Weighted Distinct Sampling: Cardinality Estimation for SPJ Queries

Yuan Qiu, Yilei Wang, Ke Yi, Feifei Li, Bin Wu, Chaoqun Zhan

HKUST

Alibaba Group

Select-Project-Join Queries

- Relational Algebra
  - $\pi_A(\sigma_\phi(R_1 \bowtie R_2 \bowtie \cdots \bowtie R_m))$
- SQL
  - `select (distinct) A`
  - `from R1, R2, …, Rm`
  - `where Phi`
- Example: Find customers who placed an order after 2020-01-01

```
- SELECT (DISTINCT) o_custkey FROM orders
    WHERE o_orderdate > 2020-01-01
```

Select-Project-Join Queries

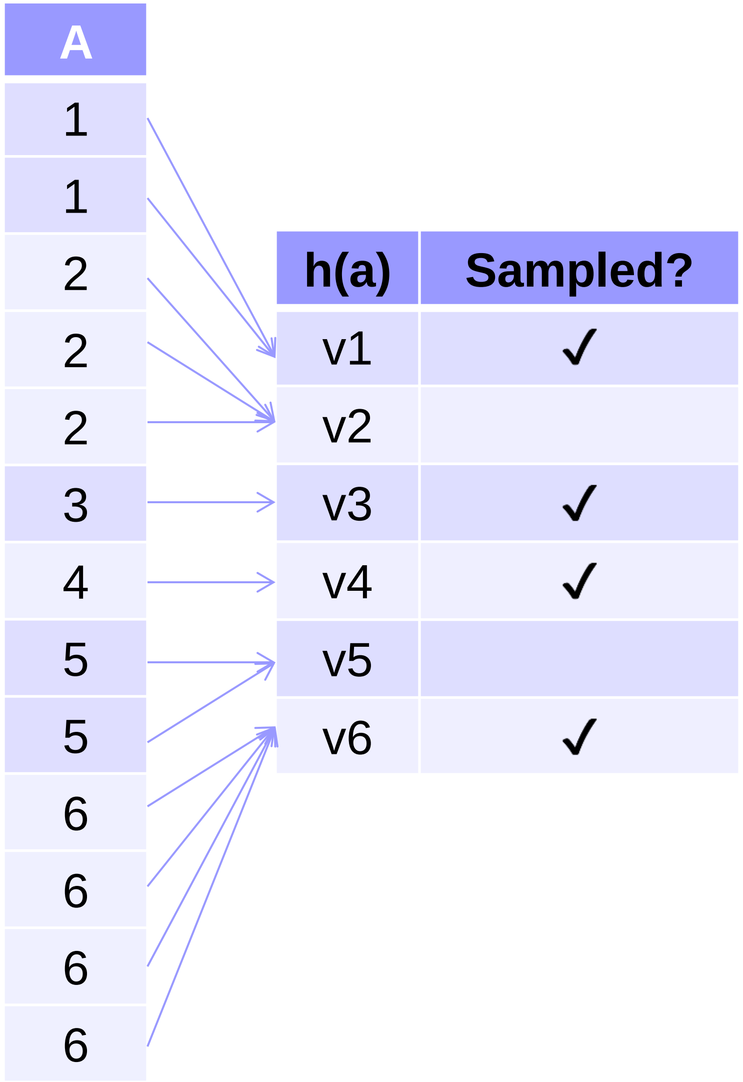
- Relational Algebra
  - $\pi_A(\sigma_\phi(R_1 \bowtie R_2 \bowtie \cdots \bowtie R_m))$
- SQL
  - `select (distinct) A`
  - `from R1, R2, …, Rm`
  - `where Phi`
- Example: Find customers who placed an order after 2020-01-01
  - And the order contains an item of price more than 100
  - `SELECT (DISTINCT) o_custkey FROM orders, lineitem`
    `WHERE o_orderdate > 2020-01-01 AND l_extendedprice > 100`

Cardinality Estimation for S / P / J Queries

- Selection (σ)
  - Selectivity estimation
  - Sampling, Assumptions (uniform, independent, …), …
- Projection (π)
  - If duplicates are not removed, cardinality is not affected (`select A from R`)
  - Otherwise, distinct count estimation (`select distinct A …` / `select A, agg() … group by A`)
  - Summary (FM, HyperLogLog, KMV, …), Sampling (uniform, distinct, …)
- Join ($\bowtie$) / Selection + Join ($\bowtie_\theta$)
  - Join size estimation
  - Sketch (AMS, Count Sketch, …), Sampling (Ripple Join, Wander Join, Two-Level Sampling, …), …
- What about Selection + Projection (+ Join)?

