750,000 SONGS
AMERICANS USE 3,138,420 GB OF INTERNET DATA

On 100+ GB memory

High Throughput & Low Latency!

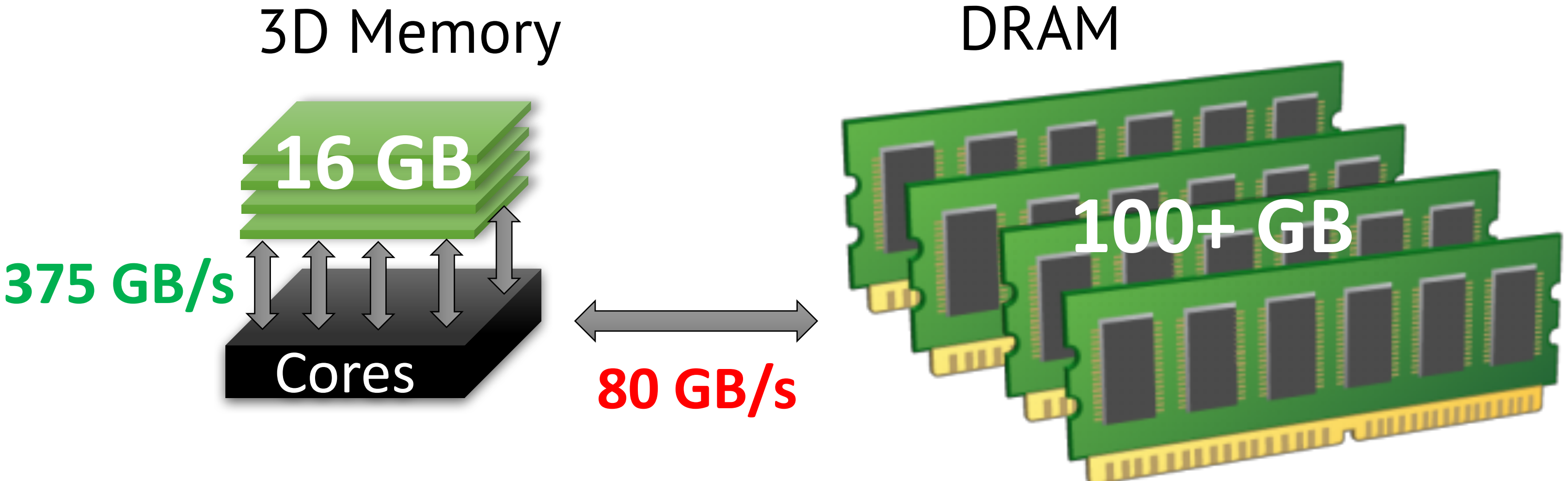
Hybrid Memory: 3D Memory + DRAM

DRAM

- Larger capacity, but lower bandwidth

3D Memory

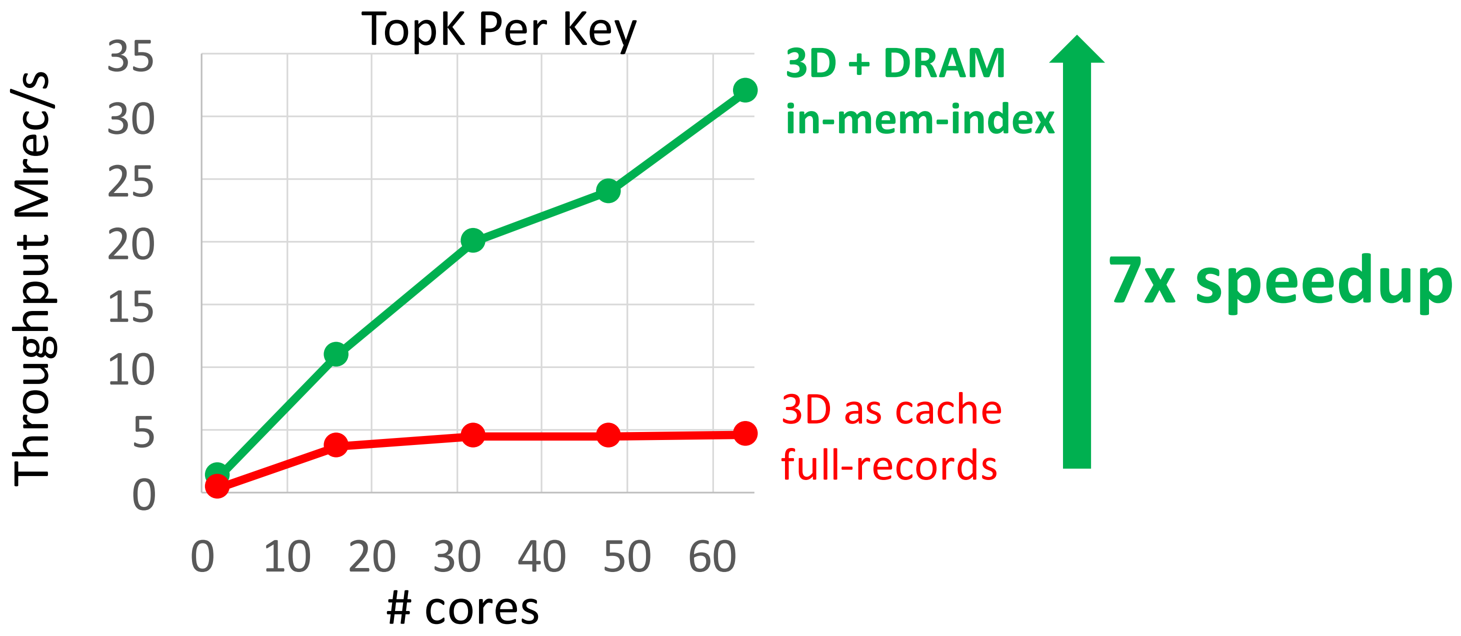
- Higher bandwidth, but smaller capacity
- NO latency benefit (Unlike cache: SRAM+DRAM)
- Same as DRAM without high parallelism or sequential access
- As cache of DRAM? → Poor performance…



Can hybrid mem speed up stream analytics?

Yes!   StreamBox-HBM

- The first stream engine optimized for 3D memory + DRAM on real hardware
- Achieves the best reported throughput on single node (win-avg:110MRec/s)
- Speeds up stream analytics by 7x



7x speedup

Challenges

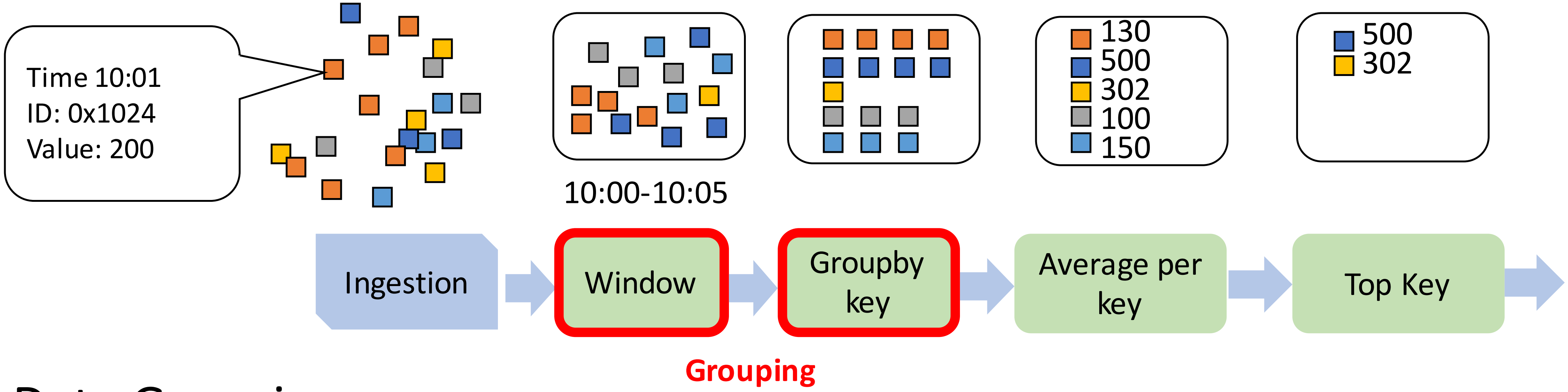
1. Hash Grouping performs poorly on 3D memory

2. 3D memory is capacity limited

3. How to dynamically map streaming data to hybrid mem?

Challenge 1: Hash Grouping performs poorly on 3D memory

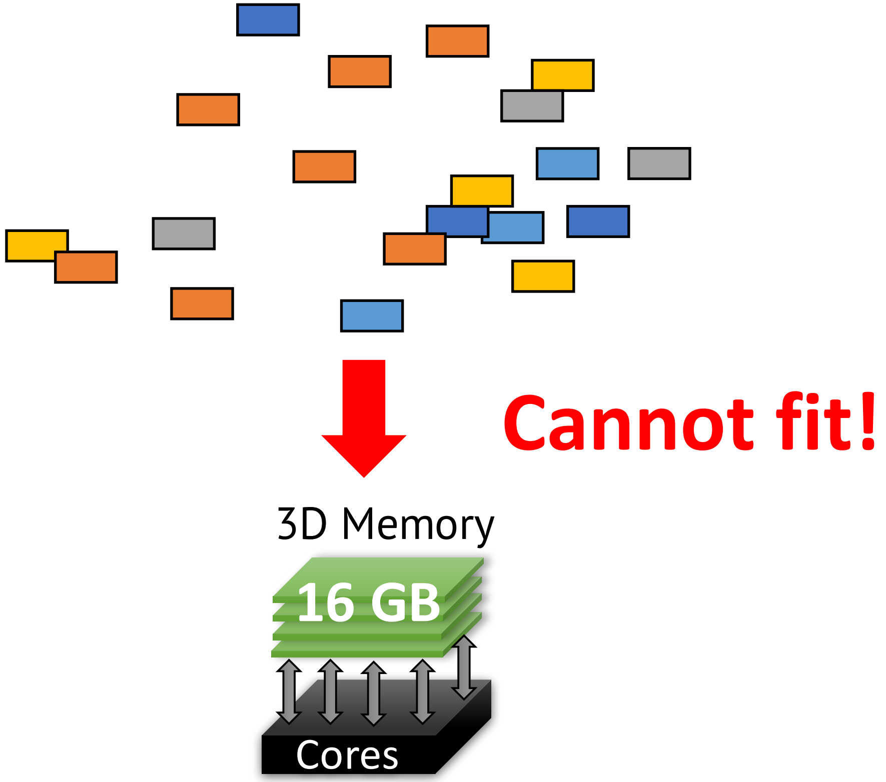
- Operators: computations consume/produce streams
- Pipeline: a graph of streaming operators



- Data Grouping
  - A set of **very common** and **expensive** operators that reorganize records
  - Hash with random access in existing engines → Performs poorly on 3D memory…

Challenge 2: 3D memory is capacity limited

- Streaming data
  - High data volume (100+ GB)
- 3D Memory
  - Capacity limited (~ 16 GB)



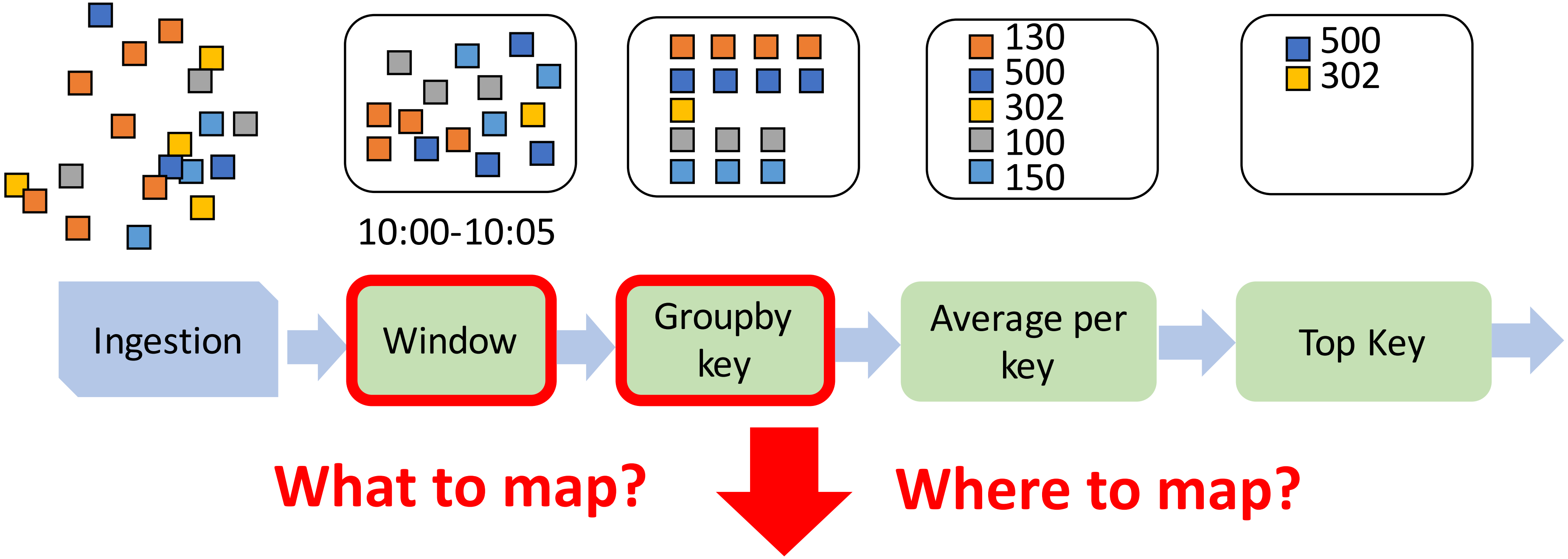
- 3D memory is NOT large enough to hold all streaming data….

