Scalable Data Analytics: On the Role of Stratified Data Sharding

Srinivasan Parthasarathy
Data Mining Research Lab
Ohio State University
srini@cse.ohio-state.edu

The Data Deluge: Data Data Everywhere

facebook

twitter

LinkedIn

YouTube

Instagram

180 zettabytes will be created in 2025 [IDC report]

2

Data Storage is Cheap

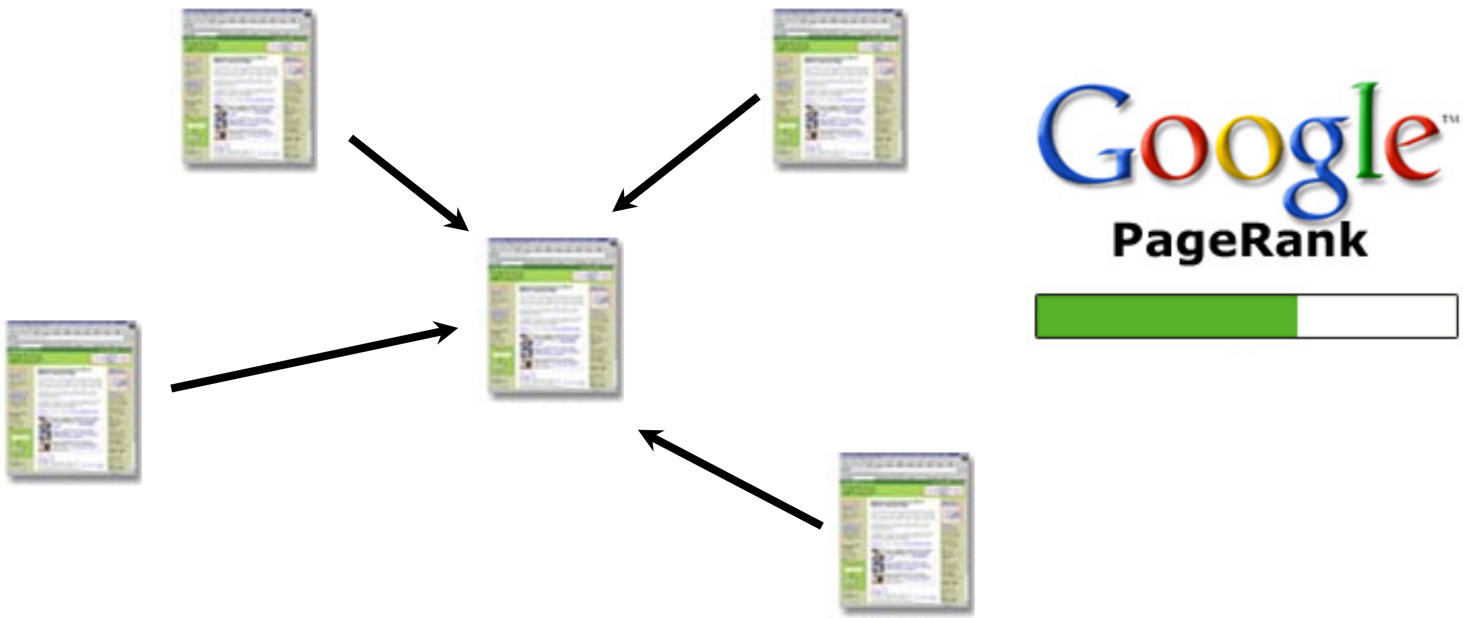
600$

to buy a disk drive that can store all of the world's music

[McKinsey Global Institute Special Report, June' 11]

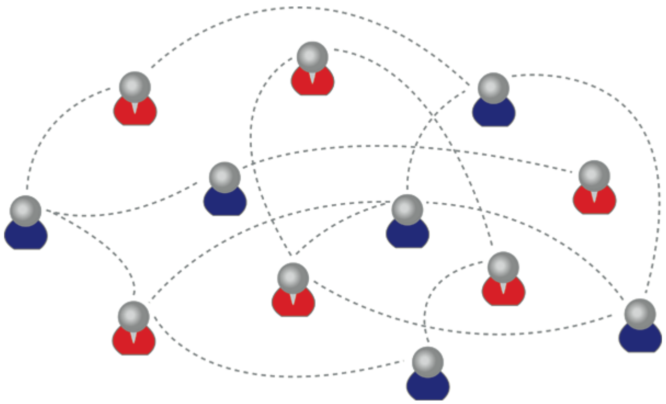
Data does not exist in isolation.

Data almost always exists in connection with other data – integral part of the value proposition.

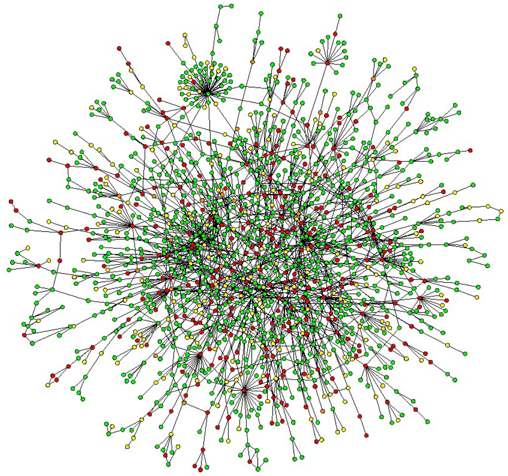


"There's gold in them there mountains of data"

