Parallel Real-Time OLAP on Multi-Core Processors and Cloud Architectures

Frank Dehne
Carleton University, Ottawa, Canada
IBM Center For Advanced Studies, Toronto, Canada

Joint work with
R. Bordawekar (IBM Yorktown), **J. Dale** (IBM Littletown), **R. Grosset** (IBM Toronto), **M. Genkin** (IBM Toronto), **S. Jou** (IBM Toronto), **P. Jain** (IBM Littletown), **Q. Kong** (Dalhousie), **M. Petitclerc** (IBM Laval), **A. Rau-Chaplin** (Dalhousie), **D. Robillard** (Carleton), **F. Thomas** (IBM Ottawa), **H. Zaboli** (Carleton), **R. Zhou** (Carleton)

Carleton UNIVERSITY

IBM

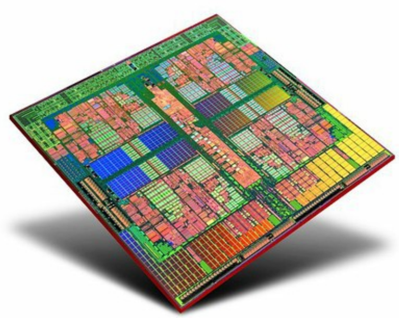
About the Speaker

Research Program:

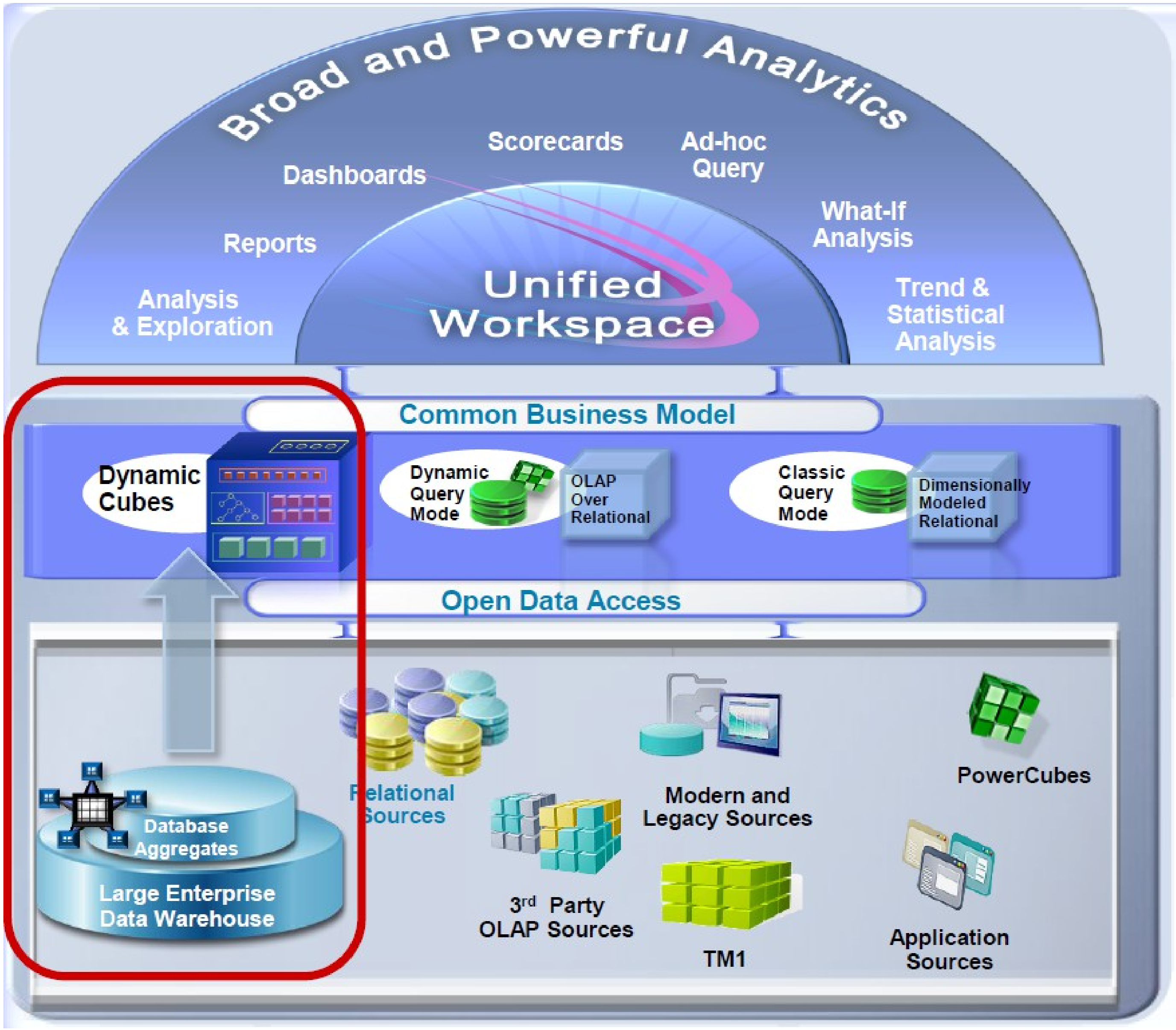
- The design and implementation of efficient *parallel* algorithms.
- The interrelationship between the theoretical analysis of parallel algorithms and the performance observed on current parallel architectures
- The use of efficient parallel algorithms for large-scale data analytics and computational biology

Current Projects

- Auto-tuned parallel algorithms for multi-core processors, GPUs, clusters & clouds.
- Parallel large-scale data analytics: online analytical processing (OLAP).
- Parallel computational biology: protein-protein interaction prediction.



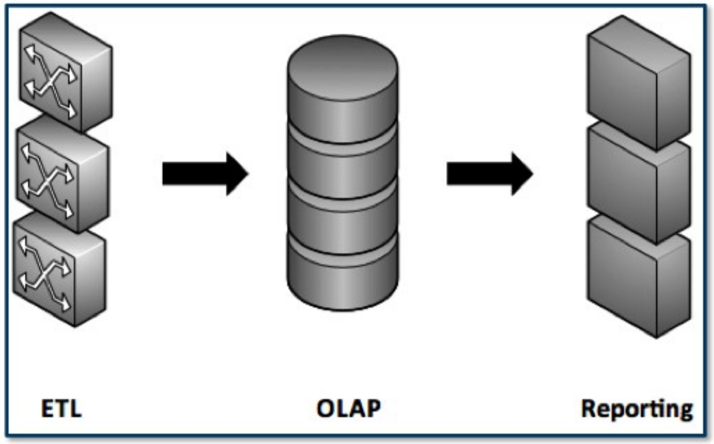
Online Analytical Processing (OLAP)



IBM/COGNOS

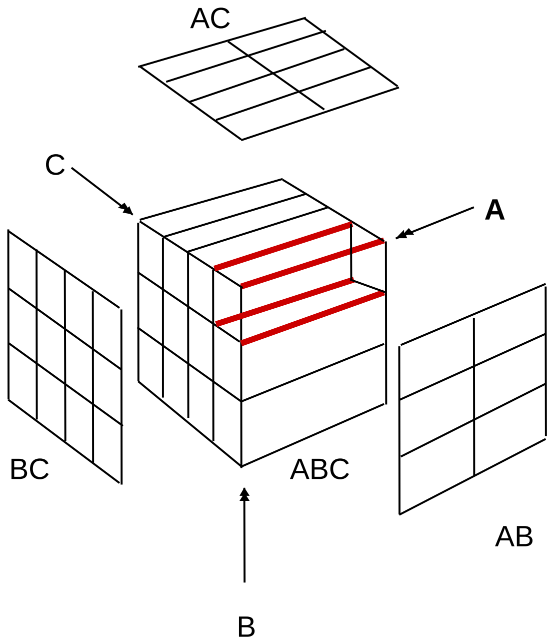
Insight
Workspace
Report/Studio

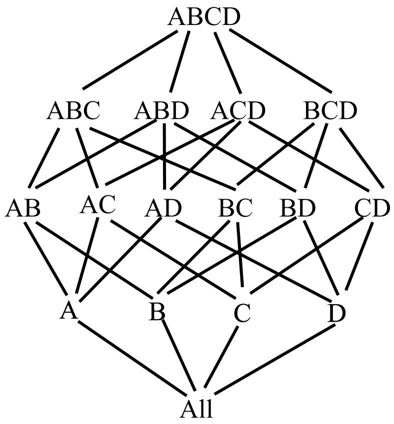
Online Analytical Processing (OLAP)



Operations:

- roll-up
- drill-down
- slice
- dice





Traditional: Data Cube

Pre-compute group-bys to improve query response time.

Static or Batch Updates

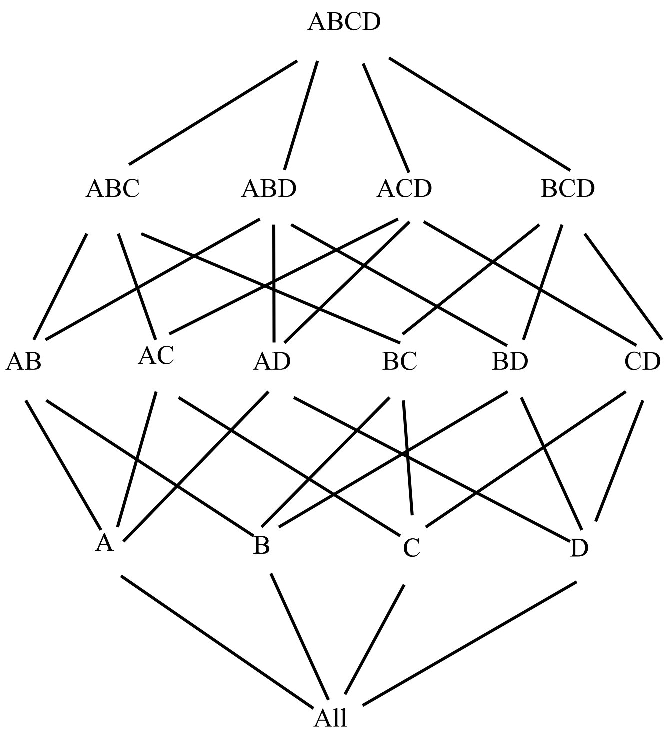
OLAP vs. OLTP

	OLTP System	OLAP System
Source of data	Operational data	Consolidated data
Purpose of data	Business operations	Planning, decision support
Type of data	Snapshot of ongoing business	Multi-dimensional views of “historic” data
Updates	Small and fast	Periodic long-running batch jobs
Queries	Relatively simple, involving few data records	Often complex, involving aggregations of large data sets
Processing speed	Typically very fast	Depends on amount of data involved; batch updates and complex queries may take many hours

Source: AcceleratedAnalytics.com

The Five V's Of “Big Data”

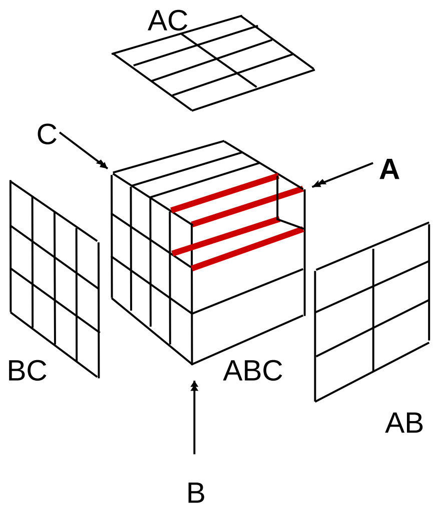
- Volume
- **Velocity**
- Variety
- Veracity
- Value



Real-Time OLAP

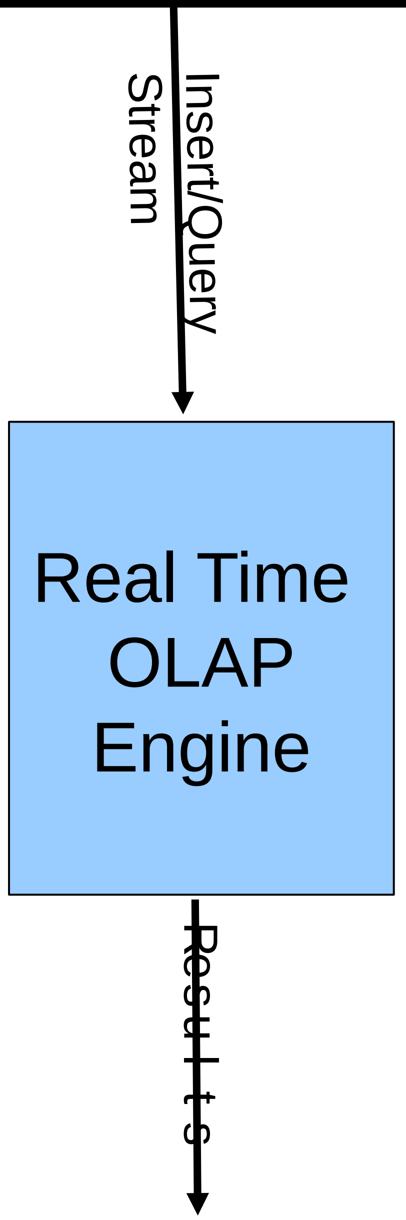
- Avoid static data cube structure and batch updates.
- **Stream** of OLAP *insert* and *query* operations.
- Inserts are **immediate.**

Queries operate on latest **up-to-date** data set.



Real-Time OLAP

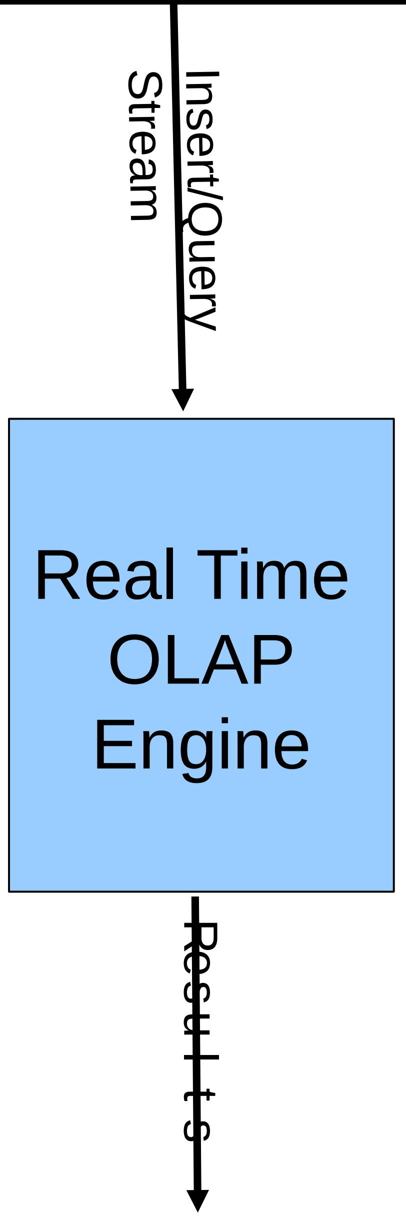
- Problem: **Performance**.
- Data cube was introduced to improve performance!



Real-Time OLAP

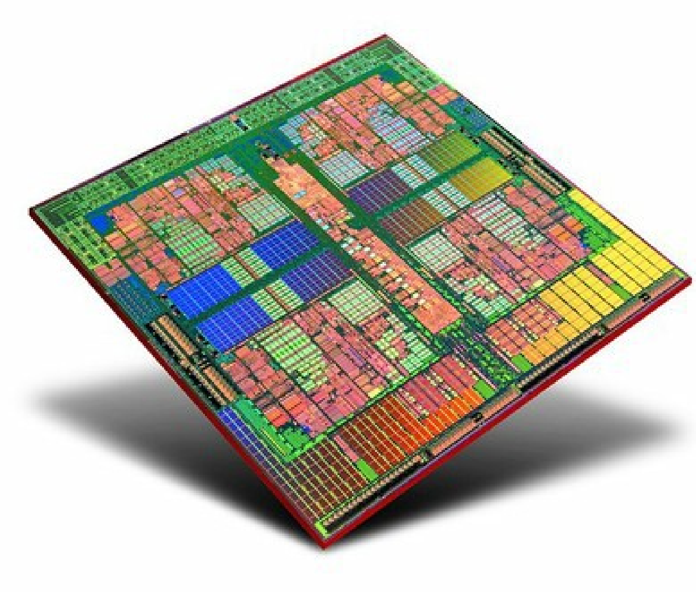
Research Question:

Can **parallel computing** be used to improve performance for real-time OLAP?



