

Scalable Histograms on Larger Probabilistic Data

Mingwang Tang and Feifei Li



September 21, 2015

New Challenges

- Large scale data size
- Distributed data sources
- Uncertainty



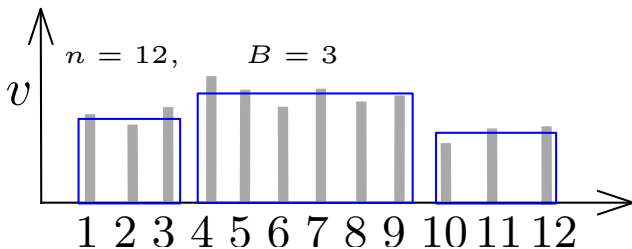
Data synopsis on large probabilistic data

- Scalable histograms on large probabilistic data

Histograms on deterministic data

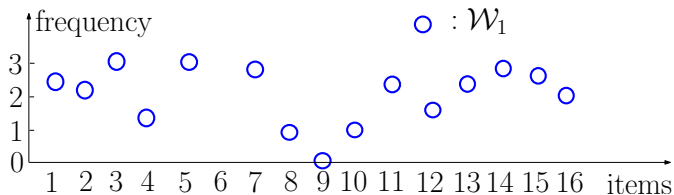
- V-optimal histogram: Given a frequency vector $\vec{v} = \{v_1, \dots, v_n\}$, where v_i is the frequency of item i in $[n]$, a space budget B , it seeks to minimize the SSE error:

$$\min \left\{ \sum_{k=1}^B \sum_{i=s_k}^{e_k} (v_i - \hat{b}_k)^2 \right\} \quad (1)$$



- Optimal B-bucket histogram takes $O(Bn^2)$ time.

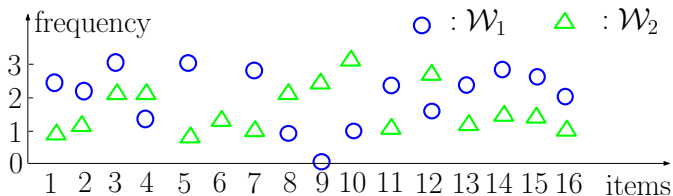
- Probabilistic database \mathcal{D} on domain $[n] = \{1, \dots, n\}$



- $\mathcal{D} = \{g_1, g_2, \dots, g_n\}$ where

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\} \quad (2)$$

- Probabilistic database \mathcal{D} on domain $[n] = \{1, \dots, n\}$



- $\mathcal{D} = \{g_1, g_2, \dots, g_n\}$ where

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\} \quad (2)$$

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\}$$

Tuple Model

- Each tuple $t_j = \langle (t_{j1}, p_{j1}), \dots, (t_{j\ell_j}, p_{j\ell_j}) \rangle$. Each t_{jk} is drawn from $[n]$ for $k \in [1, \ell_j]$.
- $1 - \sum_{k=1}^{\ell_j} p_{jk}$ specify the possibility that t_j generates no item.

t_1	$\{(1, 0.2), (3, 0.3), (7, 0.2)\}$
t_2	$\{(3, 0.3), (5, 0.1), (9, 0.4)\}$
t_3	$\{(3, 0.5), (10, 0.4), (13, 0.1)\}$
\vdots	\vdots
$t_{ \tau }$	\vdots

Probabilistic Data Models

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\}$$

Tuple Model

- Each tuple $t_j = \langle (t_{j1}, p_{j1}), \dots, (t_{j\ell_j}, p_{j\ell_j}) \rangle$. Each t_{jk} is drawn from $[n]$ for $k \in [1, \ell_j]$.
- $1 - \sum_{k=1}^{\ell_j} p_{jk}$ specify the possibility that t_j generates no item.

t_1	$\{(1, 0.2), (3, 0.3), (7, 0.2)\}$
t_2	$\{(3, 0.3), (5, 0.1), (9, 0.4)\}$
t_3	$\{(3, 0.5), (10, 0.4), (13, 0.1)\}$
\vdots	\vdots
$t_{ \tau }$	\vdots