Challenge 3: managing two types of memory

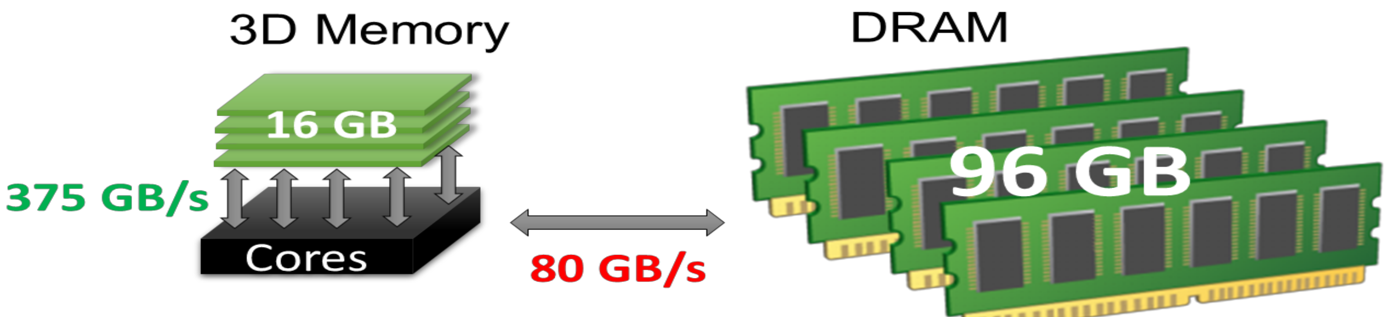
- How to **dynamically** map data/operators to two types of memory?

Unbounded data

Various queries



Hybrid memory: benefit & limitation



StreamBox-HBM Solutions

1. Hash grouping performs poorly on 3D memory

- → Solution 1: Use high parallel Sort for grouping

2. 3D memory is capacity limited

- → Solution 2: Only use 3D memory to store in-memory indexes

3. How to manage two types of memory?

- → Solution 3: Balance two limited resource with a single knob

Solution 1: Parallel Sort for Grouping

Known duals of Grouping: Hash vs. Sort

- DRAM: Hash is the best [VLDB’09, VLDB’13, SIGMOD’15]
- Contribution: 3D memory reverses the debate. Sort outperforms Hash.

Sort is **worse** than Hash on algorithmic complexity

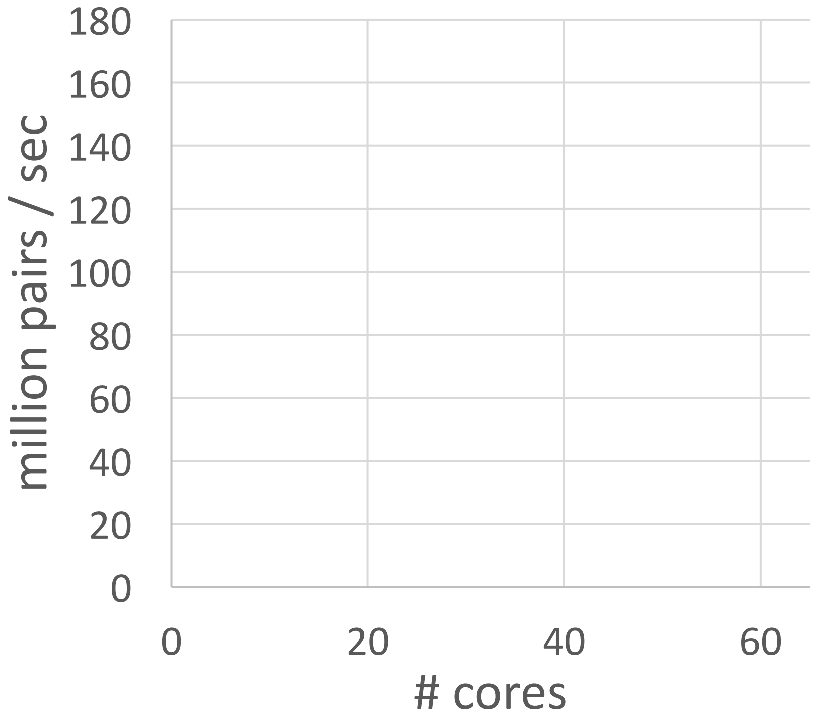
- O(NlogN) vs. O(N)

Yet, Sort **outperforms** Hash after we exploit all:

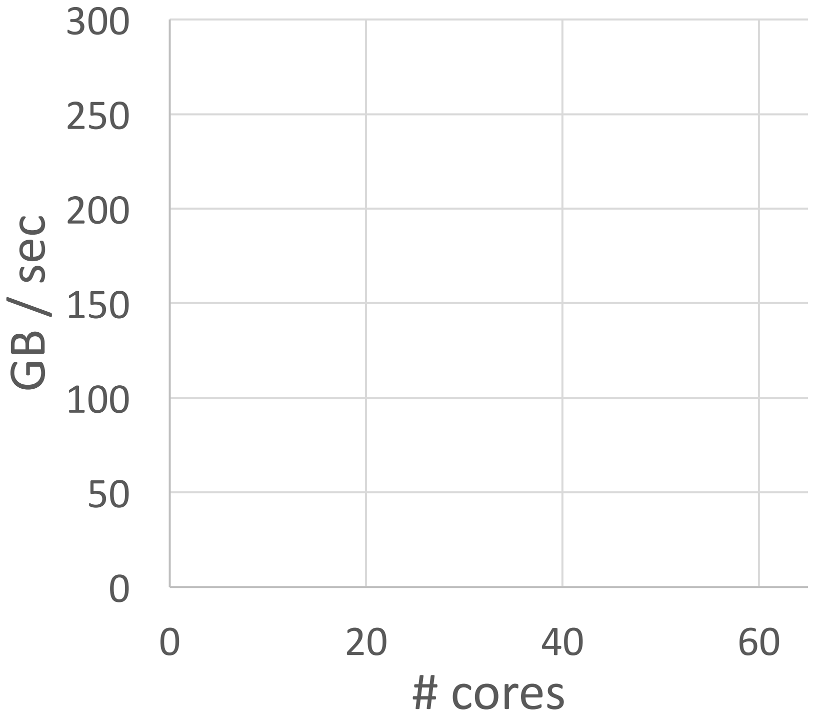
- Abundant memory bandwidth
- High task parallelism
- Wide SIMD (avx512)

[VLDB’09] Sort vs. hash revisited: Fast join implementation on modern multi-core cpus. [VLDB’13] Multi-core, main-memory joins: Sort vs. hash revisited
[SIGMOD’15] Rethinking simd vectorization for in-memory databases

