Scalable Histograms on Larger Probabilistic Data

Mingwang Tang and Feifei Li

THE UNIVERSITY OF UTAH®
School of Computing, University of Utah

August 24, 2014
Introduction

New Challenges

- Large scale data size
- Distributed data sources
- Uncertainty

Data synopsis on large probabilistic data

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

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

Optimal B-bucket histogram takes $O(Bn^2)$ time.
Probabilistic database $\mathcal{D}$ on domain $[n] = \{1, \ldots, n\}$

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

$$g_i = \{(g_i(W), \Pr(W))|W \in \mathcal{W}\}$$ (2)
Probabilistic database $\mathcal{D}$ on domain $[n] = \{1, \ldots, n\}$

\[ \mathcal{D} = \{g_1, g_2, \ldots, g_n\} \text{ where} \]

\[ g_i = \{(g_i(W), \Pr(W)) | W \in \mathcal{W}\} \] (2)
\[ g_i = \{(g_i(W), \Pr(W)) | W \in \mathcal{W}\} \]

**Tuple Model**

- Each tuple \( t_j = \langle (t_{j1}, p_{j1}), \ldots, (t_{j\ell_j}, p_{j\ell_j}) \rangle \). 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.

\[
\begin{array}{l|l}
  t_1 & \{(1, 0.2), (3, 0.3), (7, 0.2)\} \\
  t_2 & \{(3, 0.3), (5, 0.1), (9, 0.4)\} \\
  t_3 & \{(3, 0.5), (10, 0.4), (13, 0.1)\} \\
  \vdots & \vdots \\
  t_1^\tau & \vdots \\
\end{array}
\]
\[ g_i = \{(g_i(W), \Pr(W)) | W \in \mathcal{W}\} \]

**Tuple Model**

- Each tuple \( t_j = \langle (t_{j1}, p_{j1}), \ldots, (t_{j\ell_j}, p_{j\ell_j}) \rangle \). 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.

<table>
<thead>
<tr>
<th>( t_j )</th>
<th>( (1, 0.2), (3, 0.3), (7, 0.2) )</th>
<th>( (3, 0.3), (5, 0.1), (9, 0.4) )</th>
<th>( (3, 0.5), (10, 0.4), (13, 0.1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_2 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_3 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \cdots )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_{</td>
<td>\mathcal{T}</td>
<td>} )</td>
<td></td>
</tr>
</tbody>
</table>
\( g_i = \{(g_i(W), \Pr(W)) | W \in \mathcal{W}\} \)

### Value Model

- Each tuple \( t_j = \langle j : f_j = ((f_{j_1}, p_{j_1}), \ldots, (f_{j_{\ell_j}}, p_{j_{\ell_j}})) \rangle \), \( j \) is drawn from \([n]\).
- \( \Pr(f_j = 0) = 1 - \sum_{k=1}^{\ell_j} p_{jk} \)

| \( t_1 \) | \( \{< 1, (50, 0.2), (7, 0.1), (14, 0.2) >\} \) |
| \( t_2 \) | \( \{< 2, (6, 0.4), (7, 0.3), (15, 0.3) >\} \) |
| \( t_3 \) | \( \{< 3, (10, 0.3), (15, 0.2), (20, 0.5) >\} \) |
| \( \ldots \) | \( \ldots \) |
| \( t_n \) | \( \ldots \) |
\[ g_i = \{(g_i(W), \Pr(W)) | W \in W\} \]

**Value Model**

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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>( {&lt; 1, (50, 0.2), (7, 0.1), (14, 0.2) &gt;} )</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>( {&lt; 2, (6, 0.4), (7, 0.3), (15, 0.3) &gt;} )</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>( {&lt; 3, (10, 0.3), (15, 0.2), (20, 0.5) &gt;} )</td>
</tr>
<tr>
<td>( \cdots )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>( t_n )</td>
<td>( \cdots )</td>
</tr>
</tbody>
</table>
Histograms on Probabilistic data

Possible world semantic

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

Expectation based histogram

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

- [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
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:

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

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

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
**Pmerge Method**

Pmerge 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.

![Diagram of partition and sub-domain boundaries](image-url)
**Pmerge Method**

**Pmerge** 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.
**Pmerge Method**

Pmerge 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.

![Diagram showing partition and merge phases with frequency and domain value values](image)
**Pmerge Method**

**Pmerge** 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.

![Diagram](image)
Recursive Merging Method

- **Pmerge** method:
  - Approximation quality: \( \text{Pmerge} \) produces a \( \sqrt{10} \) approximation in \( O(N + Bn^2/m + B^3 m^2) \) time.

- Recursive merging (**RPmerge**):
  - Partition \([n]\) into \( m^\ell \) subdomains, producing \( Bm^\ell \).
  - Using \( \ell \) 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.
Distributed and Parallel PMERGE

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.
Distributed and Parallel PMERGE

Probabilistic database $D$ and $m$ sub-domains

$\tau_1$

$\tau_\ell$

$\tau_\beta$

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.
Distributed and Parallel PMERGE

Probabilistic database $\mathcal{D}$

$\tau_1$

$\tau_\ell$

$\tau_\beta$

$m$ sub-domains

A, B

1

\[ h(i) = \lceil \frac{i}{\lceil n/m \rceil} \rceil \]

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.
Distributed and Parallel PMERGE

Probalistic database $D$

$m$ sub-domains

Recursive merging

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

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

Estimate $A_k, B_k$ arrays using quantile sampling

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

item: 1 2 3 4 ...
Sampling $A, B$ arrays in the partition phase

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

Estimate $A_k, B_k$ arrays using quantile sampling

$$E[f_i] = \begin{bmatrix} 2 & 3 & 5 & 9 & \cdots \end{bmatrix}$$

item: 1 2 3 4 ...
Pmerge Based on Sampling

Sampling $A, B$ arrays in the partition phase

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

Estimate $A_k, B_k$ arrays using quantile sampling

$$E[f_i] = \begin{array}{cccccc}
3 & 3 & 3 & 3 & 3 & 3 \\
\end{array}$$

item: 1 2 3 4 ...
PMERGE Based on Sampling

Sampling $A$, $B$ arrays in the partition phase

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

Estimate $A_k$, $B_k$ arrays using quantile sampling

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

$$E[f_i]$$

item: 1 2 3 4 ...
Estimate $F_2 = \sum_{i=s_k}^{j} \left( \sum_{\ell=1}^{\beta} E \mathcal{W}, \ell [g_i] \right)^2$ using AMS Sketch techniques and binary decomposition of domain $[s_k, e_k]$.

(a) binary decomposition

$$F_2 = M''_k$$

(b) local Q-AMS

AMS

$$F_2 = 2\epsilon M''_k$$
Outline

1. Optimal $B$-buckets Histograms
2. Approximate Histograms
3. $P$MERGE Based on Sampling
4. Experiments
Generate tuple model and the value model dataset using the client id field of 1998 WorldCup dataset and atmospheric measurements from the SAMOS project.

The default experimental parameters:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B$</td>
<td>number of buckets</td>
<td>400</td>
</tr>
<tr>
<td>$n$</td>
<td>domain size</td>
<td>100k (600k)</td>
</tr>
<tr>
<td>$\ell$</td>
<td>depth of recursive pmerge</td>
<td>2</td>
</tr>
</tbody>
</table>
- $n$: 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