Probabilistic Data Models

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\}$$

Value Model

- Each tuple $t_j = \langle j : f_j = ((f_{j1}, p_{j1}), \dots, (f_{j\ell_j}, p_{j\ell_j})) \rangle$, j is drawn from $[n]$.
- $\Pr(f_j = 0) = 1 - \sum_{k=1}^{\ell_j} p_{jk}$

t_1	$\{ \langle 1, (50, 0.2), (7, 0.1), (14, 0.2) \rangle \}$
t_2	$\{ \langle 2, (6, 0.4), (7, 0.3), (15, 0.3) \rangle \}$
t_3	$\{ \langle 3, (10, 0.3), (15, 0.2), (20, 0.5) \rangle \}$
•	•
•	•
•	•
t_n	

Probabilistic Data Models

$$g_i = \{(g_i(W), \Pr(W)) \mid W \in \mathcal{W}\}$$

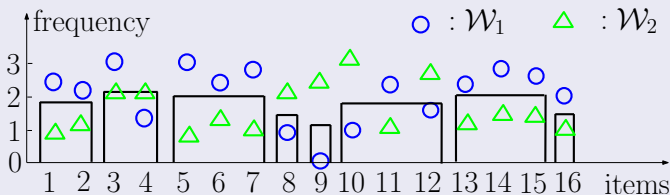
Value Model

- Each tuple $t_j = \langle j : f_j = ((f_{j1}, p_{j1}), \dots, (f_{j\ell_j}, p_{j\ell_j})) \rangle$, j is drawn from $[n]$.
- $\Pr(f_j = 0) = 1 - \sum_{k=1}^{\ell_j} p_{jk}$

t_1	$\{ \langle 1, (50, 0.2), (7, 0.1), (14, 0.2) \rangle \}$
t_2	$\{ \langle 2, (6, 0.4), (7, 0.3), (15, 0.3) \rangle \}$
t_3	$\{ \langle 3, (10, 0.3), (15, 0.2), (20, 0.5) \rangle \}$
\vdots	\vdots
\vdots	\vdots
\vdots	\vdots
t_n	

Histograms on Probabilistic data

Possible world semantic



- g_i : frequency of item i becomes random variable across possible worlds

Expectation based histogram

$$\mathcal{H}(n, B) = \min \left\{ \mathbf{E}_{\mathcal{W}} \left[\sum_{k=1}^B \sum_{j=s_k}^{e_k} (g_j - \hat{b}_k)^2 \right] \right\}.$$

- [ICDE09] G. Cormode et al., Histograms and wavelets on probabilistic data, ICDE 2009
- [VLDB09] G. Cormode et al., Probabilistic histograms for probabilistic data, VLDB 2009

Efficient computation of bucket error

- The optimal B bucket histogram takes $O(Bn^2)$ time.
- [TKDE10] shows that the minimal error of a bucket $b = (s, e, \hat{b})$ is:

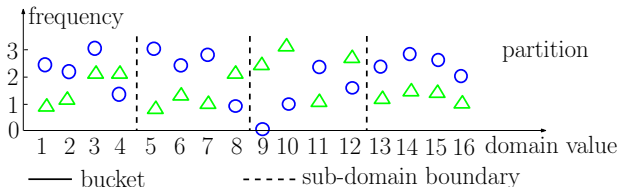
$$SSE(b, \hat{b}) = \sum_{i=s}^e \mathbf{E}_{\mathcal{W}}[g_i^2] - \frac{1}{e-s+1} \mathbf{E}_{\mathcal{W}}[\sum_{i=s}^e g_i]^2. \quad (3)$$

by setting $\hat{b} = \frac{1}{e-s+1} \mathbf{E}_{\mathcal{W}} [\sum_{i=s}^e g_i]$.