Parallel Computing

Multi-core Processors



shared memory

Cloud / Cluster

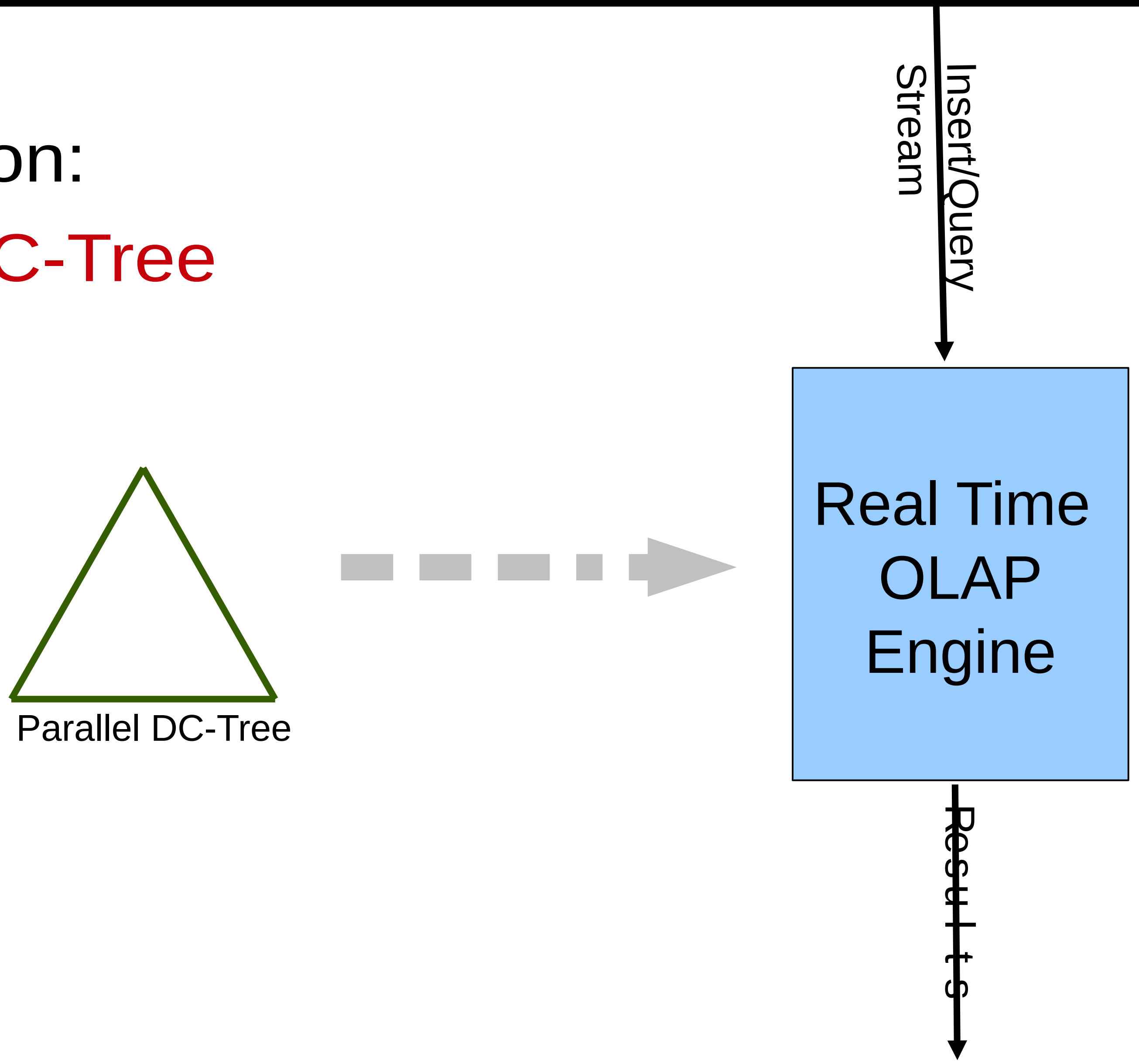
distributed memory

Parallel real-time OLAP on multi-core processors

Parallel Real-Time OLAP

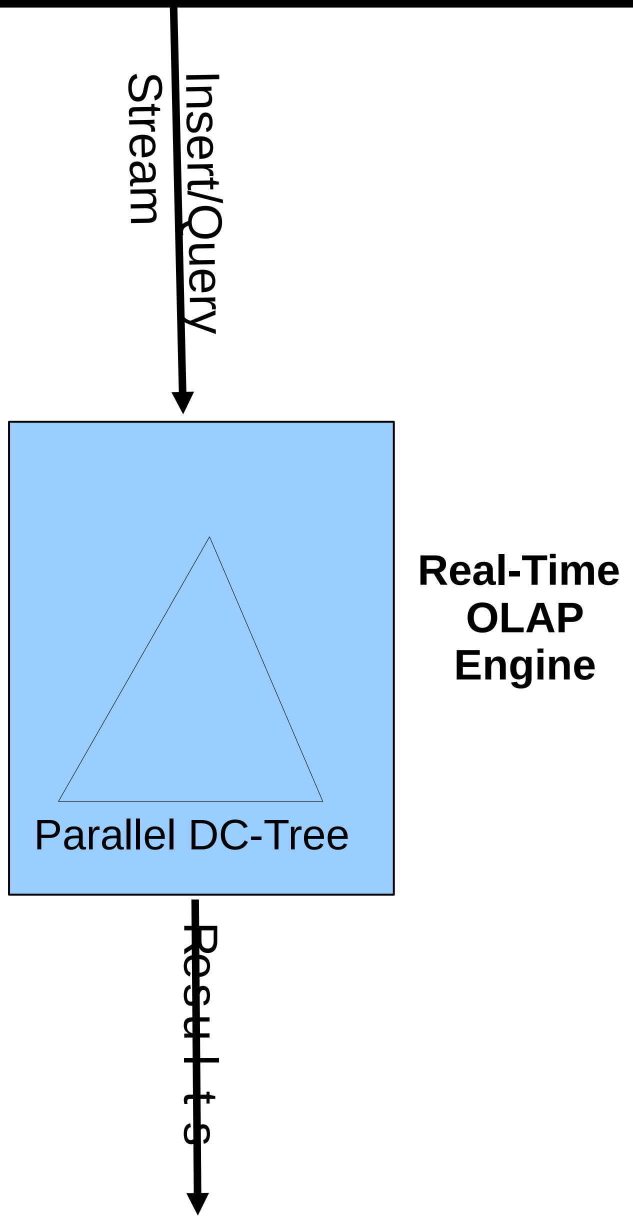
Our solution:

Parallel DC-Tree



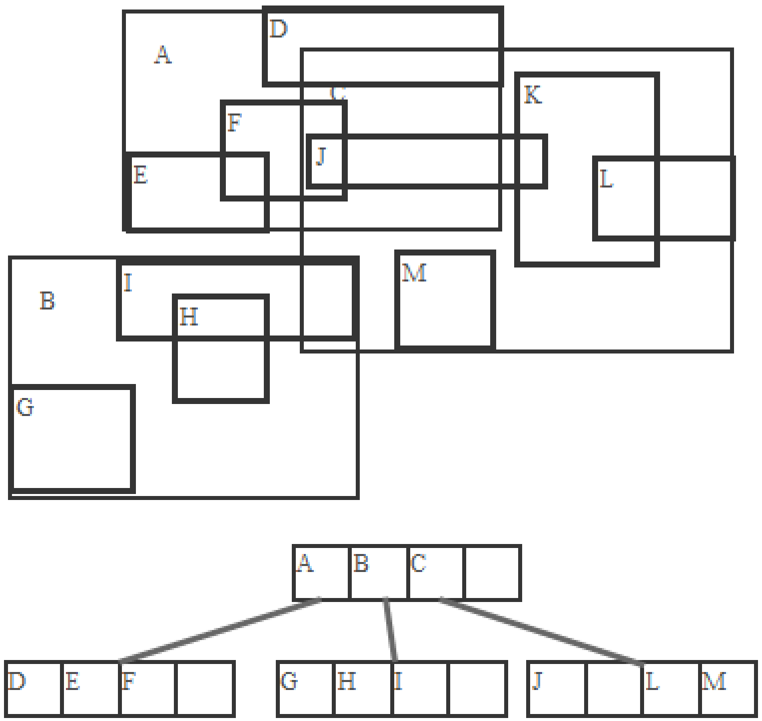
Parallel DC-Tree

- Multidimensional tree data structure.
- Operations: *insert* and *query*.
- Enhanced for data aggregation and dimension hierarchies (Kriegel et.al., ICDE 2000)
- Enhanced for multi-core parallel computing (Dehne et.al., CCGrid 2012)



Sequential DC-Tree

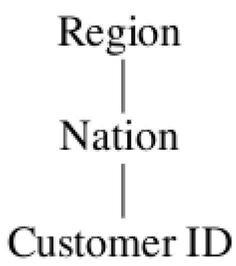
- Ester, Kohlhammer, Kriegel (ICDE 2000)
- Adaptation of R-tree for OLAP
- Replaces total ordering by ***concept-hierarchies***.
- Replaces minimum bounding rectangles (MBR) by ***minimum describing sets*** (MDS)
- Adds internal ***directory nodes***



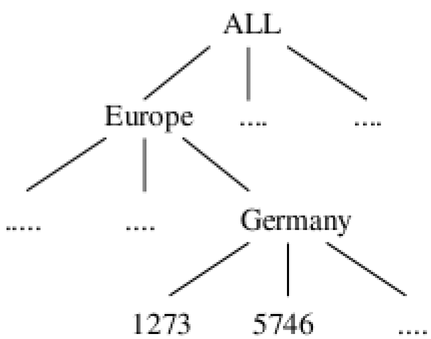
R-Tree

Concept hierarchies

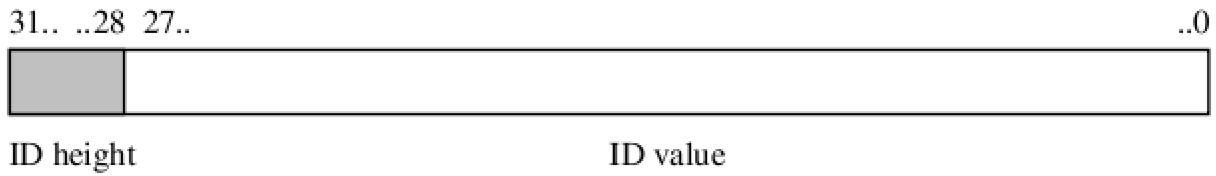
- Hierarchy Schema for dimension *Customer* :



- Conceptual Hierarchy for dimension *Customer* :

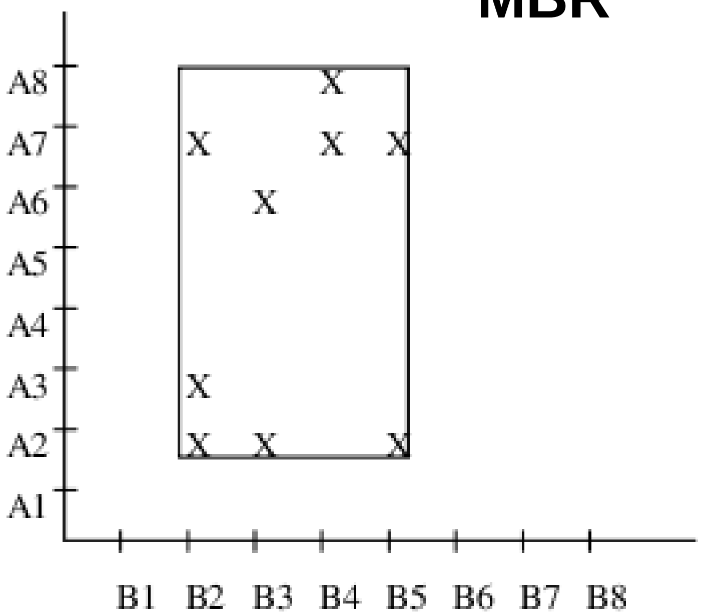


- Data representation:

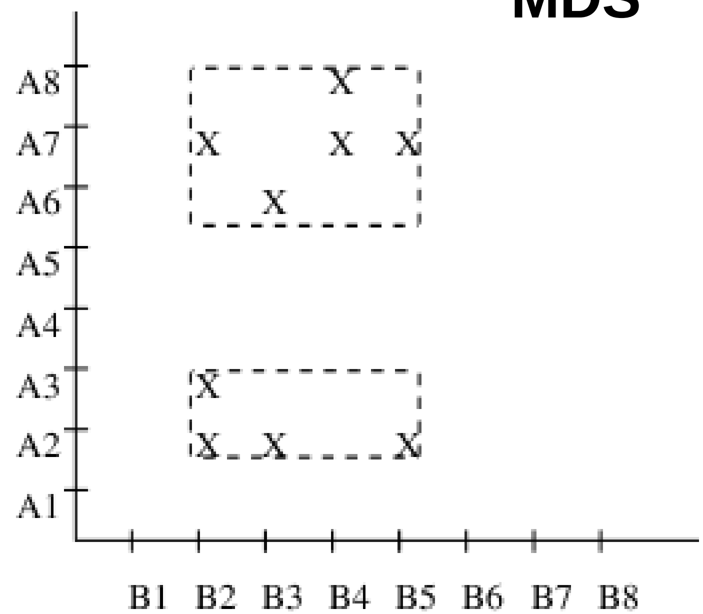


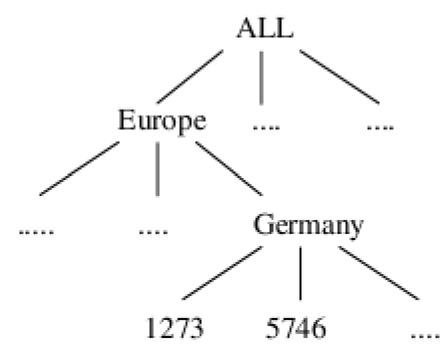
Minimum describing sets (MDS)

MBR



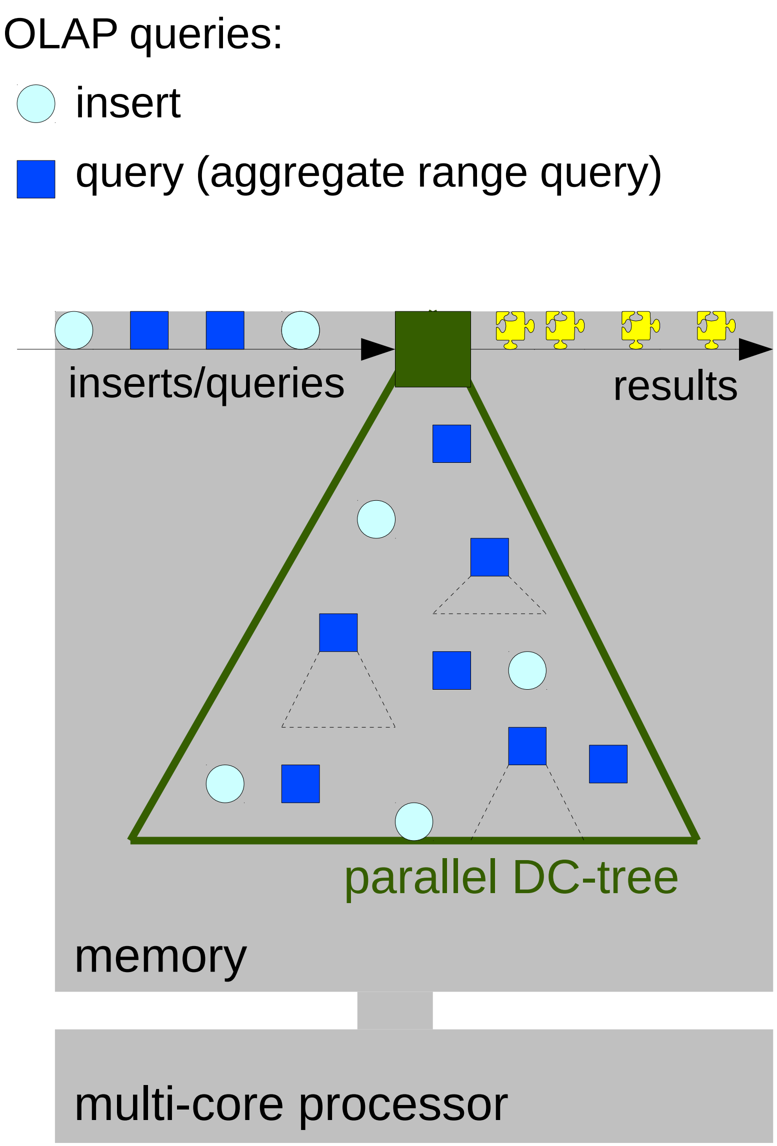
MDS





Parallel DC-Tree

- Dehne, Zaboli (CCGrid 2012)
- Parallelization:
  - Insert and query operations are executed concurrently.
  - Query operations that need to search multiple subtrees of a node are split into multiple concurrent processes.