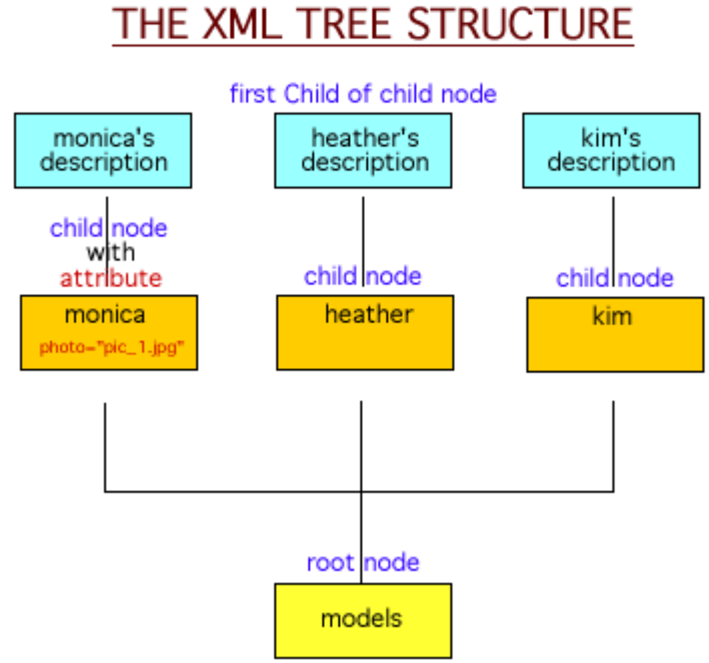
- Gill Press, Forbes Contributor

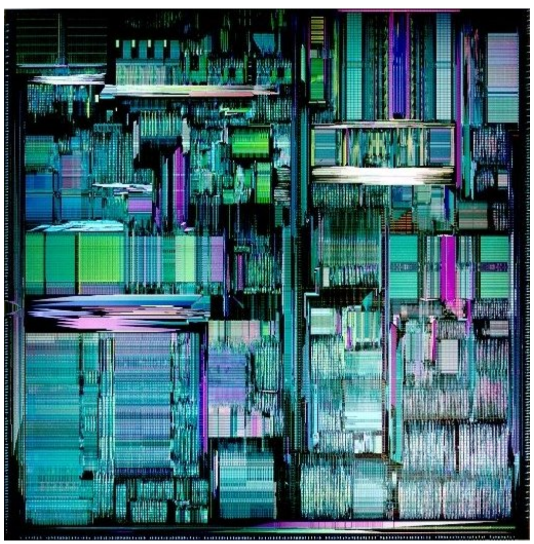


Social networks

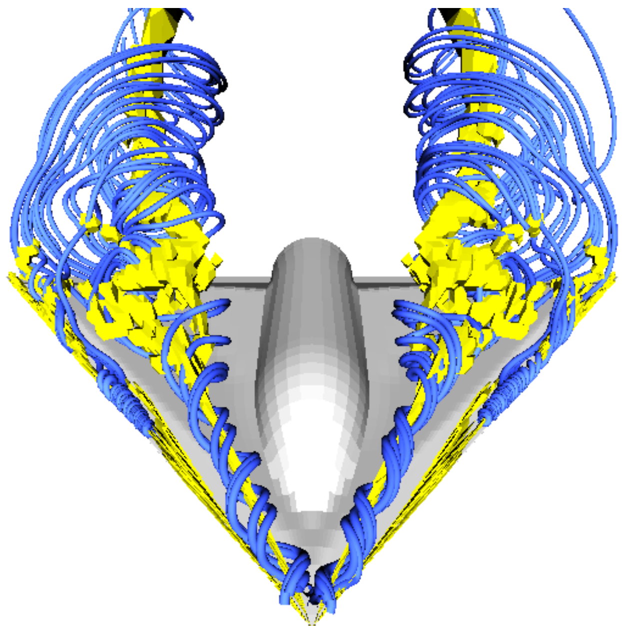


Protein Interaction

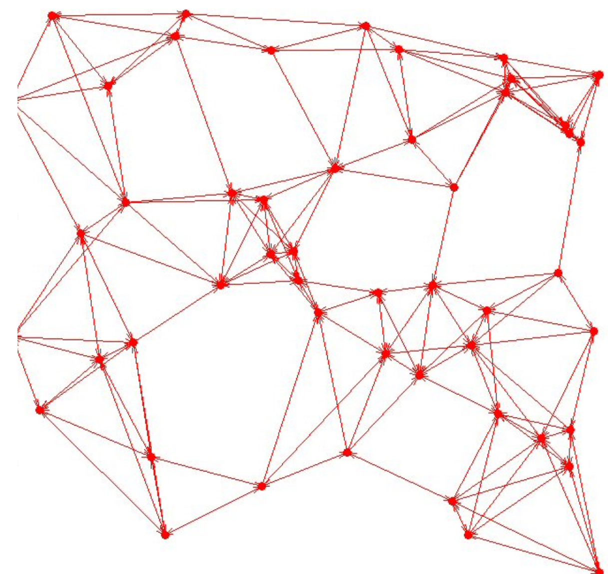




VLSI networks



Scientific Simulations



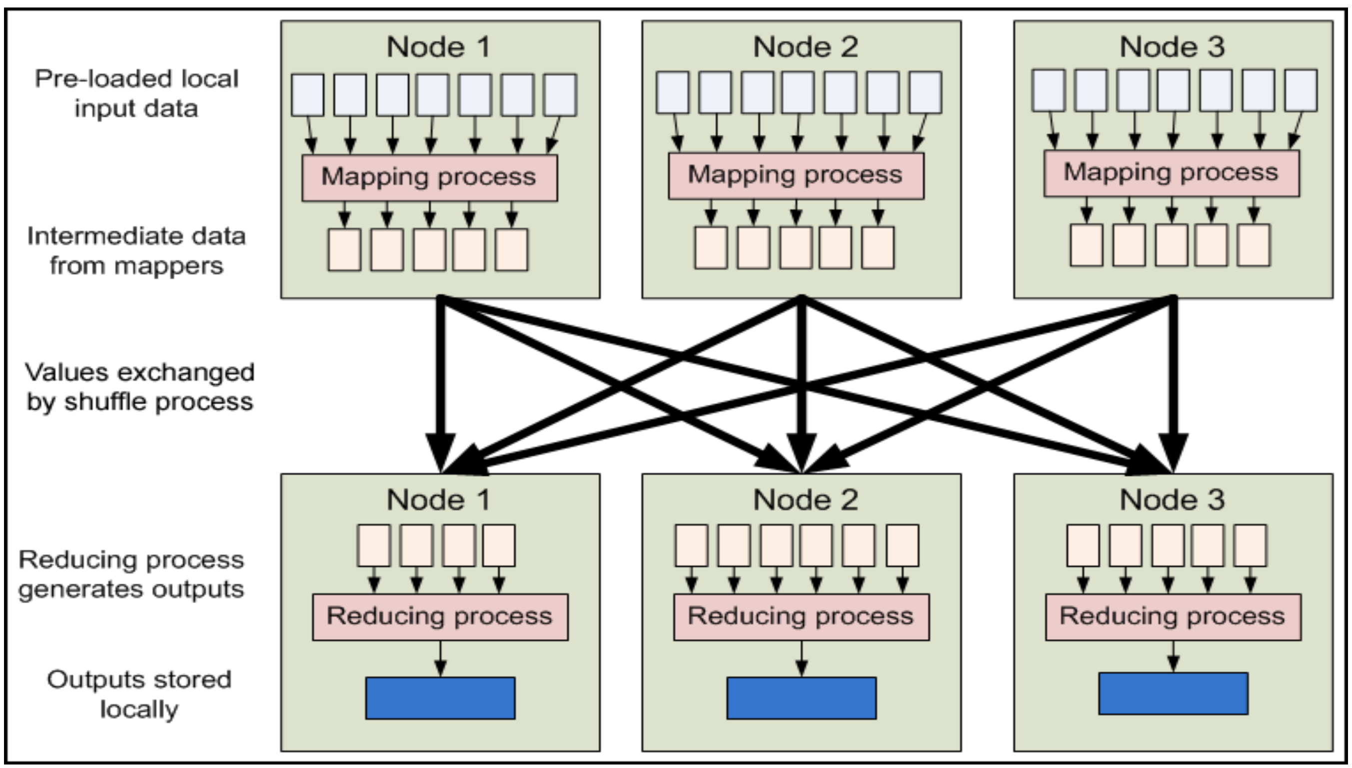
Neighborhood graphs

Big Data Challenge: All this data is only useful if we can extract **interesting** and **actionable** information from large complex data stores **efficiently**

Projected to be a $200B industry in 2020. [IDC report]

Distributed Data Processing is Central to Addressing the Big Data Challenge

MapReduce, MPI, Spark, etc.



Source: blog.mayflower.de

However distributed data processing itself can pose challenges!

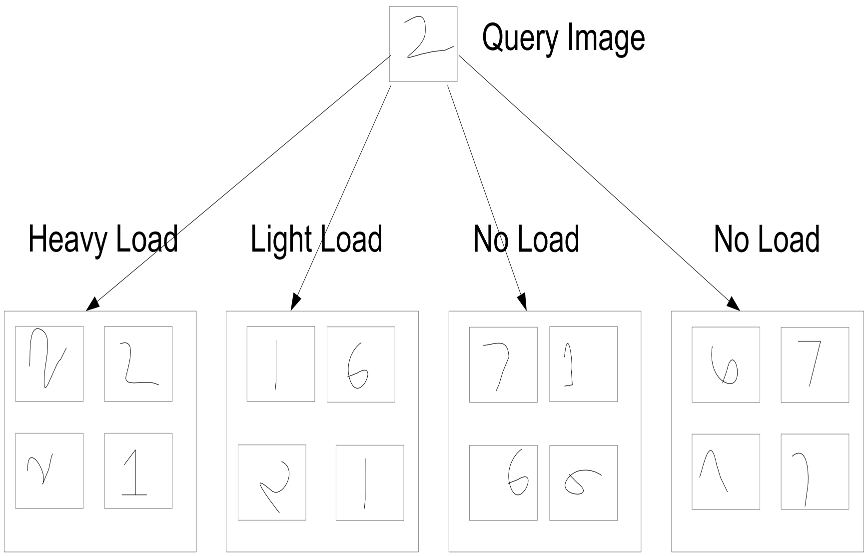
The Case for Stratified Data Sharding of Complex Big Data

THE OHIO STATE UNIVERSITY

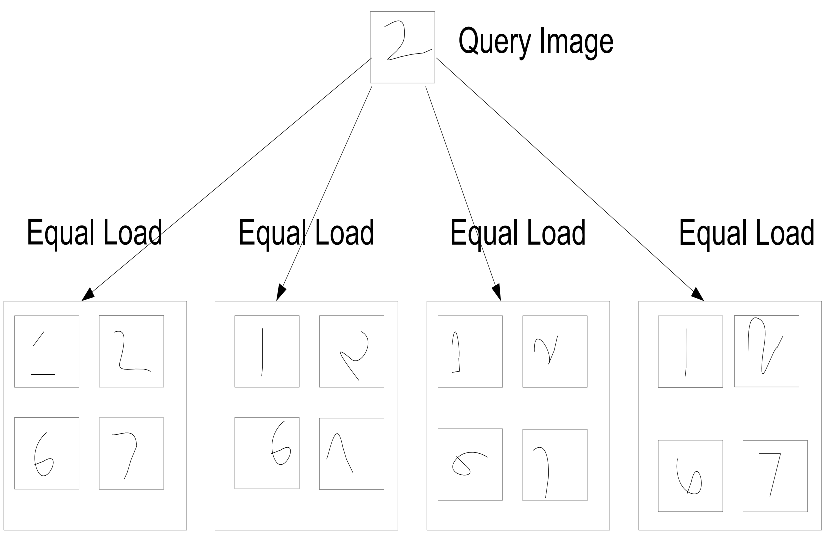
Key Challenge: Data Placement (Sharding)

- Locality of reference
  - Placing related items in proximity improves efficiency
- Mitigating Impact of Data Skew
  - Critical for big data workloads!
- Interactive Response Times
  - Operate on a sample with statistical guarantees
- Heterogeneity and Energy Aware
  - Heterogeneous compute and storage resources are ubiquitous

Example – Image retrieval



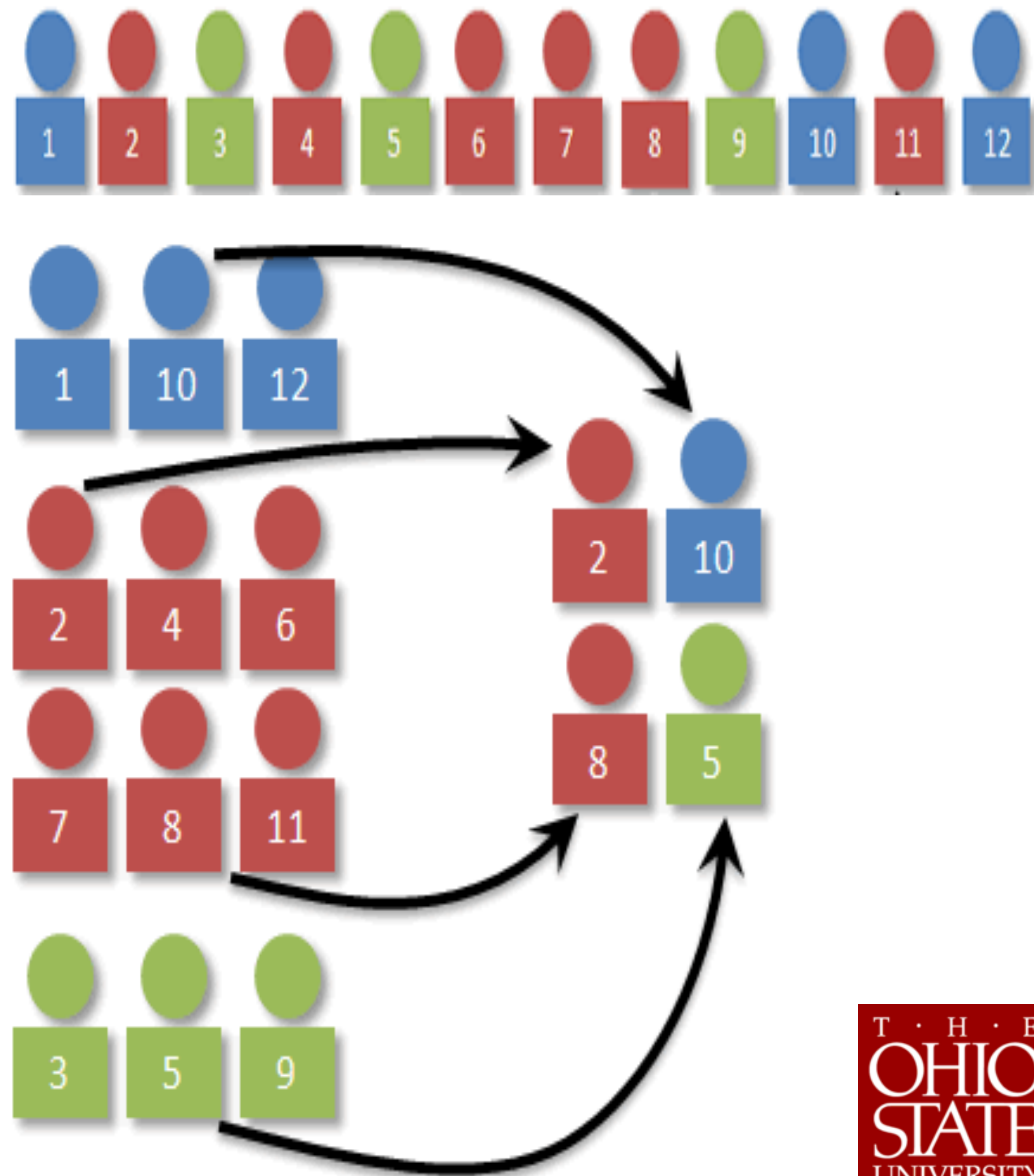
- Random partitioning
- Load imbalance



- Stratified partitioning
- Load imbalance mitigated

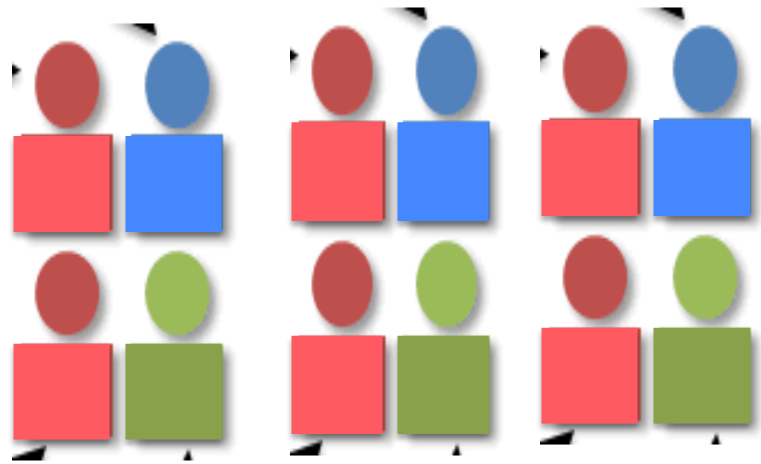
Stratified Sampling in a Slide

- Roots in Stratified Sampling (Cochran'48)
- Group related data into "homogeneous strata"
- Sample each strata
  - Proportional Allocation (shown)
  - Optimal Allocation



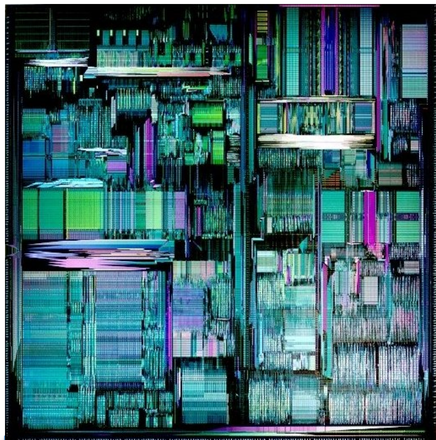
But, here we want to partition/shard

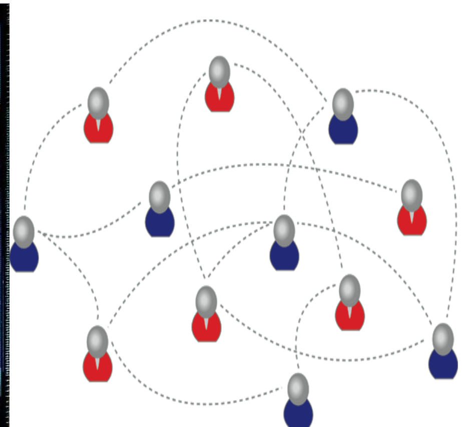
- For Locality
    - Elements within a strata are placed together
- For Mitigating Skew
    - Each partition is a proportionally allocated stratified sample
- For Interactivity
    - Optimally allocate one partition
    - Proportionally allocate the rest
- Accounting for Energy/Heterogeneity
    - More on this later -- time permitting

