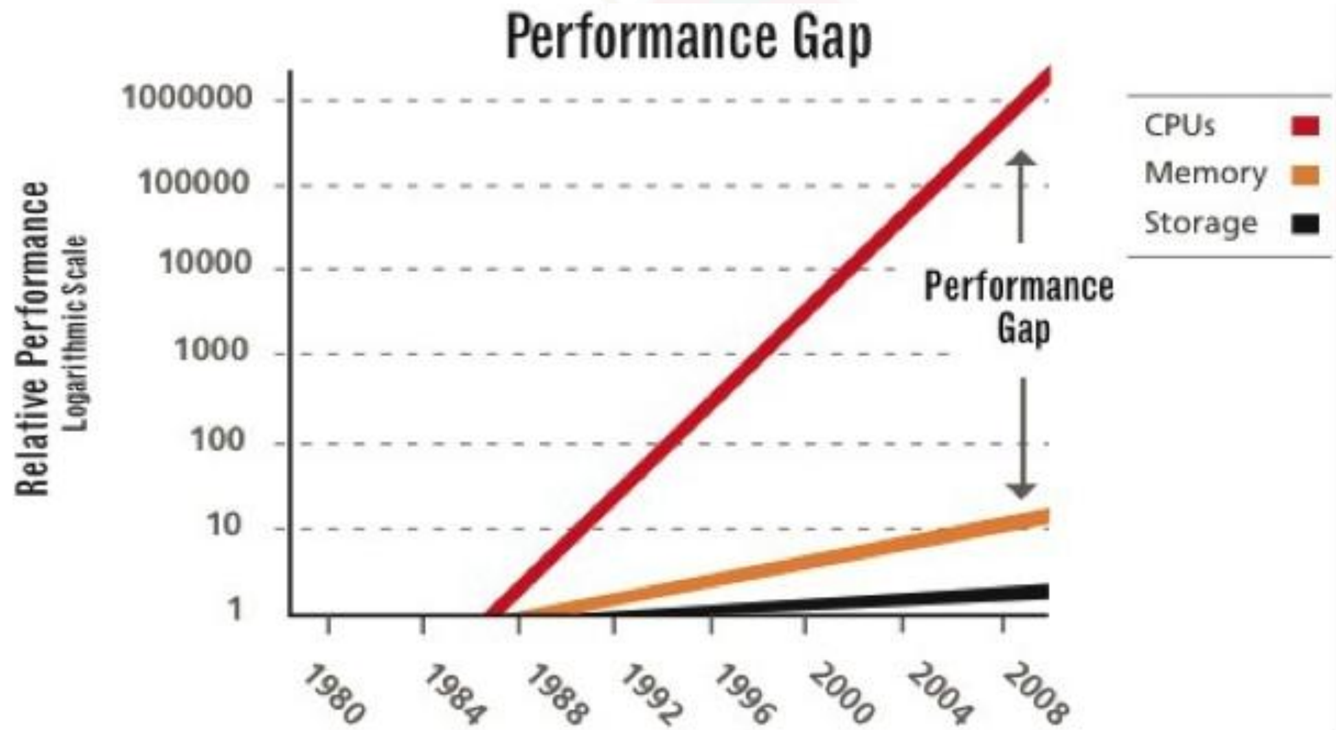
In-Situ Bitmaps Generation and Efficient Data Analysis based on Bitmaps

Yu Su, Yi Wang, Gagan Agrawal
The Ohio State University

Motivation – HPC Trends

- Huge performance gap
  - ✓ CPU: extremely fast for generating data
  - ✓ Disk, Network: very slow to store or transfer data
  - ✓ Memory: not large enough to hold data



In-Situ Analysis – What and Why

- Process of transforming data at run time
  - Analysis
  - Classification
  - Reduction
  - Visualization
- In-Situ has the promise of
  - Saving more information dense data
  - Saving I/O or network transfer time
  - Saving disk space
  - Saving time in analysis

Key Questions

- How do we decide what data to save?
  - This analysis cannot take too much time/memory
  - Simulations already consume most available memory
  - Scientists cannot accept much slowdown for analytics
- How insights can be obtained in-situ?
  - Must be memory and time efficient
- What representation to use for data stored in disks?
  - Effective analysis/visualization
  - Disk/Network Efficient

Quick Answers

- How do we decide what data to save?
  - Use Bitmaps!
- How insights can be obtained in-situ?
  - Use Bitmaps!!
- What representation to use for data stored on disks?
  - Bitmaps!!!

Specific Issues

- Bitmaps as data summarization
  - Utilize extra computer power for data reduction
  - Save memory usage, disk I/O and network transfer time
- In-Situ Data Reduction
  - In-Situ generate bitmaps
    - ✓ Bitmaps generation is time-consuming
    - ✓ Bitmaps before compression has big memory cost
- In-Situ Data Analysis
  - Time steps selection
    - ✓ Can bitmaps support time step selection?
    - ✓ Efficiency of time step selection using bitmaps
- Offline Analysis:
  - Only keep bitmaps instead of data
  - Types of analysis supported by bitmaps