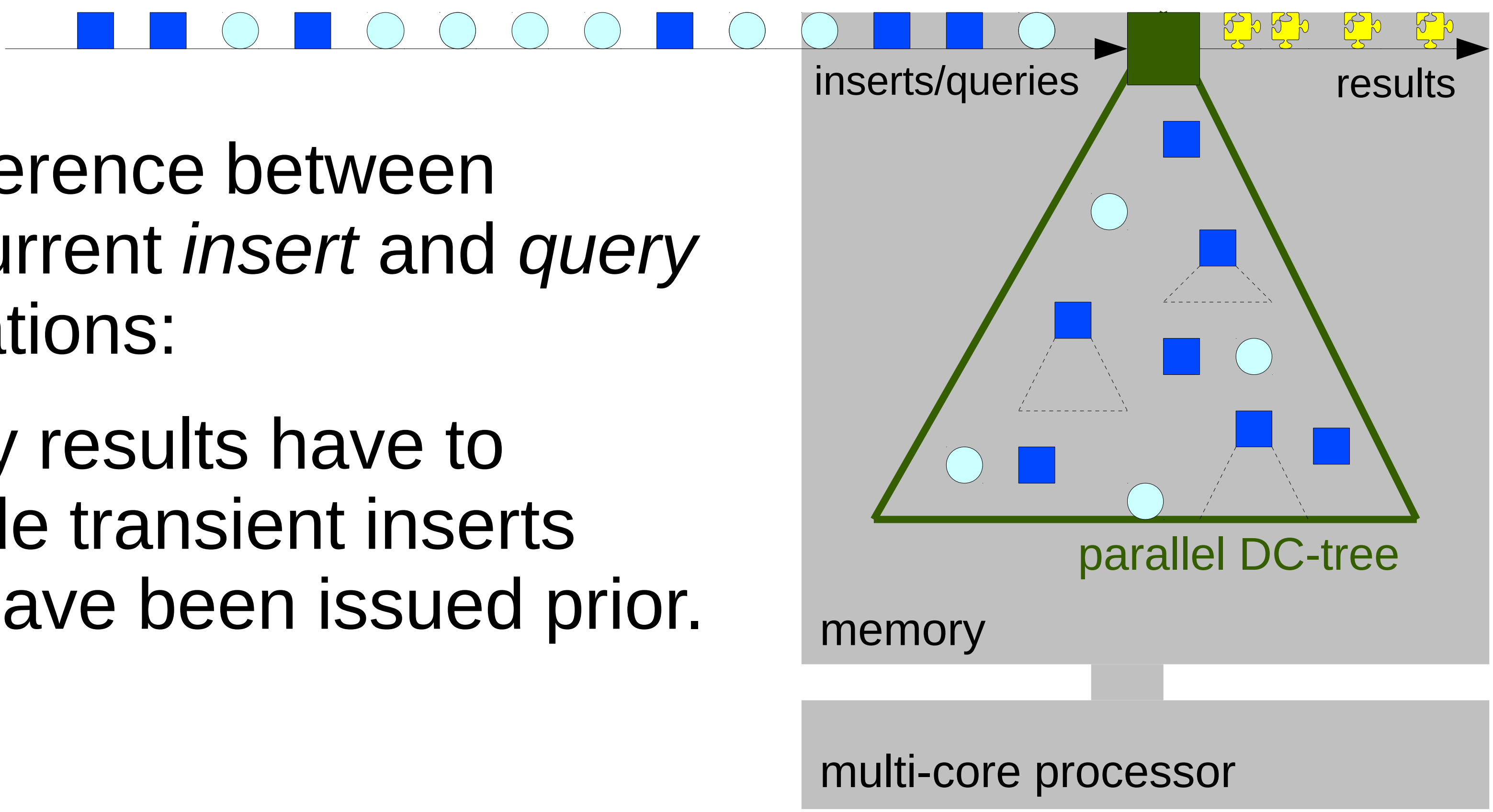


Parallel DC-Tree

Main Problem

OLAP queries:

- insert
- query (aggregate range query)



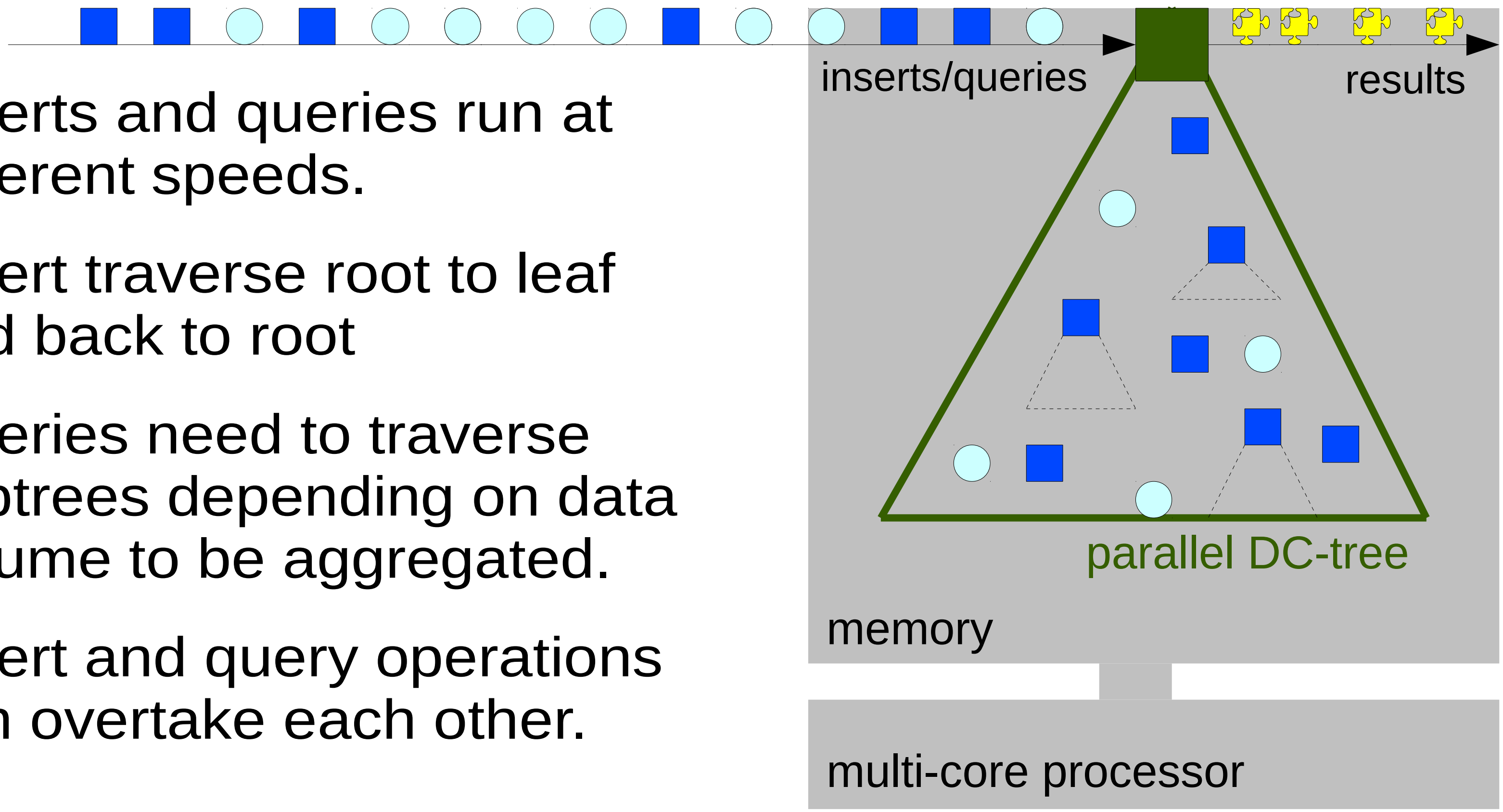
Interference between concurrent *insert* and *query* operations:

Query results have to include transient inserts that have been issued prior.

Race Conditions

OLAP queries:

- insert
- query (aggregate range query)



- Inserts and queries run at different speeds.
- Insert traverse root to leaf and back to root
- Queries need to traverse subtrees depending on data volume to be aggregated.
- Insert and query operations can overtake each other.

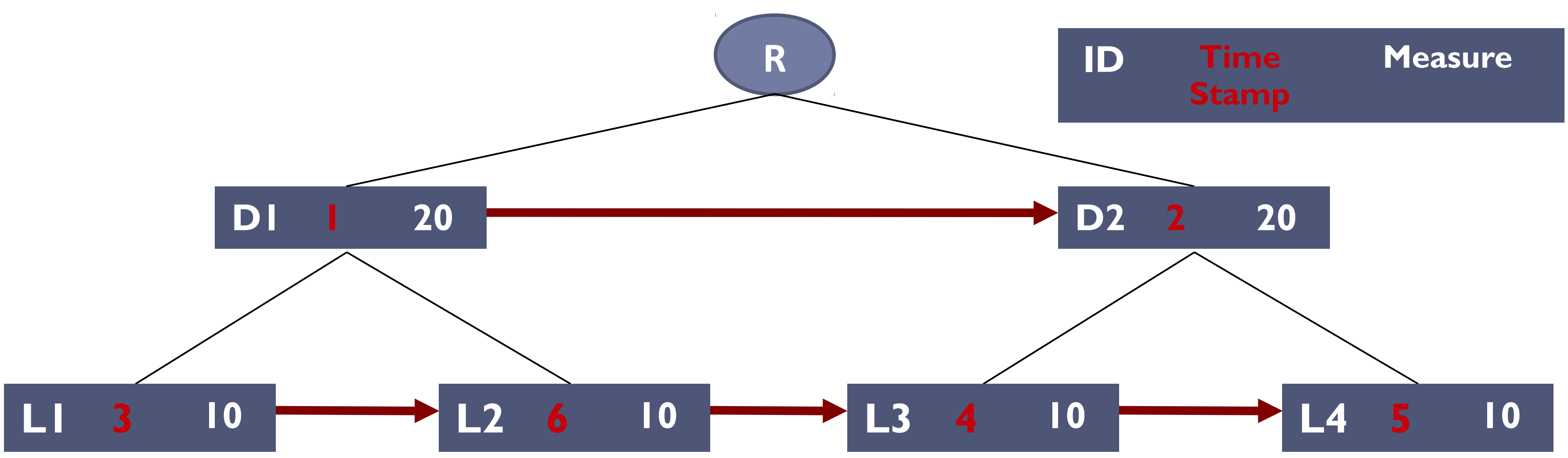
Parallel Insert & Query Operations

Solution:

Add to data structure

- Right sibling links
- Timestamps

- Lengthy case analysis
- Most challenging case:
  - Insert creates node split. Transient query needs to detect and re-calculate.

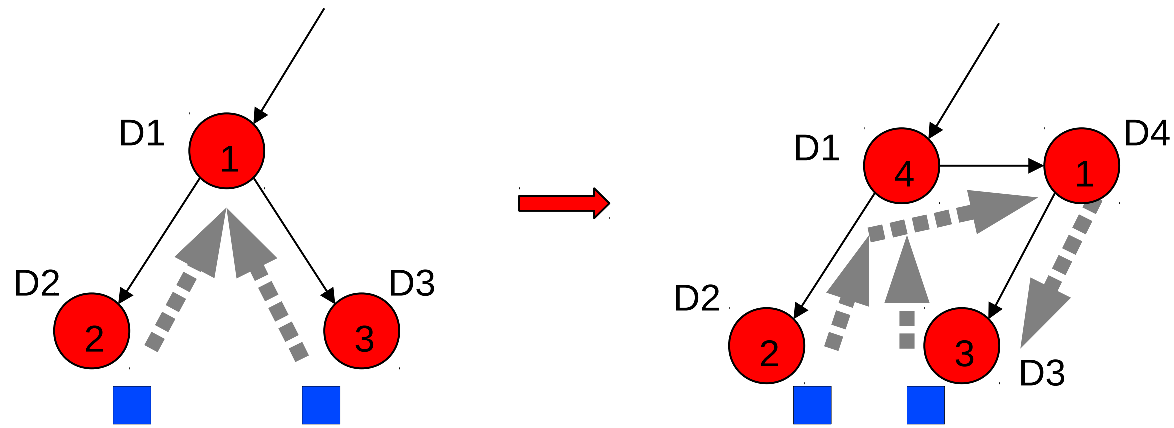


Parallel Insert & Query Operations

CASE:

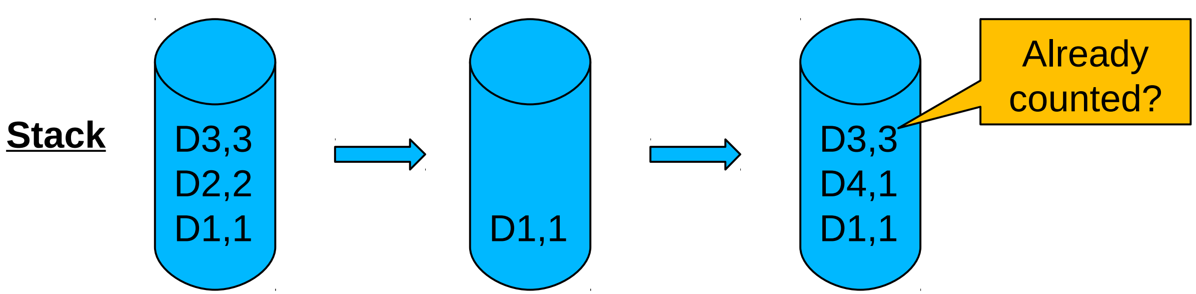
- Insert creates a directory node split
- Concurrent query returns back up the tree and finds tree structure changed.

Insert



New node gets old time stamp !