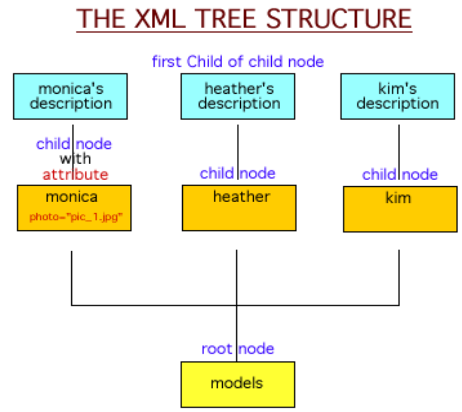


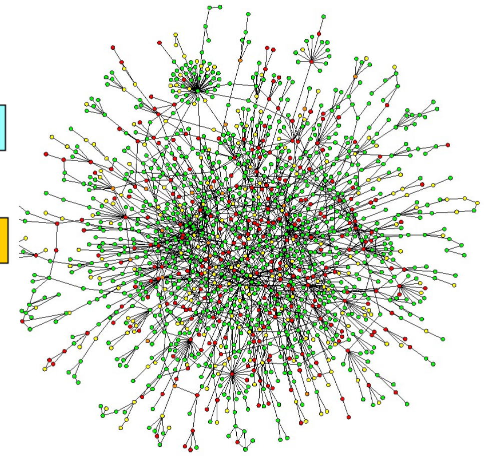
Apps have varying requirements: ONE SIZE **DOES NOT** FIT ALL!

Our Vision: Stratified Data Placement









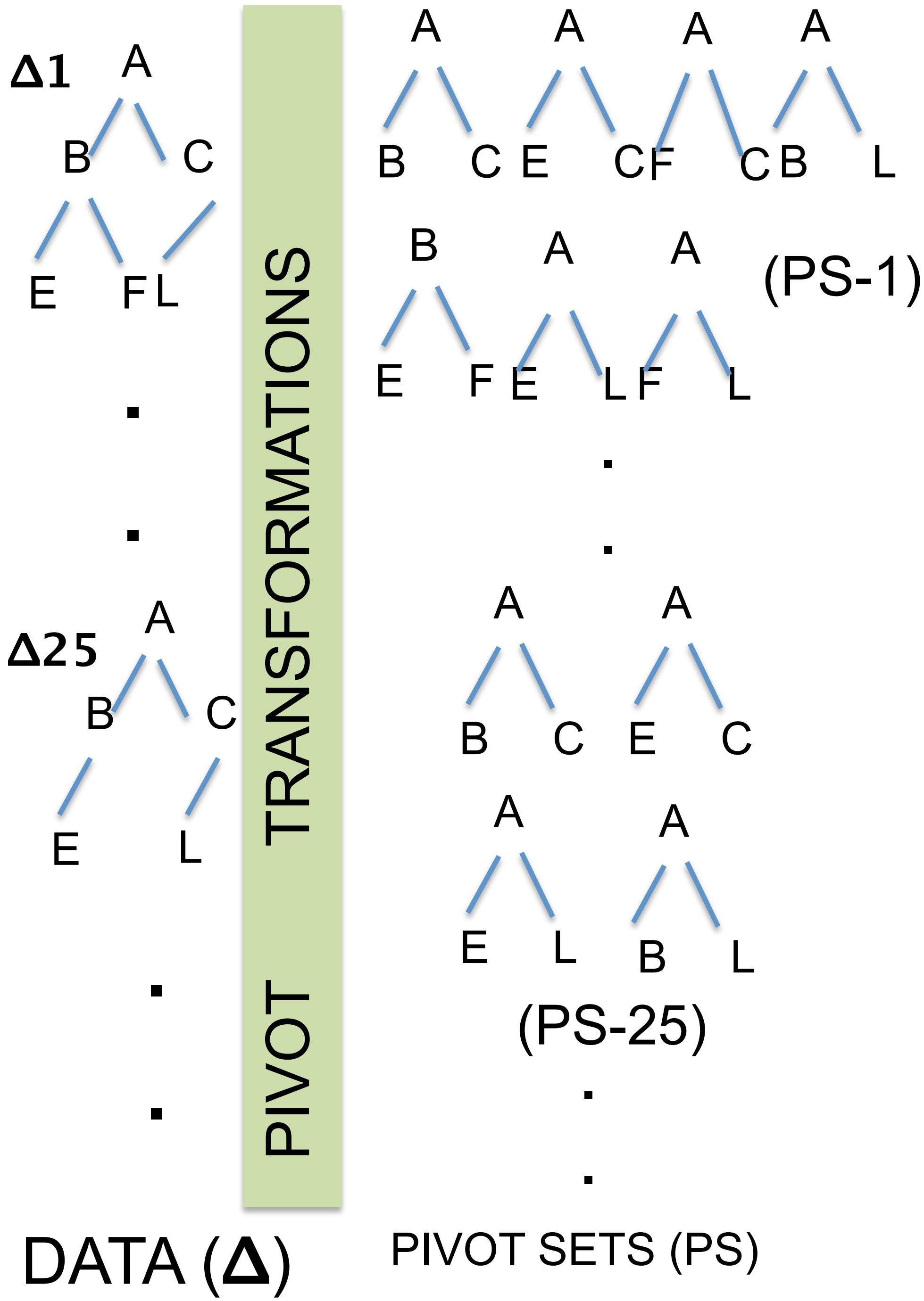
STRATIFIED DATA SHARDING & PLACEMENT

HADOOP/SHARK/ Azure (HDFS/RDD/Blob)	Key Value Stores e.g. Memcached Redis	MPI & Partitioned Global Addresses Space Systems (PGAS) e.g. Global Arrays

Key Challenge: Creating Strata (of Complex Data)

- What about Clustering?
  - Non-trivial for data with complex structure
  - Potentially expensive
  - Variable sized entities

- 4-step approach [ICDE'13]
  1. Convert complex data into a (multi-)set of pivotal elements that *capture features-of-interest*
  2. Compute sketch of set (minwise hashing)
  3. Use sketches to group into strata (sketchsort/sketchcluster)
  4. Partition strata according to application needs (e.g. skew, balance, locality)



Step 1: Pivotization

Problem: Need to simplify complex representation.

Key Idea: Think Globally Act Locally

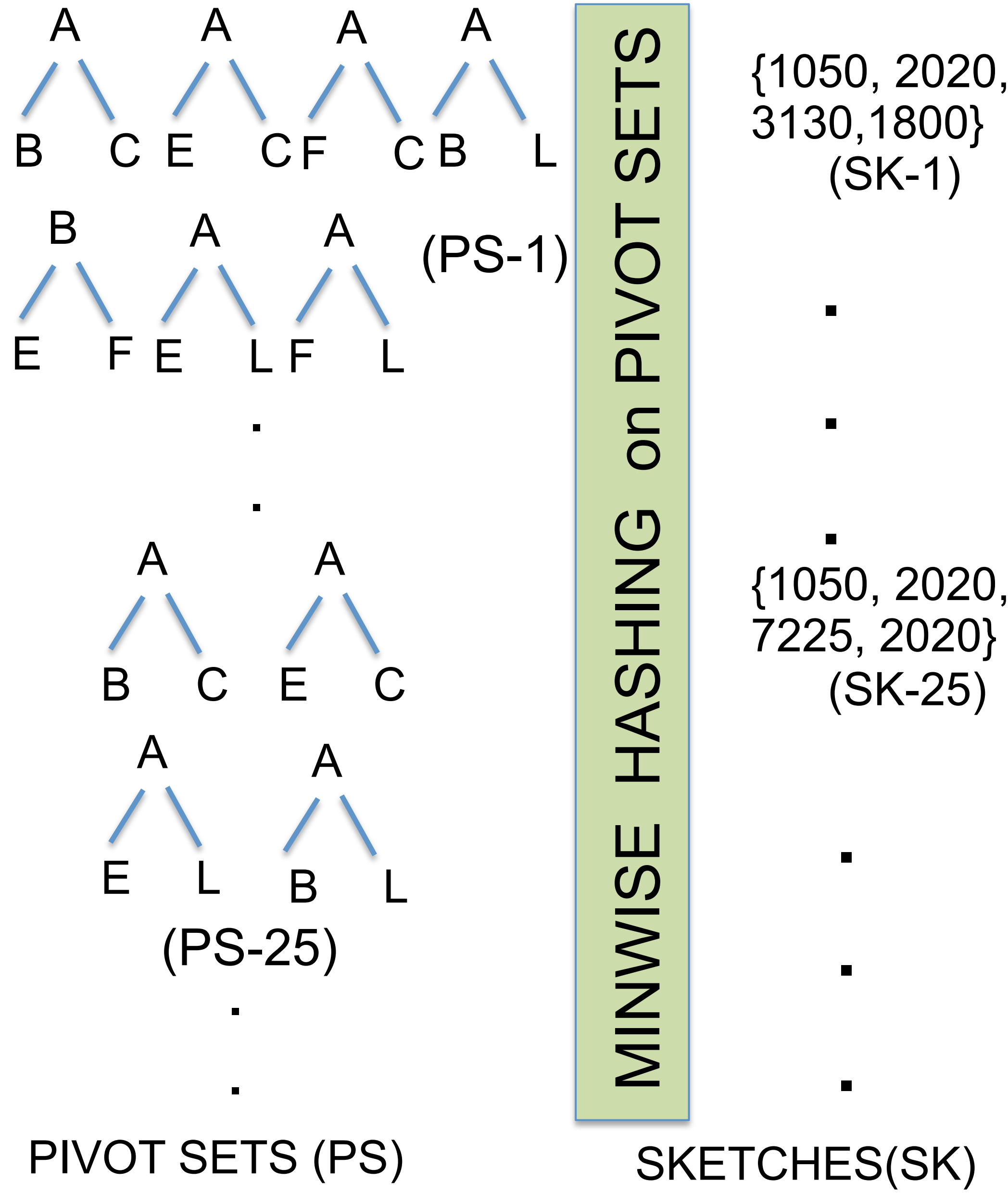
- Sets of localized features that collectively captures global picture

Solution: Specific to Data & Domain

- Documents/Text
  - Shingling [Broder 1998]
- Trees (XML, Linguistic data)
  - Wedge pivots [Tatikonda'10]
- Graphs (Web, Social, Molecules)
  - Adjacency lists [Buehrer'08], Wedge Decompositions [Seshadri'11], Graphlets [Pruzlj'09]
- Spatial/vector data
  - LSH[Indyk'99, Chariker'02, Satuluri'12]
- Images/Simulation/Sequential data
  - Kernels (Leslie'03), KLSH (Kulis'2010)

Step 2. Sketching

- **Problem:** Pivot sets may be variable length, similarity computation is expensive: O(n^2)
- **Key Idea:** Use Sketching
- **Solution:** Locality Sensitive Hashing *[Broder'98, Indyk'99, Charikar'01]*
  - *Resulting representation is fixed-length (k)*
  - *Tradeoff: Representation Fidelity vs. Sketch size*
  - *Can handle kernel functions [Kulis'09] and statistical priors [Satuluri'12, Chakrabarti'15, '16]*



Minwise Hashing (Broder et al 98)

Universe $\longrightarrow$ { dog, cat, lion, tiger, mouse}

$\pi_1 \longrightarrow$ [cat, mouse, lion, dog, tiger]