Solution 1: Parallel Sort for Grouping



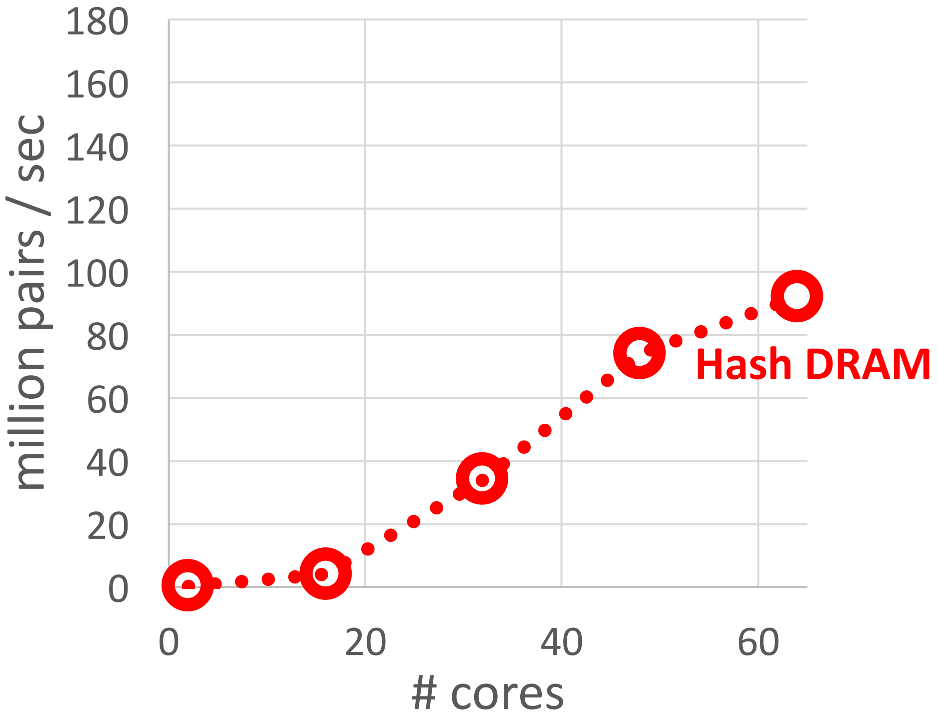
Throughput



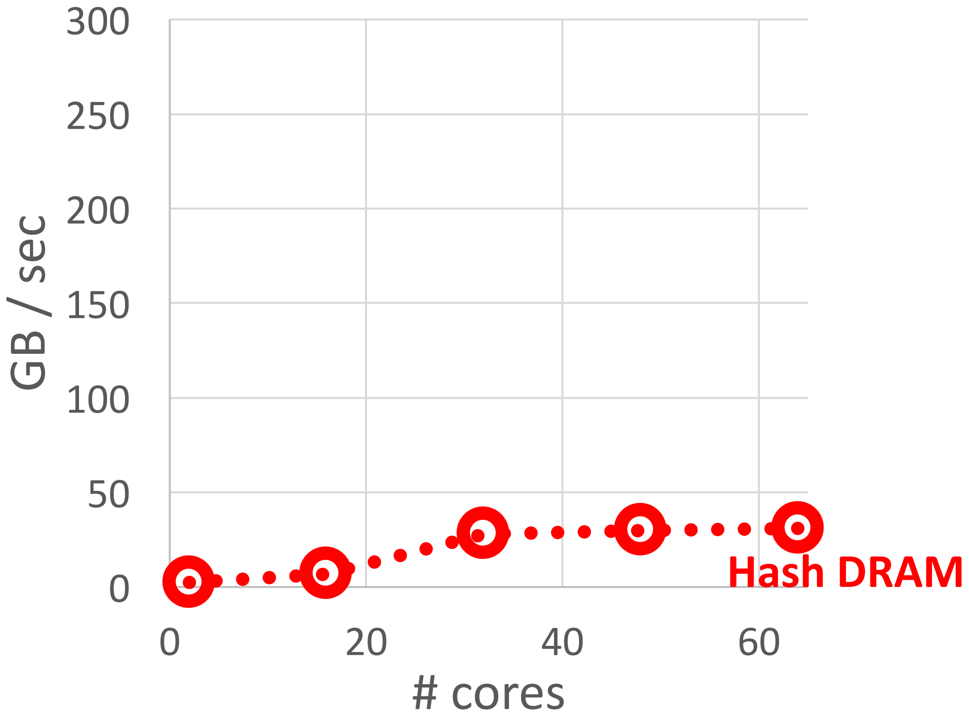
Mem bandwidth

Sort outperforms Hash on 3D memory

Solution 1: Parallel Sort for Grouping



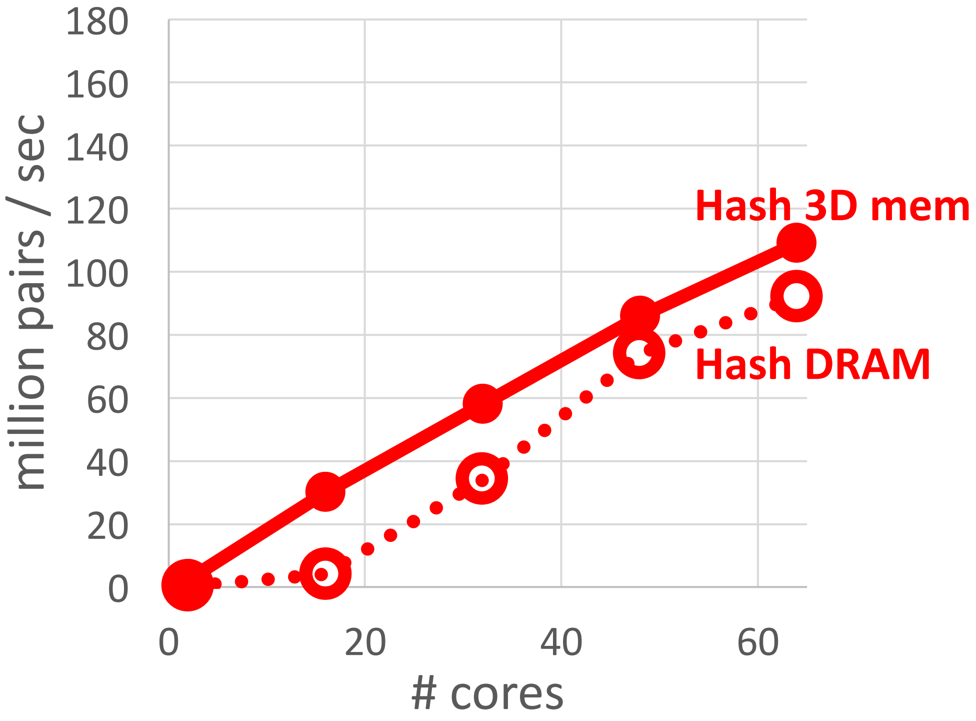
Throughput



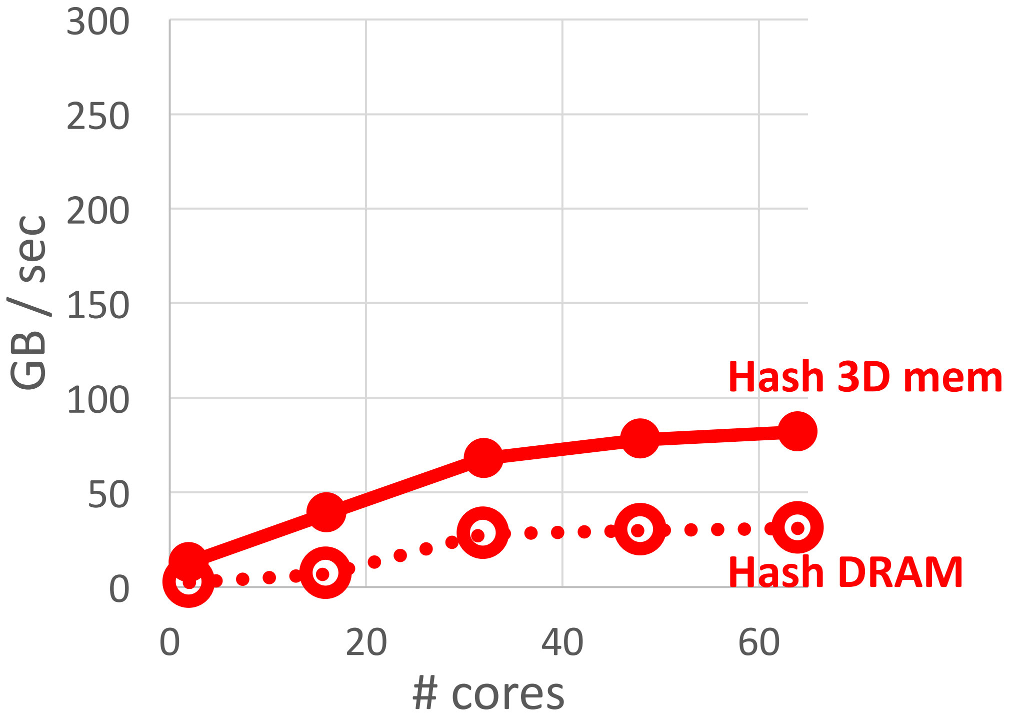
Mem bandwidth

Sort outperforms Hash on 3D memory

Solution 1: Parallel Sort for Grouping



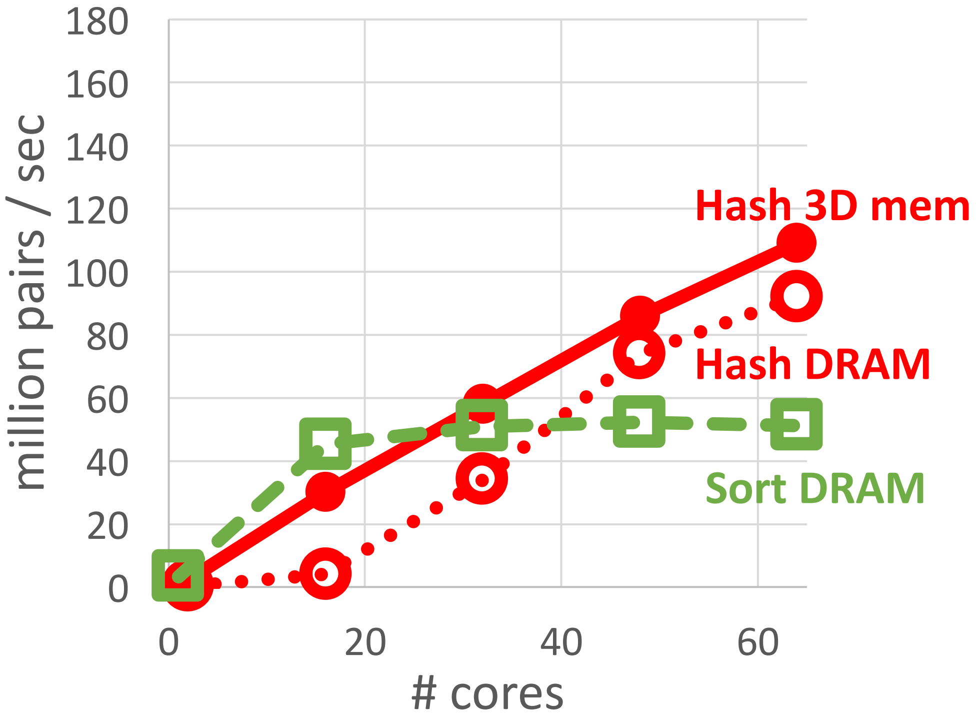
Throughput



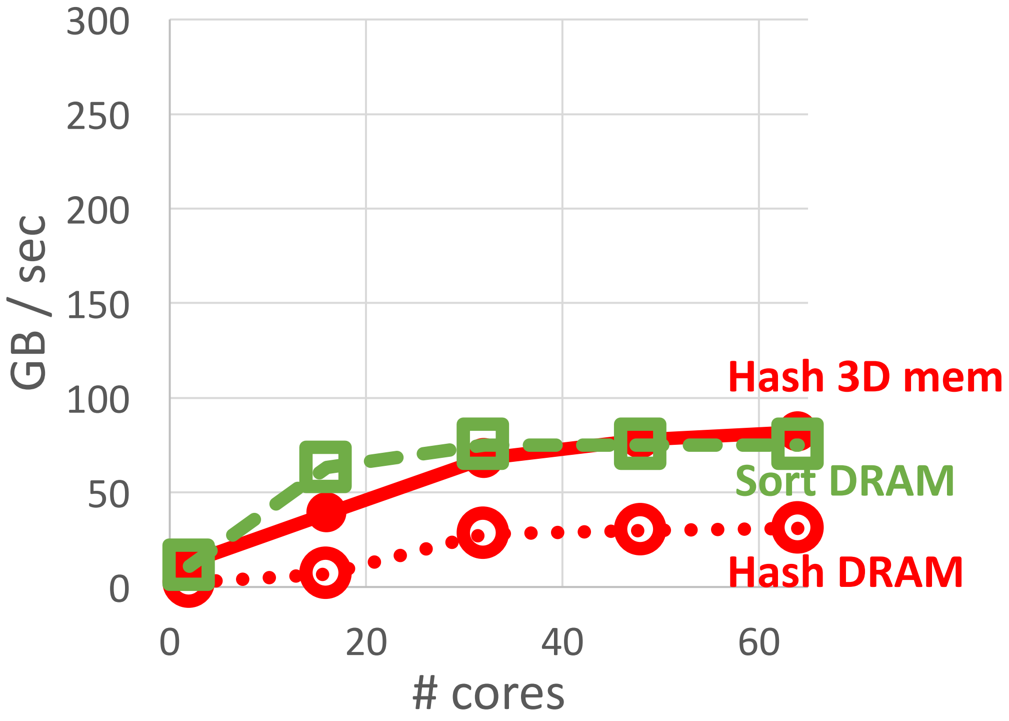
Mem bandwidth

Sort outperforms Hash on 3D memory

Solution 1: Parallel Sort for Grouping



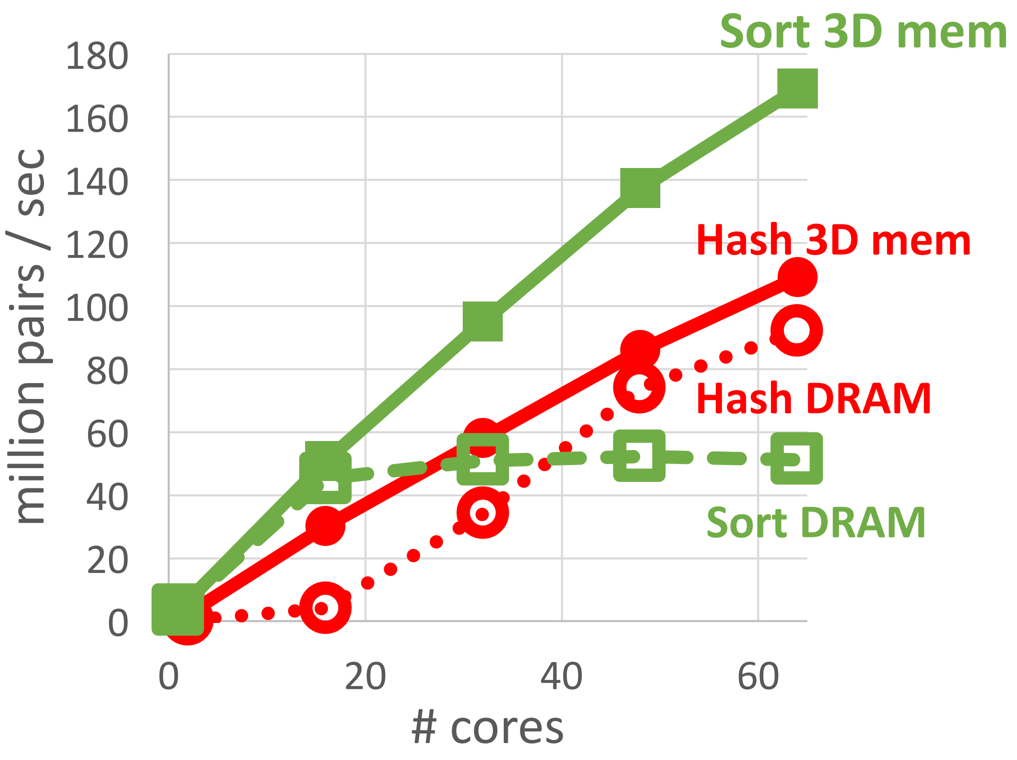