Query

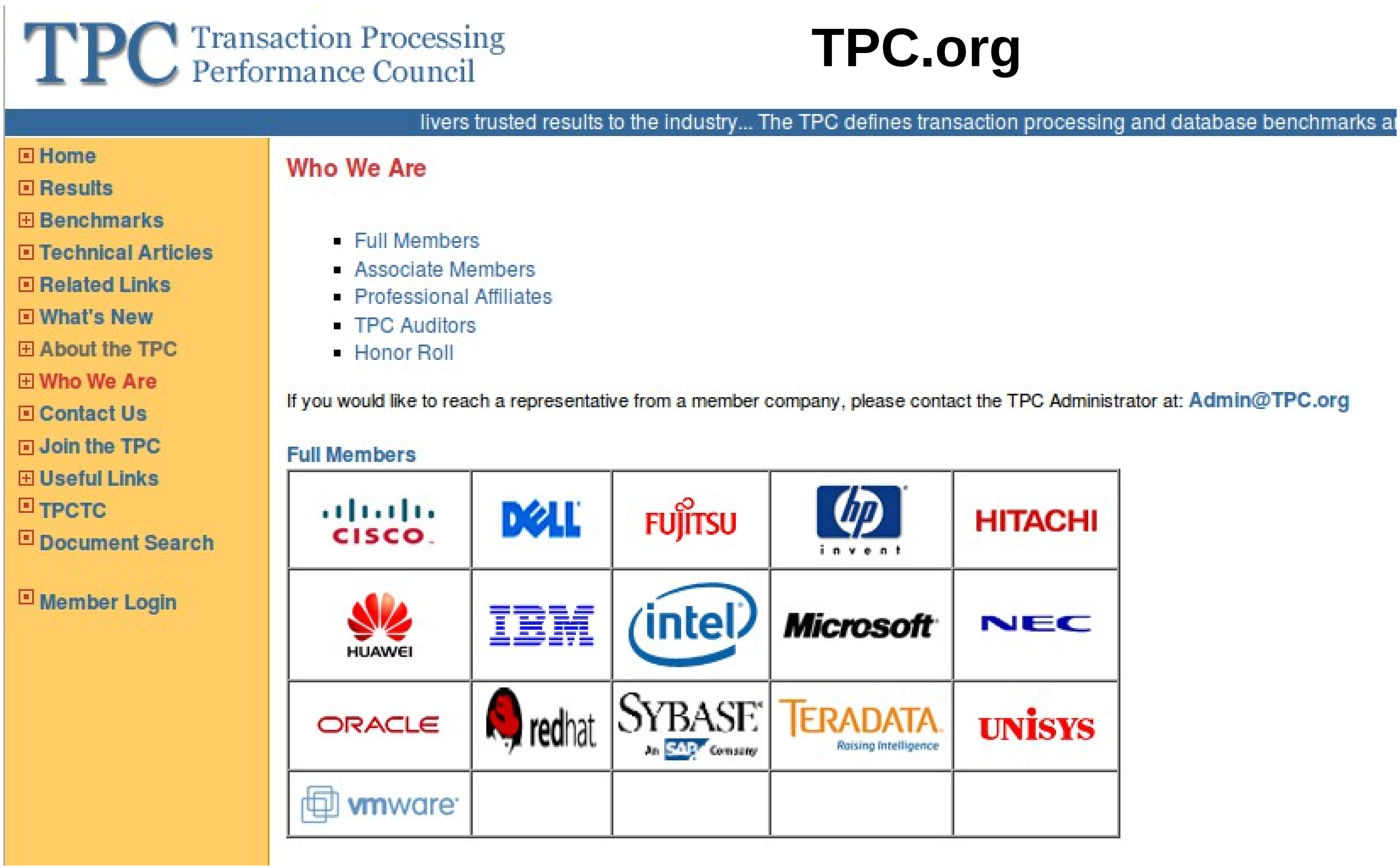


Parallel DC-Tree Performance

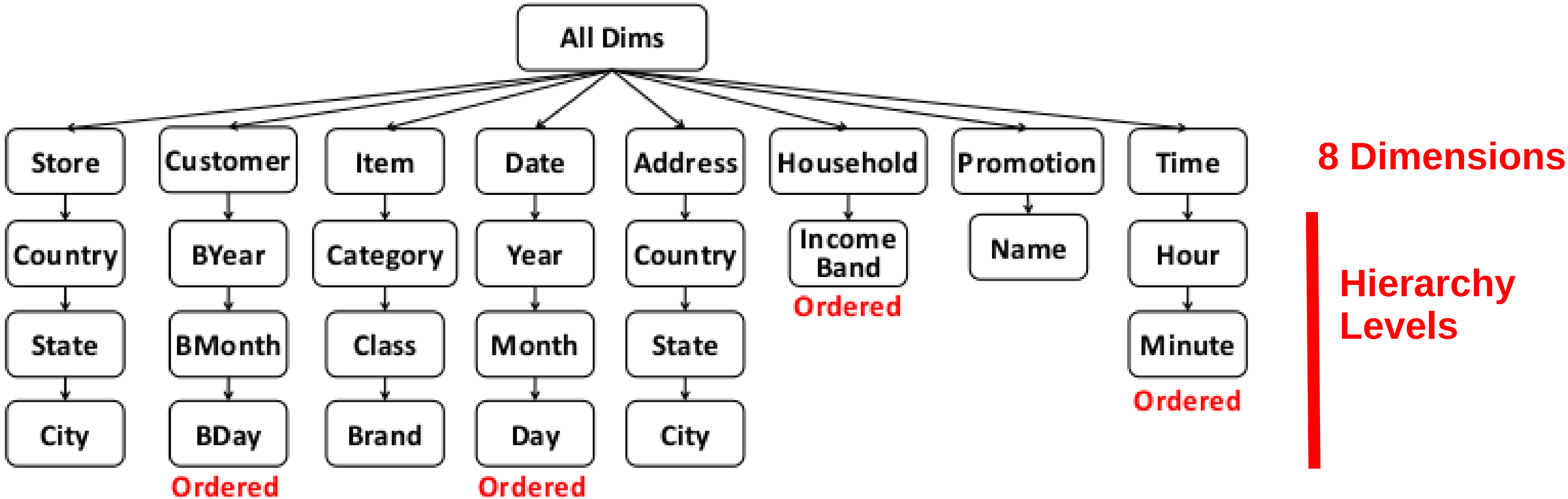
- Transaction Processing Performance Council TPC-DS (Decision Support) Benchmark.
- Two processor architectures:

1. Intel Sandy Bridge, **4 Cores**, 8 Hardware Threads (Hyperthreading), 16 GB Memory.
2. Intel Xeon Westmere EX (2 Sockets), **20 Cores**, 40 Hardware Threads (Hyperthreading), 256 GB Memory.

TPC-DS Benchmark



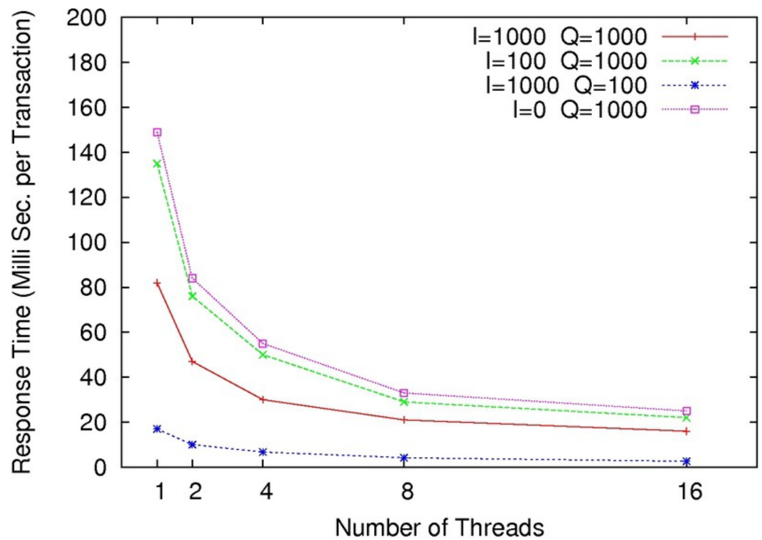
TPC-DS Benchmark

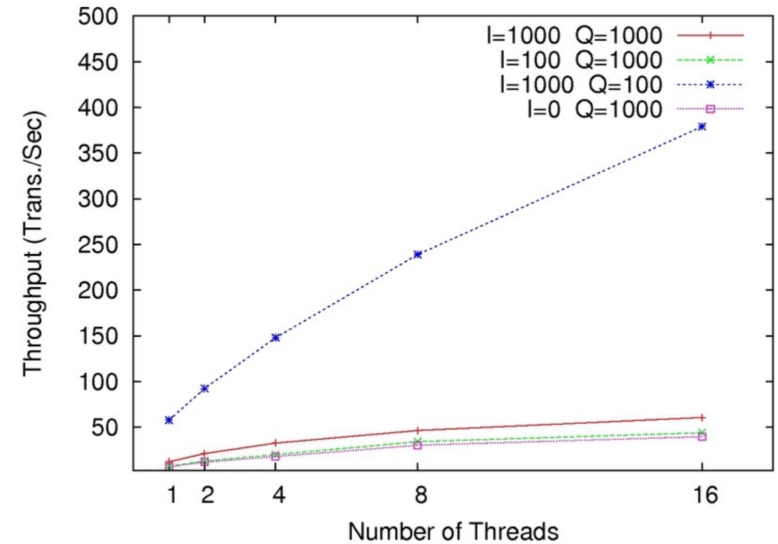


Intel Sandy Bridge, 4 Cores

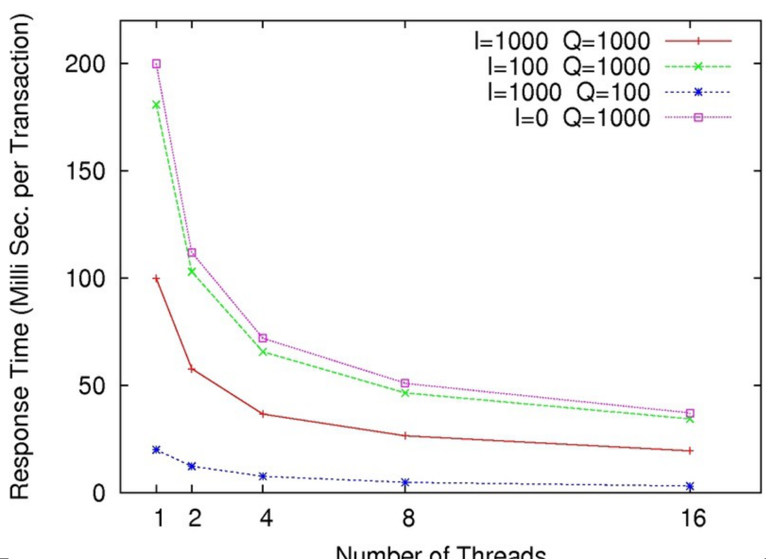
DB initialized with 400,000 records. I = # inserts and Q = # queries in the input stream.

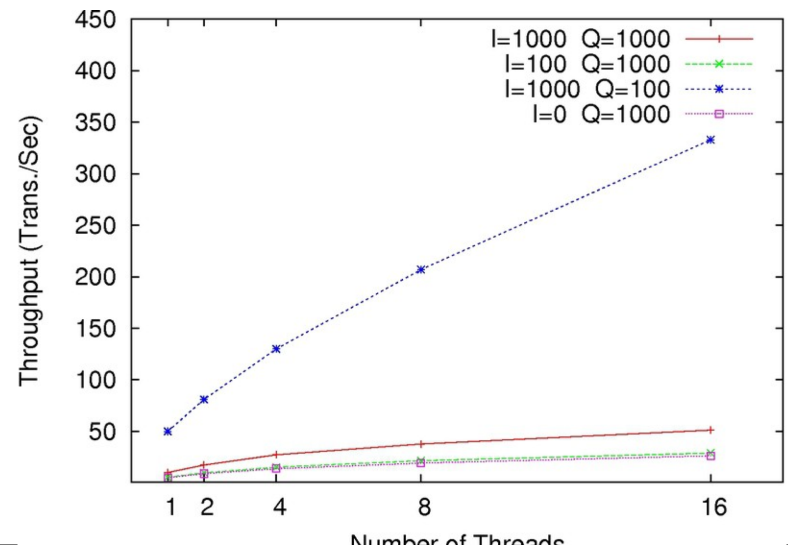
5% query coverage





25% query coverage



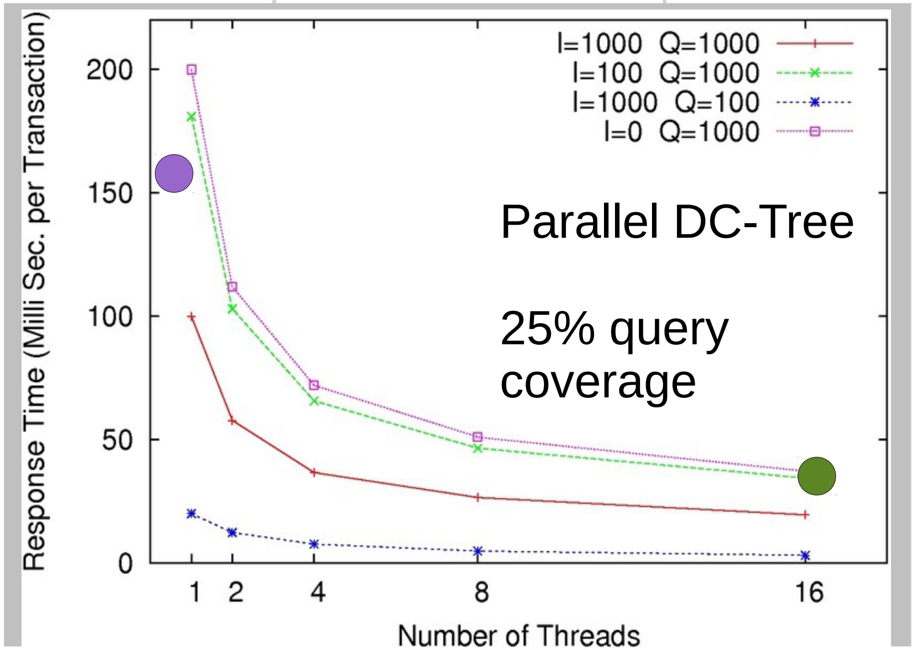


Intel Sandy Bridge, 4 Cores

Comparison with multi-threaded MySQL

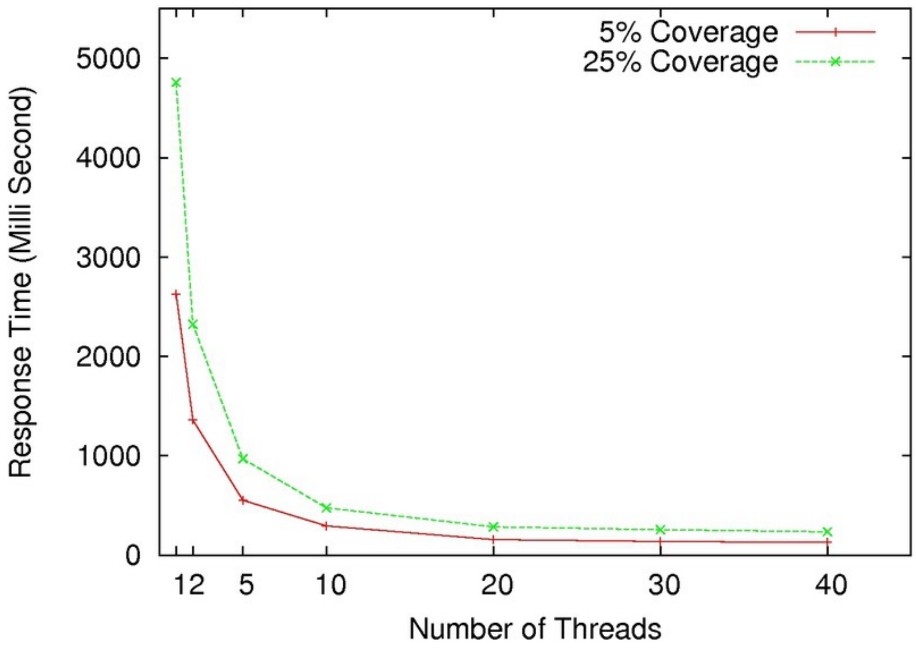
DB initialized with 400,000 records.
Stream of 1000 queries.

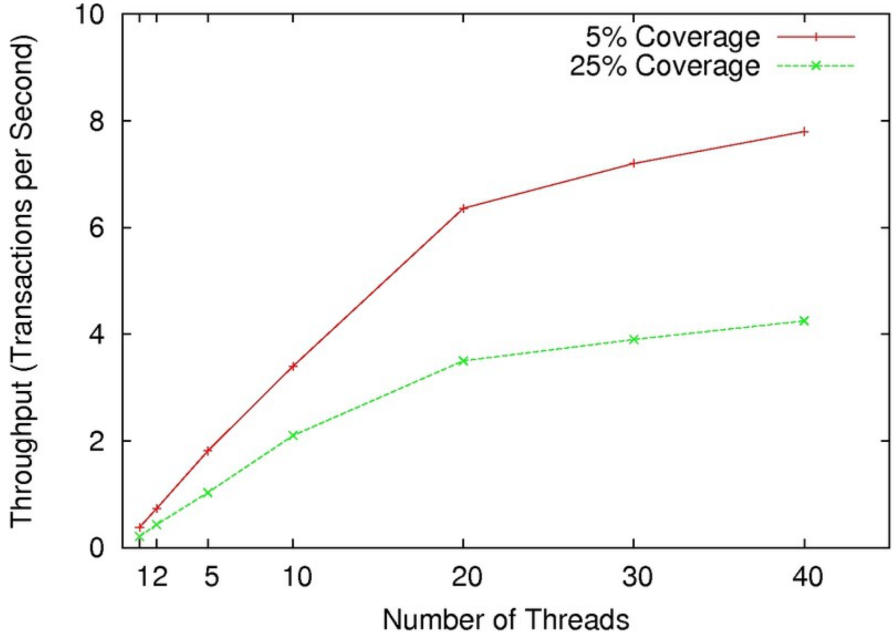
	1000 queries	1000 queries	1000 queries	1000 queries
	1% coverage	25% coverage	75% coverage	100% coverage
Parallel DC-Tree	4 sec.	38 sec	53 sec	5 sec.
MySQL multithreaded	5 sec.	161 sec.	245 sec.	415 sec.
(both using all cores)				



Intel Xeon Westmere EX, 20 Cores

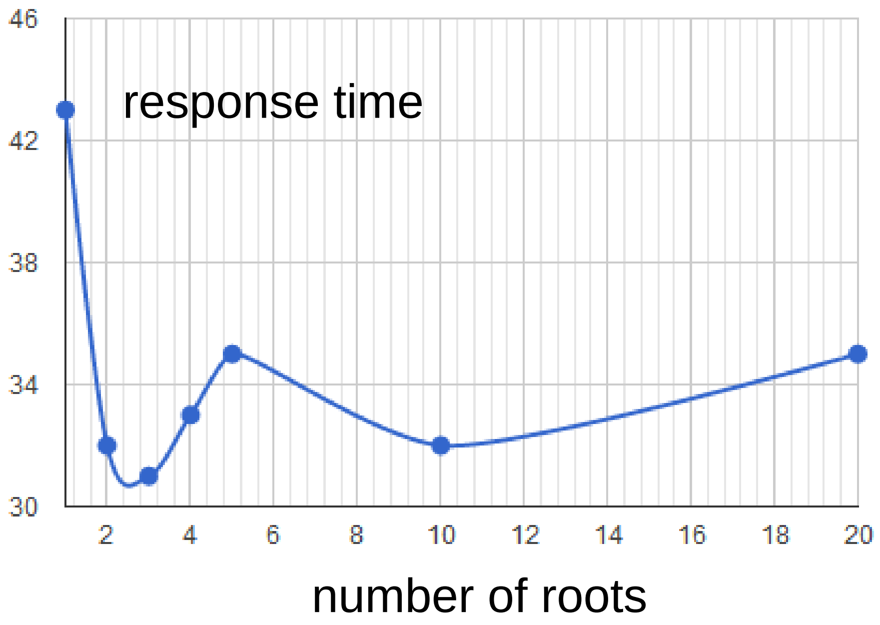
100 GB data set (10 Mil. Records)
10,000 queries
1,000 insertions

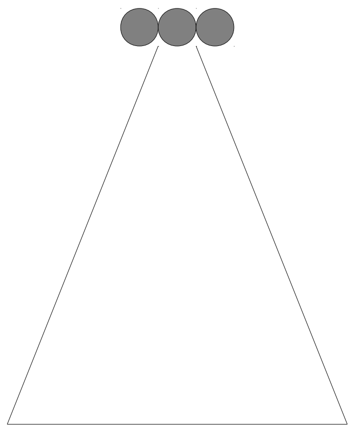




Intel Xeon Westmere EX, 20 Cores

Hotspot at root requires multiple root copies...





