RETHINKING DATABASE ALGORITHMS FOR PCM

Shimin Chen (HP Labs), Phillip Gibbons (Intel Labs), Suman Nath (Microsoft Research)

PHASE CHANGE MEMORY (PCM)

Emerging byte-addressable non-volatile memory technology

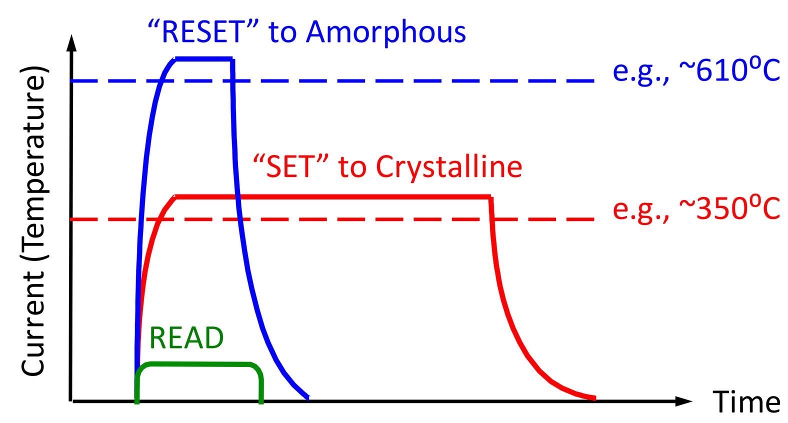
	DRAM	PCM*	NAND Flash
Page size	64B	64B	4KB
Page read latency	20-50ns	~ 50ns	~ 25 µs
Page write latency	20-50ns	~ 1 µs	~ 500 µs
Write bandwidth	~GB/s per die	50-100 MB/s per die	5-40 MB/s per die
Erase latency	N/A	N/A	~ 2 ms
Endurance	∞	$10^6 - 10^8$	$10^4 - 10^5$
Read energy	0.8 J/GB	1 J/GB	1.5 J/GB [28]
Write energy	1.2 J/GB	6 J/GB	17.5 J/GB [28]
Idle power	~100 mW/GB	~1 mW/GB	1–10 mW/GB
Density	1×	2 – 4×	4×

Sources (all public): [Doller'09] [Lee et al. '09] [Qureshi et al.'09] [Tseng et al. '06]. *PCM data is forecasted data

- Compared to DRAM, PCM has better density and scalability; PCM has similar read latency but longer write latency
- Compared to NAND Flash, PCM is byte-addressable, has orders of magnitude lower latency and higher endurance

CHALLENGE: PCM WRITES

- Limited endurance
  - Wear out quickly for hot spots
- High energy consumption
  - 6-10X more energy than a read
- High latency & low bandwidth
  - SET/RESET time > READ time
  - Limited instantaneous electric current level, requires multiple rounds of writes: Writes 20X slower than reads



PCM Write Optimizations in literature: [Cho&Lee'09] [Lee et al '09] [Yang et al'07] [Zhou et al'09]

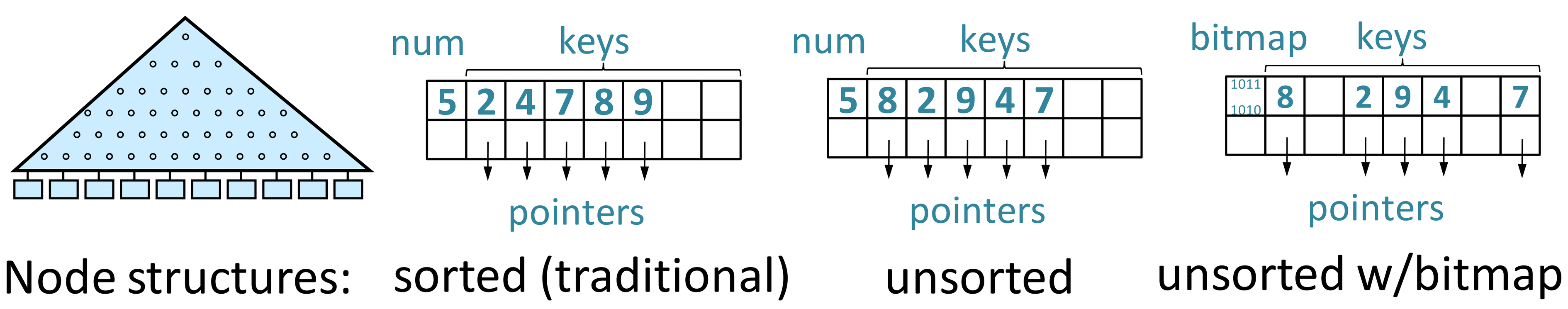
- Baseline: several rounds of writes for a cache line
  - Which bits in which rounds are hard wired
- Optimization: data comparison write:
  - Goal: write only modified bits rather than entire cache line
  - Approach: read-compare-write
- Skipping rounds with no modified bits