$\pi_2 \longrightarrow$ [lion, cat, mouse, dog, tiger]

A = { mouse, lion }

$mh_1(A) = \min(\pi_1\{ \text{mouse, lion} \}) = \text{mouse}$

$mh_2(A) = \min(\pi_2\{ \text{mouse, lion} \}) = \text{lion}$

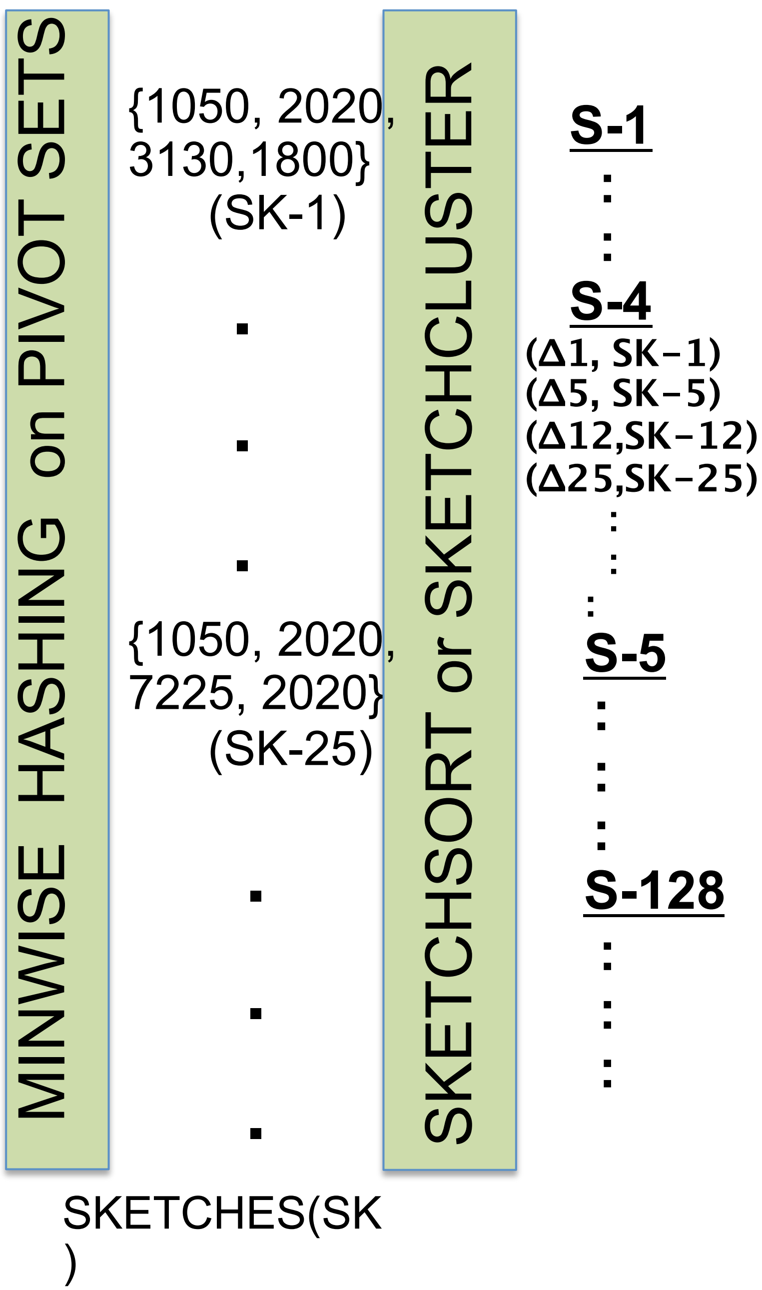
Key Fact

For two sets A, B, and a min-hash function *$mh_i()$*:

$$Pr[mh_i(A) = mh_i(B)] = Sim(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Unbiased estimator for *Sim* using *k* hashes:

$$S\hat{i}m(A, B) = \tfrac{1}{k} \sum_{i=1:k} I[mh_i(A) = mh_i(B)]$$



Step 3: Stratification

Problem: Group related entities into strata

Key Idea: Inspired by W. Cochran's work on stratified sampling [1940s]

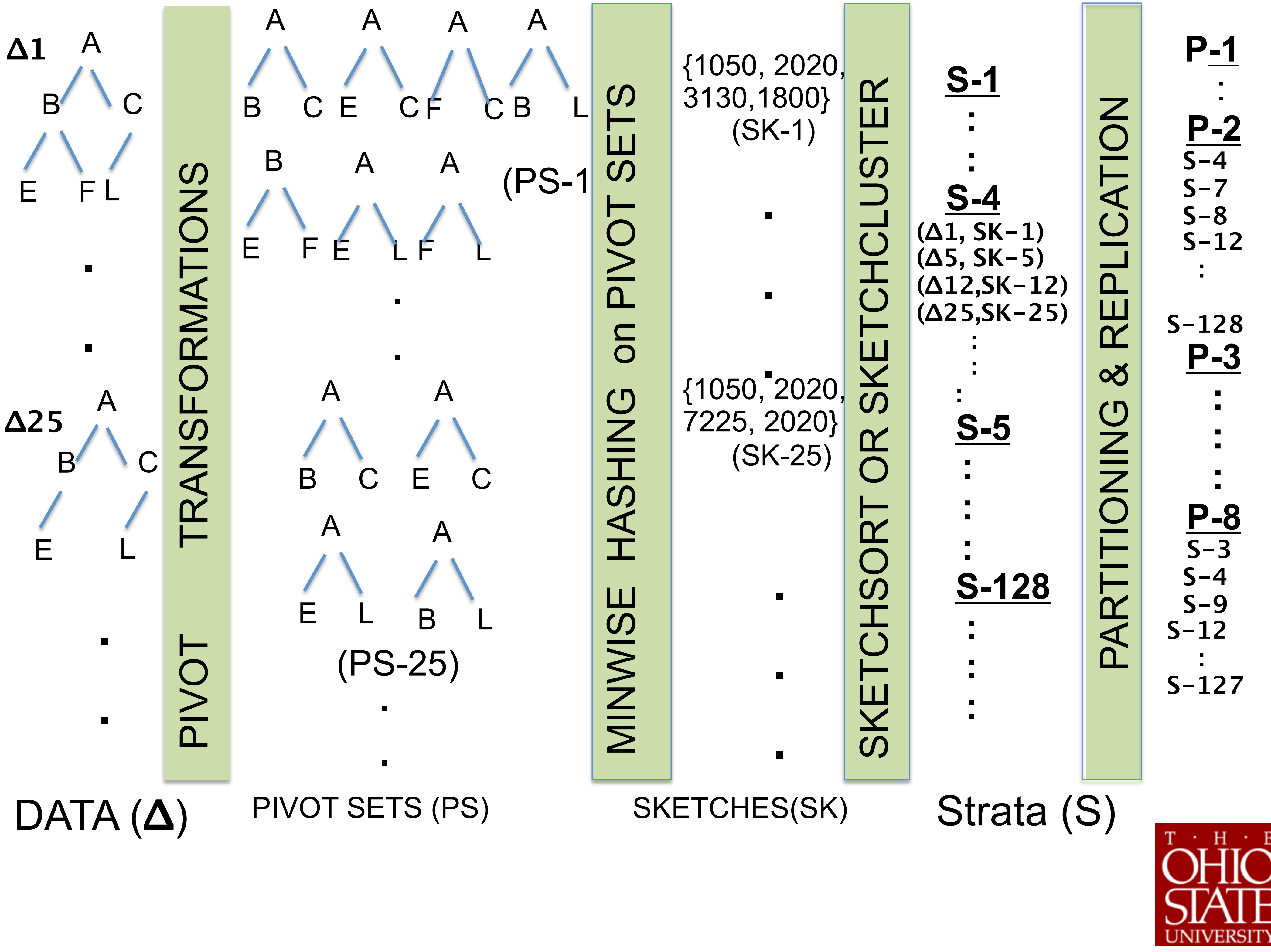
Solutions:

- Sort pivot sets directly (skip sketch step) – **Pivot Sort**
- Directly use output of LSH/Minwise Hash – **SketchSort**
- Cluster sketches with fast variant of k-modes – **SketchCluster**

Step 4: Sharding and Placement

- **Problem:** How to partition stratified data?
- **Key Ideas:** Guided by application hints and system state.
- **Solutions:**
  1. **Proportional Allocation**: Split each stratum uniformly proportionally across all partitions → mitigates skew
  2. **Optimal Allocation** for first strata, proportional for rest [C77]
  3. **All-in-One** : Place each stratum in its entirety within a partition

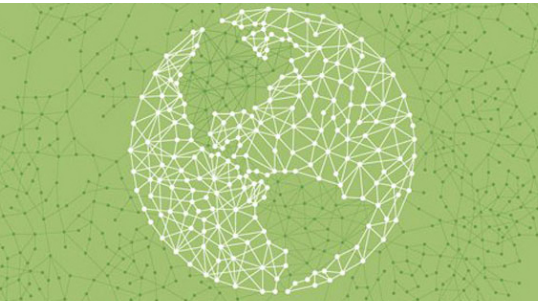
IMPORTANT NOTE: We use sketches to create strata – but partitioning happens on original data.



Δ1
Δ25
PIVOT TRANSFORMATIONS
(PS-1)
(PS-25)
MINWISE HASHING on PIVOT SETS
{1050, 2020, 3130,1800} (SK-1)
{1050, 2020, 7225, 2020} (SK-25)
SKETCHSORT OR SKETCHCLUSTER
S-1
S-4
(Δ1, SK-1)
(Δ5, SK-5)
(Δ12,SK-12)
(Δ25,SK-25)
S-5
S-128
PARTITIONING & REPLICATION
P-1
P-2
S-4
S-7
S-8
S-12
S-128
P-3
P-8
S-3
S-4
S-9
S-12
S-127
DATA (Δ)
PIVOT SETS (PS)
SKETCHES(SK)
Strata (S)