Intel Xeon Westmere EX, 20 Cores

100 GB data set (10 Mil. Records)
10,000 queries
1,000 insertions

Data	Thread	Total Query Time (Seconds) 25% Coverage	Total Query Time (Seconds) 5% Coverage	Speedup 25% coverage	Speedup 5% coverage
100G 10000 Queries 1000 Insertions Number of Data Nodes: 10000000 Number of Splits: 1029793 Number of Expansions: 13130732 Total PDC-Tree Building Time: 18829.6 Seconds. Total Number of Directory Nodes: 1266186	640	2526.6	1333.45	19.92662076	20.40923919
	320	2523.34	1334.62	19.95236472	20.39134735
	160	2508.33	1319.18	20.07176089	20.63001258
	80	2510.89	1328.64	20.05129655	20.4831256
	40	2529.48	1356.58	19.90393282	20.06125698
	20	3096.49	1619.84	16.25924837	16.80085687
	10	5536.33	3014.01	9.093858206	9.029399372
	5	10437.4	5697.09	4.823672562	4.776947529
	2	25920.6	14073.6	1.942339298	1.933741189
	1	50346.6	27214.7	1	1
	Average Visited Nodes Ratio	0.0442987	0.0089575		

Response time
5 sec. -> .25 sec.

Response time
2.7 sec. -> .13 sec.

Parallel real-time OLAP on multi-core processors

- **Published in *ACM/IEEE CCGrid 2012***

Parallel real-time OLAP on multi-core processors

- **Published in *ACM/IEEE CCGrid 2012***
- ***IBM Research Impact Of The Year Award***

Parallel real-time OLAP on multi-core processors

- **Published in *ACM/IEEE CCGrid 2012***
- ***IBM Research Impact Of The Year Award***
- ***IBM patent submission***
- ***IBM implemenation group for TM1***

Parallel real-time OLAP

- **Published in *ACM/IEEE CCGrid 2012***
- ***IBM Research Impact Of The Year Award***
- ***IBM patent submission***
- ***IBM implemenation group for TM1***

- **New 3 year funded project (2013-2016): scale up to cloud**
- **$1M hardware (private cloud at Carleton)**
- **Carleton/IBM Data Science Institute**

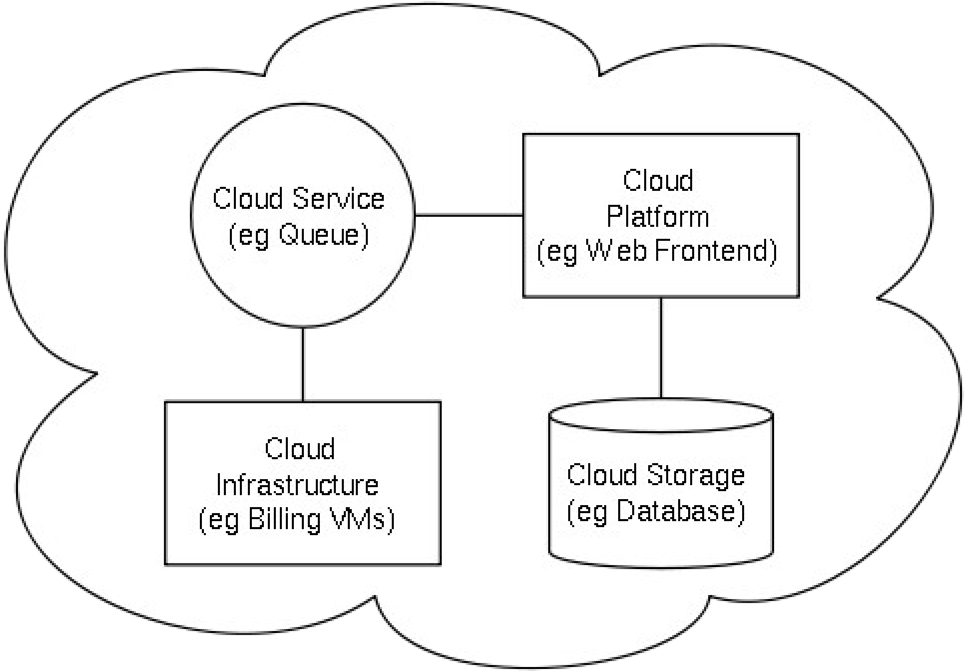
Parallel real-time OLAP on cloud architectures

Version 1

Presented at ***IEEE BigData 2013***

Cloud Computing Architecture

- Large scale compute cluster
- Virtual machines on demand
- Elastic: dynamic addition of compute resources
- Dedicated storage devices (e.g. S3 buckets)

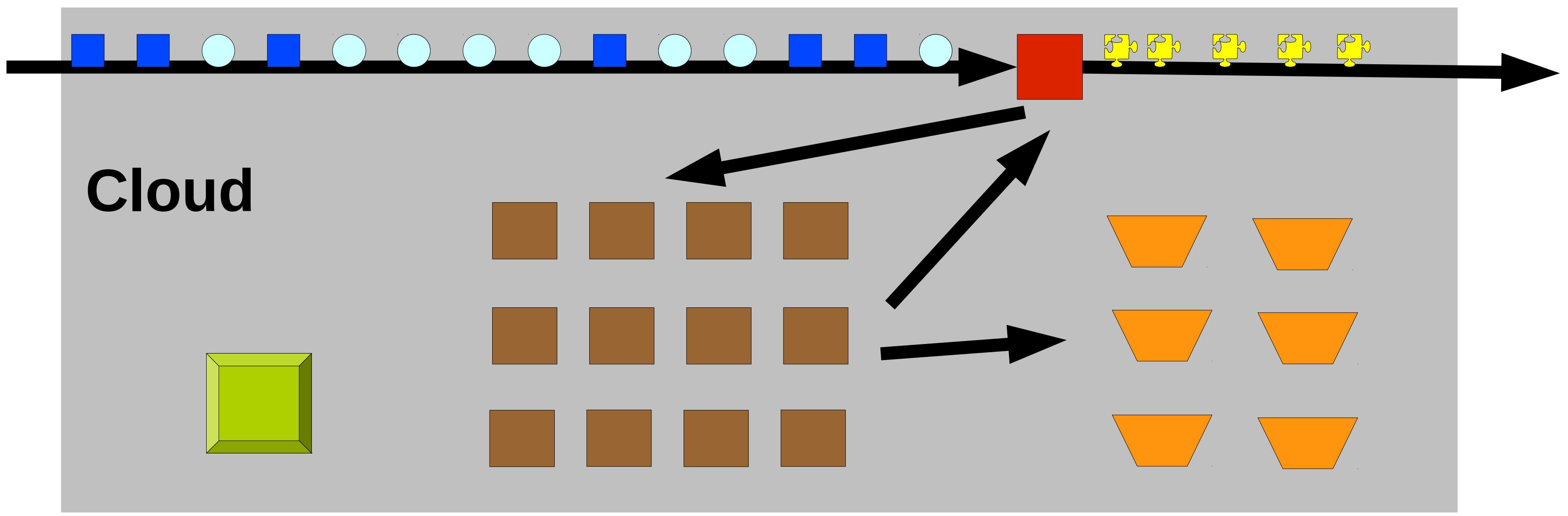


Real-Time OLAP “In The Cloud”

OLAP operations:

- insert
- query (aggregate range query)

Pointers to results (on S3)



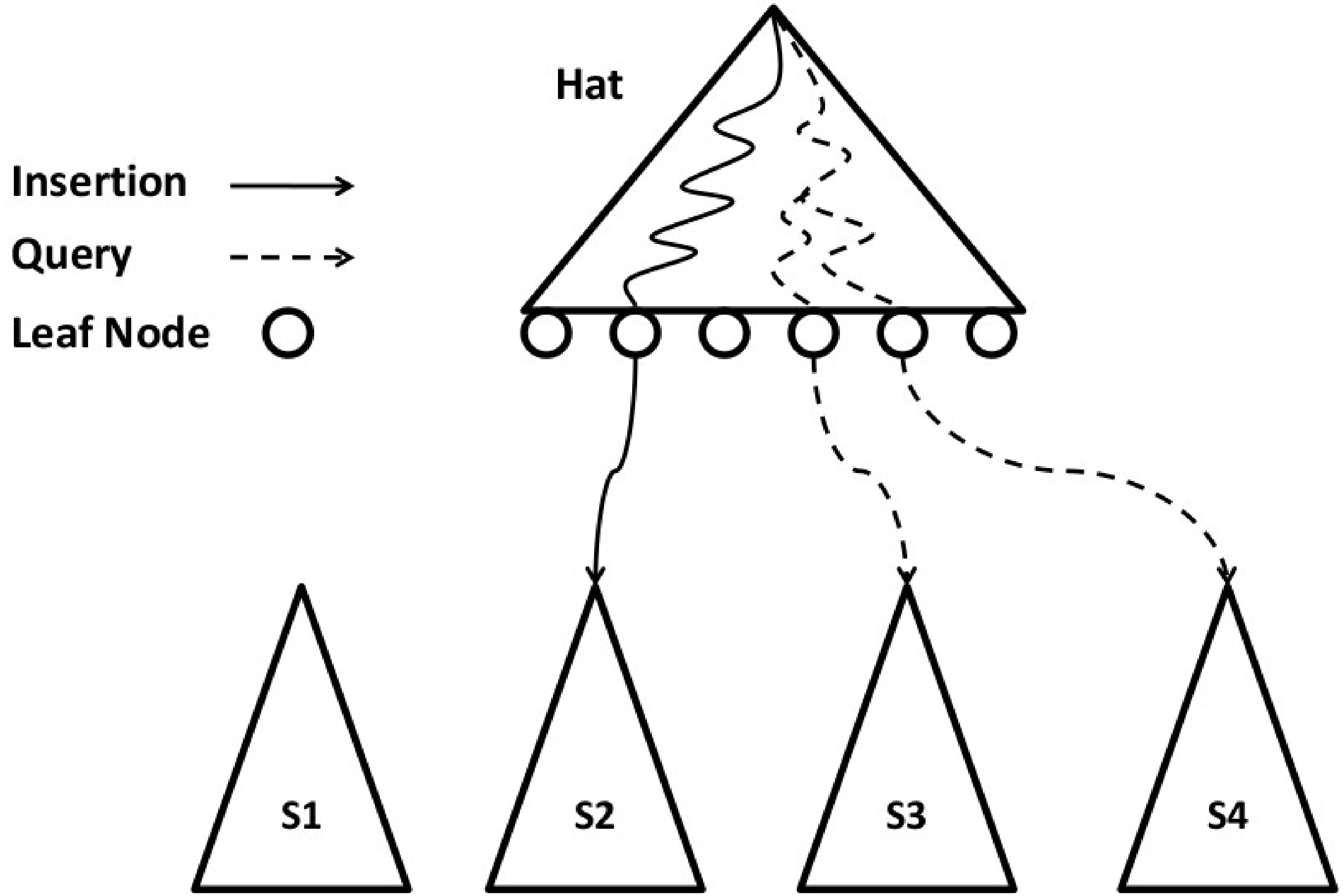
Master (multi-core)

Worker (multi-core)

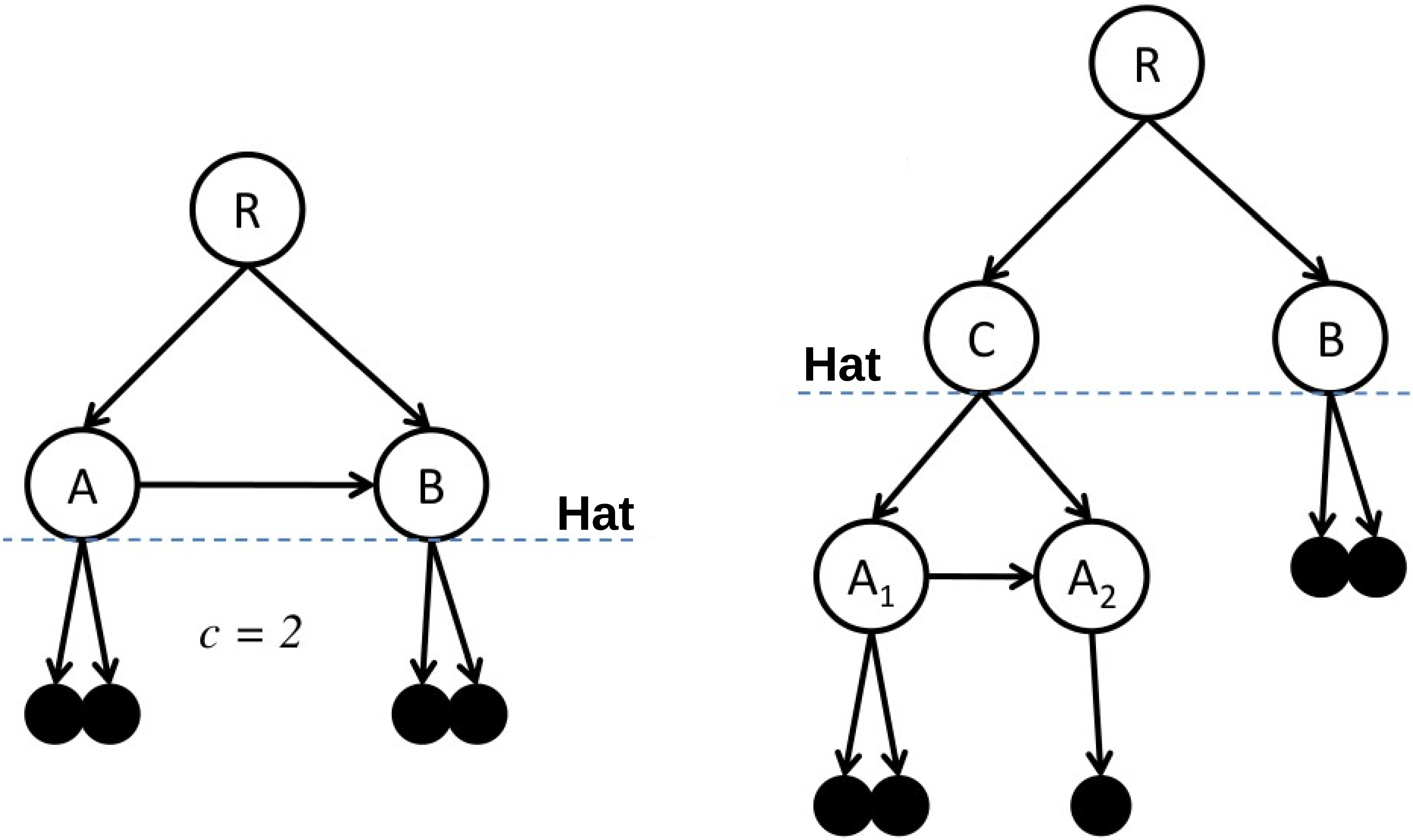
Storage device (S3)