Throughput



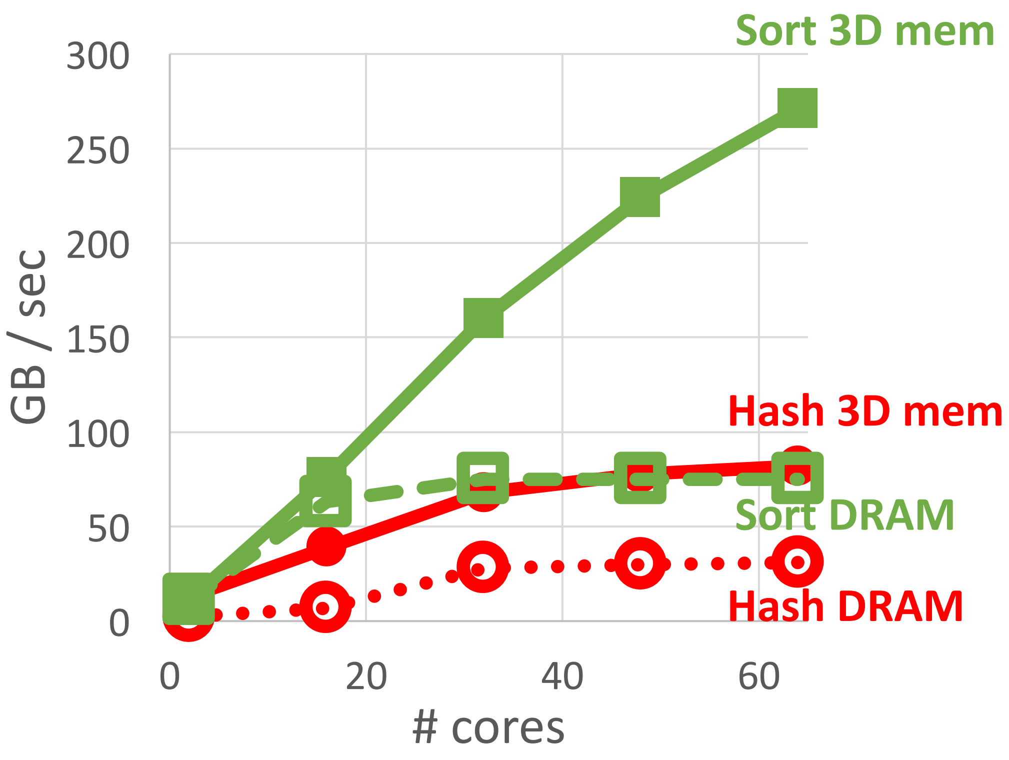
Mem bandwidth

Sort outperforms Hash on 3D memory

Solution 1: Parallel Sort for Grouping



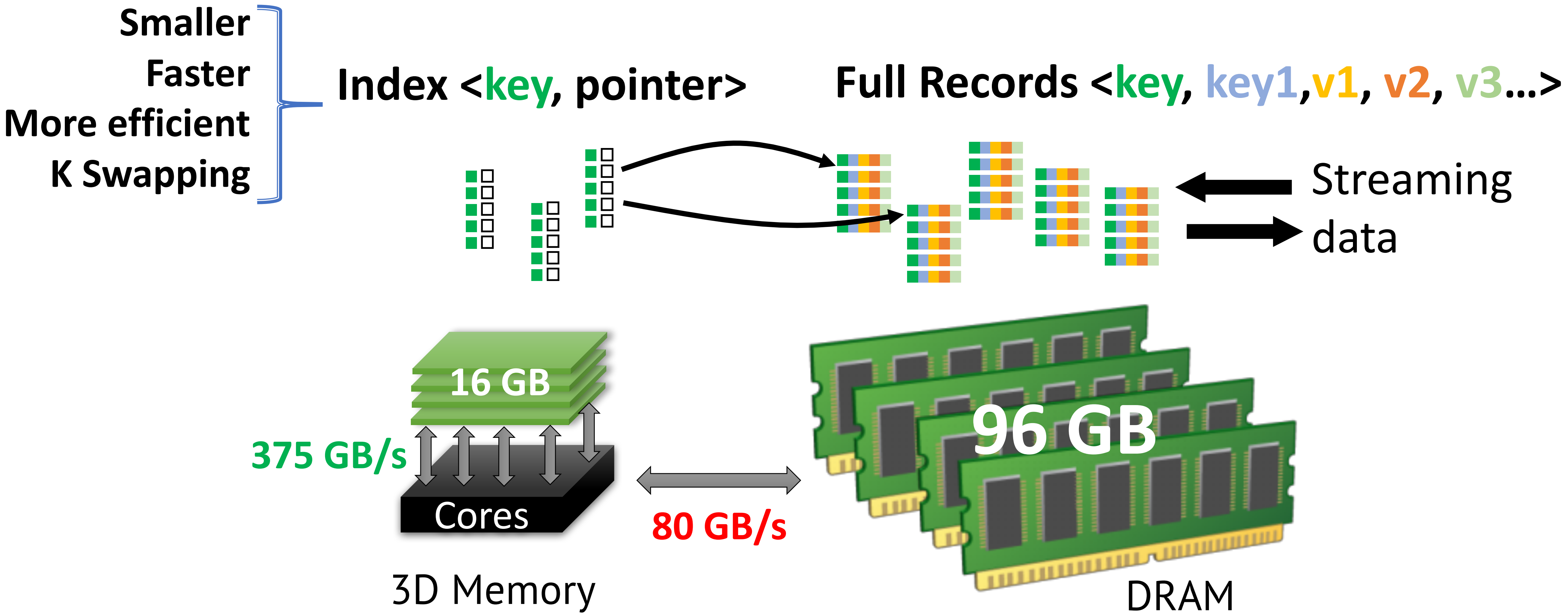
Throughput



Mem bandwidth

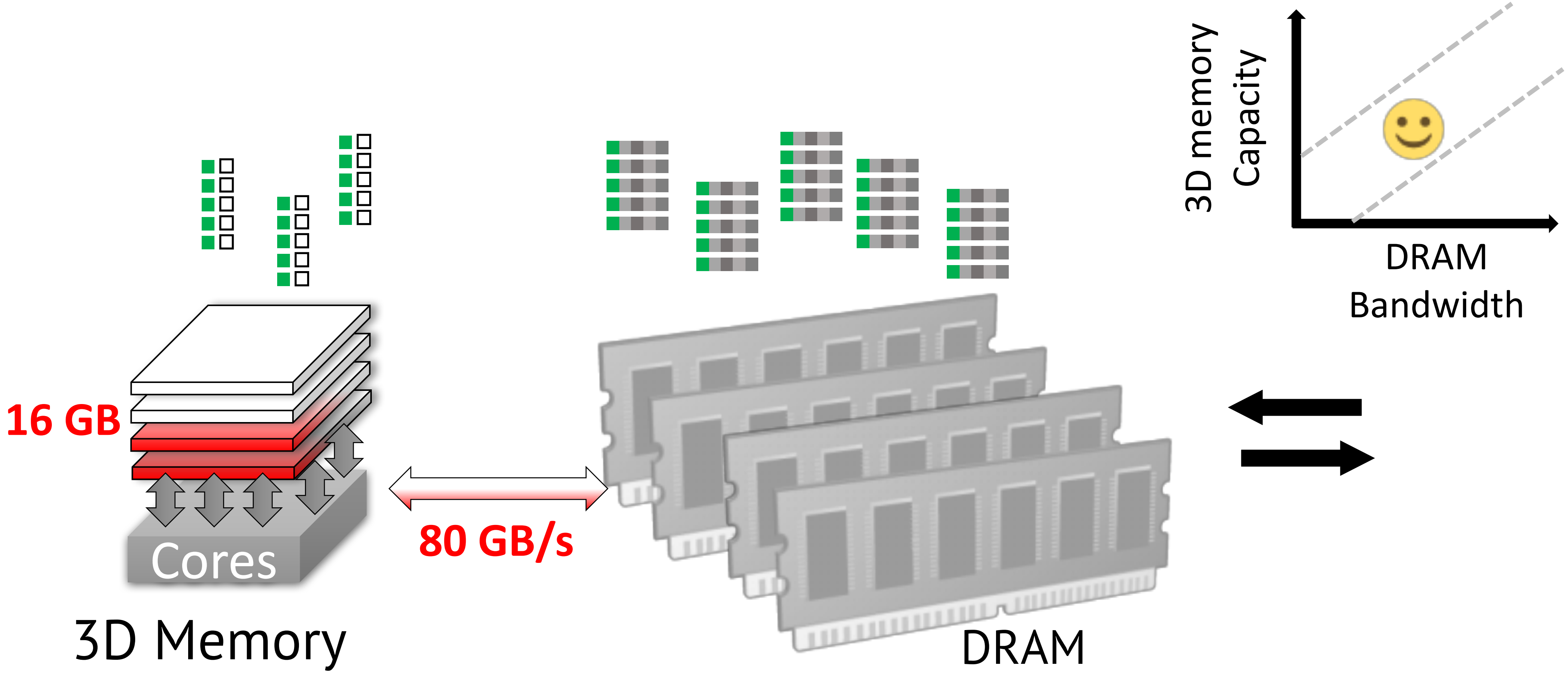
Sort outperforms Hash on 3D memory

Solution 2: Only use 3D memory for in-memory index

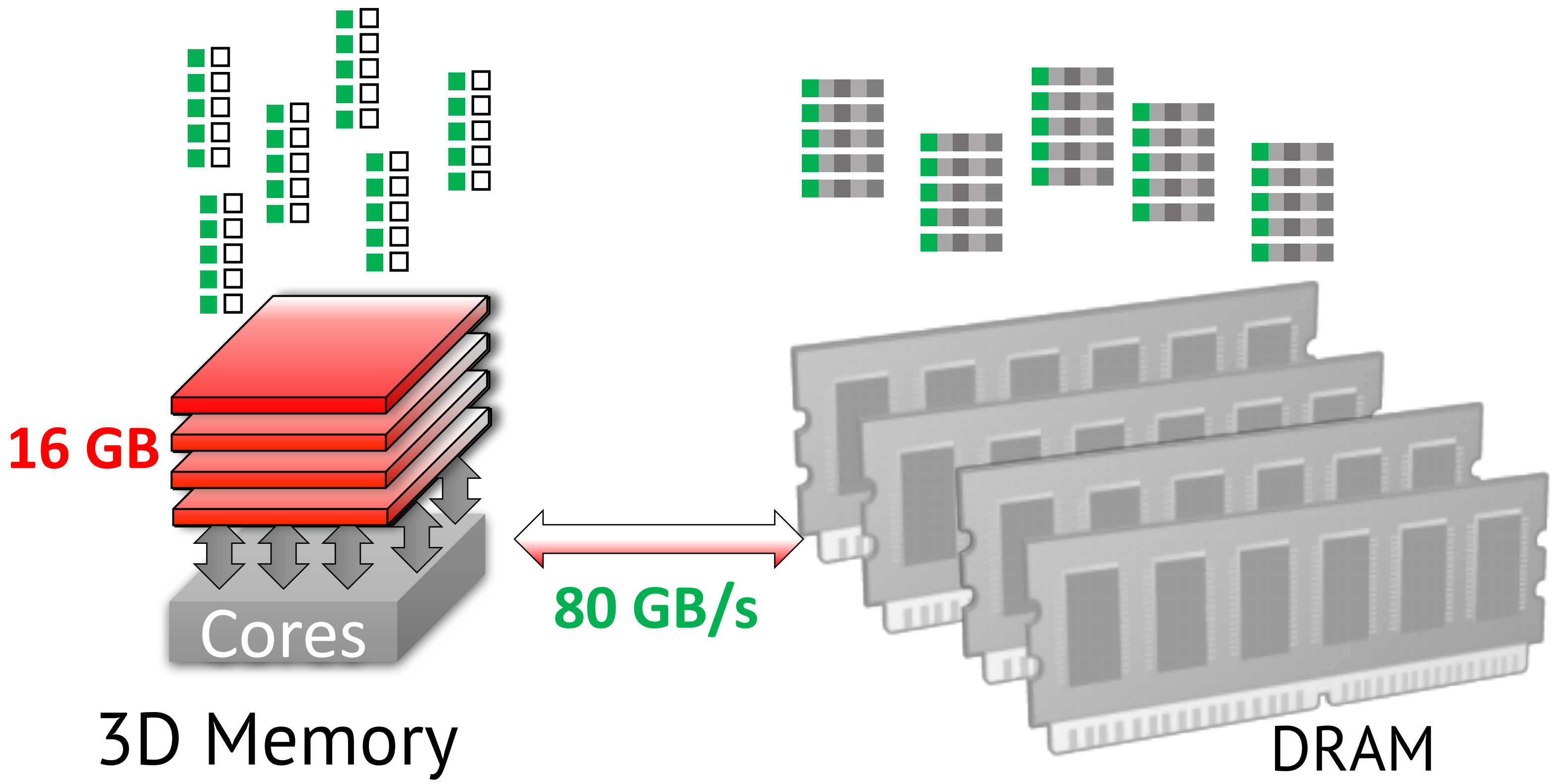


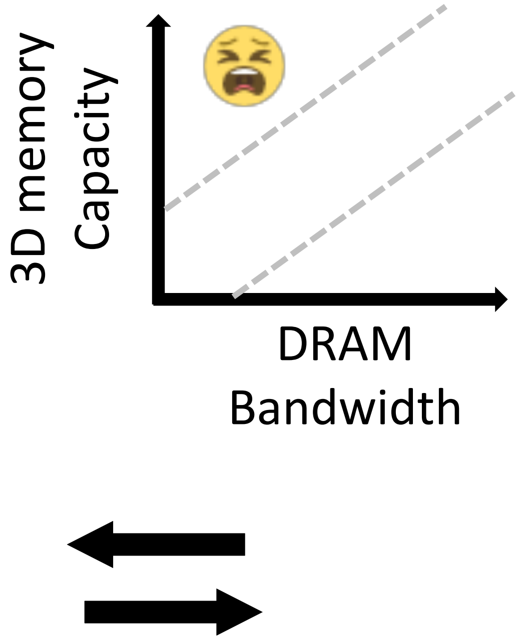
Minimize the use of precious 3D mem's capacity while exploit high bandwidth

Solution 3: balance two limited resources



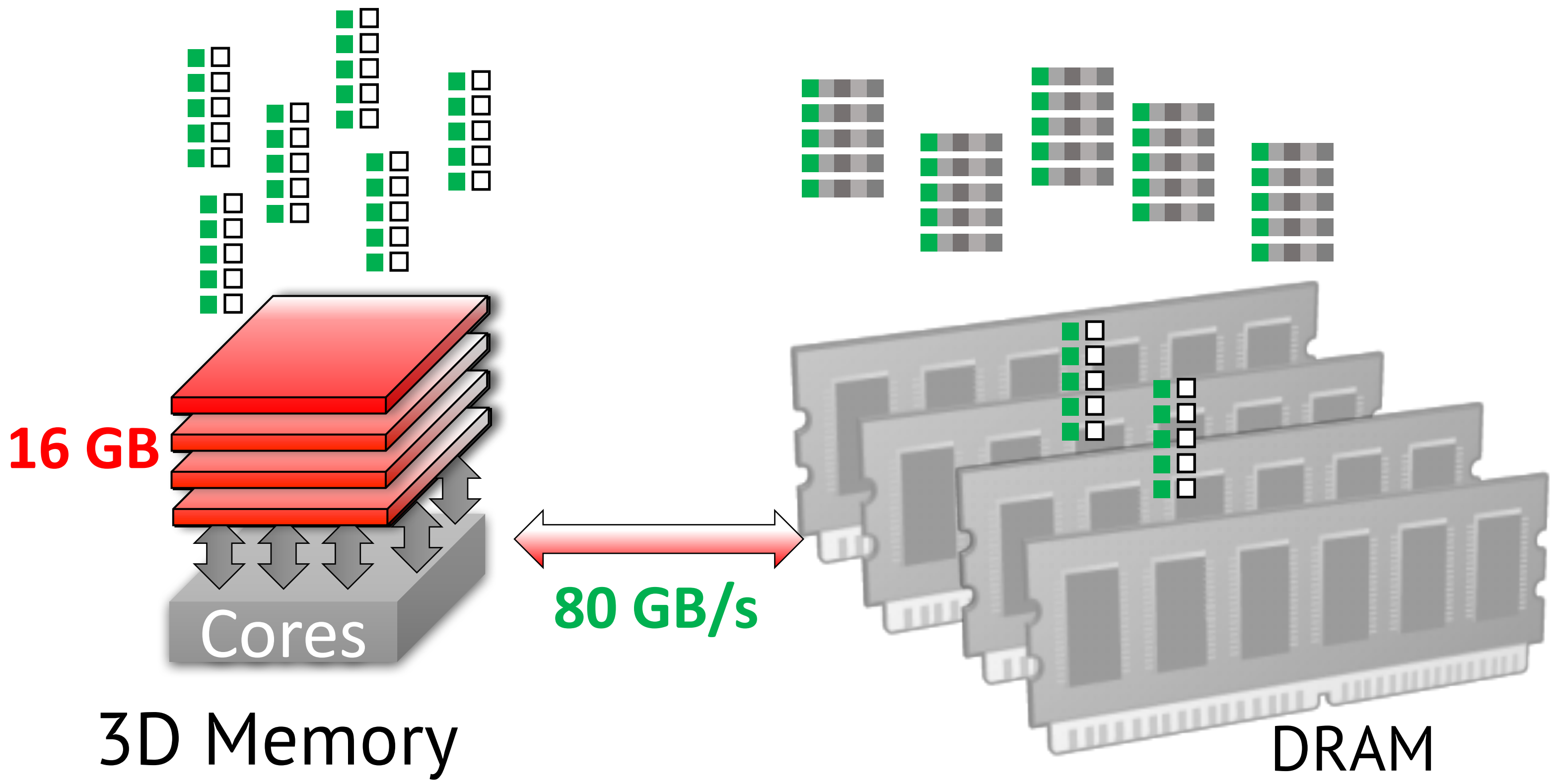
Solution 3: balance two limited resources