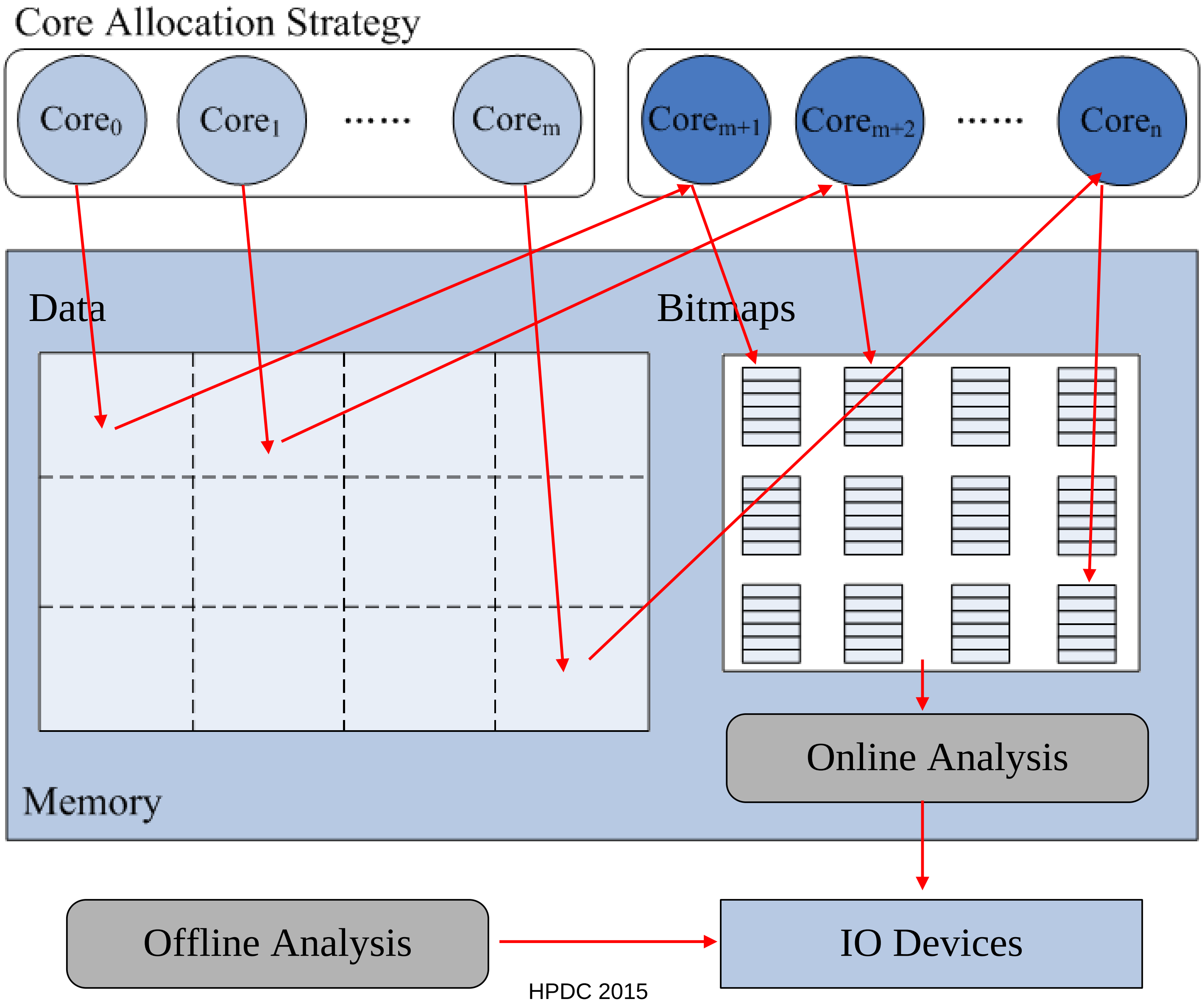
Background: Bitmaps

- Widely used in scientific data management

ID	Dimension		Value	v0	v1	v0	v1	v2	v3
	D0	D1		[1, 2]	[3, 4]	= 1	= 2	= 3	= 4
0	0	0	4	0	1	0	0	0	1
1	0	1	1	1	0	1	0	0	0
2	0	2	2	1	0	0	1	0	0
3	0	3	2	1	0	0	1	0	0
4	1	0	3	0	1	0	0	1	0
5	1	1	4	0	1	0	0	0	1
6	1	2	3	0	1	0	0	1	0
7	1	3	1	1	0	1	0	0	0
Dataset				1st Level Indices		2nd Level Indices			

- Suitable for floating value by binning small ranges
- Run Length Compression (WAH, BBC)
- Bitmaps can be treated as a small profile of the data