Zookeeper

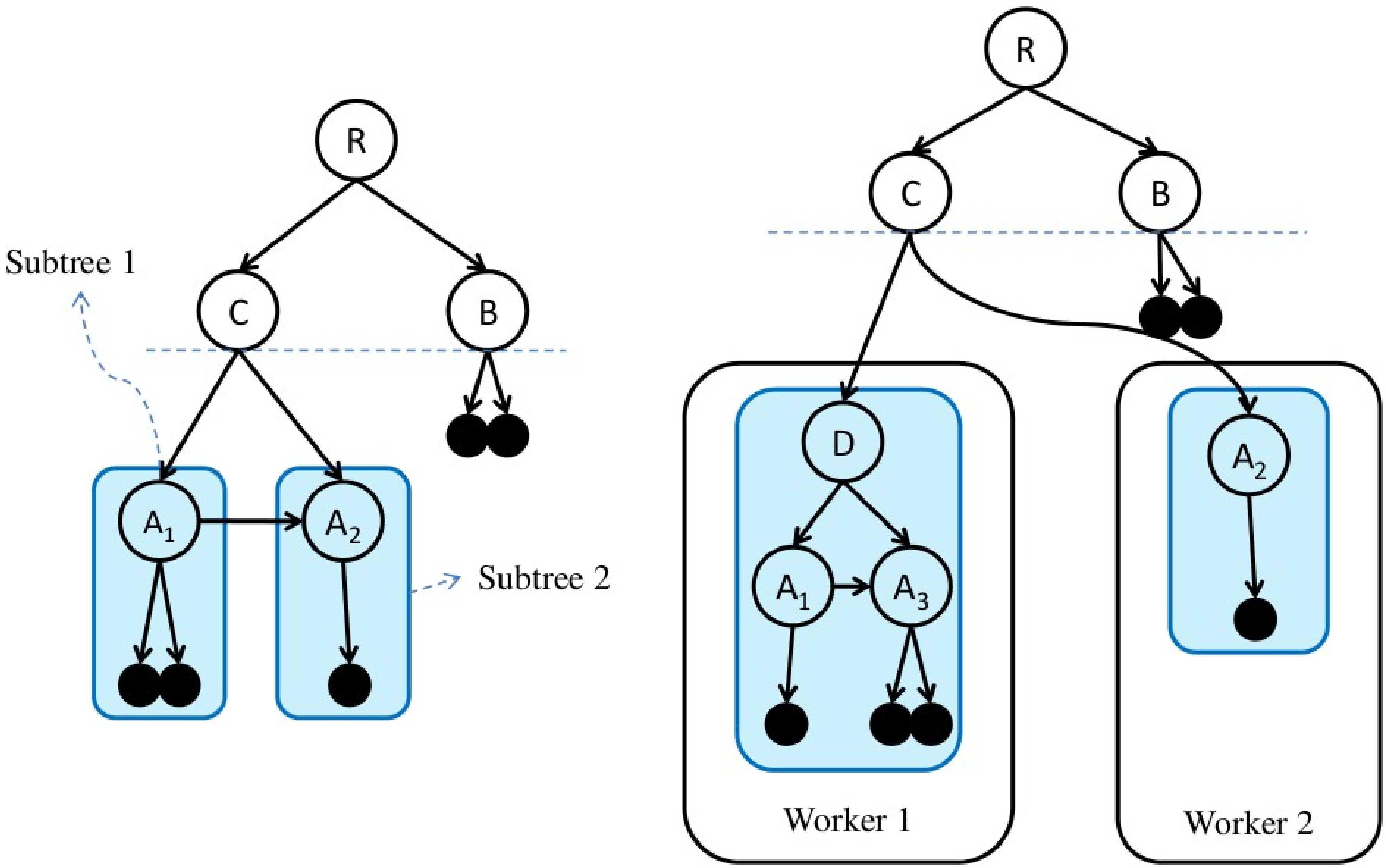
Real-Time OLAP "In The Cloud"



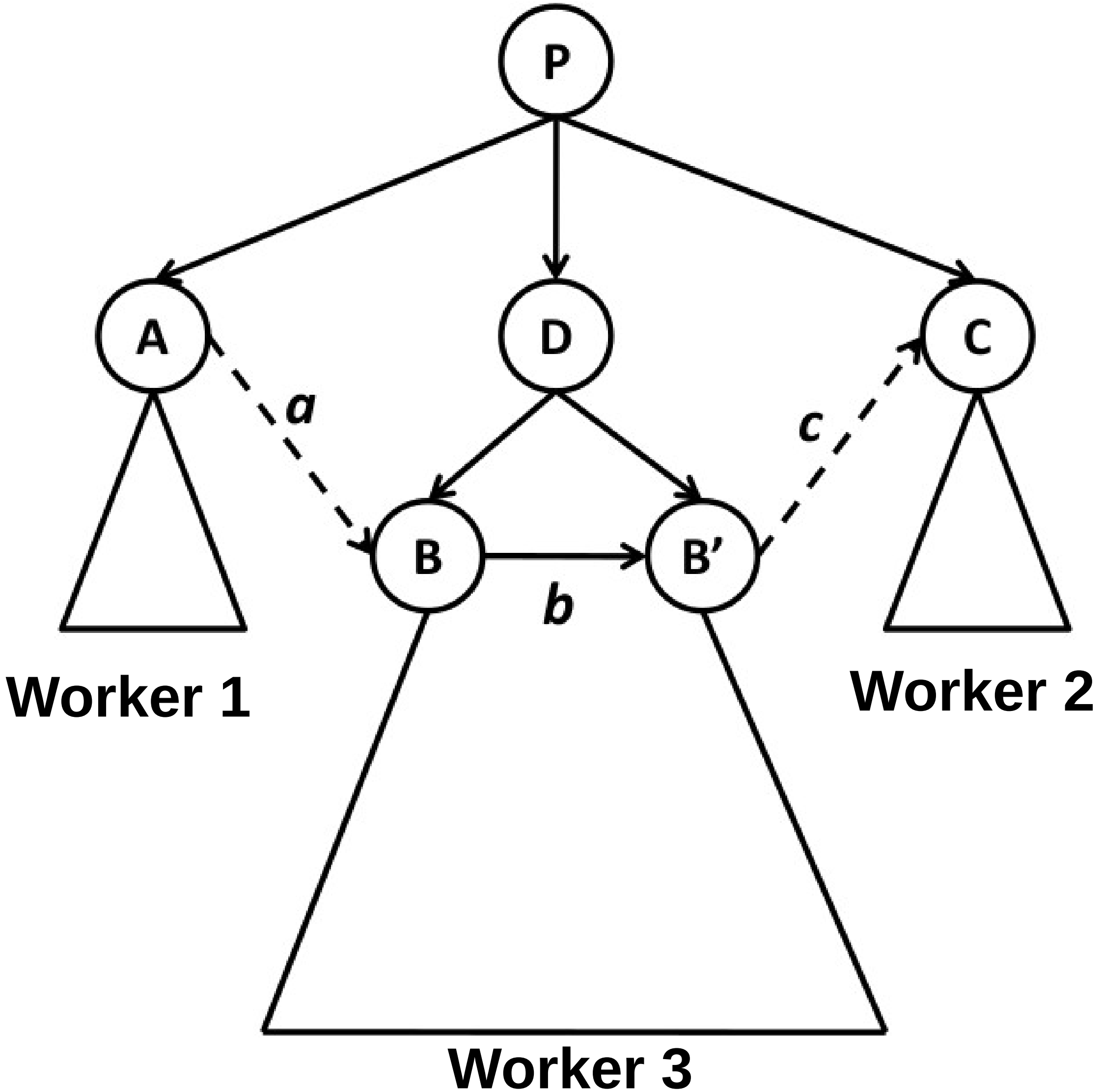
"Elastic" System Growth



“Elastic” System Growth



Correctness

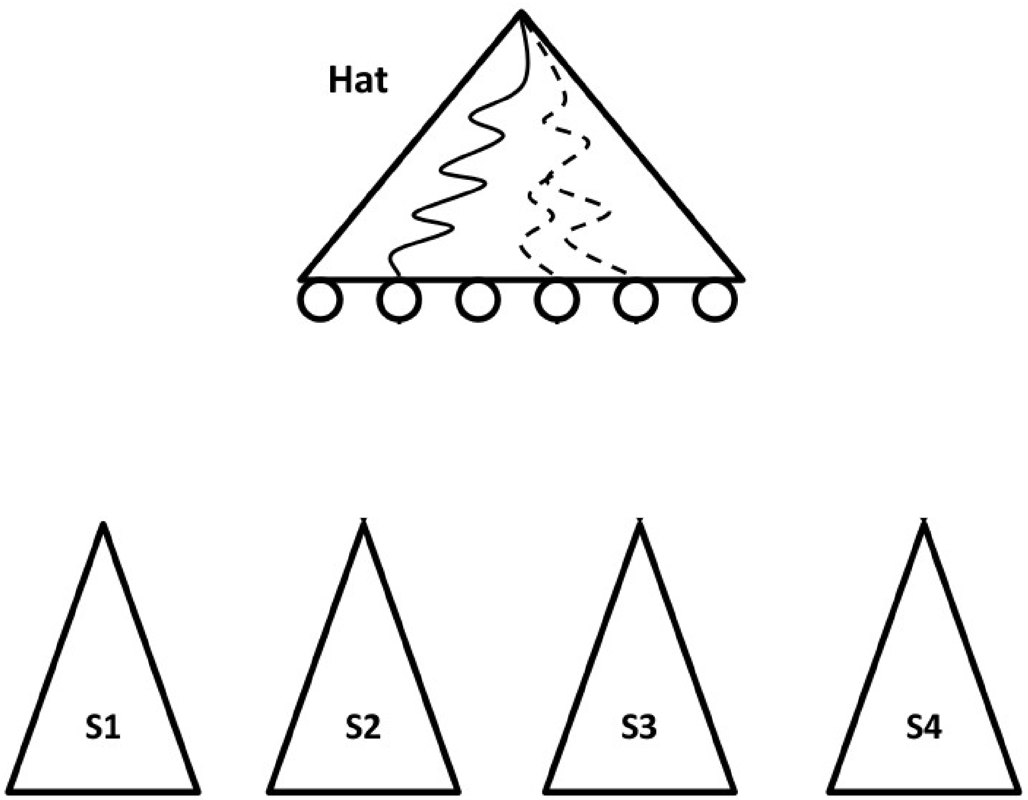


Theorem:

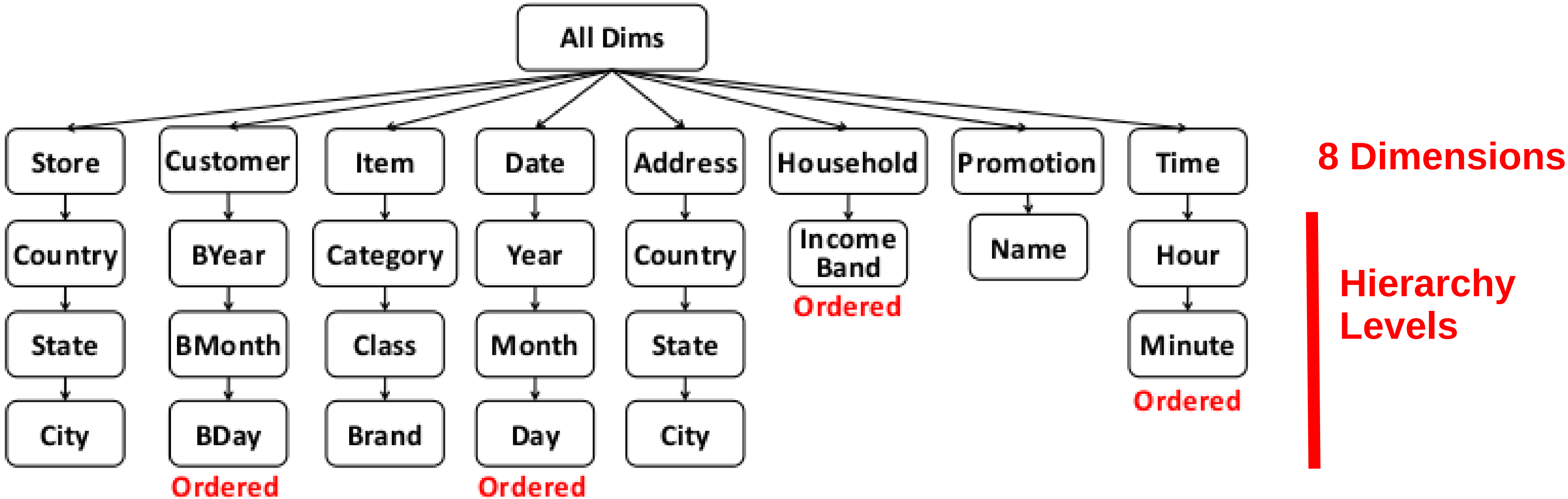
Horizontal links *a* and *c* between workers are <u>not</u> needed.

Load Balancing

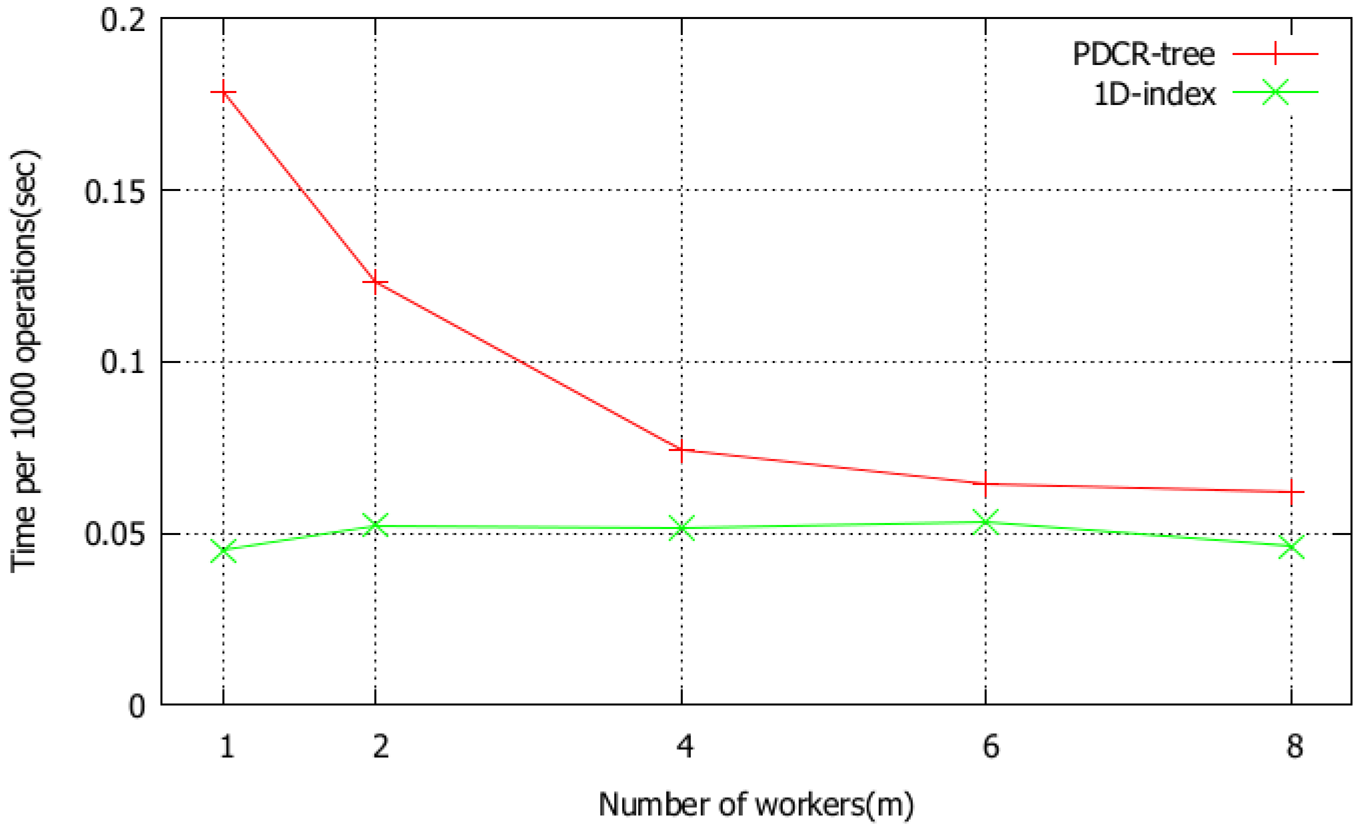
- Insertion/query load and memory usage
- More subtrees than workers
- Global statistics in Zookeeper
- Migrate subtrees
- Split subtrees



TPC-DS Benchmark



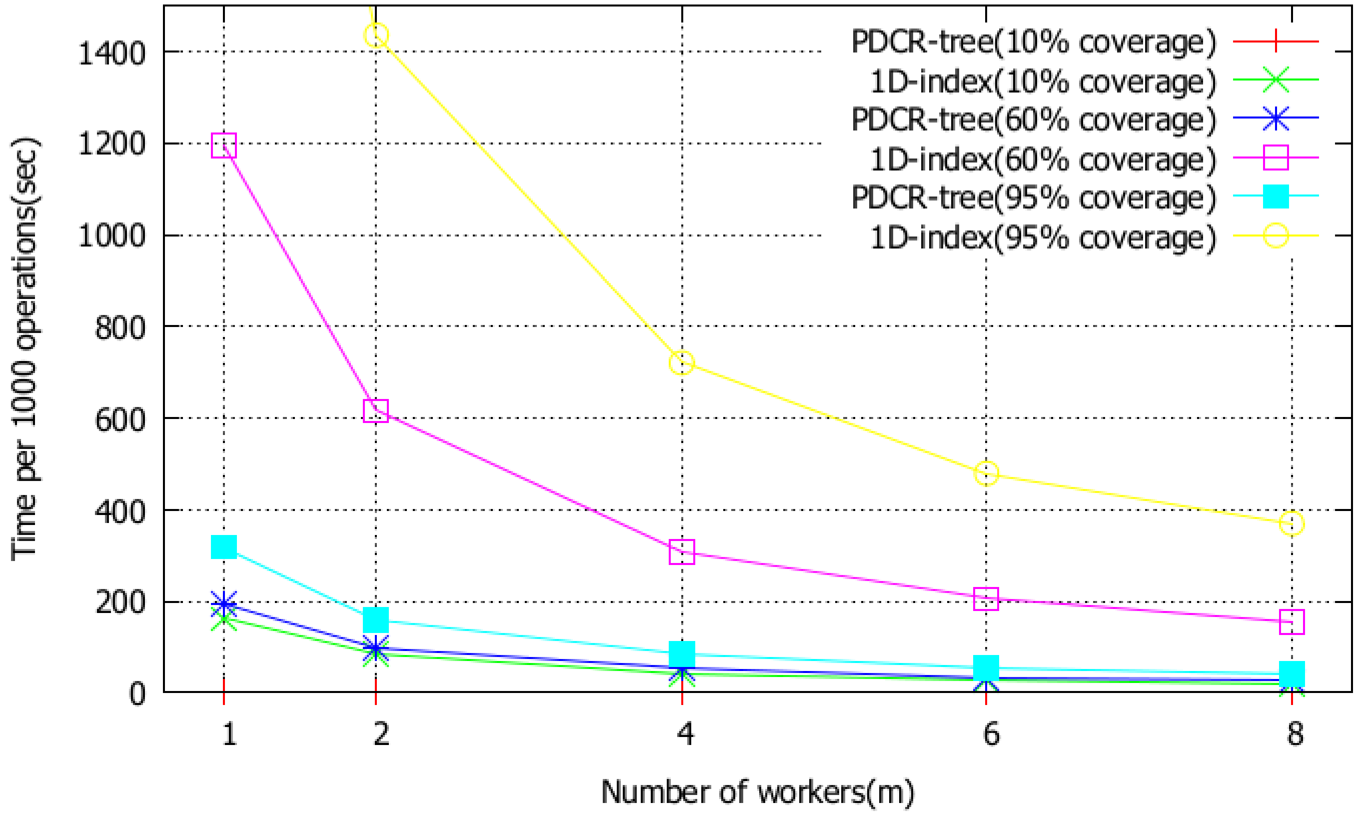
Performance



Time for 1000 insertions as a function of the number of workers. ($N = 40Mil$, $d = 8$, $1 \leq m \leq 8$)