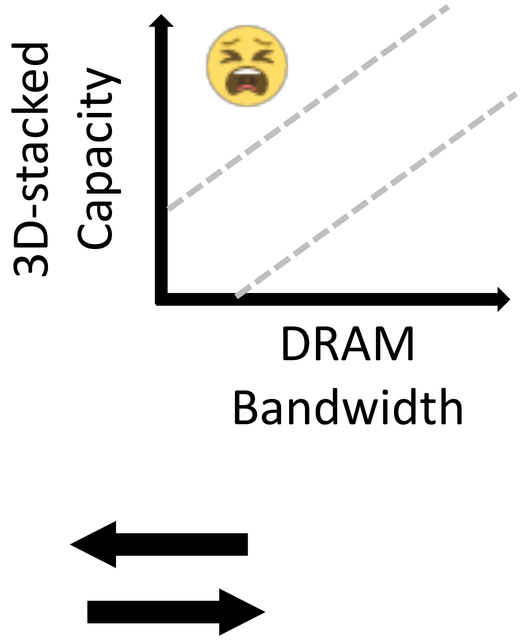




High pressure on 3D Memory capacity

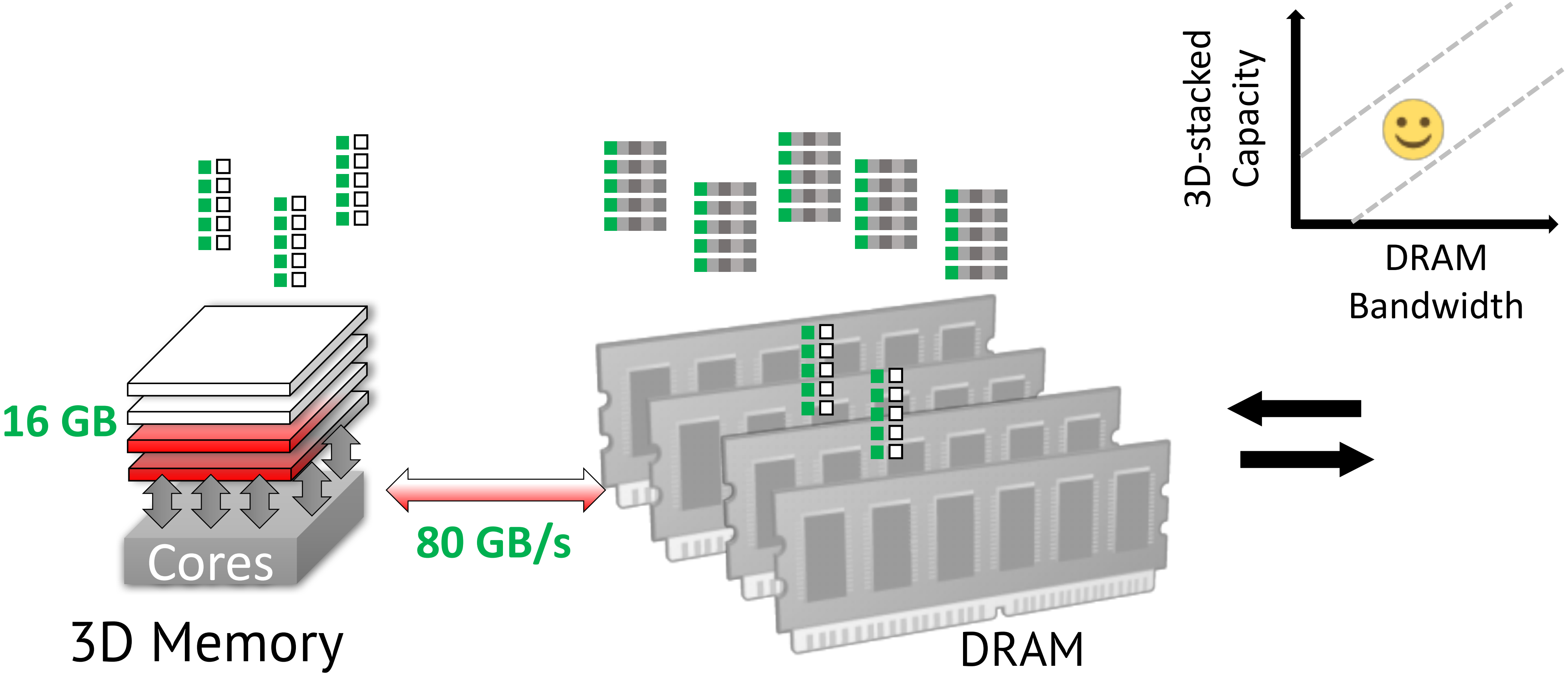
Solution 3: balance two limited resources





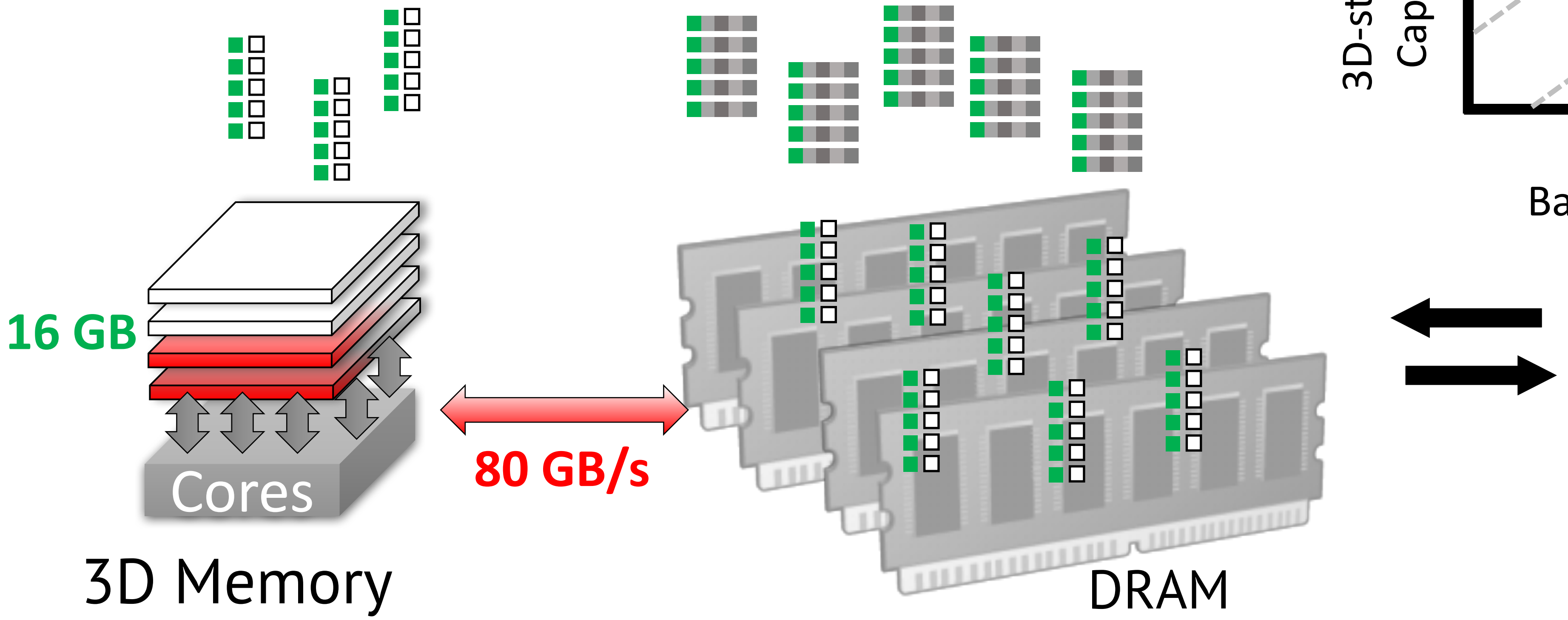
High pressure on 3D Memory capacity → indexes on DRAM

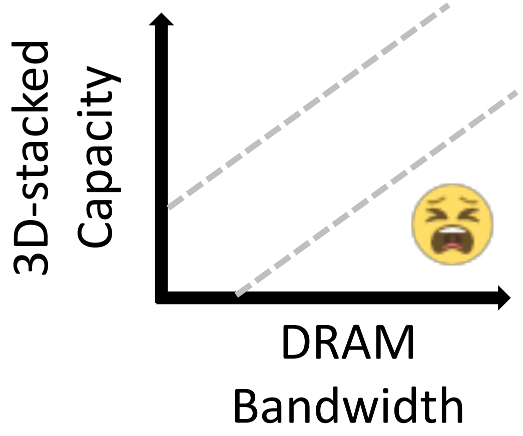
Solution 3: balance two limited resources



Pressure rebalanced

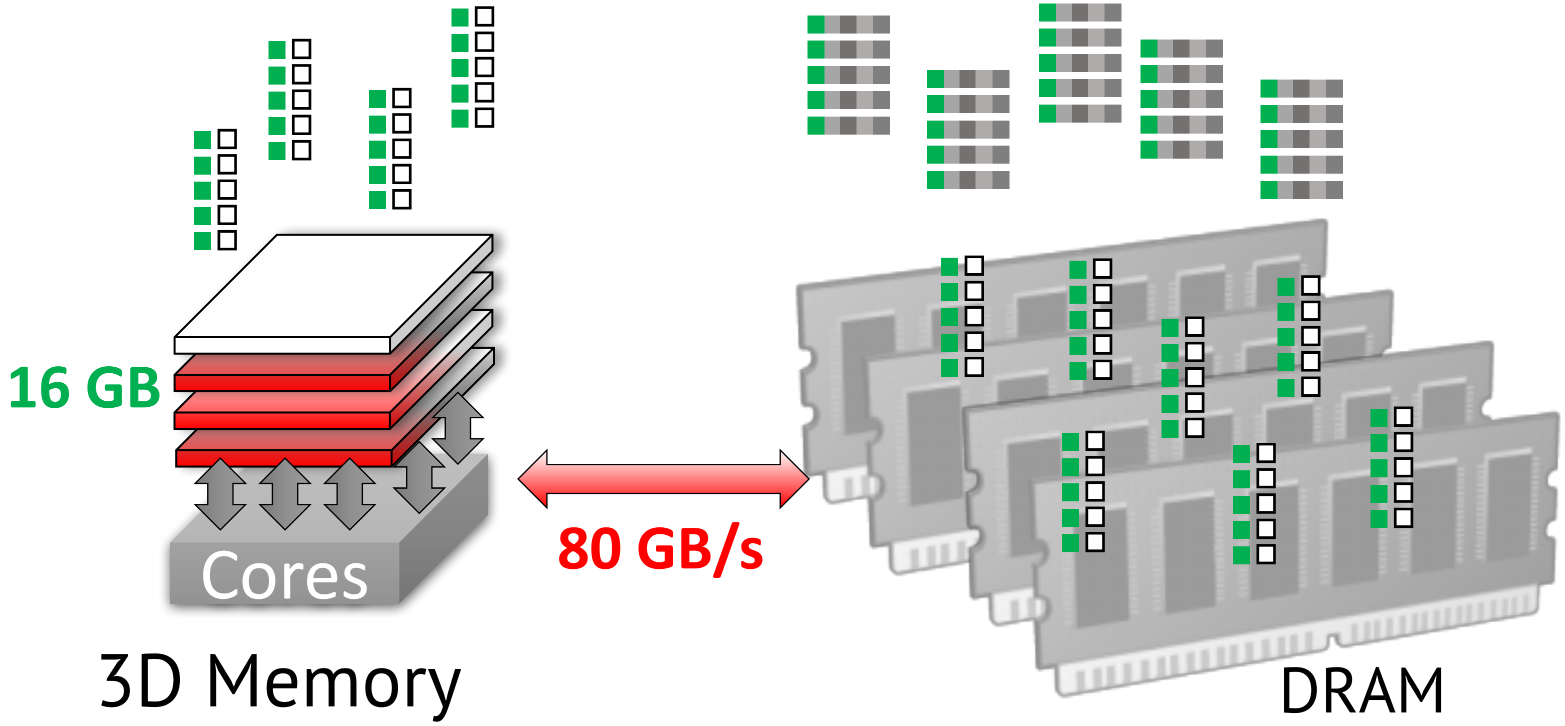
Solution 3: balance two limited resources