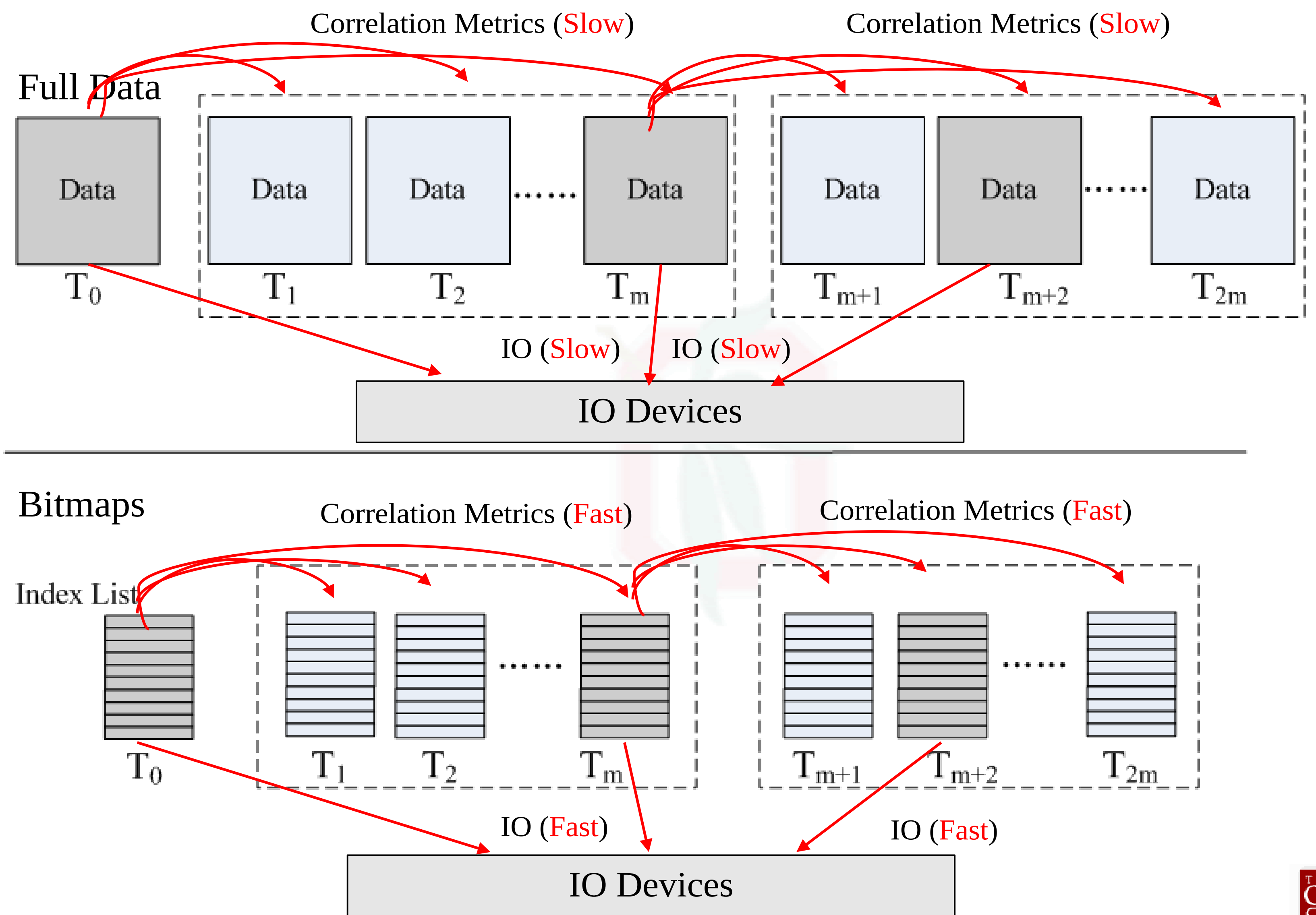
In-Situ Bitmaps Generation



In-Situ Bitmaps Generation

- Parallel index generation
  - Save the data loading cost
  - Multi-Core based index generation
- Core allocation strategies
  - Shared Cores
    - ✓ Allocate all cores to simulation and bitmaps generation
    - ✓ Executed in sequence
  - Separate Cores
    - ✓ Allocate different core sets to simulation and bitmaps generation
    - ✓ A data queue is shared between simulation and bitmaps generation
    - ✓ Executed in parallel
- In-place bitvector compression
  - Scan data by segments
  - Merge segment into compressed bitvectors