Empirical Evaluation

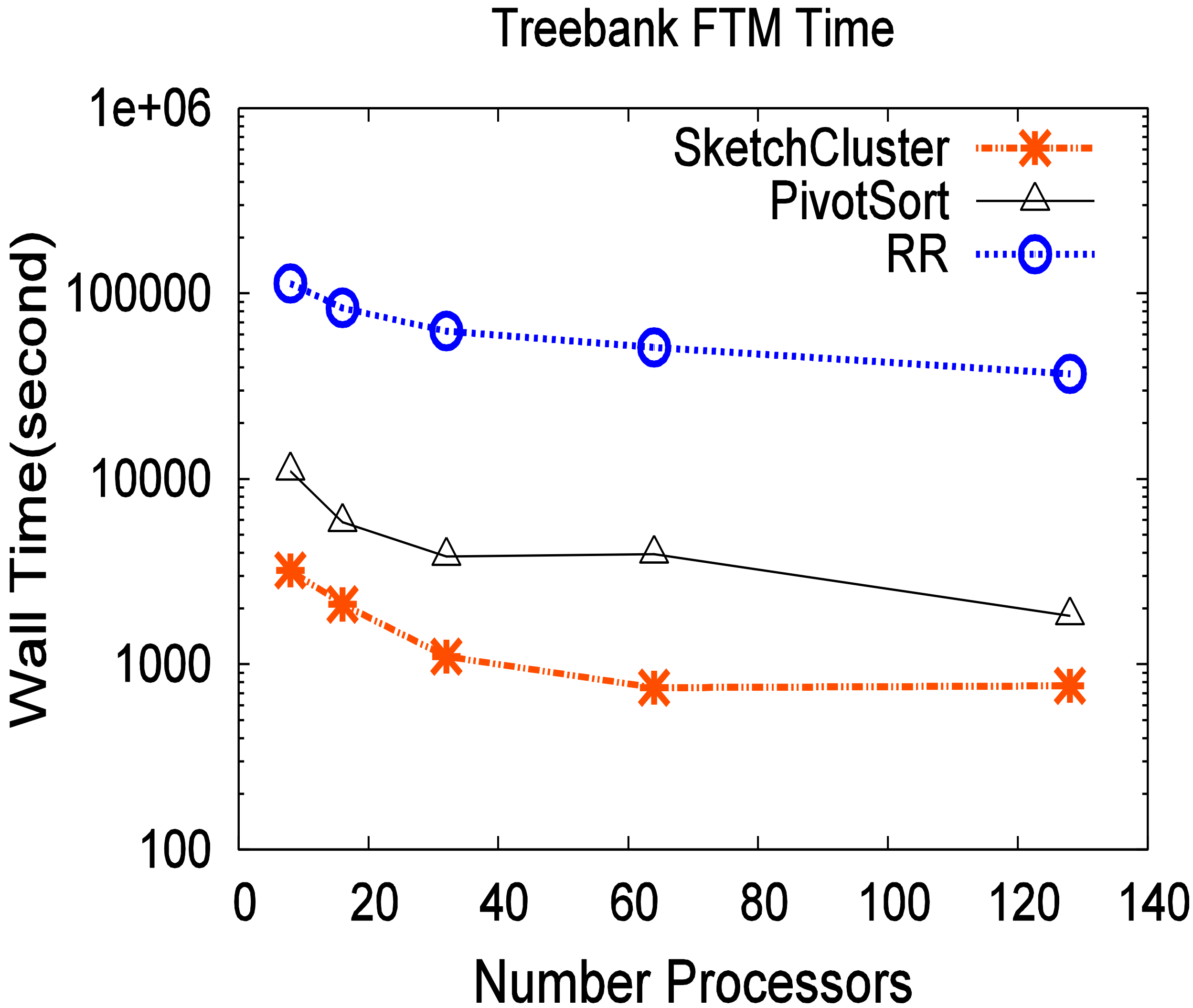
- We report wall clock times
- All times include cost of placement
- Evaluations on several key analytic tasks
  - Top-K algorithms [Fagin], Outlier Detection [Ghoting'08, Otey'06], Frequent Tree[Zaki'05, Tatikonda'09] and Graph Mining [Buehrer'06, Yan'02, Nijlsson'04], XML Indexing [Tatikonda'07], Community detection in Social/Biological data [Ucar'06, Satuluri'11], Web Graph Compression [Chellapilla'08-09; Vigna'11, LZ'77], Itemset Mining [Buehrer-Fuhry'15]
  - All applications are run straight out of the box – the only thing the user specifies relates to locality, skew, and interaction.



Frequent Tree Mining

[Tatikonda'09]

- Used Widely
  - Transactions, graphs, trees
- Approach
  1. Distribute Data
     - Proportional Allocation
  2. Run Phase 1
  3. Exchange Meta Data
  4. Run Phase 2
  5. Final Reduction
- Sharding mainly impacts steps 1-3. Steps 3 and 5 are sequential.



Proposed approaches shows 100X gains

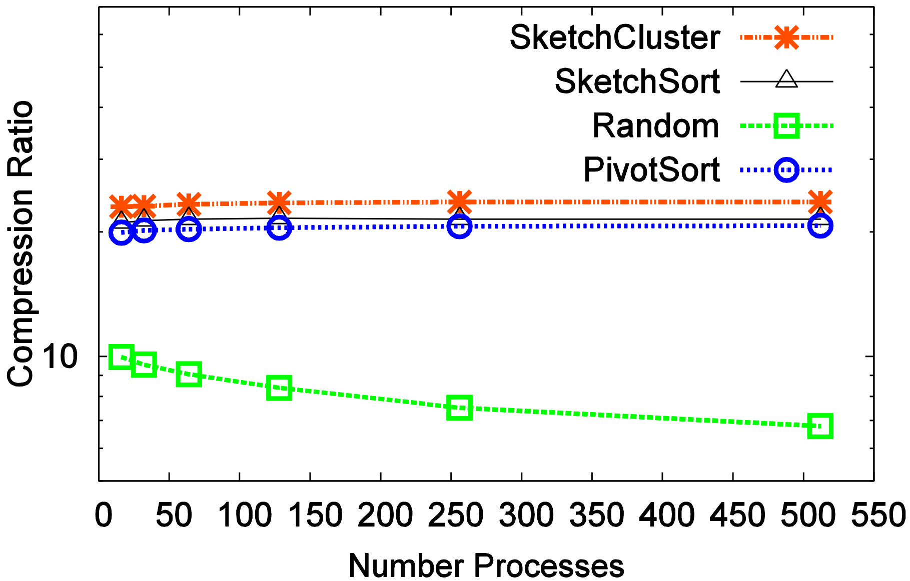
FTM Phase 1: Drilling Down

•

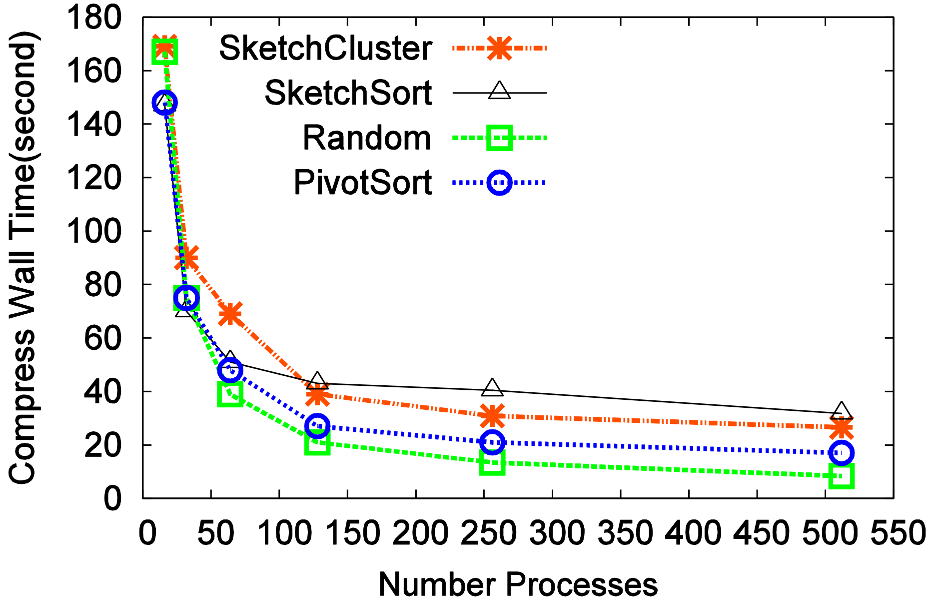
- Data Dependent Workload Skew is mitigated
- Payload-aware sharding helps!

WebGraph Compression [Vigna et al 2011]

Arabic WG Compression Ratio



Arabic WG Compression Time



Critical application for search companies

Key Requirement: Locality

Approach:

- Distribute data via placement
- Run compression algorithm in parallel
- Parameters (similar to FTM)
  - Use adjacency/triangle pivots
  - Use All-in-one partitioning

A segway and drill down (ICDE'15): Localized Approximate Miner (LAM)

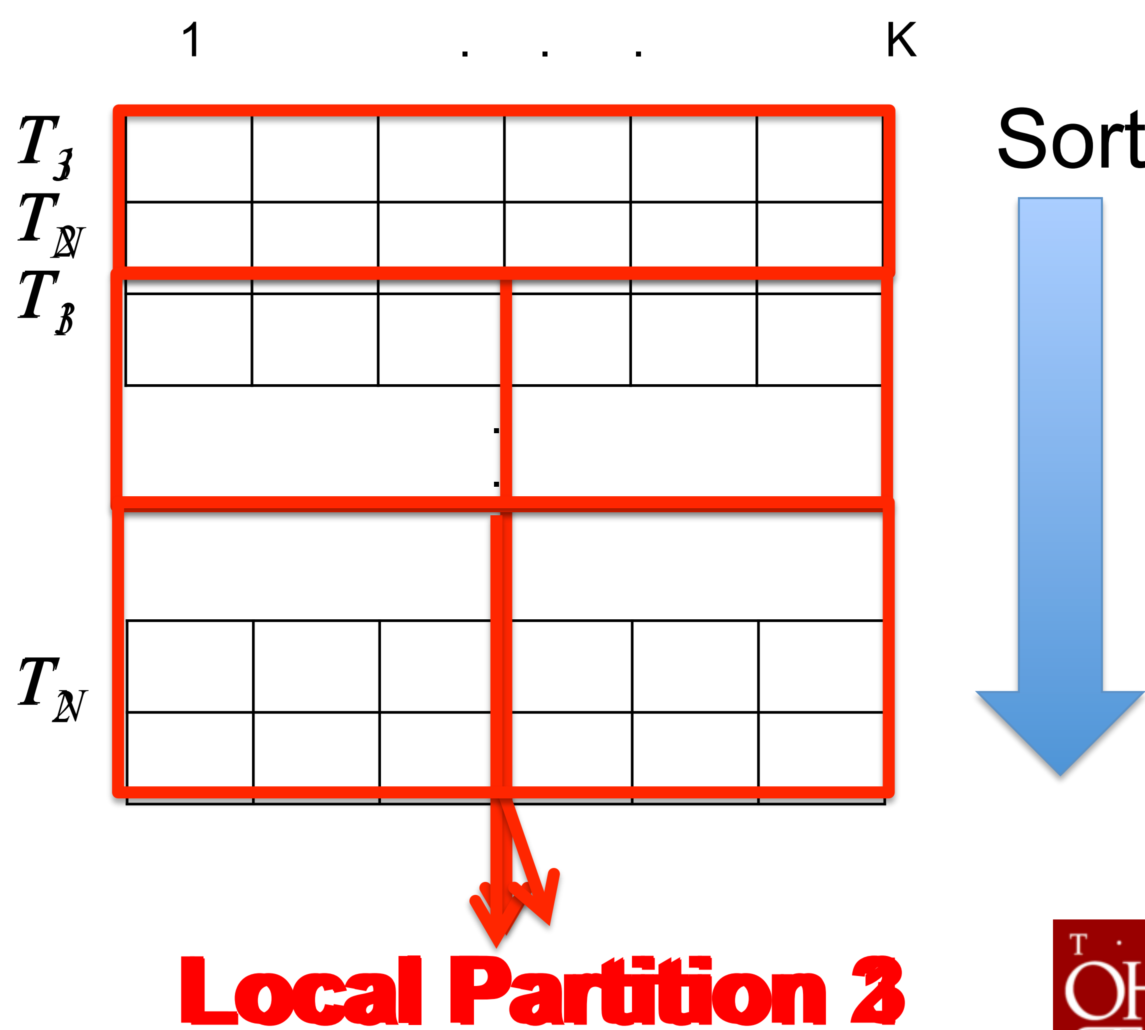
- First bounded space & time pattern mining algorithm; $O(|D| \log |D|)$
- Parameter-free
- Scales with compute resources
  - Near-linear in cores & machines
- Scales with data size
  - Billions of transactions & items
  - E.g. 67 min on one machine; 1 min on a cluster
- Two parallel phases: *Localize*, *ApproxMining*

LAM Phase 1: Localize [SketchSort and Stratify!]