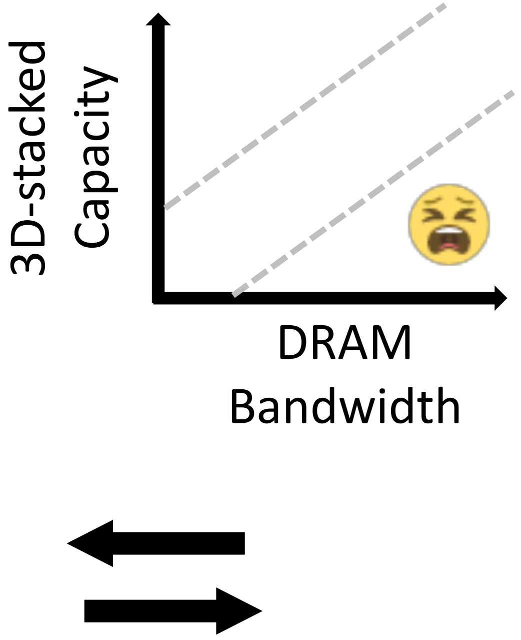




High pressure on DRAM bandwidth

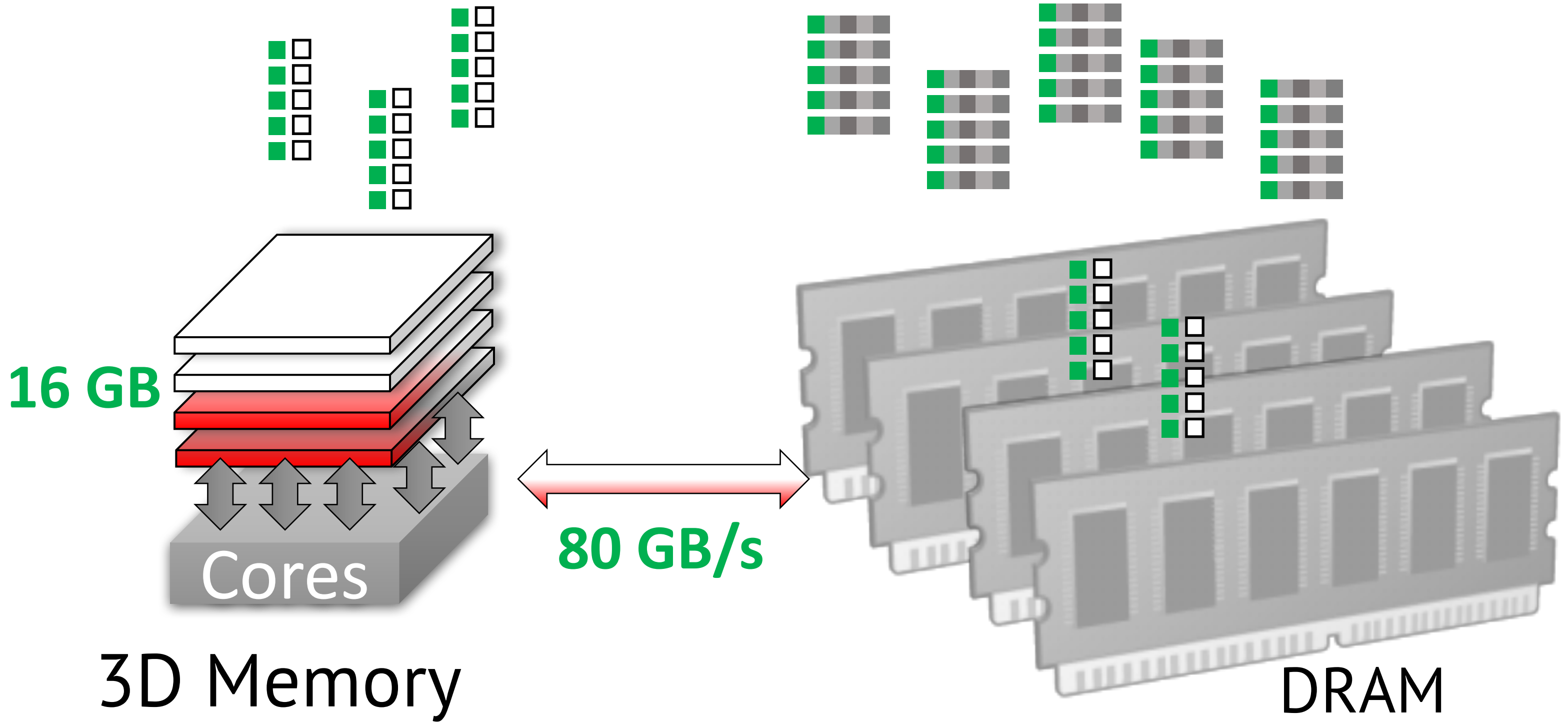
Solution 3: balance two limited resources

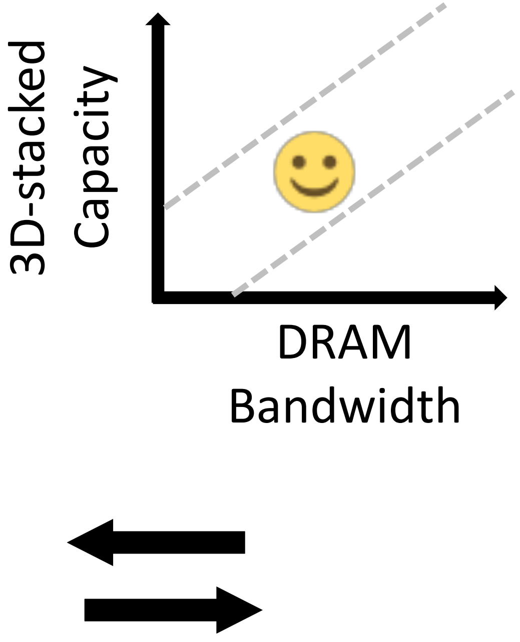




High pressure on DRAM bandwidth → more indexes on 3D memory

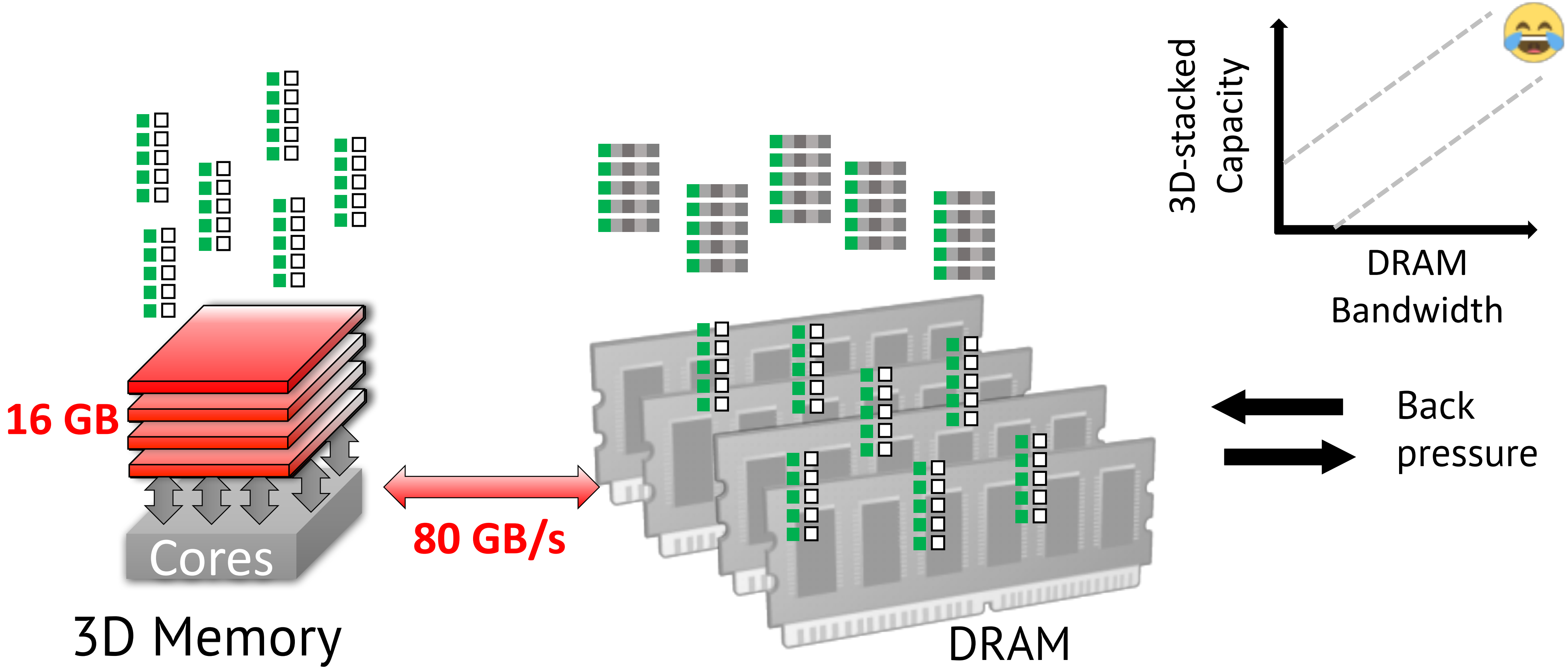
Solution 3: balance two limited resources





Pressure rebalanced

Solution 3: balance two limited resources



High pressure on both… → reach hardware limit → limit data ingestion

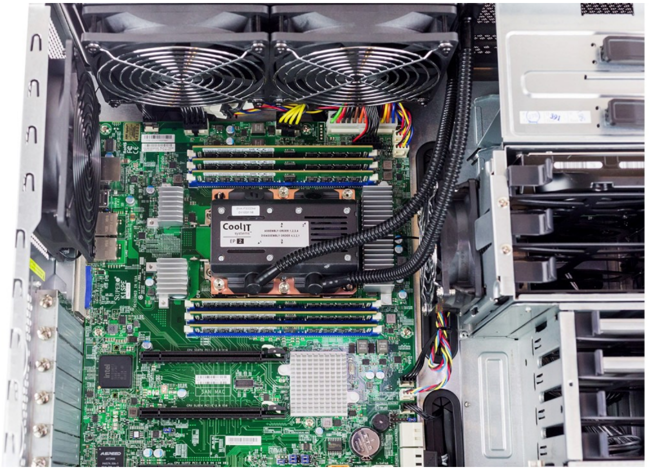
Other optimizations

- Customized memory allocator
- Customized task scheduler for high pipeline and data parallelism
- High parallel merge-sort kernels using avx-512
- Dynamically handle key changes
- Parallel aggregation
- Co-design RDMA ingestion with memory management and task scheduling
- Task parallelism to utilize all CPU cores
- …

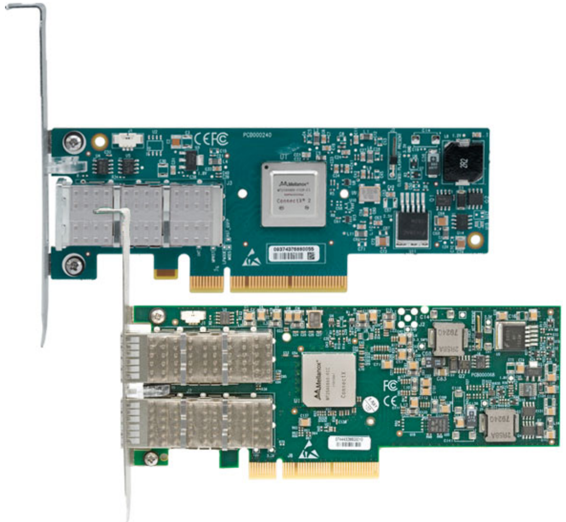
StreamBox-HBM Implementation

- Based on our prior work StreamBox [USENIX ATC'17]
- Implement on real hardware (Intel KNL) with RDMA network
  - 61K lines of C++11, of which 38K lines are new
  - Open source: http://xsel.rocks/p/streambox

16GB 3D memory
96GB DRAM
64 cores @1.3GHz



Ninja Developer Platform (KNL)



