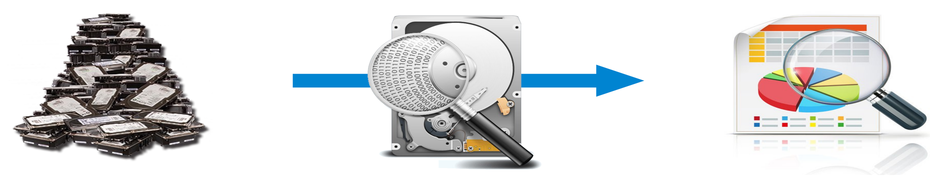
Sampling-based Randomized Algorithms for Big Data Analytics

Matteo Riondato (matteo@cs.brown.edu, http://cs.brown.edu/~matteo) – Advisor: Prof. Eli Upfal, Dept. of Computer Science, Brown University, Providence, RI

1. Motivation



=
cleaning, inspecting, transforming, modeling, ...

Needs fast algorithms ⟶ challenging due to Big Data { Volume, Variety

Volume: data is large and grows fast
Variety: no. of "questions" to answer using data is large (e.g., itemsets)

cost(analytics algorithm) = cost(Volume) + cost(Variety)

e.g., cost(APriori) = cost(size dataset) + cost(no. of patterns)

Smart algorithms may cut cost(Variety) but cost(Volume) always takes over

Idea: cut cost(Volume) by analyzing only a subset of the dataset

2. Thesis Statement and Contributions

We use VC-dimension to obtain (probabilistically-guaranteed) high-quality approximations for many data analytics tasks by processing only a small random sample of the data

Data Analytics Task	Contributions & Comparison with prev. work
Frequent Itemsets and Association Rules	Sampling algorithm – smaller sample size
	(MapReduce+sampling) alg. – more scalable
	Stat. test for false positives – more power
Betweenness Centrality	Sampling algorithm – smaller sample size
	Tighter analysis of existing algorithm
Database Query Selectivity	Sampling algorithm – smaller sample size

3. Approximations and Limitations of Classic Approach

Tradeoff between sample size and quality of approximation is well studied

Given $(u_i)_{1\le i\le n}, u_i\in[0,1]$, $(\tilde{u}_i)_{1\le i\le n}$ is (ε,δ)-approximation of $(u_i)_{1\le i\le n}$ if

$$\boxed{\Pr(\exists i \text{ s.t. } |u_i-\tilde{u}_i|>\varepsilon)<\delta}$$

Our goal: compute sample size $|\mathcal{S}|$ to obtain (ε,δ)-approximation

Classic bounds can not handle Big Data Variety (i.e., high values of n)
E.g., with Chernoff bound + Union bound, sample size is

$$|\mathcal{S}|=O\left(\frac{1}{\varepsilon^2}\left(\ln n+\ln\frac{1}{\delta}\right)\right)$$

Dependency on $\ln n$ is too much for typical data analytics tasks
E.g., in Frequent Itemsets mining, $\ln n$ is number of items, can be O(10^4).

VC-dimension overcomes this issue: $|\mathcal{S}|=O\left(\frac{1}{\varepsilon^2}\left(\mathsf{VC}((u_i)_{1\le i\le n})+\ln\frac{1}{\delta}\right)\right)$

4. Vapnik-Chervonenkis (VC)-Dimension

D: set of points (domain)
$\mathcal{F}$: collection of subsets of D (ranges) } $(D,\mathcal{F})$ rangeset

For any $C\subseteq D$, let $P_C=\{C\cap R\ :\ R\in\mathcal{F}\}(\subseteq 2^C)$
If $P_C=2^C$, C is shattered by F

$$\boxed{\mathsf{VC}(D,\mathcal{F})=\sup\{|C|\ :\ C\subseteq D\wedge P_C=2^C\}}$$

Theorem [VapnikChervonenkis'71, Li et al. '08]
Assume $\mathsf{VC}(D,\mathcal{F})\le d$. Let $0<\varepsilon,\delta<1$, π be a probabil. distrib. on D, and $\mathcal{S}$ be a collection of samples from D w.r.t. π with $\boxed{|\mathcal{S}|\ge\frac{1}{\varepsilon^2}\left(d+\log\frac{1}{\delta}\right)}$