Dataset D =

T_1	1,2,3,7
T_2	1,2,5,7,8
T_3	1,2,4
.	.
.	.
.	.
T_N	1,7,8,5,3,22

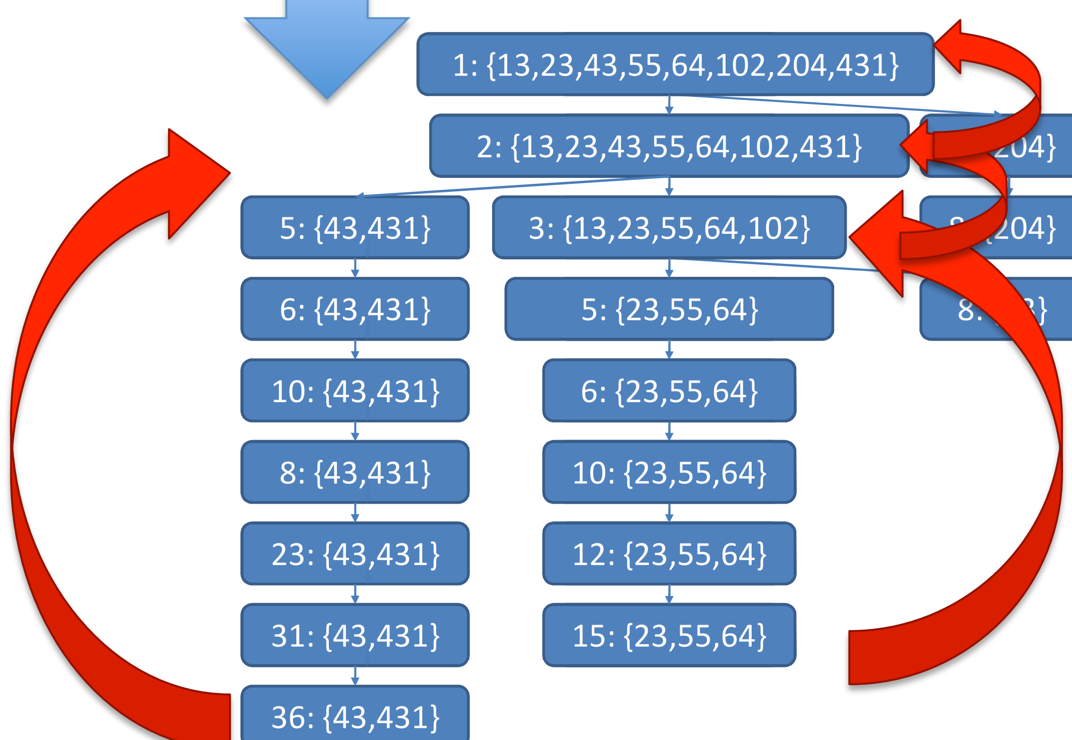
Min-wise hash Matrix M =



LAM Phase 2: Approx Mining I

Trans Id	Items
23	6,10,5,12,15,1,2,3
102	1,2,3,20
55	2,3,10,12,1,5,6,15
204	1,7,8,9,3
13	1,2,3,8
64	1,2,3,5,6,10,12,15
43	1,2,5,10,22,31,8,23,36,6
431	1,2,5,10,21,31,67,8,23,36,6

Trans Id	Items
23	1,2,3,5,6,10,12,15
102	1,2,3
55	1,2,3,5,6,10,12,15
204	1,3,8
13	1,2,3,8
64	1,2,3,5,6,10,12,15
43	1,2,5,6,10,8,23,31,36
431	1,2,5,6,10,8,23,31,36



Pattern	Transaction List	Uty.
1,2,3,5,6,10,12,15	23,55,64	14
1,2,5,6,10,8,23,31,36	43,431	8
1,2,3	13,23,55,64,102	8
1,2	12,23,43,55,64,102,431	6

LAM Phase 2: Approx Mining II

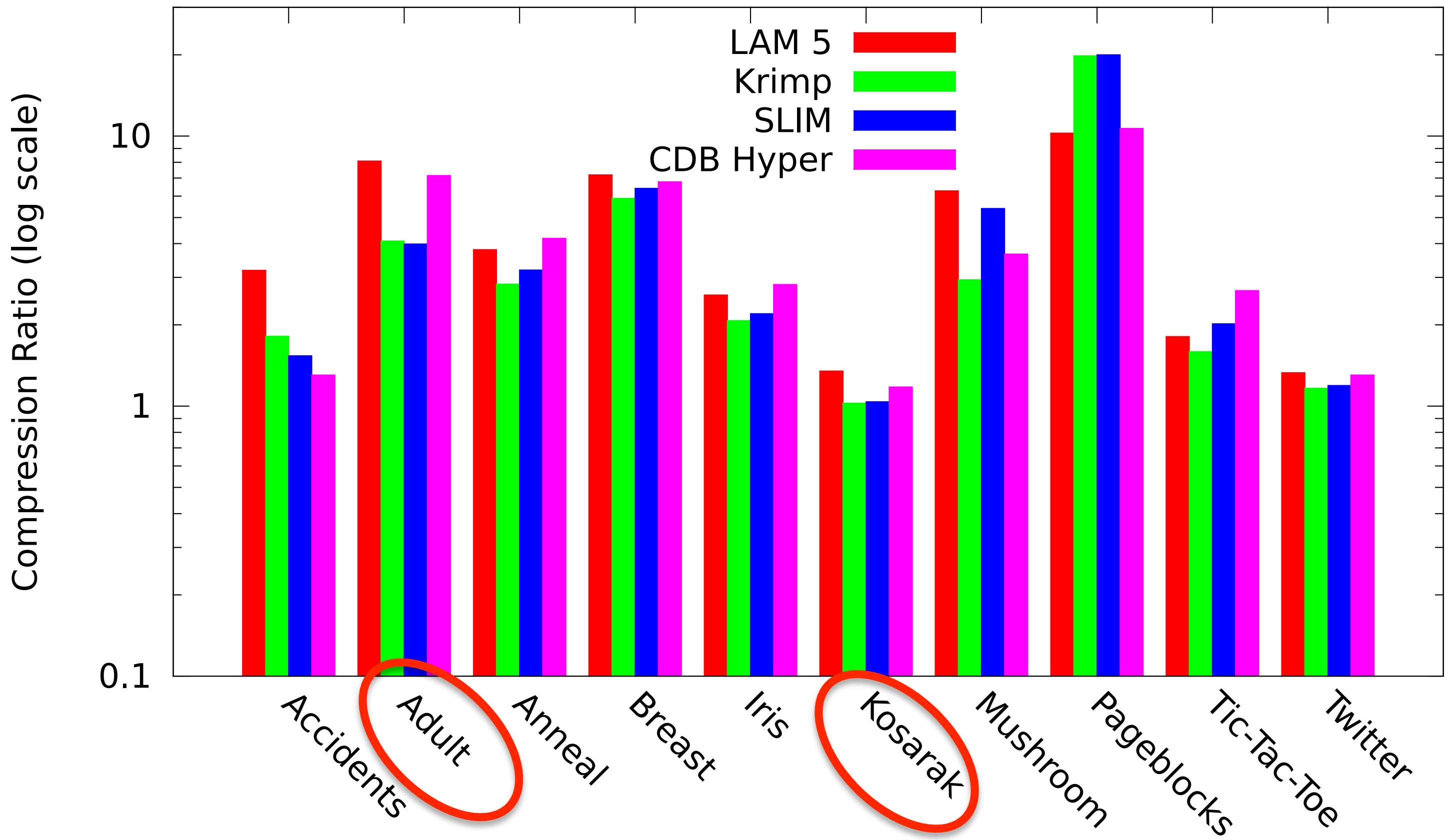
- Mined (*p*, *tlist*) pairs ordered by *utility*
- Add *p* to pattern set *P*
- In dataset *D*, Remove *p* from each row in *tlist*
- Replace with a pointer to *p* in the pattern set
- Append *P* to *D* and run LAM again on new *D*

Iterate multiple times for better compression

Experiments

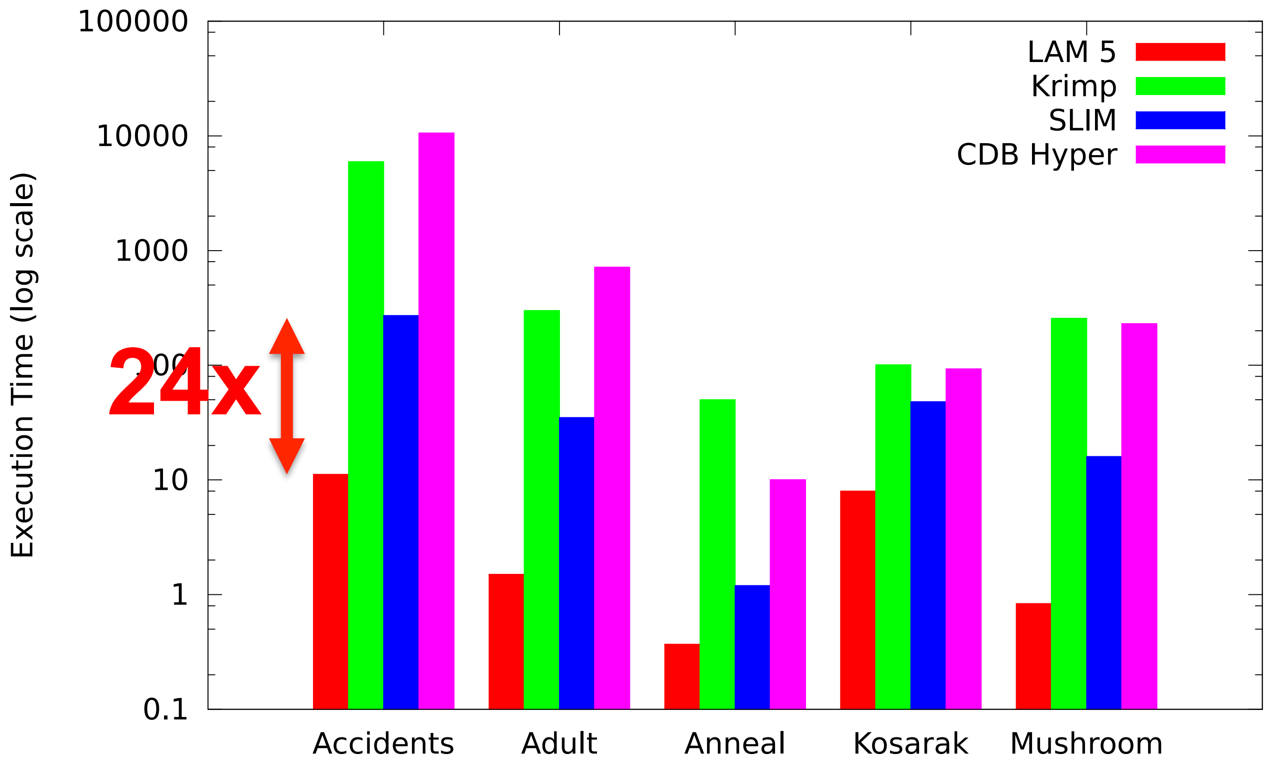
- Nine transactional datasets from UCI, FIMI
- Compare LAM to state-of-the-art
  - Krimp [Vreeken et al. 2011]
  - Slim [Smets et al. 2012]
  - CDB-Hyper [Xiang et al. 2008]
- Five web graph datasets ($|V| \sim 10^7$, $|E| \sim 10^9$)
- PLAM (*Parallel* LAM): Cluster implementation
  - Compare to Closed Itemset Mining
- Compression, execution time, scalability