40Gb/s

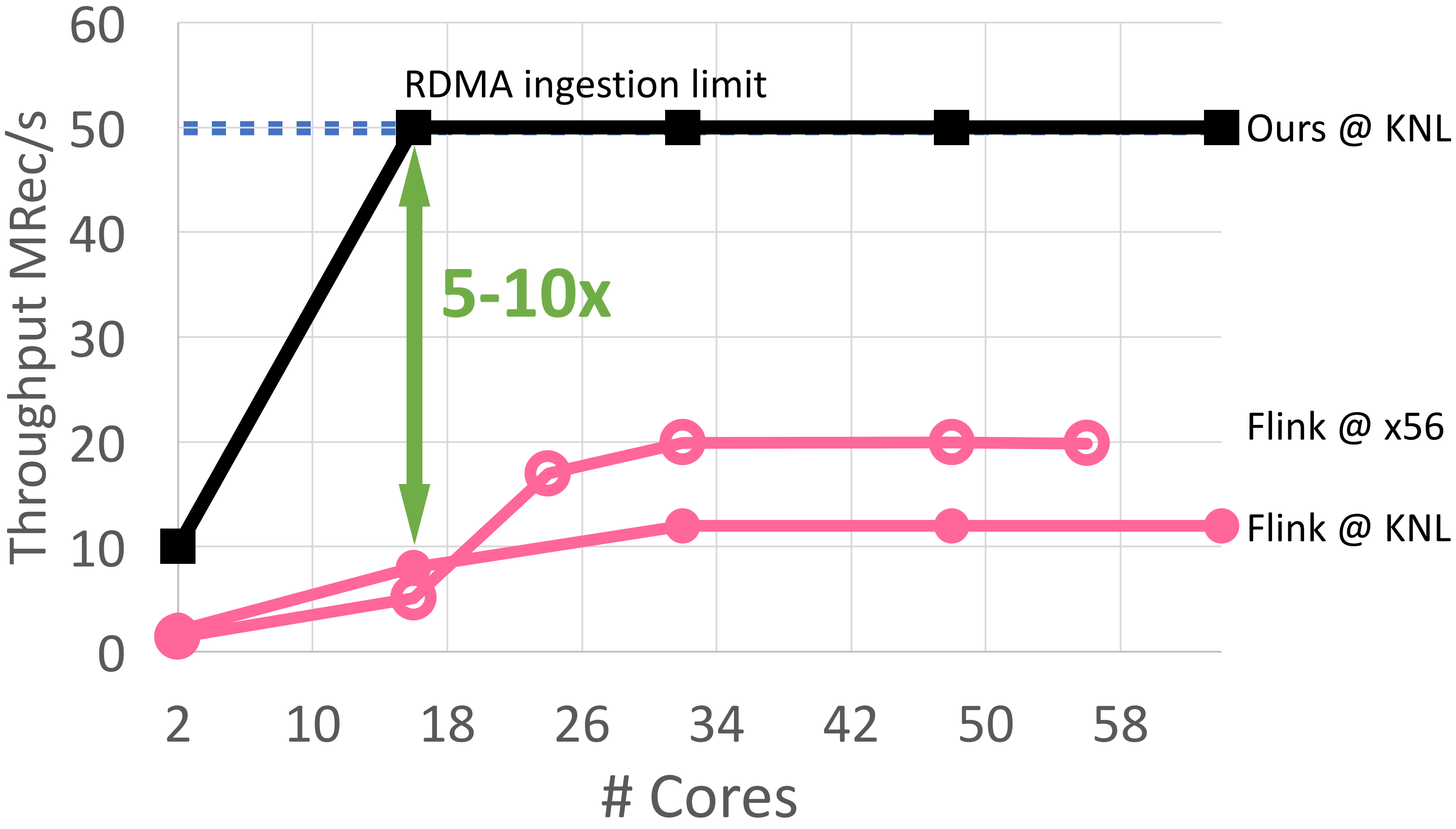
Mellanox ConnectX-2

[USENIX ATC'17] StreamBox: Modern Stream Processing on a Multicore Machine, Hongyu Miao, Heejin Park, Myeongjae Jeon, Gennady Pekhimenko, Kathryn S. McKinley, and Felix Xiaozhu Lin, in Proc. USENIX Annual Technical Conference, 2017.

Evaluation

- Comparing to widely used stream analytics engine
- Validating our key system designs

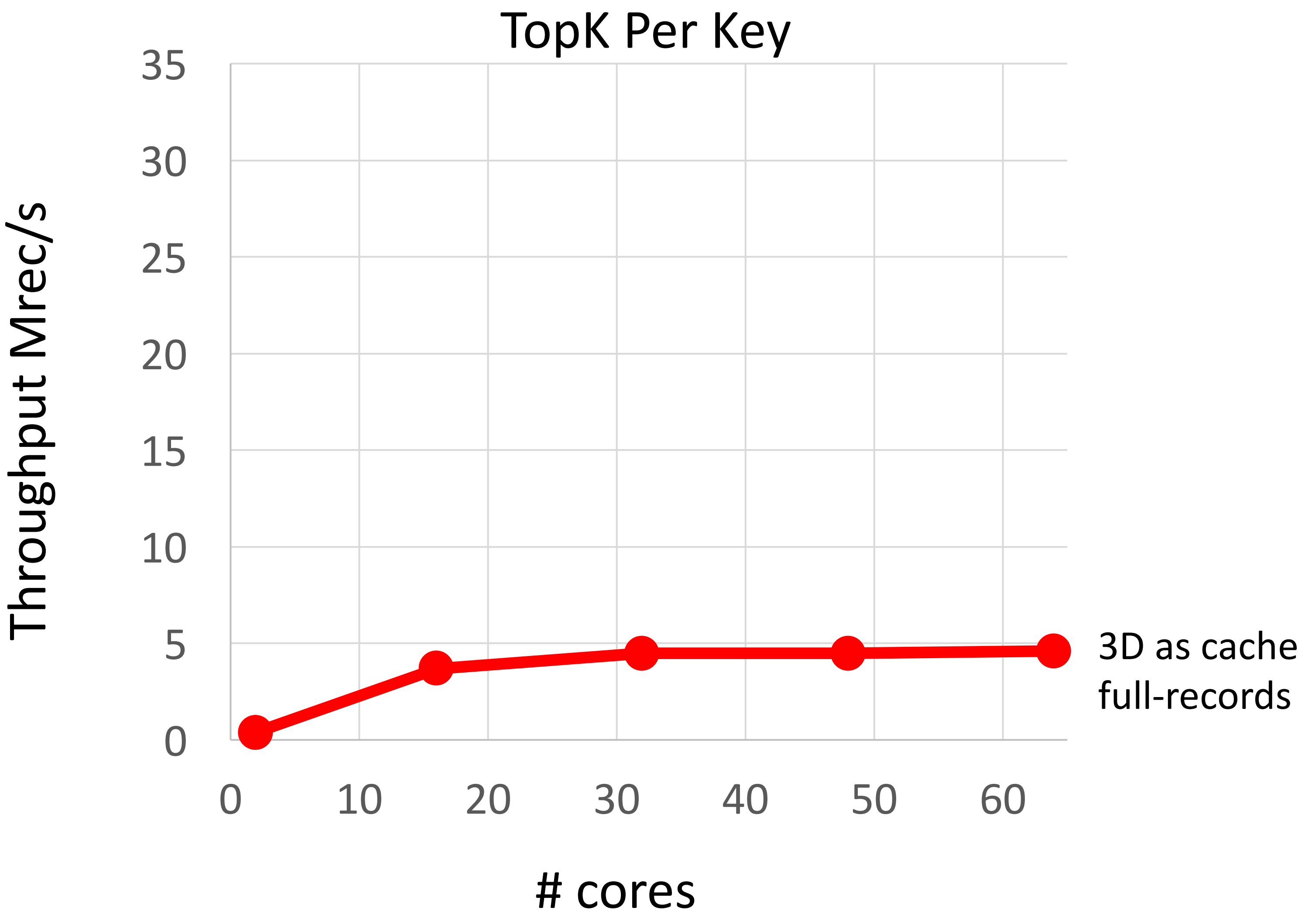
StreamBox-HBM is 10x faster than Flink



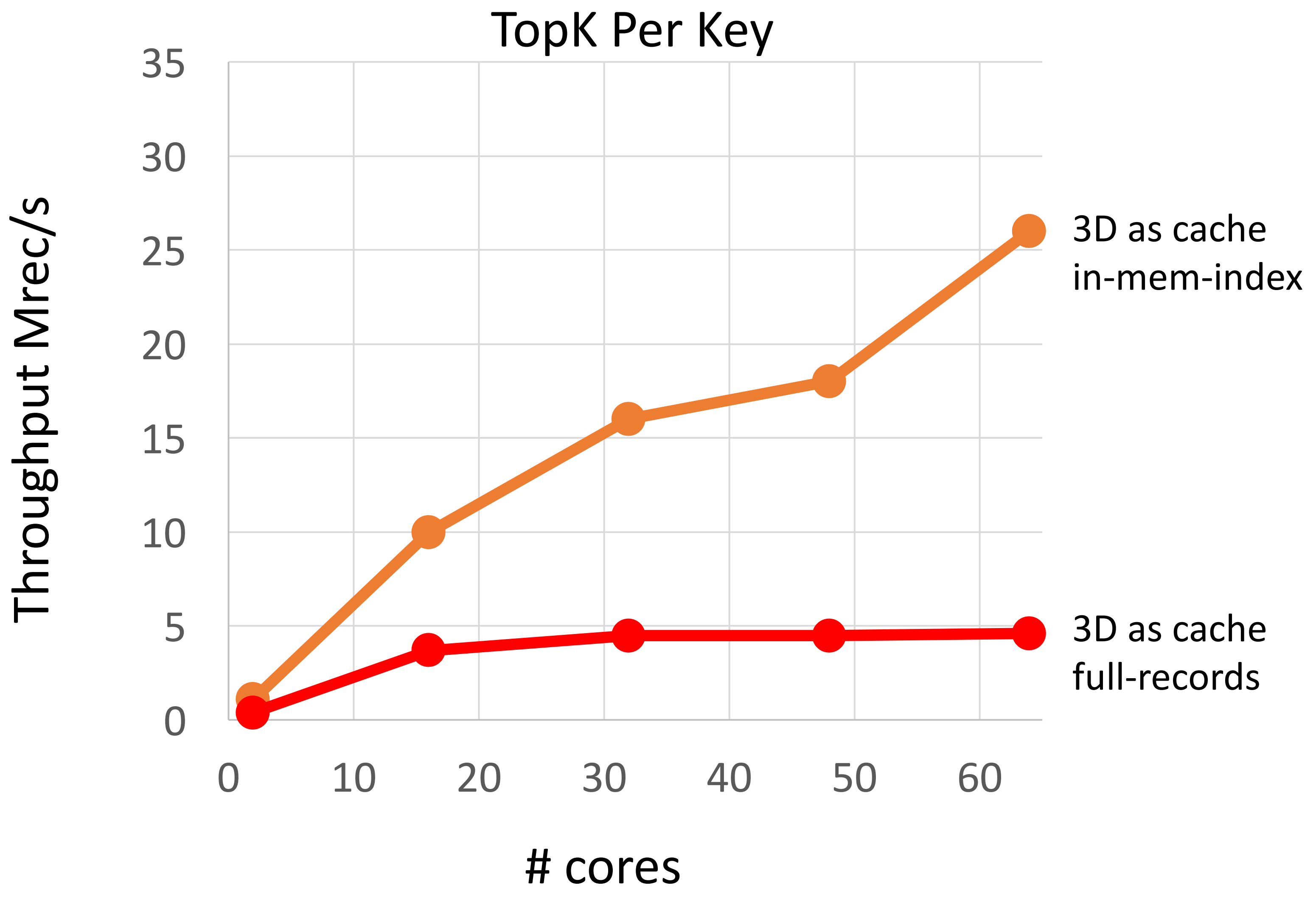
KNL: Intel Xeon Phi Knights Landing w/ HBM. 64 cores@1.3GHz. $5,000
x56: Intel Xeon E7-4830v4. 4x14 cores @2.0GHz. 256GB. $23,000