Then, with probability $\ge 1-\delta$, $\boxed{\left|\pi(R)-\frac{1}{|\mathcal{S}|}\sum_{a\in\mathcal{S}}\mathbb{1}_R(a)\right|\le\varepsilon, \forall R\in\mathcal{F}}$

i.e., can compute (ε,δ)-approximation of $(\pi(R))_{R\in\mathcal{F}}$

5. Mining Frequent Itemsets and Assoc. Rules (1)

R., Upfal, "Efficient Discovery of Association Rules and Frequent Itemsets through Sampling with Tight Performance Guarantees", ECML PKDD'12, ACM TKDD'14
R., DeBrabant, Fonseca, Upfal, "PARMA: A Parallel Randomized Algorithm for Approximate Association Rules Mining in MapReduce", ACM CIKM'12
R., Vandin, "Finding the True Frequent Itemsets", SIAM SDM'14

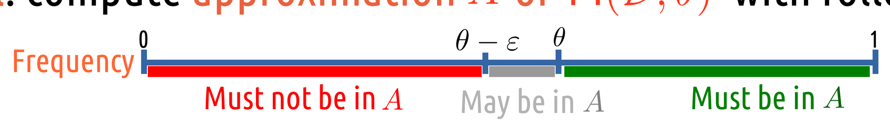
Settings: Transactional dataset $\mathcal{D}$
Frequency of itemset X in $\mathcal{D}$:
$f_{\mathcal{D}}(X)$: fraction of $\mathcal{D}$ containing X
e.g., $f_{\mathcal{D}}(\{\text{bread,milk}\})=3/5$

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Tan et al. - Introduction to Data Mining

Transaction
Items from $\mathcal{I}$
Itemset

Task: Given $\theta\in(0,1)$, compute $\mathsf{FI}(\mathcal{D},\theta)=\{(X,f_{\mathcal{D}}(X))\ :\ f_{\mathcal{D}}(X)\ge\theta\}$
Exact algorithms (APriori, ...) scan dataset multiple times: too slow
Our goal: compute approximation A of $\mathsf{FI}(\mathcal{D},\theta)$ with following properties:



Can achieve goal by computing $(\varepsilon/2,\delta)$-approximation of $(f_{\mathcal{D}}(X))_{X\subseteq\mathcal{I}}$

Algorithm
$d\leftarrow$ max integer g s.t. $\mathcal{D}$ contains $\ge g$ transactions of length $\ge g$
$r\leftarrow(2/\varepsilon^2)(d+\ln(1/\delta))$
$\mathcal{S}\leftarrow$ random sample of $\mathcal{D}$ of size r
return $\mathsf{FI}(\mathcal{S},\theta-\varepsilon/2)$

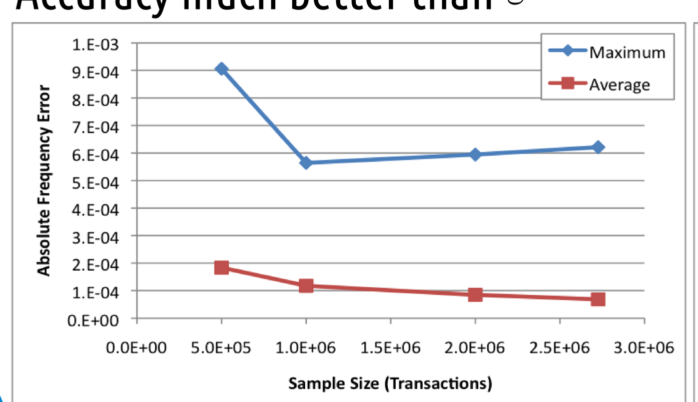
Theorem: $\mathsf{FI}(\mathcal{S},\theta-\varepsilon/2)$ has the required properties