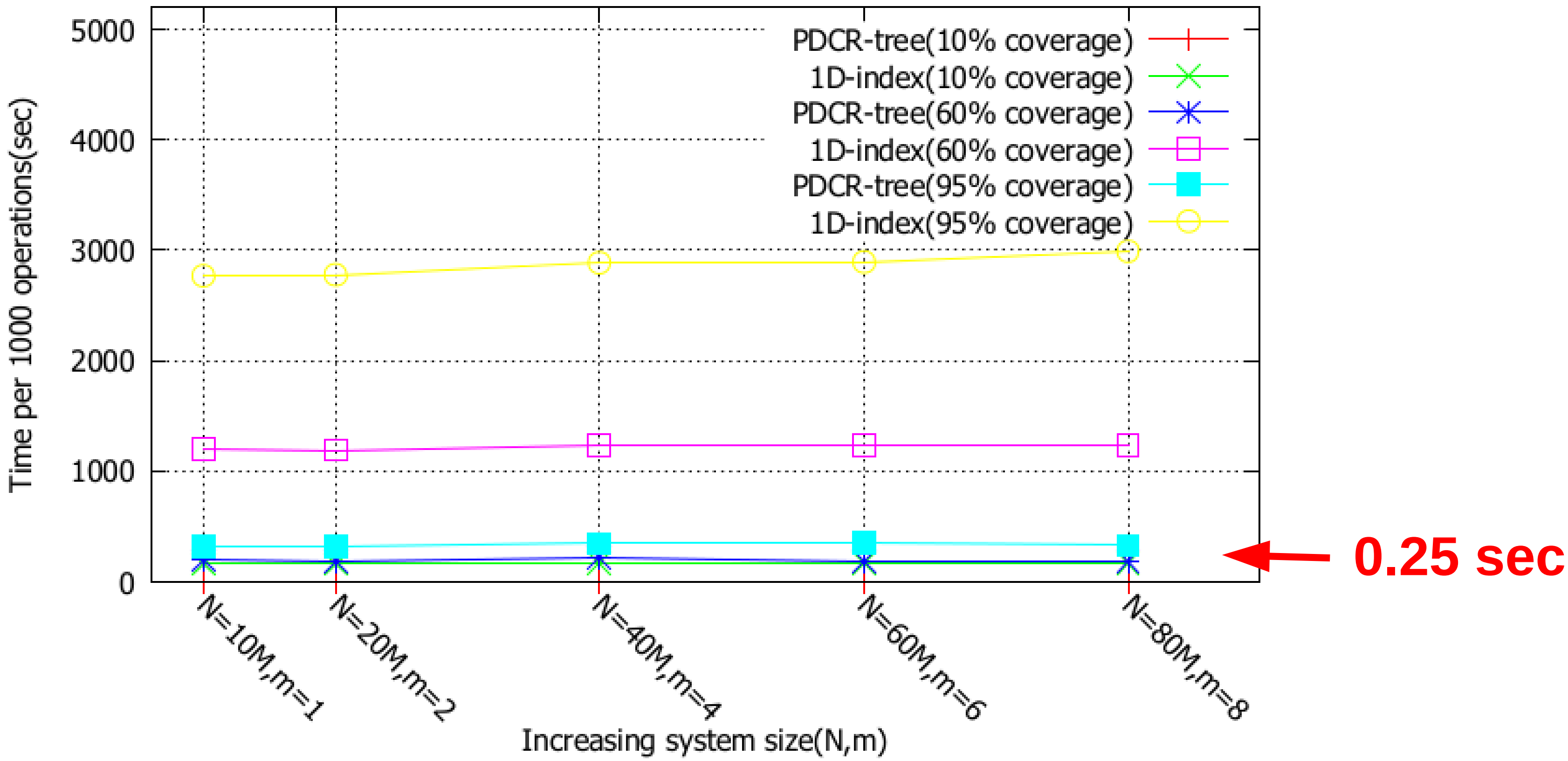
N: database size, d: # dimensions, m: # workers

Performance



Time for 1000 queries as a function of the number of workers. ($N = 40Mil$, $d = 8$, $1 \leq m \leq 8$)

Performance



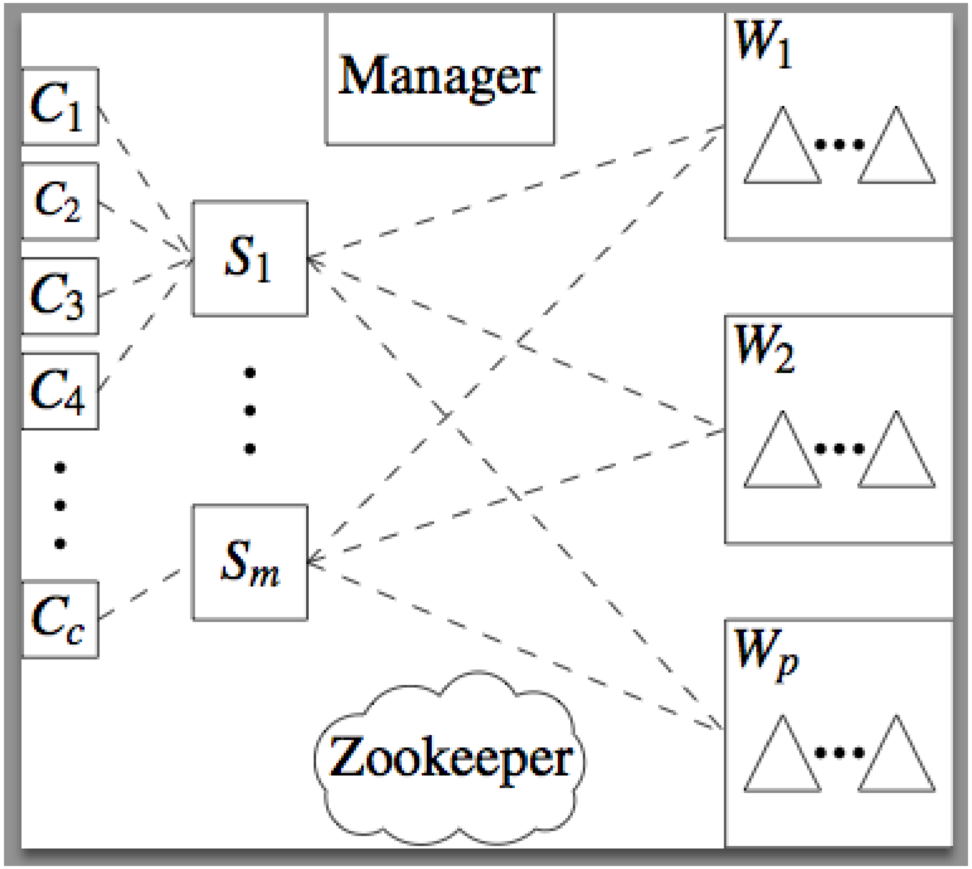
Time for 1000 queries as a function of system size: N & m combined. ($10Mil \leq N \leq 80Mil$, $d = 8$, $1 \leq m \leq 8$)

Parallel real-time OLAP on cloud architectures

Version 2

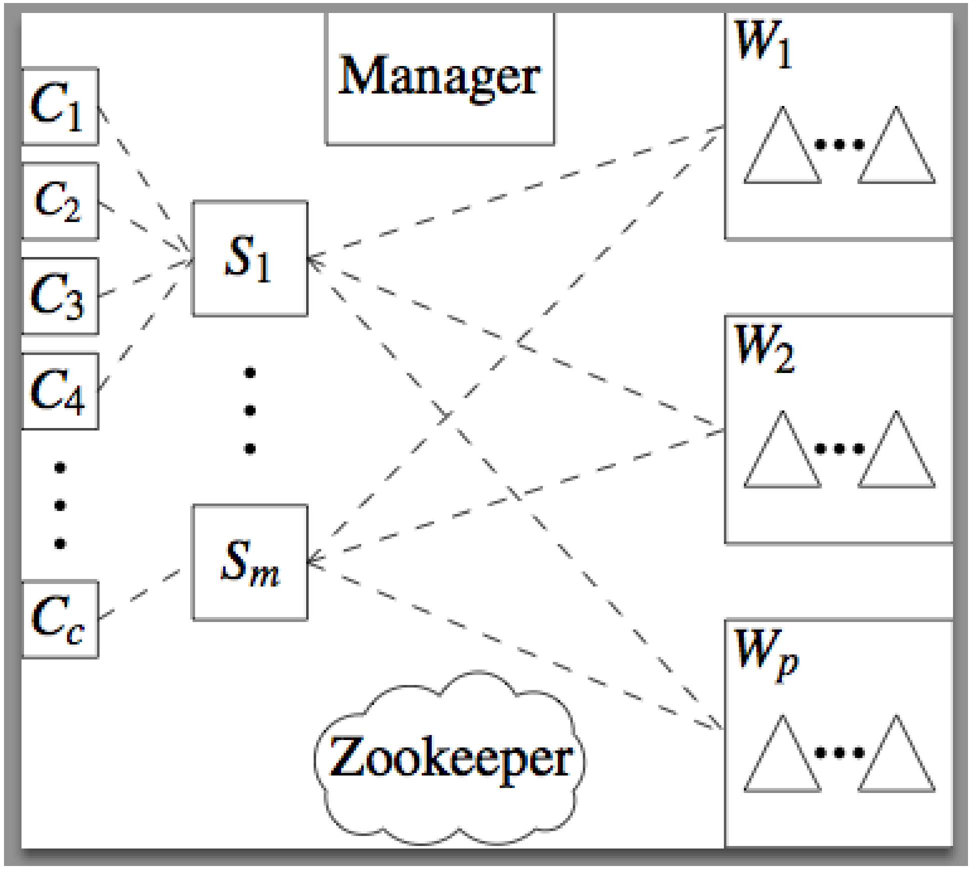
Fully Distributed (no “Master” processor)

System Overview



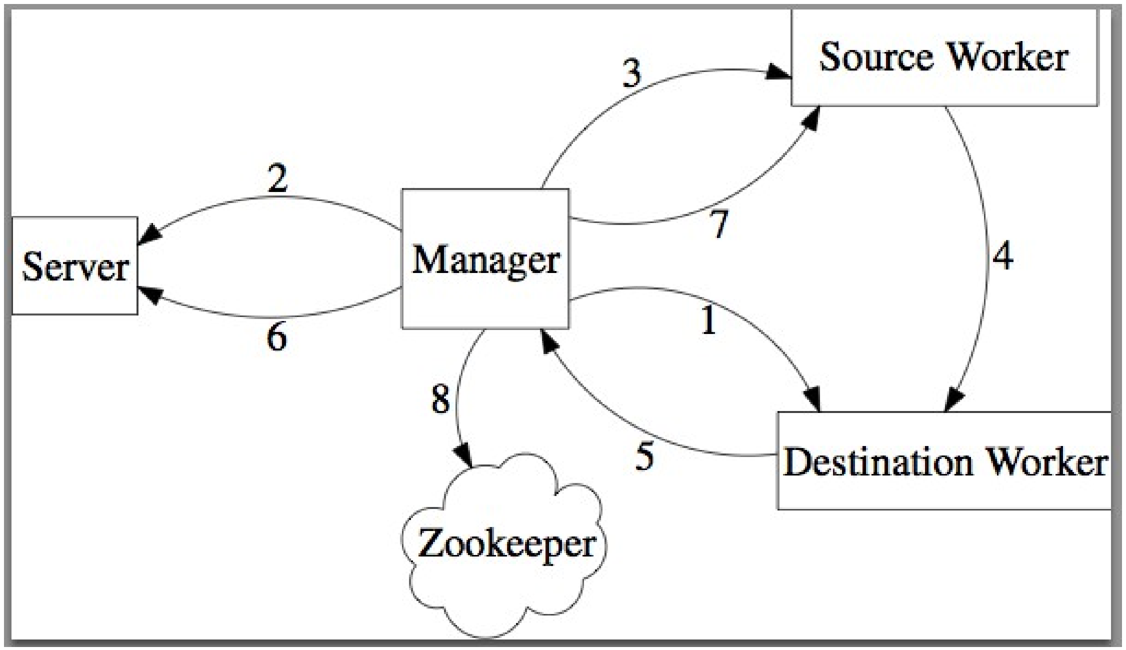
- Multiple servers
- Multiple workers
- Each worker stores multiple PDC trees
- Each server stores a local "system image" (PDC tree hat)
- Zookeper stores a global "system image" (PDC tree hat) and worker load statistics

System Overview



- Global system image stored at Zookeeper
- Servers store local system image
- Local system images get pushed and aggregated into Zookeeper
- Zookeper returns new global systems image to servers
- Strong serialization among user sessions connected to the same server (workgroup).
- "Best effort" serialization between user sessions on different servers (typical freshness bound <= 8 seconds; worst case freshness bound <= 15 seconds)

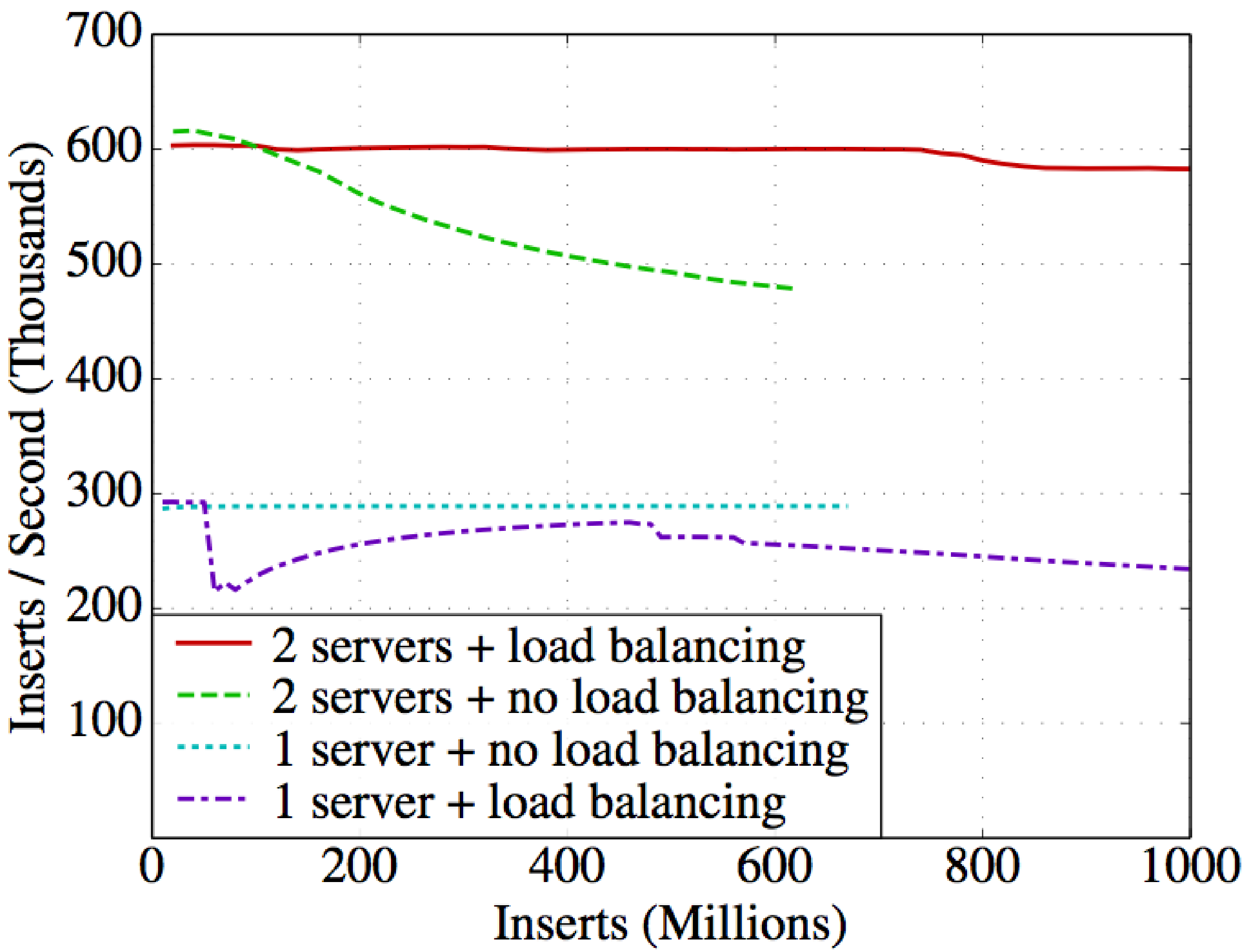
Load Balancing



- Manager queries Zookeeper and examines worker load statistics
- Manager initiates load balancing operations between workers
- Concurrent with Insert/Query operations