UCI/FIMI: Compression

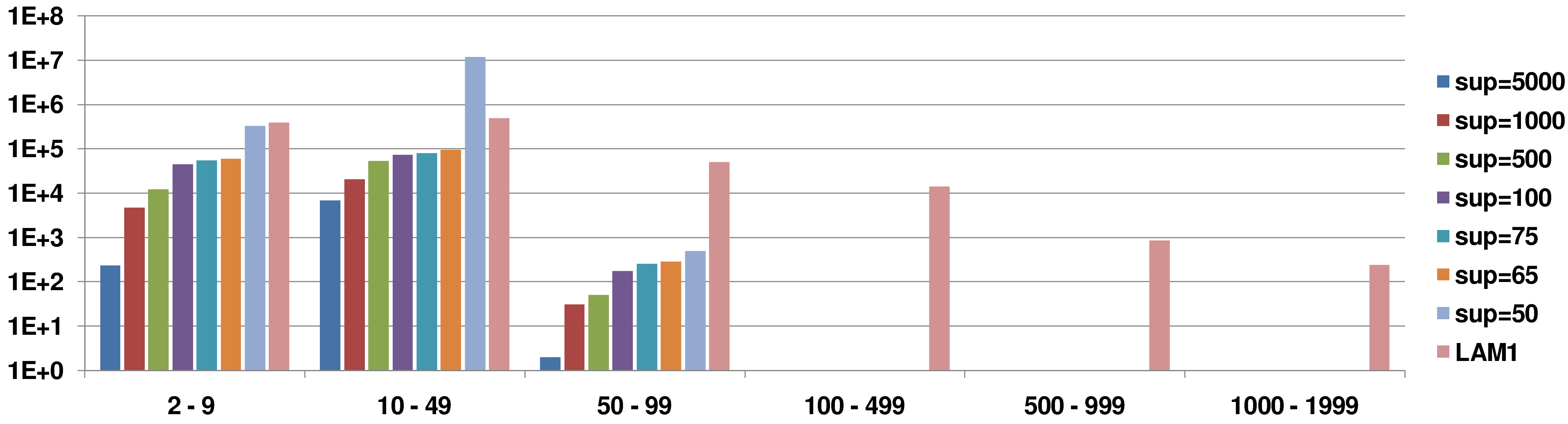


LAM achieves better compression on most datasets

UCI/FIMI: Execution time

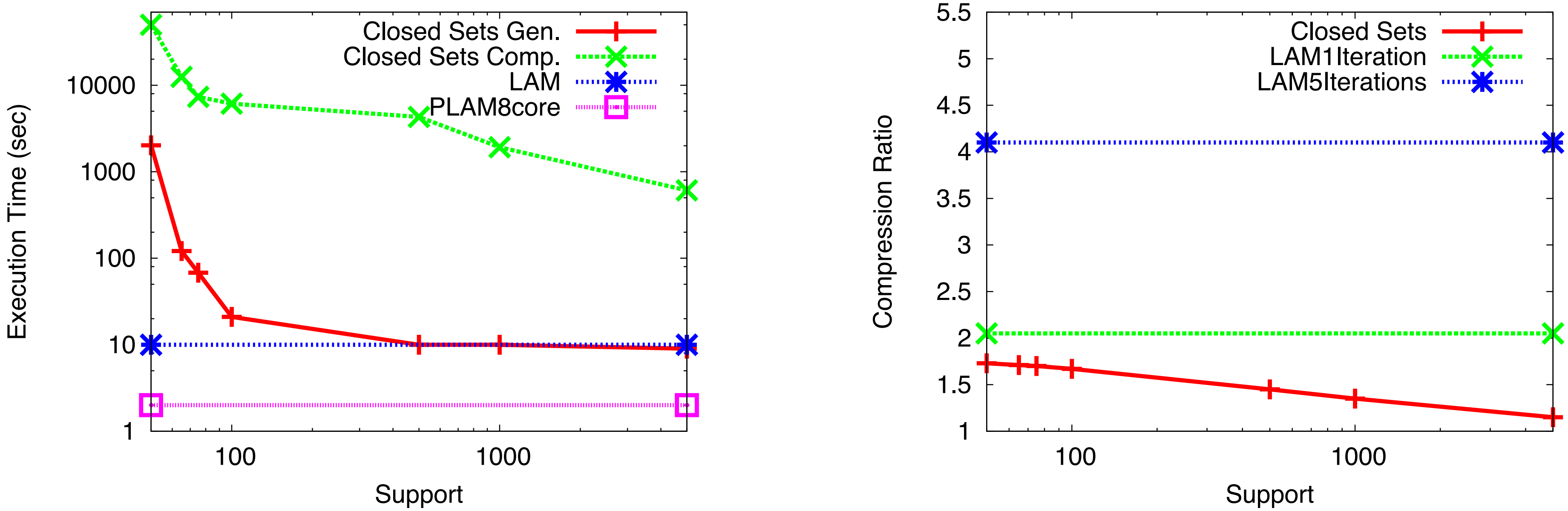


LAM is one or more orders of magnitude faster



Itemset results for various supports, grouped by set size.

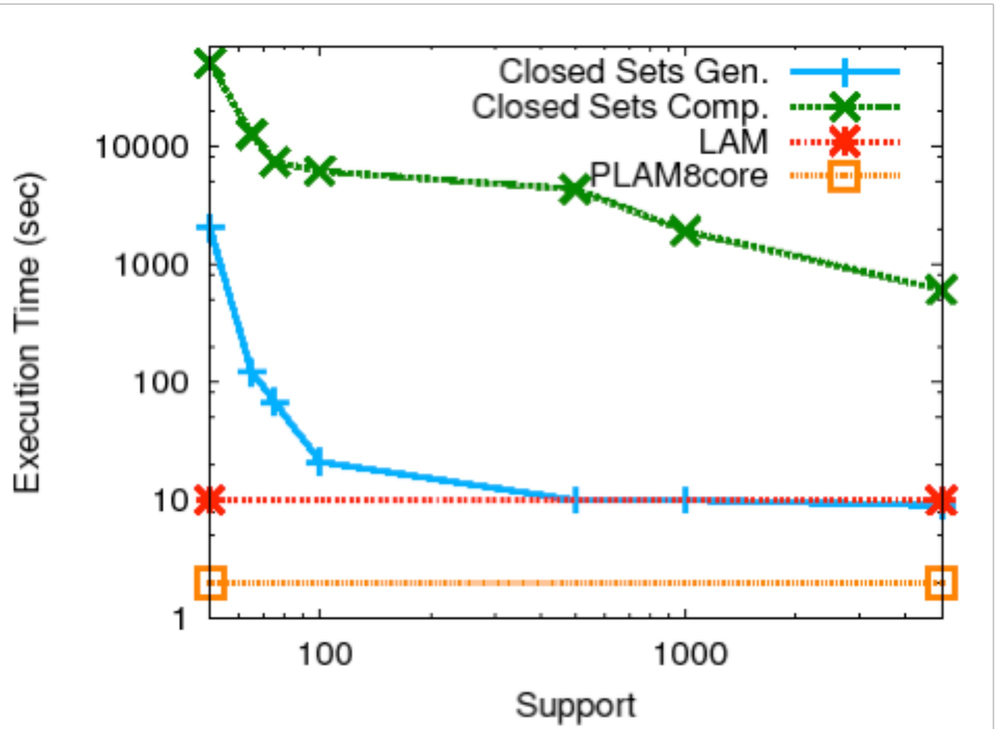
Web: Comparing LAM to Closed Sets

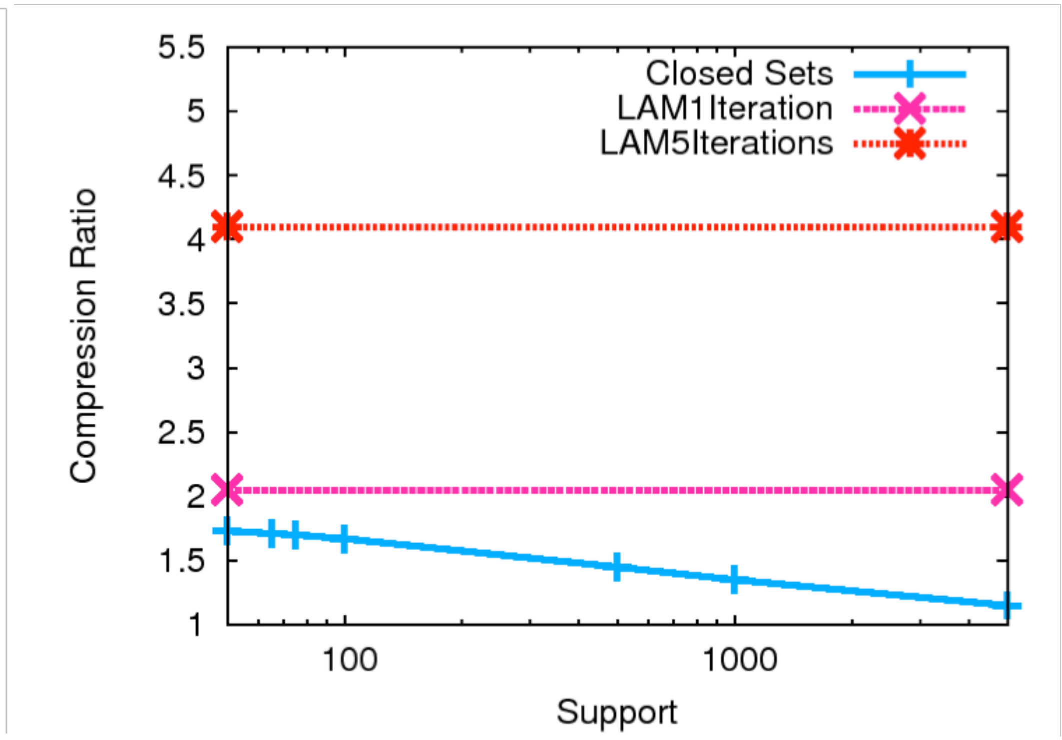


Smallest web graph dataset EU2005: |V| = 863K, |E| = 19M

- For σ < 100, Closed Sets slow at generating patterns
- Even slower at compressing
- LAM produces better compression: 2x w/ 1 iter, 4x w/ 5 iter

Web: Comparing LAM to Closed Sets

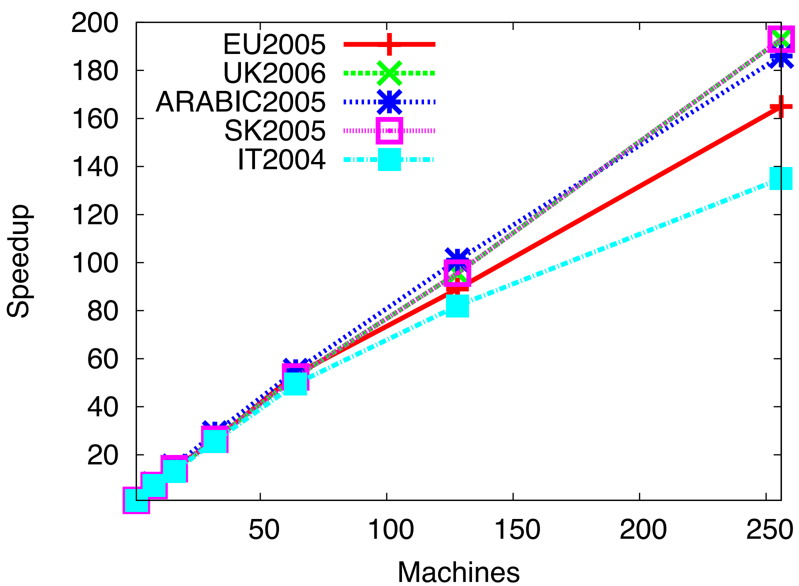


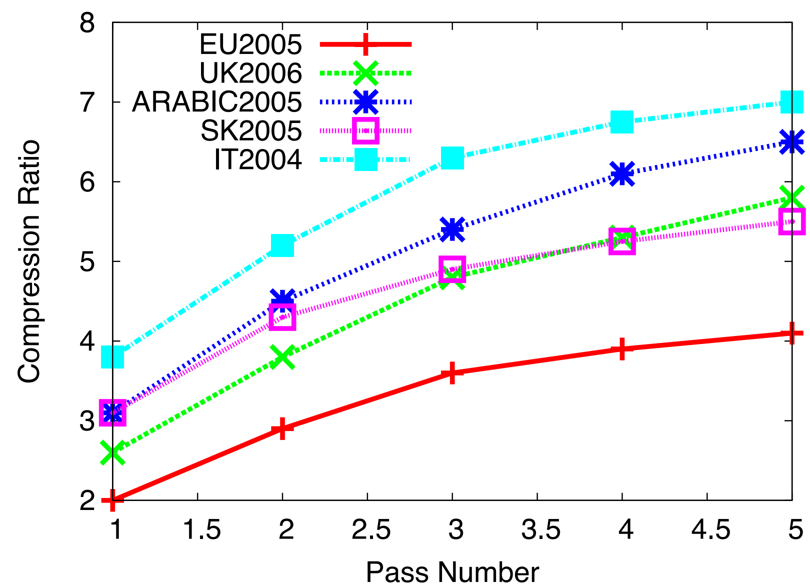


Smallest web graph dataset EU2005: |V| = 863K, |E| = 19M

Larger datasets: Better results than closed sets, in less time.