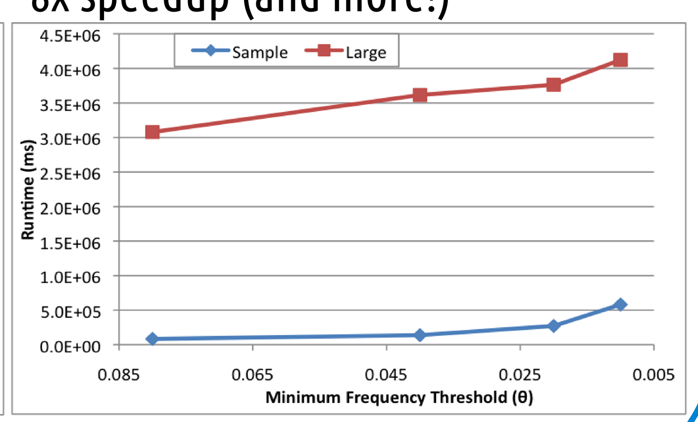
Rangeset for $\mathsf{FI}(\mathcal{D},\theta)$: $D=\mathcal{D},\mathcal{F}=\{\{\tau\in\mathcal{D}\ :\ X\in\tau\},X\subseteq\mathcal{I}\}$

Theorem: $\mathsf{VC}(\mathcal{D},\mathcal{F})\le d$

Evaluation: dataset from FIMI'04 repository

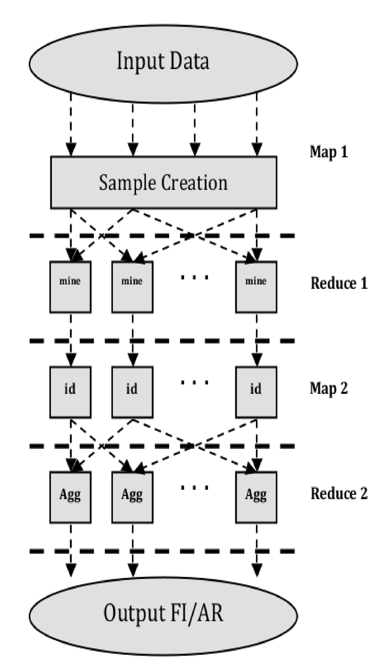
Accuracy much better than ε | 8x speedup (and more!)



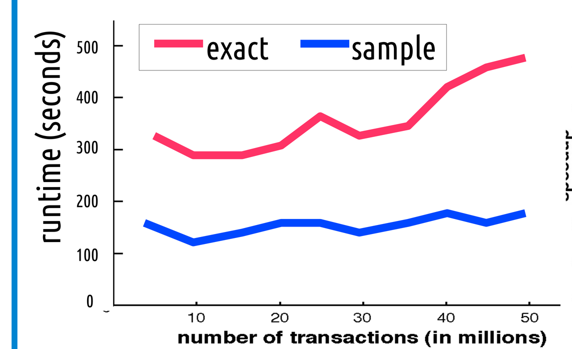


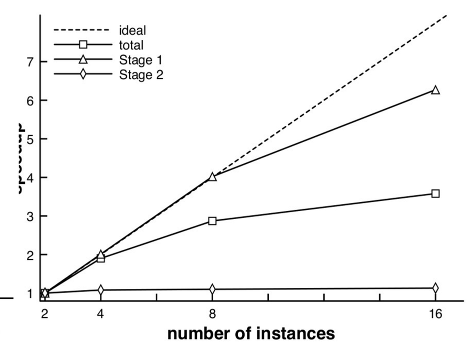
6. Mining Frequent Itemsets and Assoc. Rules (2)

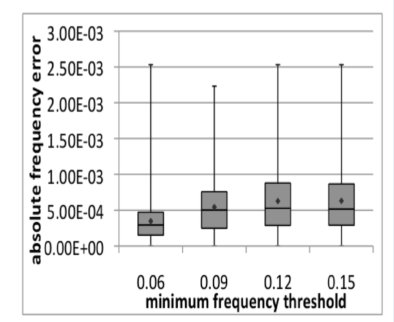
Idea: improve scalability and accuracy using MapReduce
Previous algorithm used in boosting-like approach
Algorithm
1st Map: Create many samples in parallel.
No. of samples and size to exploit resources
1st Reduce: Mine each sample on dedicated reducer
2nd Map: Identity function
2nd Reduce: Aggregate and filter mining results:
1) only itemsets that were frequent in many samples are considered frequent and sent to output
2) estimation of frequency is median of estimations