- Based on two precomputed arrays (A, B) , $SSE(b, \hat{b})$ can be computed in constant time.
- [TKDE10] G. Cormode et al., Histograms and wavelets on probabilistic data, TKDE 2010

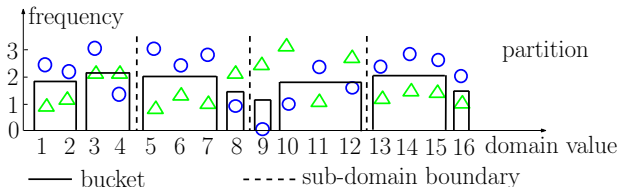
P_{MERGE} method based on partition and merge principle

- Partition phase: partition the domain n into m sub-domain of equal size and compute the local optimal B buckets for each sub-domain.
- Merge phase: merge mB input buckets from the partition phase into B buckets.



P_{MERGE} method based on partition and merge principle

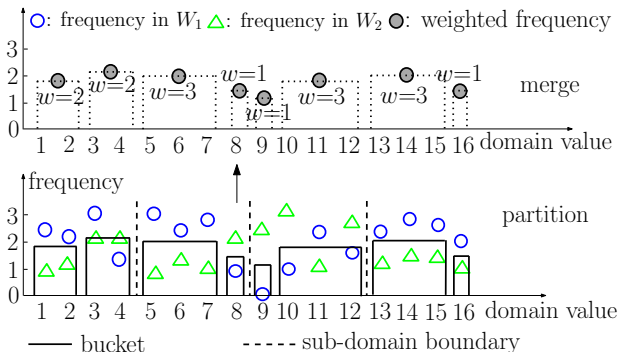
- Partition phase: partition the domain n into m sub-domain of equal size and compute the local optimal B buckets for each sub-domain.
- Merge phase: merge mB input buckets from the partition phase into B buckets.



P_{MERGE} Method

P_{MERGE} method based on partition and merge principle

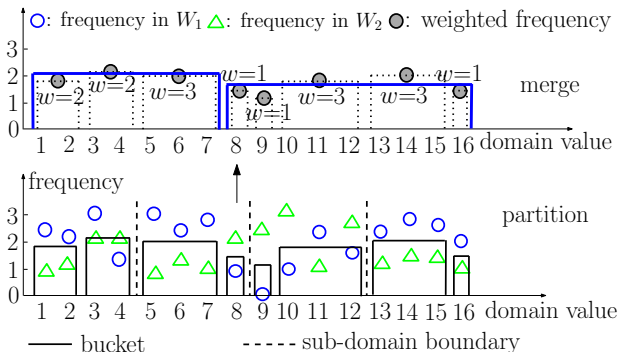
- Partition phase: partition the domain n into m sub-domain of equal size and compute the local optimal B buckets for each sub-domain.
- Merge phase: merge mB input buckets from the partition phase into B buckets.



P_{MERGE} Method

P_{MERGE} method based on partition and merge principle

- Partition phase: partition the domain n into m sub-domain of equal size and compute the local optimal B buckets for each sub-domain.
- Merge phase: merge mB input buckets from the partition phase into B buckets.

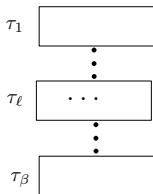


Recursive Merging Method

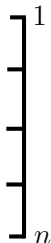
- PMERGE method:
 - Approximation quality: PMERGE produces a $\sqrt{10}$ approximation in $O(N + Bn^2/m + B^3m^2)$ time.
- Recursive merging (RPMERGE):
 - Partition $[n]$ into m^ℓ subdomains, producing Bm^ℓ .
 - Using ℓ iterations and each iteration reduce the domain size by a factor of m .
 - Takes $O(N + B \frac{n^2}{m^\ell} + B^3 \sum_{i=1}^{\ell} m^{(i+1)})$ time and the RPMERGE method gives a $10^{\frac{\ell}{2}}$ approximation of the optimal B -buckets histogram found by OPTHIST.
- In practice, PMERGE and RPMERGE always provide close to optimal approximation quality as shown in our experiments.