Web: Scalability





- Near-linear scalability to hundreds of machines
- Compression ratios increase over multiple passes

LAM: thoughts and future work

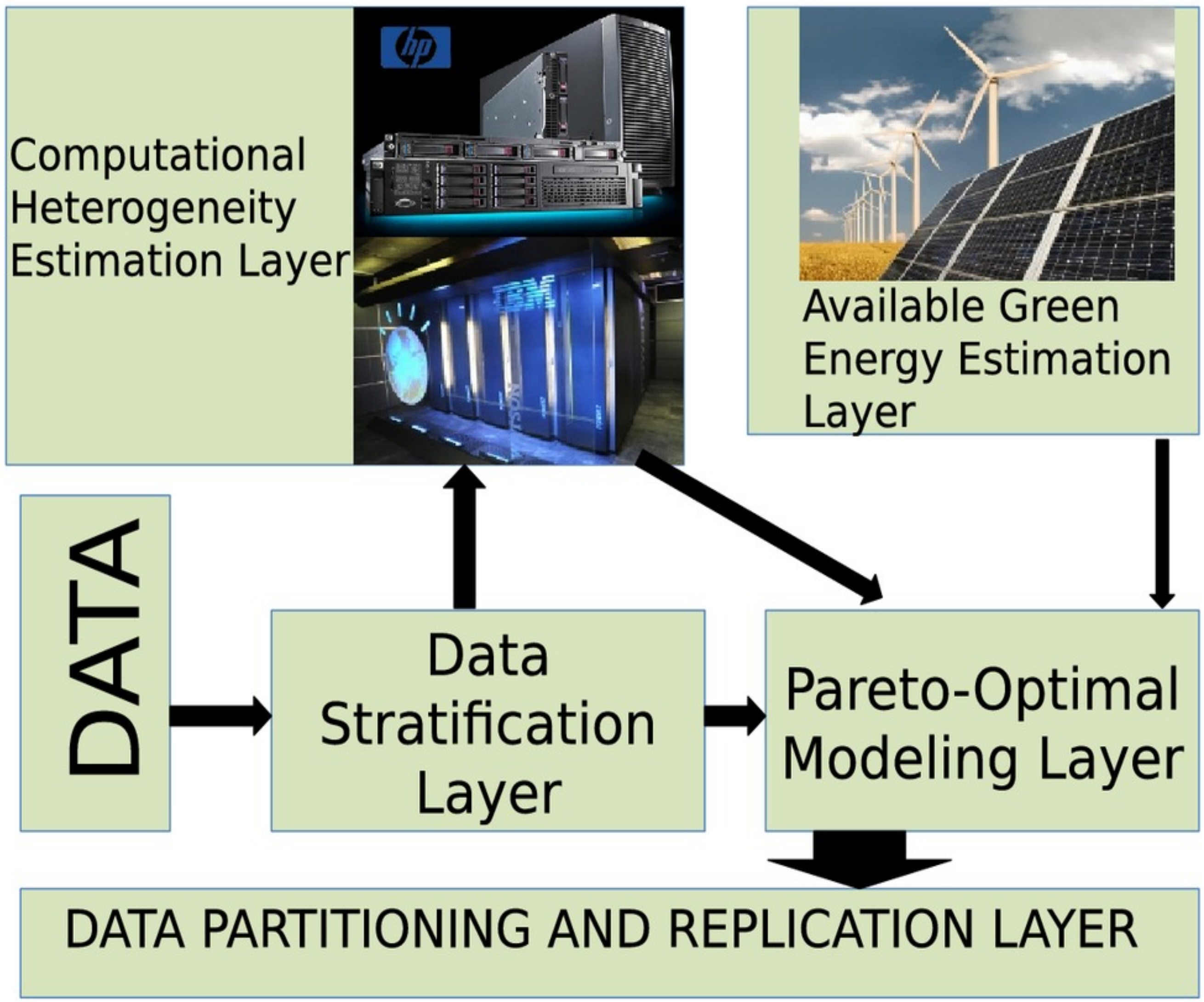
- First pattern mining algorithm to run in linearithmic time in the size of the input
- Levers Stratified Data Partitioning.
- Parameter-free – saves domain expert time
- Scales near-linearly to
  - Hundreds of cores & machines
  - Billions of transactions and items
- Future work: Can we extend similar ideas to trees, graphs and sequences?

Energy- and Heterogeneity- Aware Partitioning (ICPP'17)

- Modern Datacenters are increasingly heterogeneous
  - Computation
  - Storage
  - Green Energy Harvesting
- Sharding and placement while accounting for heterogeneity is challenging
  - Pareto Optimal Model

THE OHIO STATE UNIVERSITY

Overview of Pareto Framework



Pareto Function: Math

- Goal: Find partition size distribution that will (1) minimize the maximum execution time across partitions and (2) minimize the total dirty energy footprint

$$\underset{x_i}{minimize}\left\{max(f_i(x_i)), \sum_{i=1}^{p} g_i\,(x_i)\right\}$$

$$s.t. \sum_{i=1}^{p} x_i = N$$

For node i: $f_i(x_i) = m_i x_i + c_i$

- Predicted execution time: $f_i(x_i)$
- Regression coefficients: m_i, c_i
- Input partition size: x_i

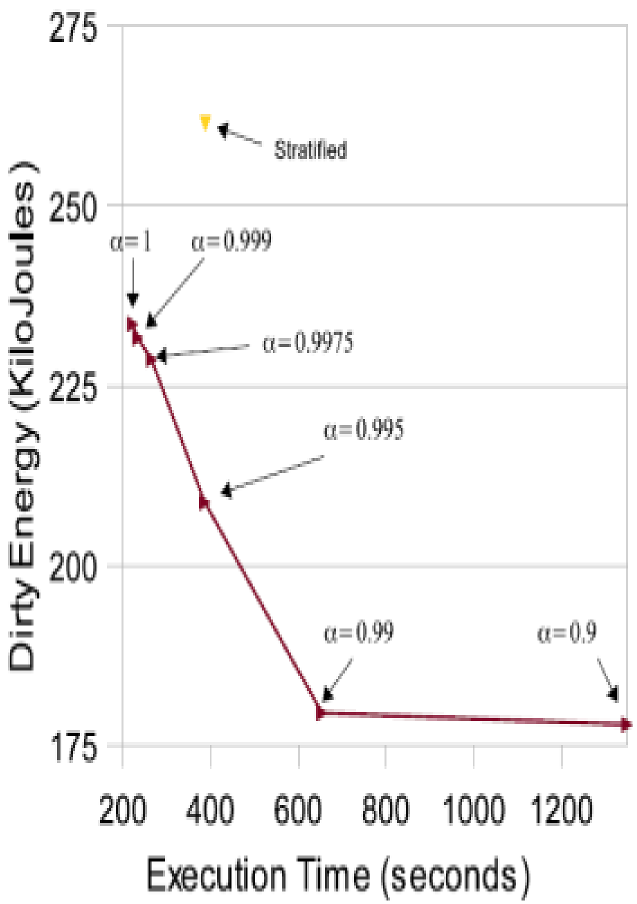
For node i, the dirty energy consumption for a job is modeled as

$$g_i(x_i) = E_i f_i(x_i) - \sum_{t=1}^{f_i(x_i)} GE_i\,(t)$$

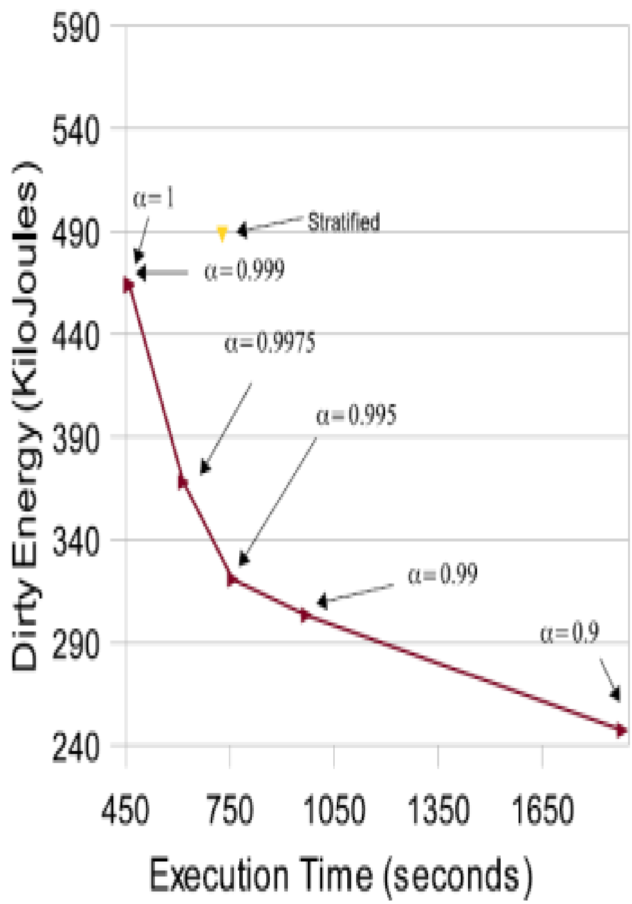
[Gori'11]

- Predicted dirty energy: $g_i(x_i)$
- Avg. total energy consumed per hour: E_i

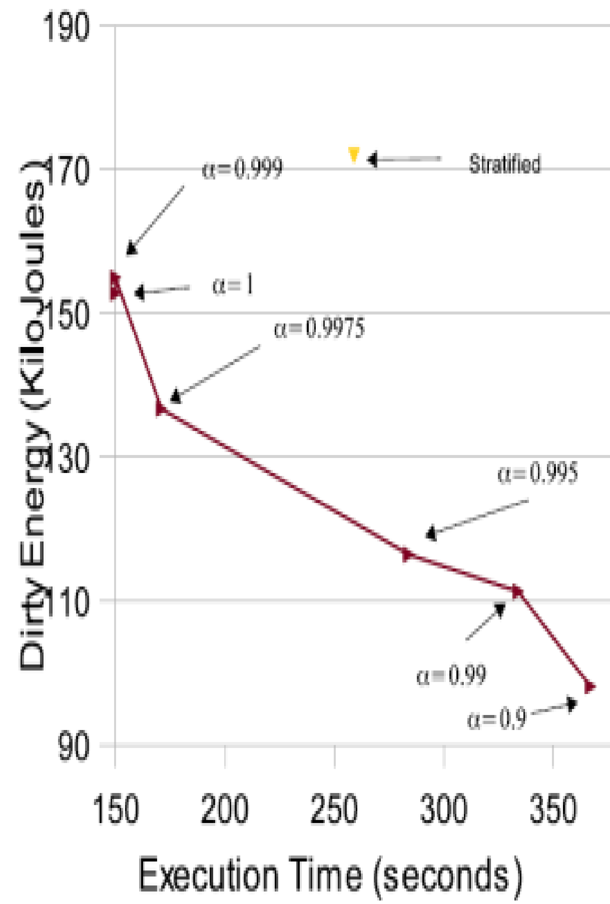
Evaluation – Pareto Frontiers



Swiss (Tree Mining)



RCV: Text Mining



UK: Web Analytics

Take Home Message

- In todays analytics world data has complex structure
- Stratitifed Data Placement has a central role to play

STRATIFIED DATA SHARDING &PLACEMENT		
HADOOP/SHARK/ Azure (HDFS/RDD/Blob)	Key Value Stores e.g. Memcached Redis	MPI & Partitioned Global Addresses Space Systems (PGAS) e.g. Global Arrays

  - Over 2 orders of magnitude improvement over state-of-art for a multitude of analytic tasks. First to explore this idea for placement.
  - Preliminary results on heterogeneous- energy-aware systems show significant promise!

Thanks

- Organizers
- Audience
- Former Students (who did all the work!)
  - Ye Wang (AirBnB), A. Chakrabarty (MSR), D. Fuhry (OSU).
- Funding agencies
  - NSF, DOE, NIH, Google, IBM