Evaluation







Important issue: avoid false positives in mining results
Settings: π: unknown process generating transactions,
$\mathcal{D}$: collection of i.i.d. samples from
Itemset X, $t_\pi(X)=\sum_{Y\subseteq\mathcal{I},X\subseteq Y}\pi(Y)$: true frequency of X

Task: Given $\theta\in(0,1)$, compute $\mathsf{TFI}(\pi,\theta)=\{(X,t_\pi(X)\ :\ t_\pi(X)\ge\theta\}$
Can't be solved exactly with finite $\mathcal{D}$
Our goal: compute $\theta'\ge\theta$ such that $\mathsf{FI}(\mathcal{D},\theta')\subseteq\mathsf{TFI}(\pi,\theta)$
At same time, try to maximize $|\mathsf{FI}(\mathcal{D},\theta')|$, i.e., minimize $\theta'-\theta$

Intuition: Variant of Knapsack to compute VC-Dim of Negative Border
NP but do not need exact solution
Results: Much more power (larger $|\mathsf{FI}(\mathcal{D},\theta')|$) w.r.t. traditional methods

7. Estimating Betweenness Centrality

R., Kornaropoulos, "Fast Approximation of Betweenness Centrality through Sampling", ACM WSDM'14

Betweenness centrality: measure of vertex importance in graphs
Settings: Graph $G=(V,E)$ $|V|=n,|E|=m$
$\mathsf{b}(v)$: fraction of Shortest Paths in G that pass through v

$$\boxed{\mathsf{b}(v)=\frac{1}{n(n-1)}\sum_{p_{uw}\in\mathbb{S}_G}\frac{\mathbb{1}_{\mathcal{T}_v}(p_{uw})}{\sigma_{uw}}}$$

$\mathbb{S}_G$: all SPs in G
$\mathcal{S}_{uv}$: SPs from u to v, $\sigma_{uv}=|\mathcal{S}_{uv}|$
$\mathcal{T}_v=\{p\in\mathbb{S}_G\ :\ v\in\mathsf{Int}(p)\}$

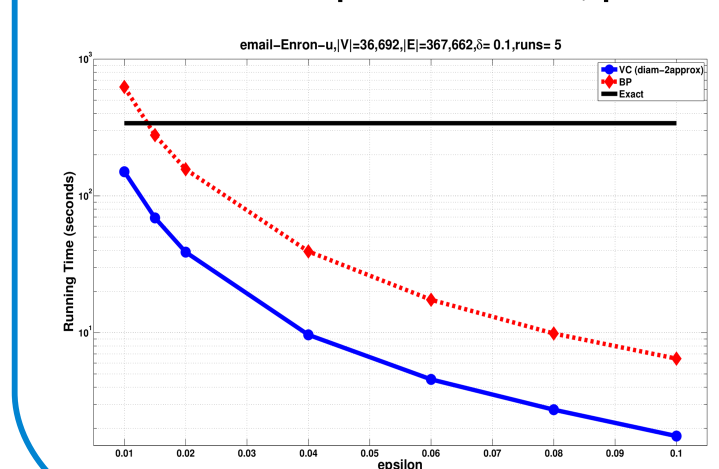
Exact algorithm for $(\mathsf{b}(v))_{v\in V}$ takes time $O(nm+n^2\log n)$ [Brandes01]
Our goal: fast computation of (ε,δ)-approximation using sampling
Algorithm
$\mathsf{VD}(G)\leftarrow\max\{|p|,p\in\mathbb{S}_G\}$
$r\leftarrow(1/2\varepsilon^2)(\lfloor\log_2(\mathsf{VD}(G)-2)\rfloor+1+\ln(1/\delta))$
$\tilde{\mathsf{b}}(v)\leftarrow 0,\forall v\in V$
for $i\leftarrow 1,\dots,r$
$(u,v)\leftarrow$ random pair of vertices
$\mathcal{S}_{uv}\leftarrow$ all SPs from u to v (BFS, Dijkstra, bidirectional search)
$p\leftarrow$ random SP from $\mathcal{S}_{uv}$
$\tilde{\mathsf{b}}(w)\leftarrow\tilde{\mathsf{b}}(w)+1/r,\forall w\in\mathsf{Int}(p)$
return $\tilde{\mathsf{b}}(v),\forall v\in V$