- Partition phase in the distributed environment

Probabilistic Database \mathcal{D}



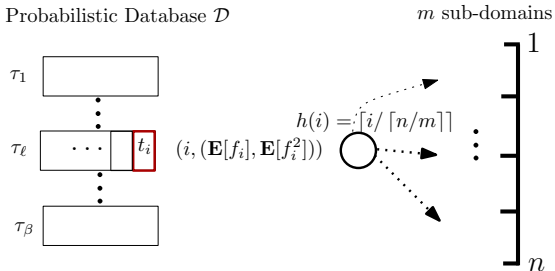
m sub-domains



Communication cost

- Computing A_k, B_k arrays in the partition phase
 - Tuple model: $O(\beta n)$ bytes.
 - Value model: $O(n)$ bytes.
- $O(Bm)$ bytes in the merge phase for both models.

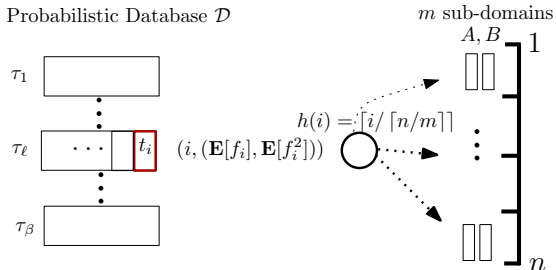
- Partition phase in the distributed environment



Communication cost

- Computing A_k, B_k arrays in the partition phase
 - Tuple model: $O(\beta n)$ bytes.
 - Value model: $O(n)$ bytes.
- $O(Bm)$ bytes in the merge phase for both models.

- Partition phase in the distributed environment



Communication cost

- Computing A_k, B_k arrays in the partition phase
 - Tuple model: $O(\beta n)$ bytes.
 - Value model: $O(n)$ bytes.
- $O(Bm)$ bytes in the merge phase for both models.

PMERGE Based on Sampling

Sampling A, B arrays in the partition phase

$$A_k[j] = \sum_{i=1}^i \mathbf{E}[f_j^2], B_k[j] = \sum_{i=1}^j \mathbf{E}[f_i]$$

Estimate A_k, B_k arrays using quantile sampling

$E[f_i]$	2	3	5	9	...
----------	---	---	---	---	-----

item: 1 2 3 4 ...

PMERGE Based on Sampling

Sampling A, B arrays in the partition phase

$$A_k[j] = \sum_{i=1}^i \mathbf{E}[f_j^2], B_k[j] = \sum_{i=1}^j \mathbf{E}[f_i]$$

Estimate A_k, B_k arrays using quantile sampling

$E[f_i]$	2	3	5	9	...
----------	---	---	---	---	-----

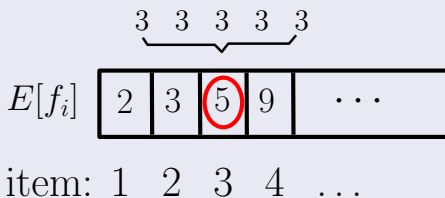
item: 1 2 3 4 ...

PMERGE Based on Sampling

Sampling A, B arrays in the partition phase

$$A_k[j] = \sum_{i=1}^j \mathbf{E}[f_i^2], B_k[j] = \sum_{i=1}^j \mathbf{E}[f_i]$$

Estimate A_k, B_k arrays using quantile sampling



PMERGE Based on Sampling

Sampling A, B arrays in the partition phase

$$A_k[j] = \sum_{i=1}^i \mathbf{E}[f_j^2], B_k[j] = \sum_{i=1}^j \mathbf{E}[f_i]$$

Estimate A_k, B_k arrays using quantile sampling

$$p = \min\left\{\Theta\left(\frac{\sqrt{\beta}}{\epsilon N}\right), \Theta\left(\frac{1}{\epsilon^2 N}\right)\right\}$$

3 ③ 3 3 3

$E[f_i]$	2	3	5	9	...
----------	---	---	---	---	-----

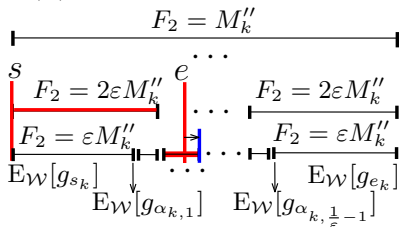
item: 1 2 3 4 ...