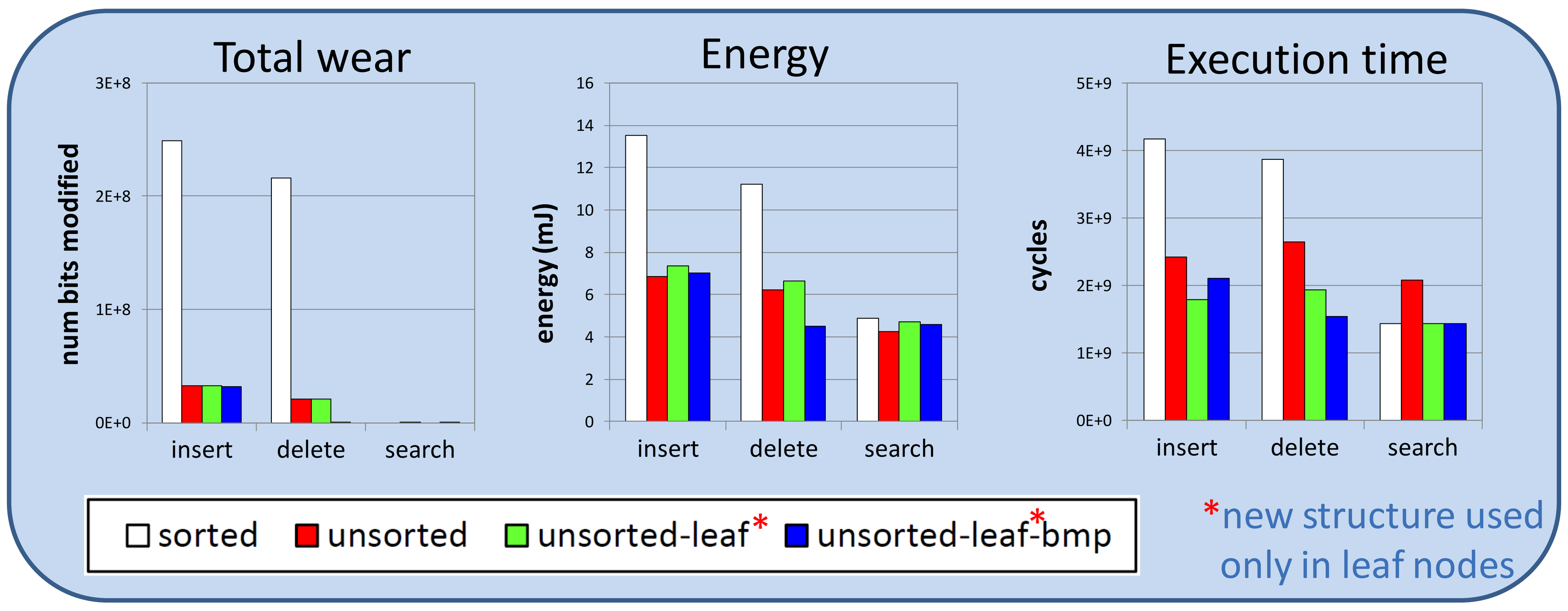
GOAL: DATABASE MINIMIZES PCM WRITES

PCM-DB: Database systems exploiting PCM as primary main memory

- Initial focus on key database algorithms:
  - B^+-Tree Index & Hash Joins
  - Structures reside in PCM main memory
  - Optional DRAM is another (transparent or explicit) cache
- Key analytical metrics:
  - Total Wear, Energy, Total PCM Access Latency
- Experimental Setup: PTLSSim extended with PCM support
  - B^+-Tree: Node size 8 cache lines, 50M entries, 75% full; Inserting / Deleting / Searching 500K random keys
  - Hash Join: 500MN joins 100MB; varying record size from 20B-100B

B^+-TREE ALGORITHMS & RESULTS

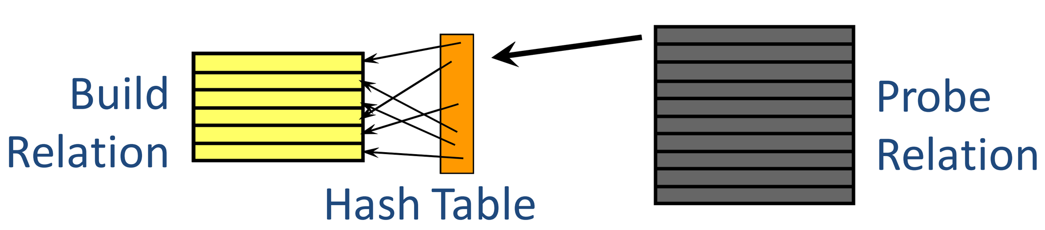




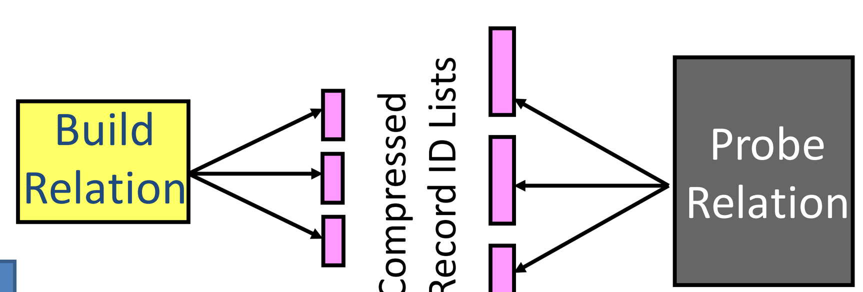
Unsorted leaf schemes achieve the best performance