Benchmark: Yahoo Stream Benchmark.
Output delay: 1 second

Poor performance without any key designs

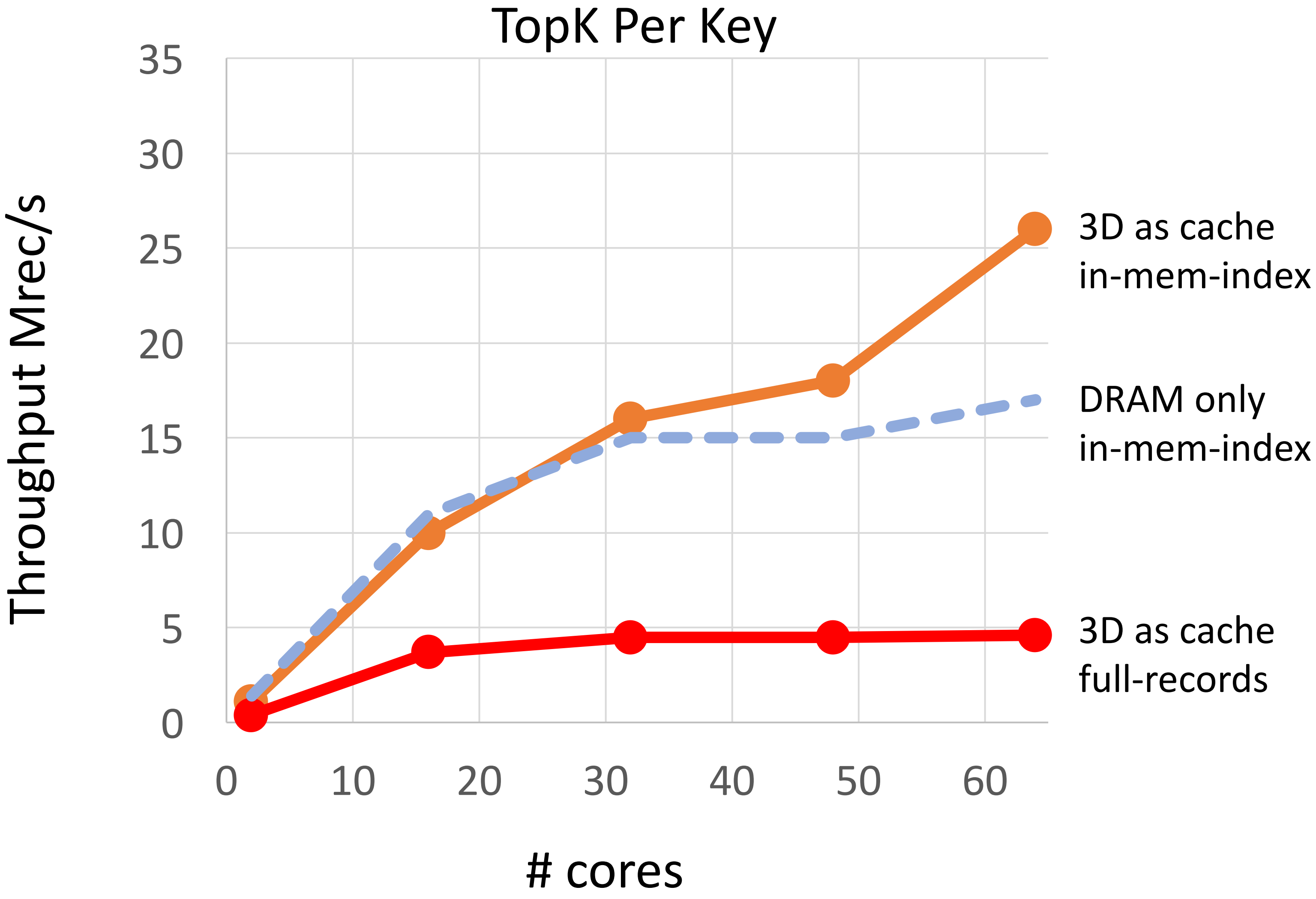


In-mem-index performs better than full-record



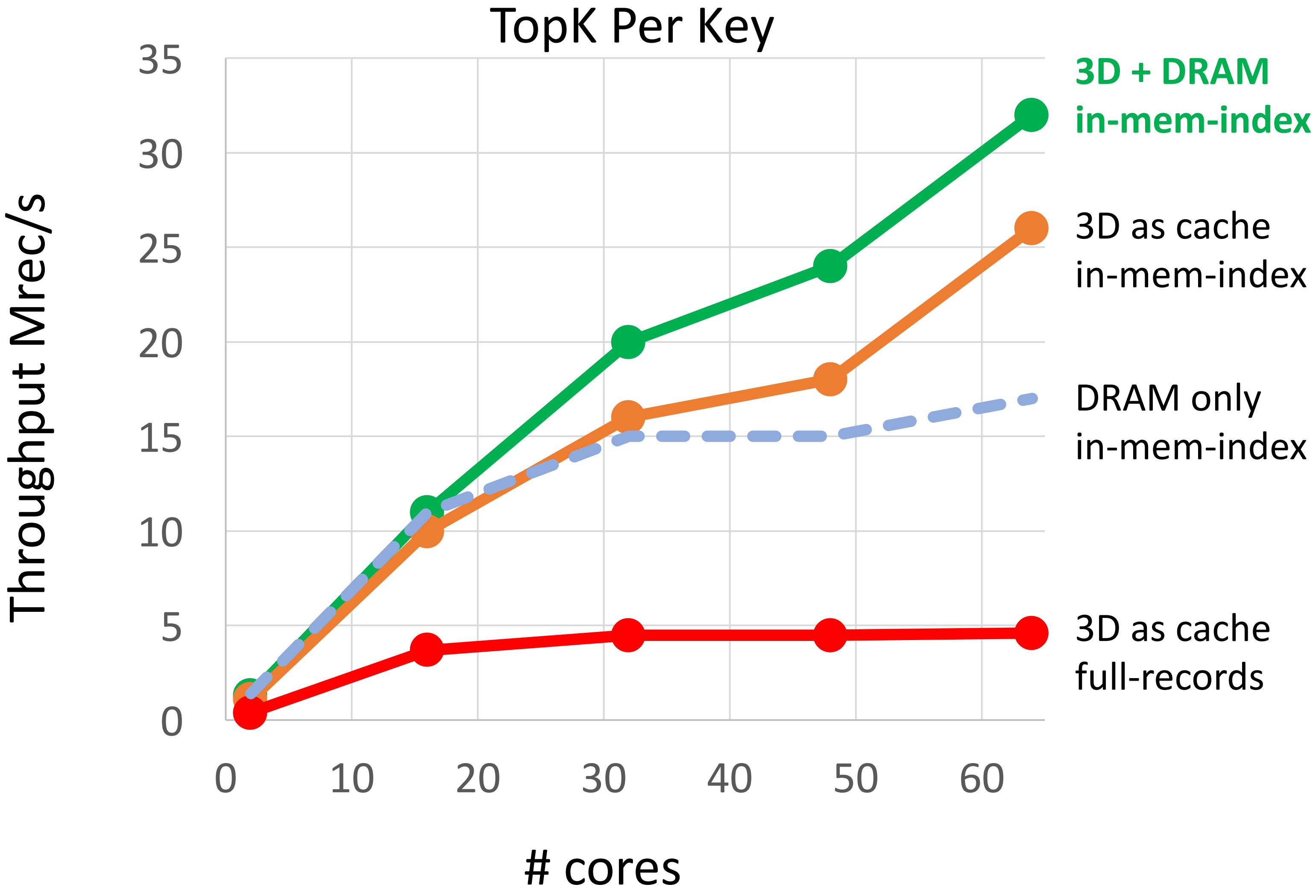
Using in-mem index

3D memory boosts performance



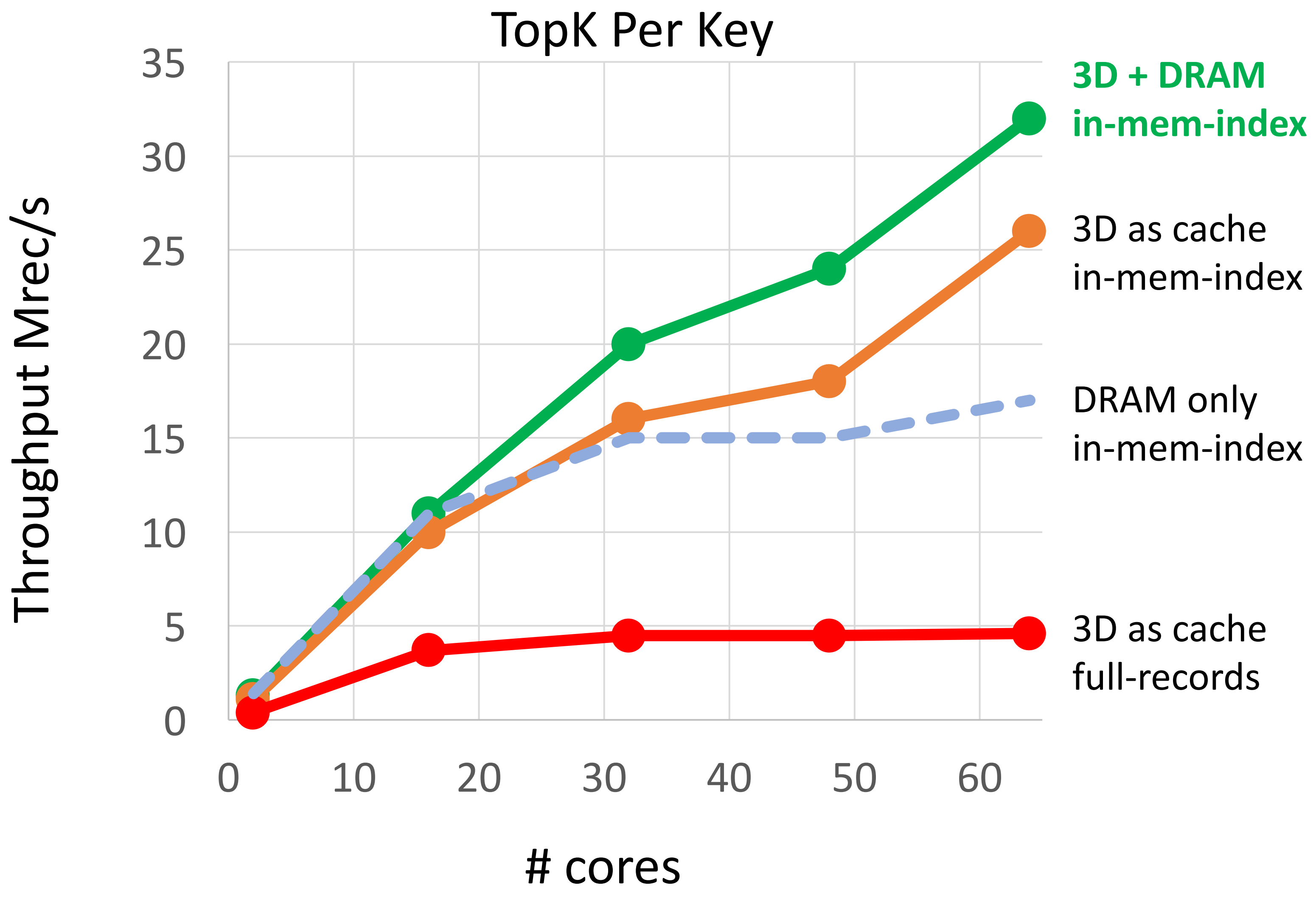
Using 3D memory

SW better manages hybrid memory than HW



SW manages hybrid memory

Performance improve with all system designs



Using all key system designs

StreamBox-HBM

The first stream engine optimized for 3D Memory + DRAM on real hardware

1. Grouping with Sort

Hash → Sort

Abundant memory

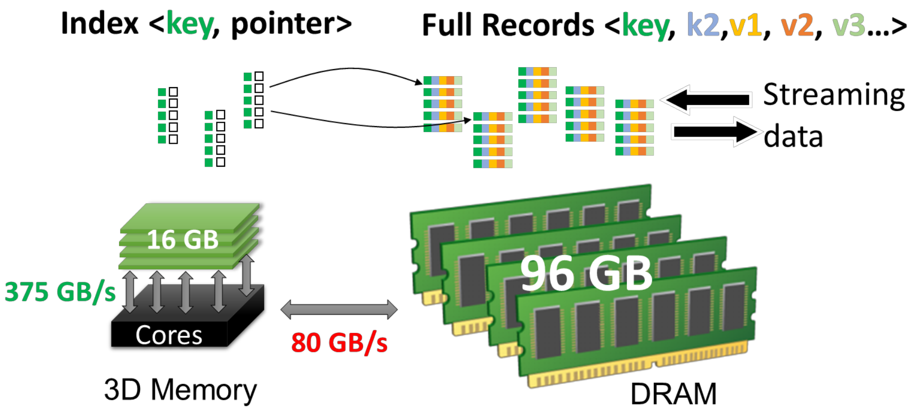
High parallelism

Wide SIMD (avx512)

Sequential access

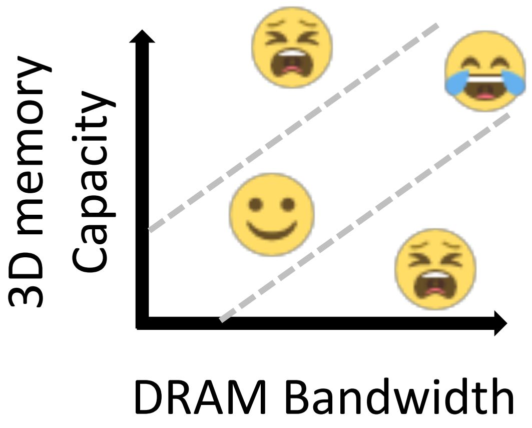
Exploit high bandwidth

2. In-memory index in 3D Memory



Minimize use of capacity

3. Mng hybrid mem



Balance limited resources

http://xsel.rocks/p/streambox

Lessons on exploiting 3D memory + DRAM

Apps

High task parallelism

Wide SIMD (avx512)

Sequential mem access

Packed data structure

Runtime

Thread pool + custom task scheduler

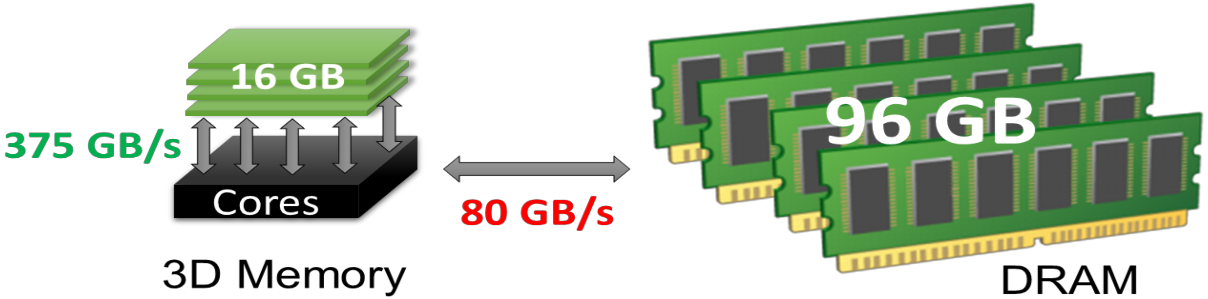
Custom mem allocator

OS kernel

Cheap VM (huge page)

RDMA network bypass kernel, free CPU

Hybrid Memory


16 GB
375 GB/s
Cores
3D Memory
80 GB/s
96 GB
DRAM