Time-Steps Selection

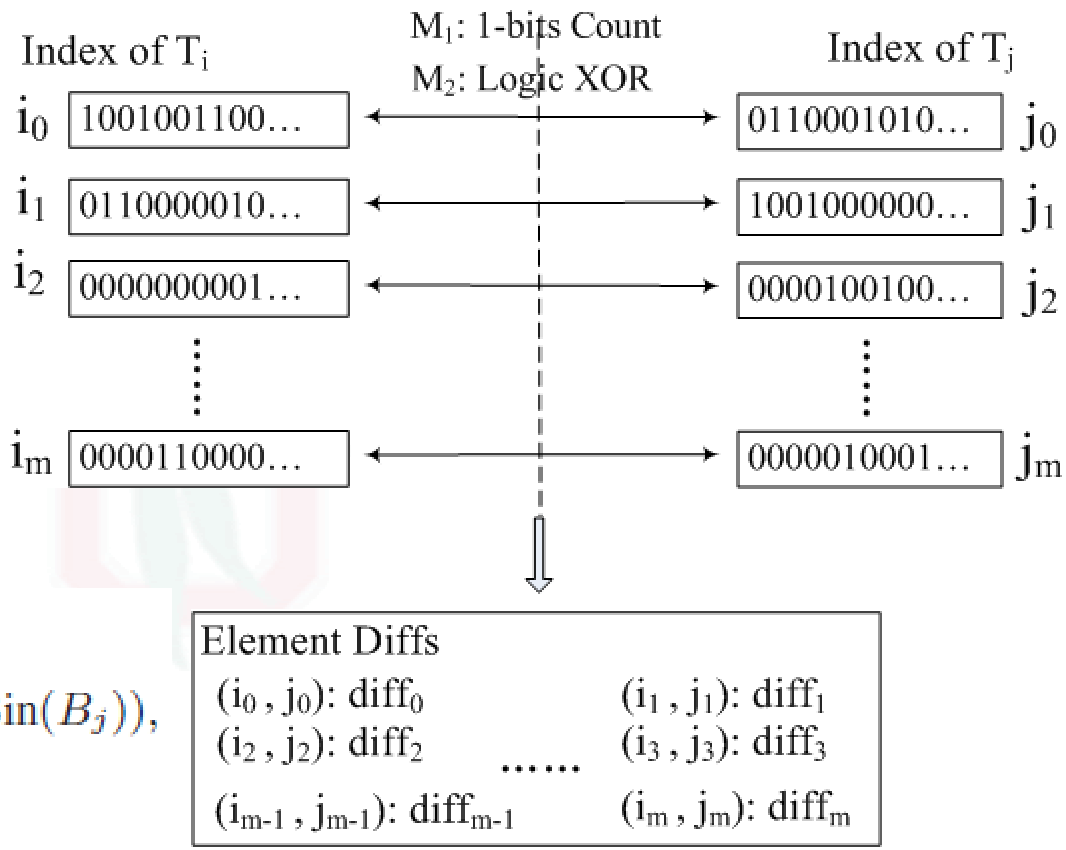


Correlation Metrics

- Earth Mover’s Distance:
  - Indicate distance between two probability distributions over a region
  - Cost of changing value distributions of data
- *Shannon*’s Entropy:
  - A metric to show the variability of the dataset
  - High entropy => more random distributed data
- Mutual Information:
  - A metric for computing the dependence between two variables
  - Low M => two variables are relatively independent
- Conditional Entropy:
  - Self-contained information
  - Information with respect to others

Calculate Earth Mover's Distance Using Bitmaps

- Divide T_i and T_j into bins over value subsets
- Generate a CFP based on value differences between bins of T_i and T_j
- Accumulate results

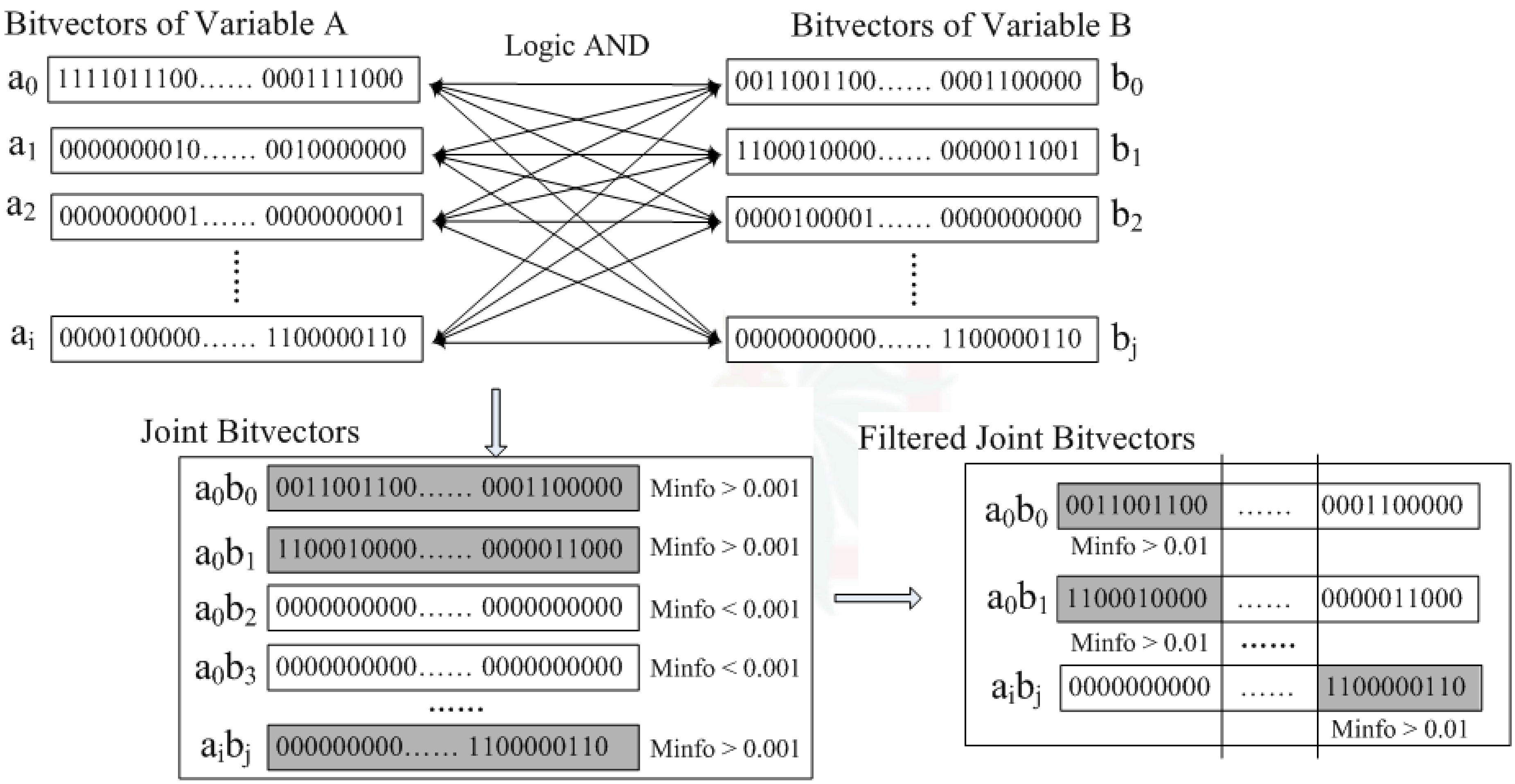


$$\mathrm{EMD} = \sum_{j=1}^{N} \mathrm{CFP}(j),$$
$$\mathrm{CFP}(j) = \mathrm{CFP}(j-1) + Diff(\mathrm{Bin}(A_j), \mathrm{Bin}(B_j)),$$
$$\mathrm{CFP}(0) = 0.$$