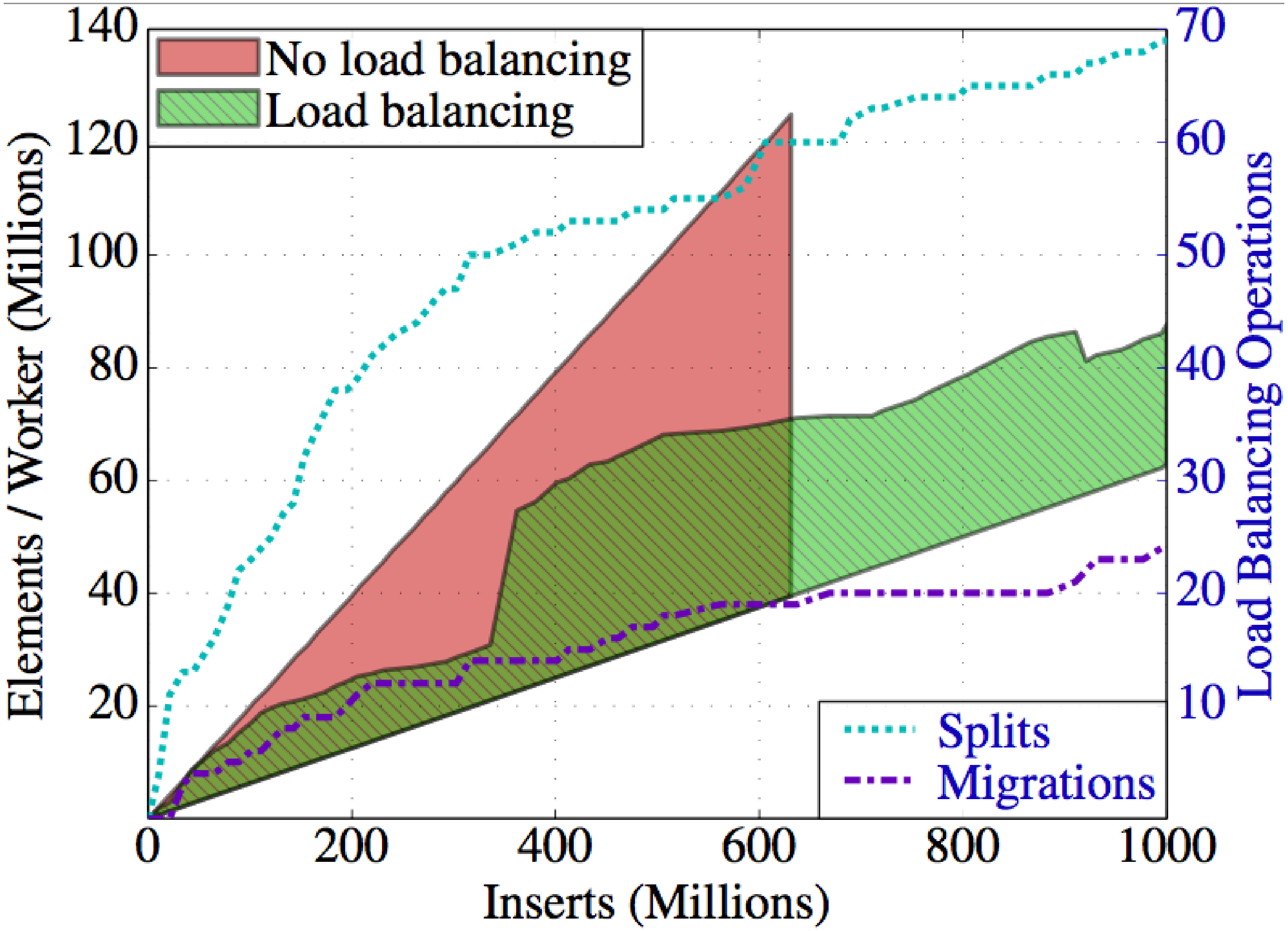
- Manager is NOT involved in Insert/Query operations

Data Ingestion Performance



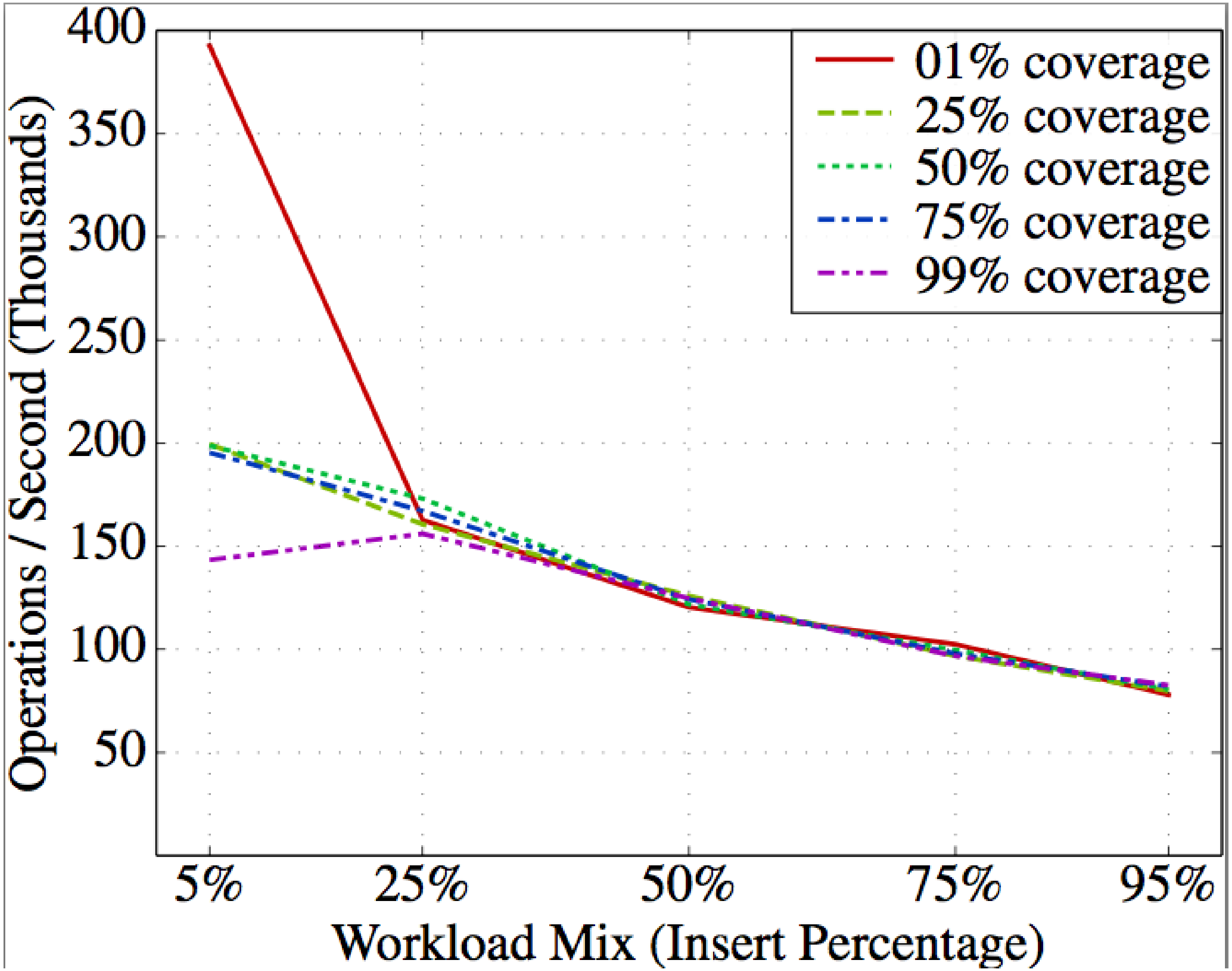
16 workers

Load Balancing



16 workers

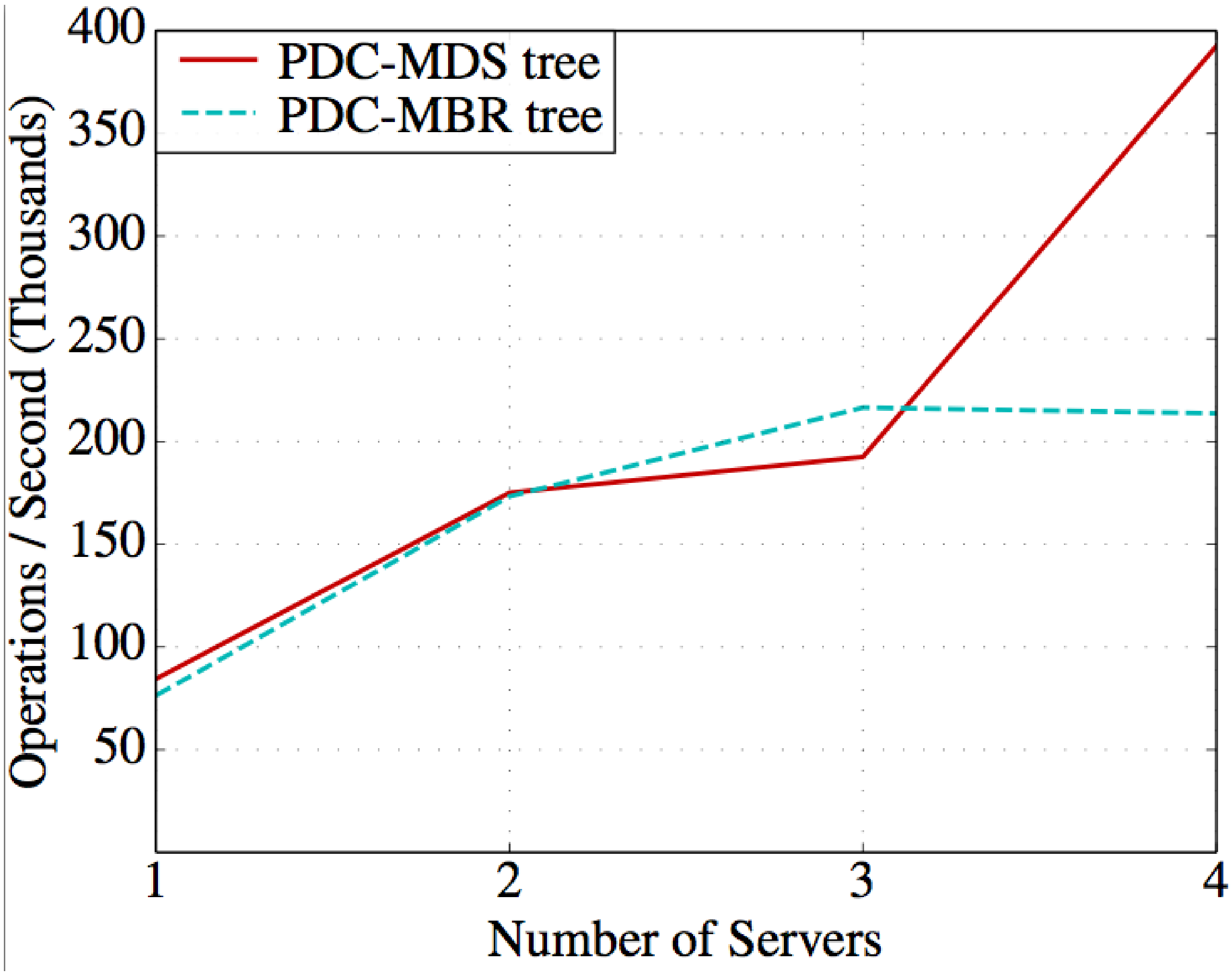
Real Time OLAP: Insert/Query Stream



Database size: 400 M

16 workers
4 servers

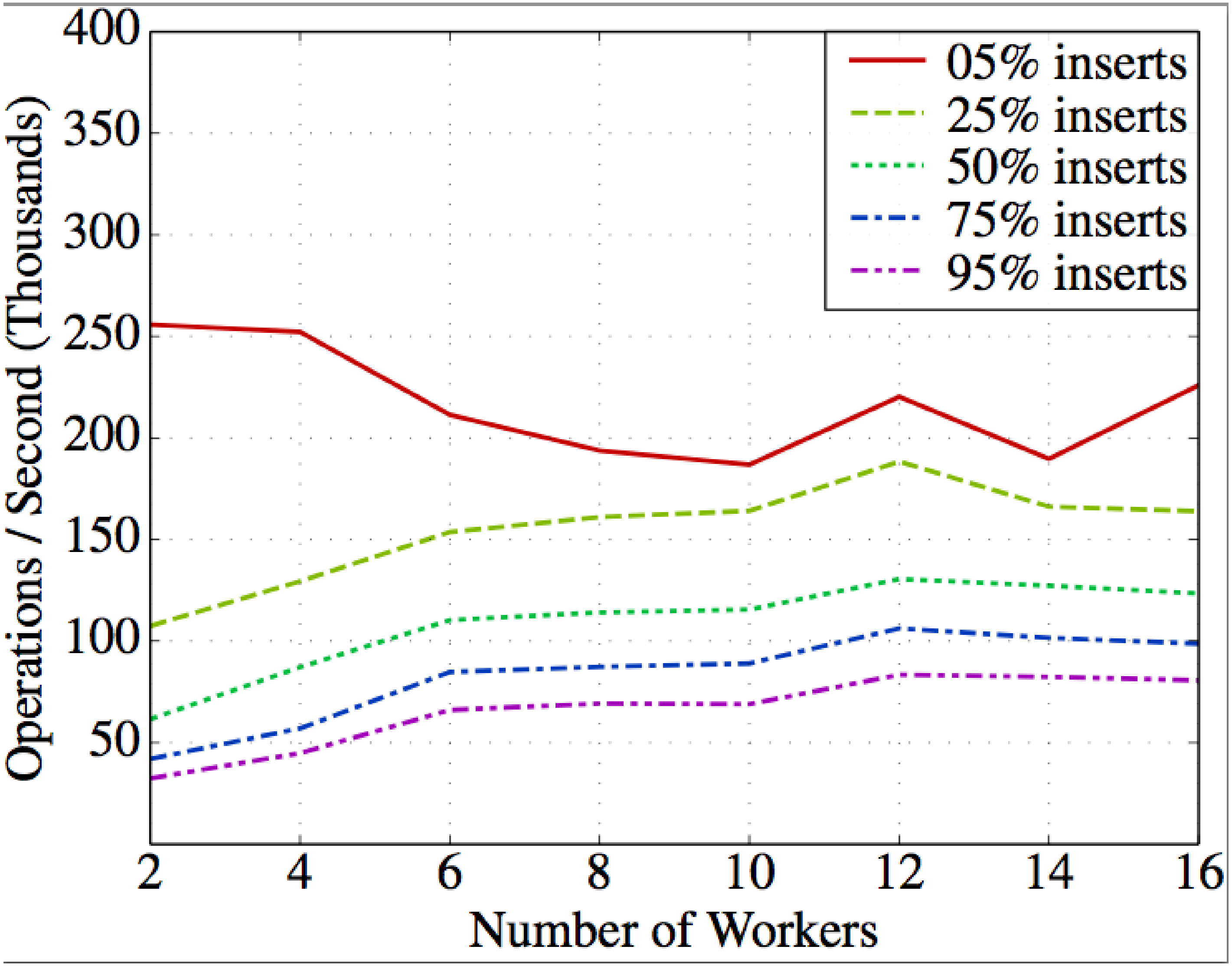
Impact Of Number Of Servers



Database size: 400 M

16 workers

System Scale-UP



Data size and
#workers are
Both increasing

25 M data items
Per worker

4 servers

average over all
query coverages
(5% - 95%)

Conclusion

Parallel data structures can enable real-time OLAP on multi-core and cloud architectures.