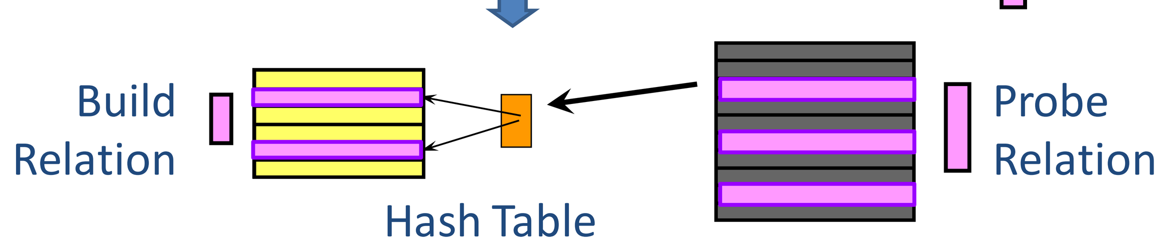
IN-MEMORY HASH JOIN ALGORITHMS & RESULTS

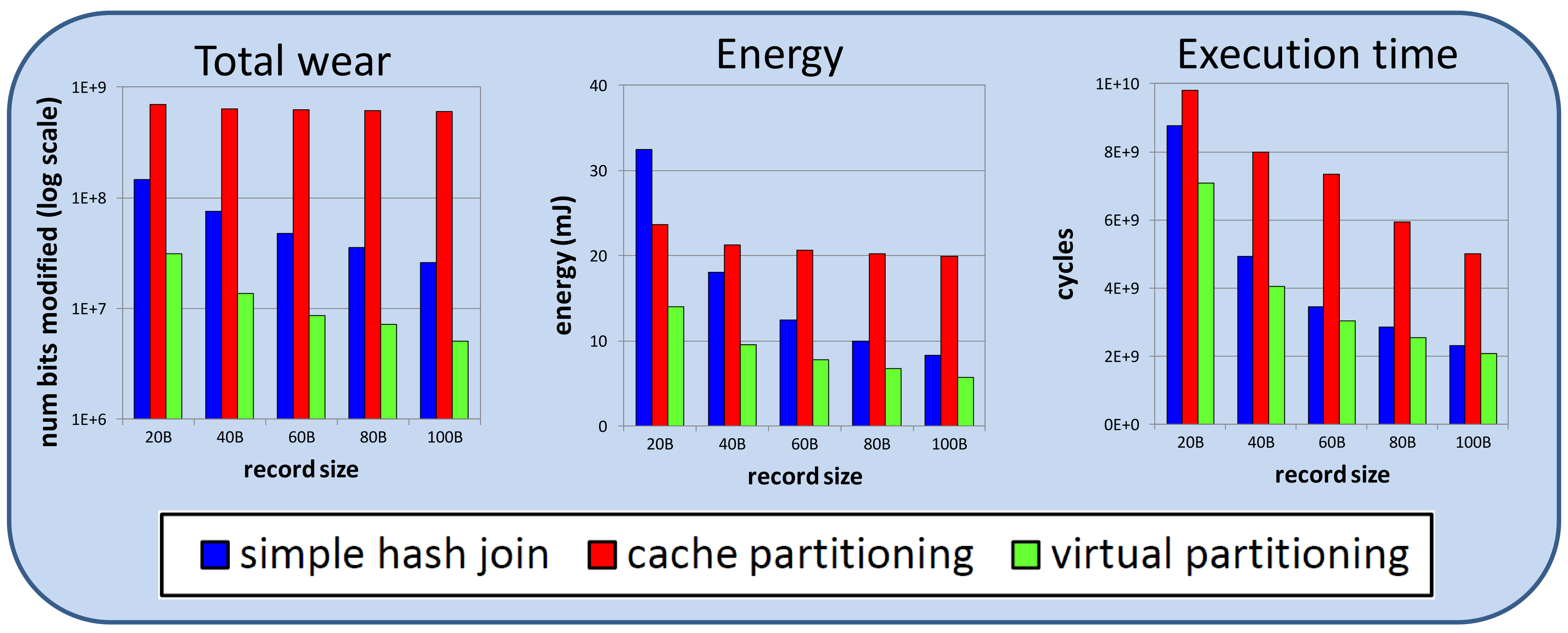
- Simple Hash Join



- Cache Partitioning
  - Partition each table into cache-sized partitions
  - Join each pair of partitions
- Virtual Partitioning (new)
  - Partition without copying
  - Join each pair of virtual partitions







Virtual Partitioning achieves the best performance

Intel Science & Technology Center for Cloud Computing

[Paper in CIDR'11]

Carnegie Mellon University · Georgia Tech · intel · Princeton University · UC Berkeley