Correlation Mining Using Bitmaps

- Correlation mining
  - Automatically suggest data subsets with high correlations
  - Correlation Analysis: keep submitting queries
  - Traditional Method
    - ✓Exhaustive calculation over data subsets (spatial and value)
    - ✓Huge time and memory cost
- Correlation mining using bitmap
  - Mutual Information
    - ✓Calculated by probability distribution (value subsets)
  - A top-down method for value subsets
    - ✓Multi-level bitmap indexing
    - ✓Go to low-level index only if high-level has high mutual info
  - A bottom-up method for spatial subsets
    - ✓ Divide bitvectors (with high correlations) into basic strides
    - ✓ Perform 1-bits count operation over strides

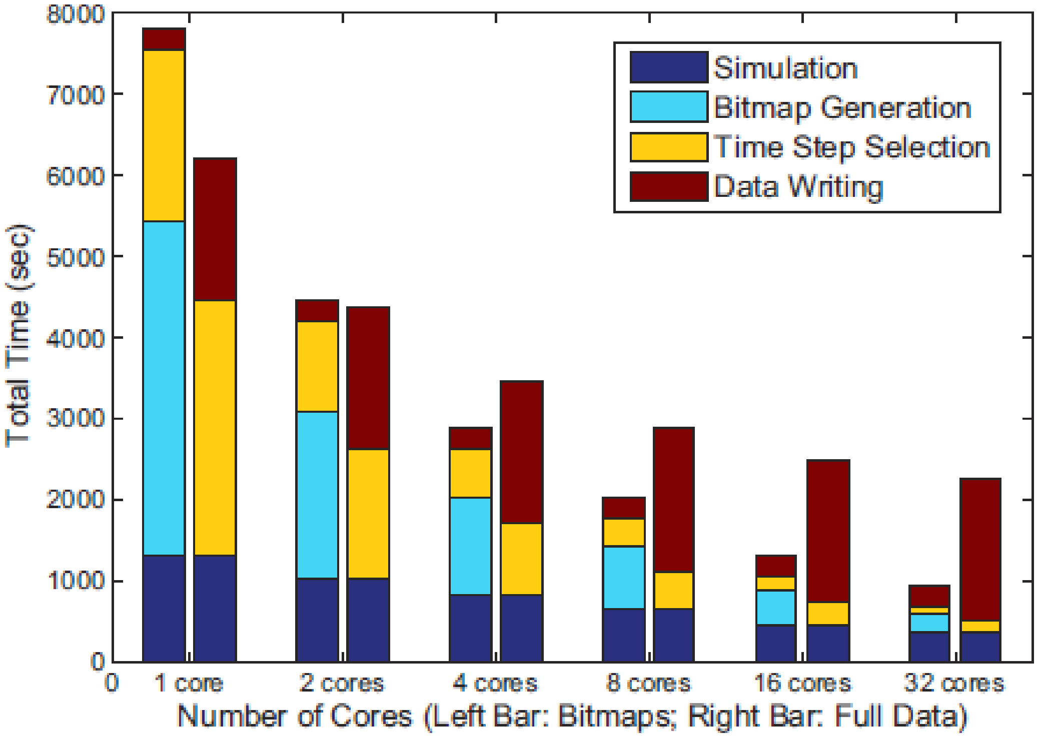
Correlation Mining



Experiment Results

- Goals:
  - Efficiency and storage improvement using bitmaps
  - Scalability in parallel in-situ environment
  - Efficiency improvement for correlation mining
  - Efficiency and accuracy comparison with sampling
- Simulations: Heat3D, Lulesh
- Datasets: Parallel Ocean Program
- Environment:
  - 32 Intel Xeon x5650 CPUs and 1TB memory
  - MIC: 60 Intel Xeon Phi coprocessors and 8GB memory
  - OSC Oakley Cluster: 32 nodes with 12 Intel Xeon x5650 CPUs and 48 GB memory

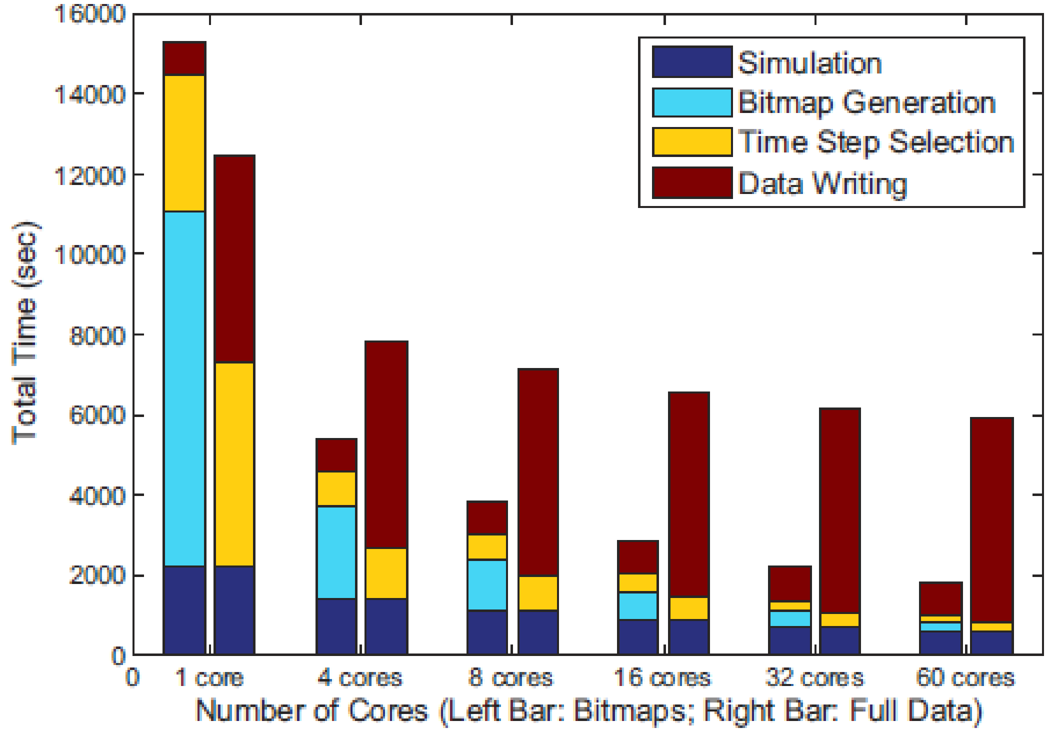
Efficiency Comparison for In-Situ Analysis - CPU