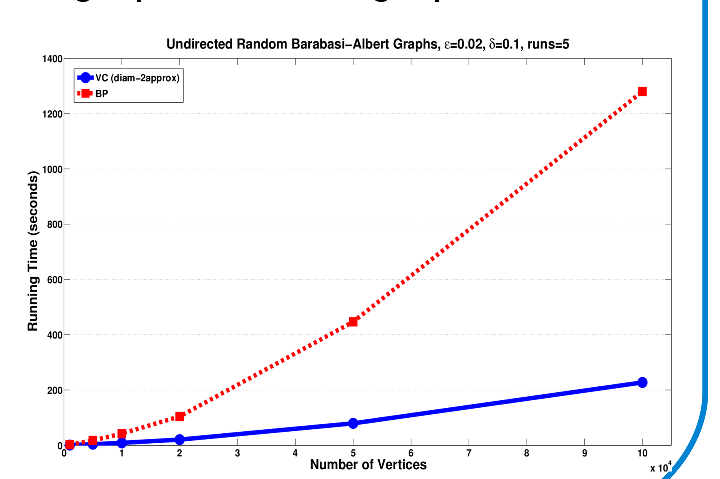
Theorem: $(\tilde{\mathsf{b}}(v))_{v\in V}$ is a (ε,δ)-approximation for $(\mathsf{b}(v))_{v\in V}$

Rangeset for betweenness centrality: $D=\mathbb{S}_G,\mathcal{F}=\{\mathcal{T}_v,v\in V\}$

Theorem: $\mathsf{VC}(\mathbb{S}_G,\mathcal{F})\le\lfloor\log_2(\mathsf{VD}(G)-2)\rfloor+1$

Evaluation: C implementation, patch for igraph, on SNAP graphs

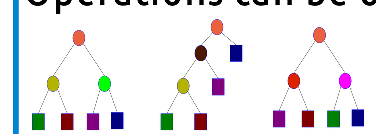




8. Estimating the Selectivity of Database Queries

R., Akdere, Cetintemel, Zdonik, Upfal, "The VC-Dimension of SQL Queries and Selectivity Estimation through Sampling", ECML PKDD'11

Database queries are composed of elementary operations (selection, joins,...)
Operations can be organized in many ways with same output (query plans)



DBMS must choose plan with smallest execution time
NP-hard problem. Must use heuristics.

Common heuristic: perform small selectivity operations first

Selectivity of a query q: $\boxed{\sigma_{\mathrm{DB}}(q)=\frac{\#\text{rows in output of }q}{\#\text{rows in input of }q}}$

Exact value not available before execution, must approximate.
DBMSs use histograms: independence, uniformity assumption ⟵ not true

Our goal: Given class of queries Q, compute sample size $|\mathcal{S}|$ such that $(\sigma_{\mathcal{S}}(q))_{q\in Q}$ is (ε,δ)-approximation to $(\sigma_{\mathrm{DB}}(q))_{q\in Q}$

Rangeset for query selectivity: $D=\mathrm{DB},\mathcal{F}=\{\text{output of }q,q\in Q\}$

Theorem: Let k= max no. of join operations in a query from Q
b= max no. of selection conditions in a query from Q
m= max no. of columns in a table
Then, $\mathsf{VC}(\mathrm{DB},\mathcal{F})=\tilde{O}(k^2mb)$

Evaluation:
Sample can fit into main memory ⟶ estimation is fast
Estimate much closer to real value than guaranteed
Beat PostgreSQL and SQLServer by orders of magnitude