PMERGE Based on Sketch

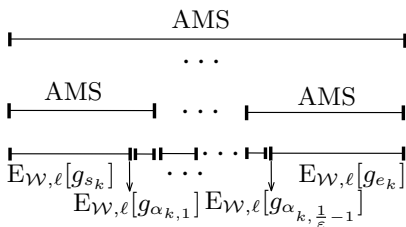
Tuple model A, B arrays

- Estimate $F_2 = \sum_{i=s_k}^j (\sum_{\ell=1}^{\beta} \mathbf{E}_{\mathcal{W},\ell}[g_i])^2$ using AMS Sketch techniques and binary decomposition of domain $[s_k, e_k]$.

(a) binary decomposition



(b) local Q-AMS



- 1 Optimal B -buckets Histograms
- 2 Approximate Histograms
- 3 PMERGE Based on Sampling
- 4 Experiments**

- Generate tuple model and the value model dataset using the client id filed of 1998 WorldCup dataset and atmospheric measurements from the SAMOS project.
- The default experimental parameters:

Symbol	Definition	Default Value
B	number of buckets	400
n	domain size	100k (600k)
l	depth of recursions	2

Running time:

- n : domain size

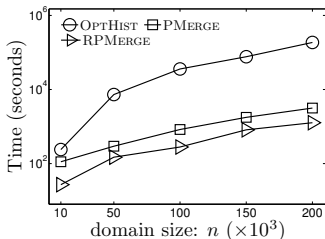


Figure: Tuple Model

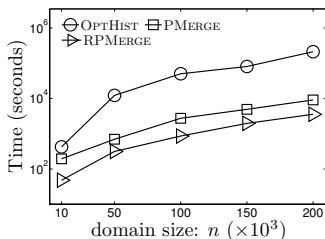


Figure: Value Model

Approximation Ratio:

- n : domain size

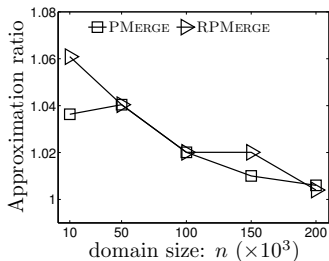


Figure: Tuple Model

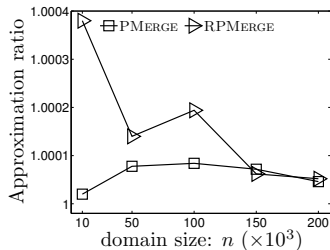


Figure: Value Model

Running time on large scale probabilistic data

- n : domain size

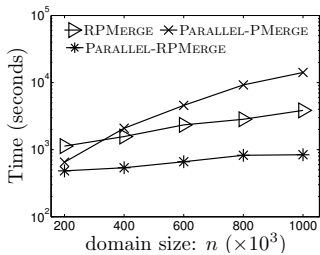


Figure: Tuple Model

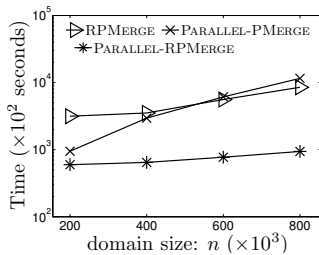


Figure: Value Model

- Conclusion
 - Novel approximation methods for constructing scalable histograms on large probabilistic data.
 - The quality of the approximate histograms are almost as good as the optimal histogram in practice.
 - Extended the techniques to distributed and parallel settings to further improve scalability.
- Future work
 - extend our study to probabilistic histograms with pdf bucket representatives and handle histogram of other error metrics

Thank You

Q and A