- Simulation: Heat3D; Processor: CPU
- Time steps: select 25 over 100 time steps
- 6.4 GB per time step (800*1000*1000)
- Metrics: Conditional Entropy

- Full Data (original):
  - Simulation: bad scalability
  - Time Step Selection: big
  - Data Writing: big and bad scalability
- Bitmaps:
  - Simulation: utilize extra computing power for bitmaps generation
  - Extra bitmaps generation time but good scalability
  - Time Step Selection Using Bitmaps: **1.38x to 1.5x**
  - Bitmaps Writing: **6.78x**
  - Overall: **0.79x to 2.38x**
  - More number of cores, better speedup we can achieve

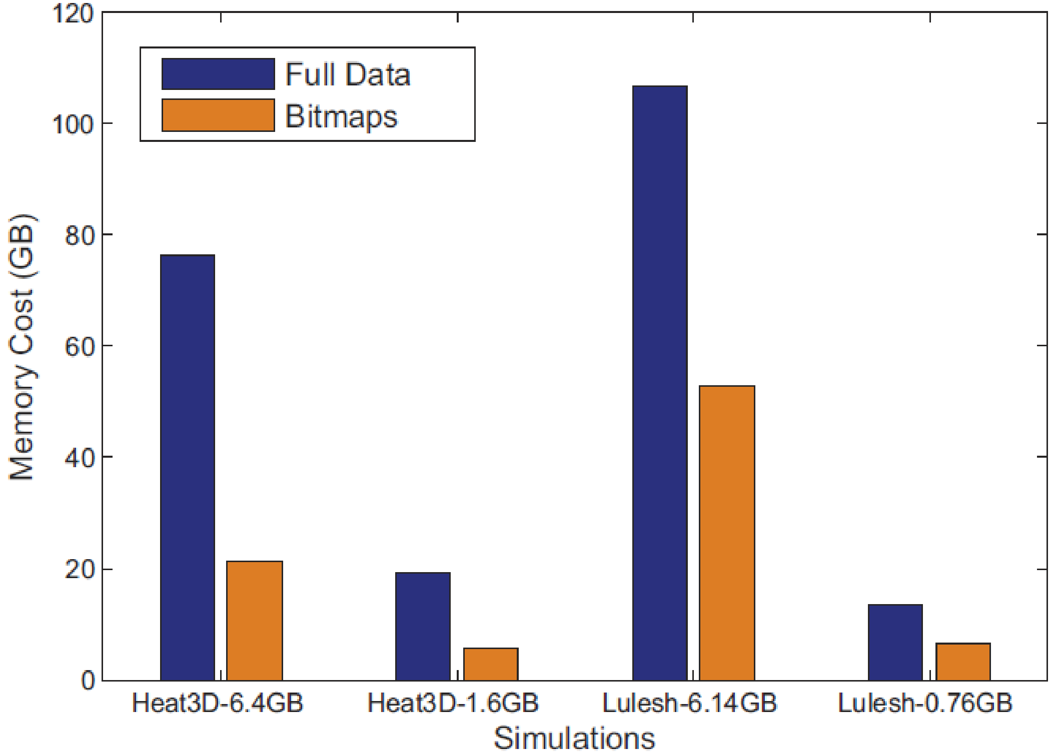
Efficiency Comparison for In-Situ Analysis - MIC



- Simulation: Heat3D; Processor: MIC
- Time steps: select 25 over 100 time steps
- 1.6 GB per time step (200*1000*1000)
- Metrics: Conditional Entropy

- MIC:
  - More cores
  - Lower bandwidth
- Full Data (original):
  - Huge data writing time
- Bitmaps:
  - Good scalability of both bitmaps generation and time step selection using bitmaps
  - Much smaller data writing time
  - Overall: **0.81x to 3.28x**

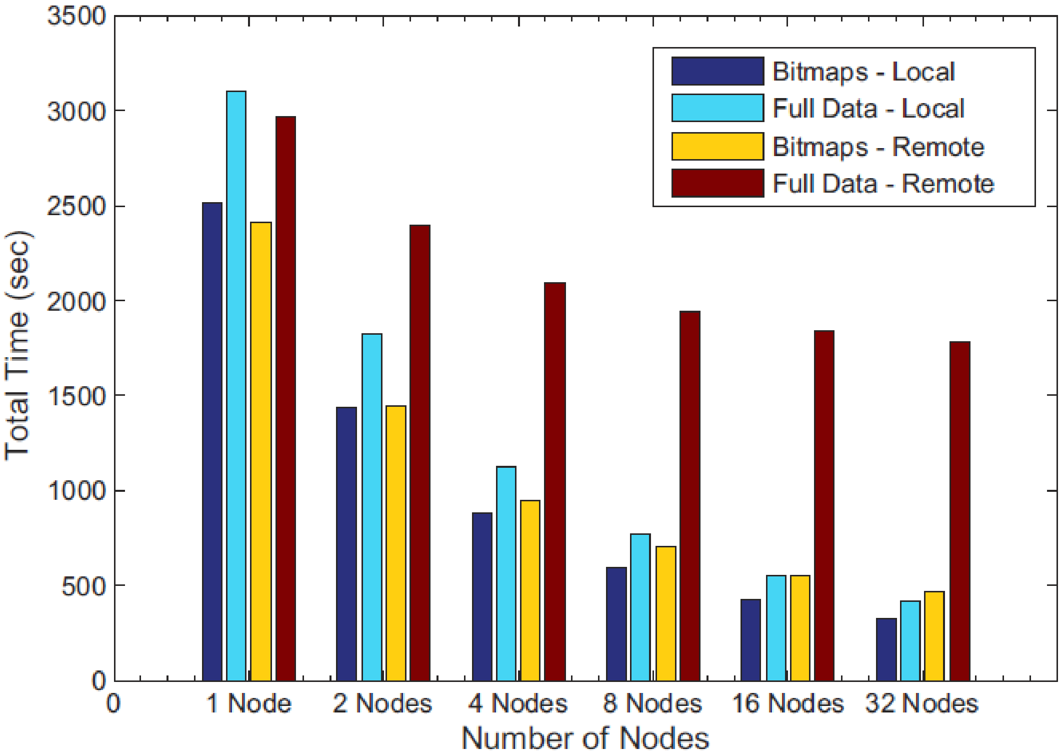
Memory Cost of In-Situ Analysis



- Simulation: Heat3D, Lulesh
- Processor: CPU, MIC
- Keep 10 time steps in memory

- Heat3D - No Indexing:
  - 12 time steps (pre, temp, cur)
- Heat3D - Bitmap Indexing:
  - 2 time steps (pre, temp)
  - 1 previous selected indices
  - 10 current indices
- Lulesh – No Indexing:
  - 11 time steps (pre, cur)
  - Huge extra memory for edges
- Lulesh – Bitmap Indexing:
  - 1 time step (pre)
  - 1 previous selected indices
  - 10 current indices
  - Huge extra memory for edges
- **2.0x to 3.59x smaller memory**
- Better as **bigger data** simulated and **more time steps** to hold

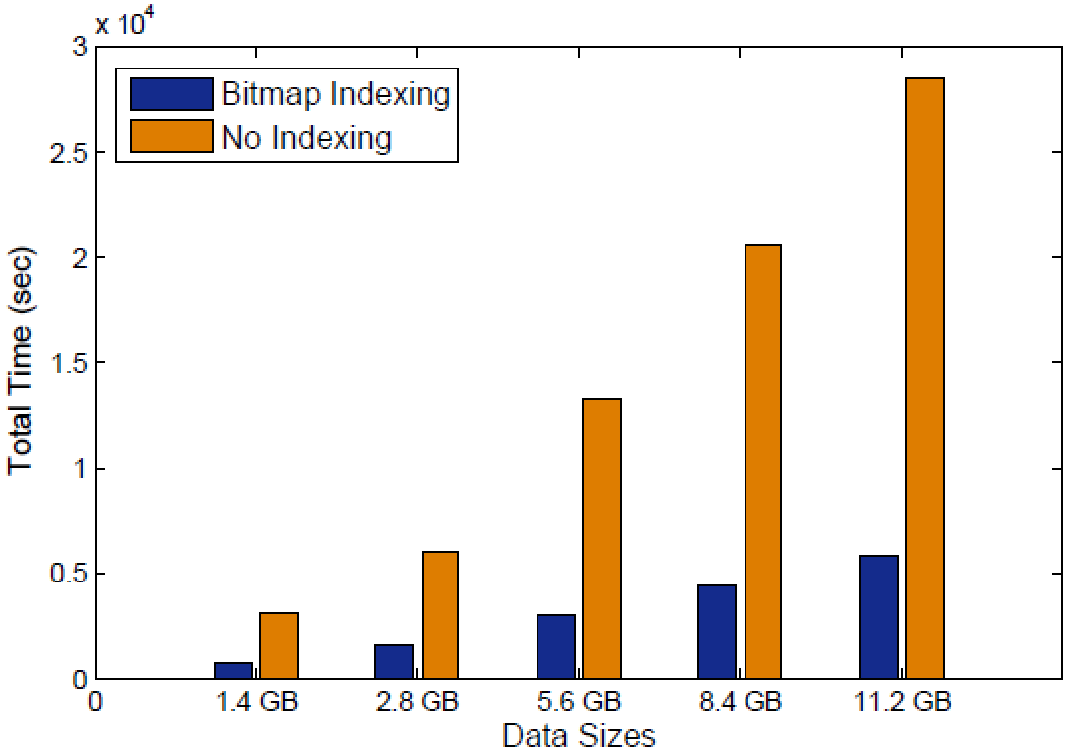
Scalability in Parallel Environment



- Select 25 time steps out of 100
- TEMP Variable: 6.4 GB per time step
- Number of nodes: 1 to 32
- Number of cores: 8

- Simulation: Heat3D
- Full Data– Local:
  - Each node write its data subblock into its own disk
- Bitmaps– Local:
  - Each node writes its bitmaps subblock into its own disk
  - Fast time step selection and local writing
  - **1.24x – 1.29x speedup**
- Full Data– Remote:
  - Different nodes send data sub-blocks to a master node
- Bitmaps – Remote:
  - Greatly alleviate data transfer burden of master node
  - **1.24x – 3.79x speedup**

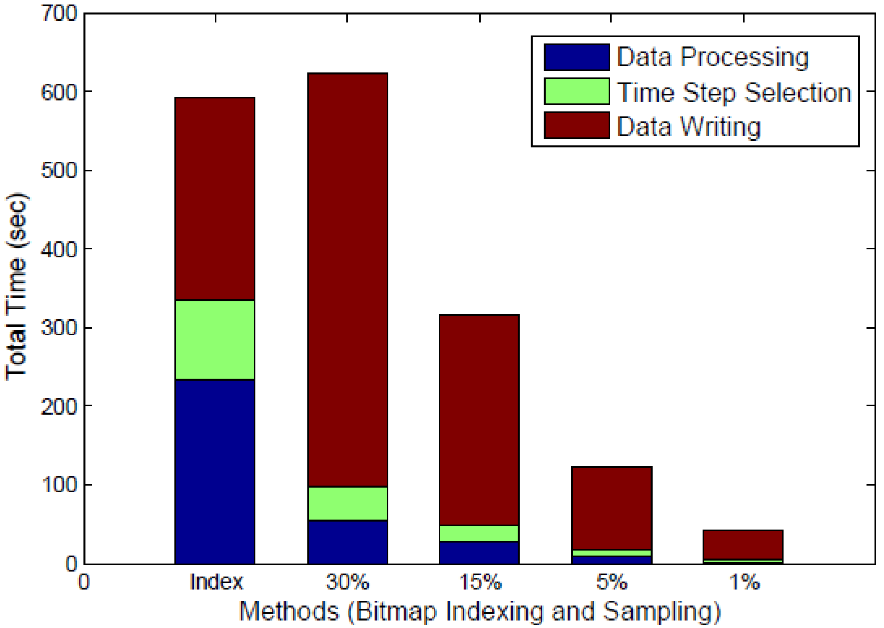
Speedup for Correlation Mining

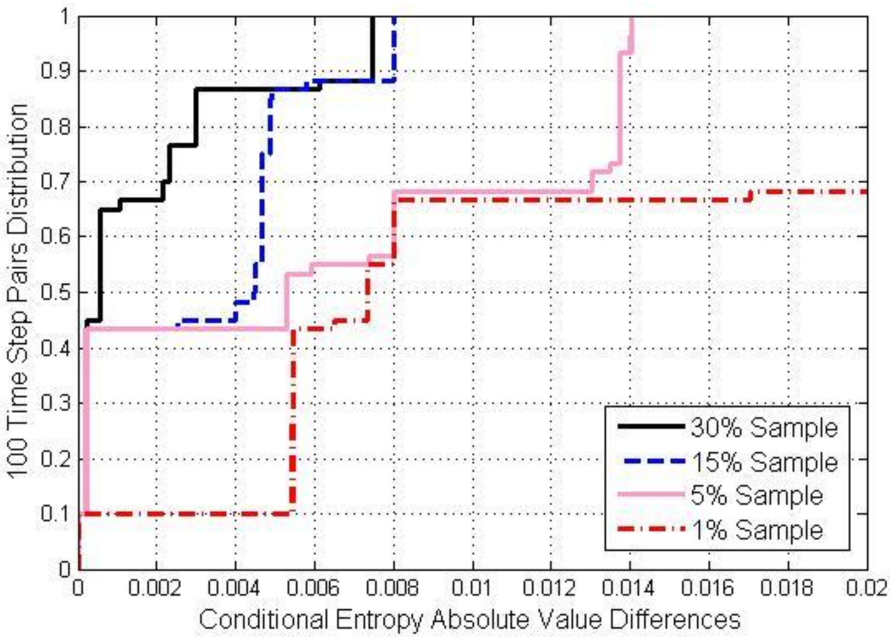


- Variables: TEMP, SALT
- Data size per variable: 1.4 GB to 11.2 GB
- Number of cores: 1

- Simulation: POP
- Full Data:
  - Big data loading cost
  - Exhaustive calculations over data subsets
  - Each calculation is time consuming
- Bitmaps:
  - Smaller data loading
  - Multi-level bitmaps to improve the mining process
  - Bitwise AND and 1-bits count operations to improve the calculation efficiency
  - **3.81x – 4.92x speedup**

In-Situ Sampling vs. Bitmaps





- Heat3D ,100 time steps (6.4 GB), 32 cores
- Bitmaps generation (binning, compression) has more time cost then down-sampling
- Sampling can effectively improve the time step selection cost
- Bitmaps generation can still achieve better efficiency if the index size is smaller than sample size

- Bitmaps: using the same binning scale, does not have any information loss
- Sampling: information loss is unavoidable no matter what sample%
- **30% - 21.03% loss**
- **15% - 37.56% loss**
- **5% - 58.37% loss**

Conclusion

- ‘Big Data’ issue brings challenges for scientific data management
- Efficient in-situ bitmaps generation
- Efficient online data analysis (time step selection) using only bitmaps
- Efficient offline data analysis (correlation mining) using only bitmaps
- Compare in-situ data sampling with in-situ bitmaps