Review: Distinct Sampling

- Projection Only:
  - Want to estimate $D = |\pi_A R|$
  - Sample each distinct value with probability p into set A_s
    - Perform sampling on hash values
  - $|A_s|/p$ is a good estimator for D
    - Unbiased
    - Variance $\frac{Dp(1-p)}{p^2} \approx \frac{D}{p}$
  - Example:
    - Suppose the sampling rate $p = 1/2$
    - Our sample is $A_s = \{1,3,4,6\}$
    - Estimate $\widehat{D} = \frac{4}{1/2} = 8$ (Actual $D = 6$)



Review: Distinct Sampling

- Selection + Projection:
  - Want to estimate $D^{\phi} = |\pi_A \sigma_\phi R|$
  - Augment each sample with τ tuples as R_s
    - Uniformly taken from all its tuples
  - Use $|\pi_A \sigma_\phi R_s|/p$ as estimator
  - Example:
    - Still $p = 1/2$ and $A_s = \{1,3,4,6\}$
    - Set $\tau = 2$, so each $a \in A_s$ is augmented by ≤ 2 tuples
    - Now our filter is $\phi \coloneqq (B < 10 * A)$
    - $\pi_A \sigma_\phi R = \{2,3,4,5,6\}$, so $D^{\phi} = 5$

A	B	sampled?	ϕ
1	10	✓	
1	20	✓	
2	30		
2	20		
2	10		✓
3	10	✓	✓
4	20	✓	✓
5	60		
5	10		✓
6	80	✓	
6	20		✓
6	60	✓	
6	30		✓

Review: Distinct Sampling

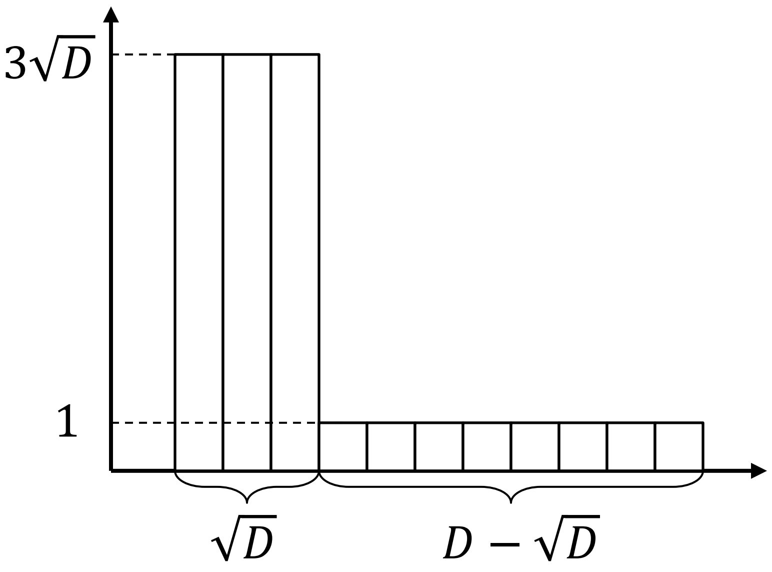
A	B	sampled?	ϕ
1	10	✓	
1	20	✓	
3	10	✓	✓
4	20	✓	✓
6	80	✓	
6	60	✓	

- Selection + Projection:
  - Want to estimate $D^{\phi} = |\pi_A \sigma_\phi R|$
  - Augment each sample with τ tuples as R_s
    - Uniformly taken from all its tuples
  - Use $|\pi_A \sigma_\phi R_s|/p$ as estimator
  - Example:
    - Still $p = 1/2$ and $A_s = \{1,3,4,6\}$
    - Set $\tau = 2$, so each $a \in A_s$ is augmented by ≤ 2 tuples
    - Now our filter is $\phi \coloneqq (B < 10 * A)$
    - $\pi_A \sigma_\phi R = \{2,3,4,5,6\}$, so $D^{\phi} = 5$
    - $\pi_A \sigma_\phi R_s = \{3,4\}$, so $\widehat{D^{\phi}} = \frac{2}{1/2} = 4$

Uniform Distinct Sampling: Problems

- If we could augment each value with ALL its tuples, the estimator would degenerate to the projection-only case.
  - Unbiased with variance $\Theta\left(\frac{D}{p}\right)$.
- However, we only stored τ tuples
  - It is possible that we failed to sample a passing tuple when there exists
  - This creates a (downward) bias
- The expected sample size is $Dp\tau$, so a problem is how to balance
  - The original paper used a heuristic
  - We show next that there are hard inputs where no setting is good

Uniform vs. Weighted Distinct Sampling



- Hard Input
  - $\sqrt{D}$ heavy hitters, each having $3\sqrt{D}$ tuples
  - $D-\sqrt{D}$ light hitters, each having 1 tuple
  - D distinct values, $\approx 4D$ tuples, use $2D$ sample budget
- Uniform Distinct Sampling: $\text{MSE} = \Omega(D)$
  - If $\tau > 2\sqrt{D}$, variance is $\Omega(D)$
  - If $\tau \le 2\sqrt{D}$, bias is $\Omega\left(\sqrt{D}\right)$
- Weighted Distinct Sampling: A simple configuration can achieve $O\left(\sqrt{D}\right)$
  - Keep ALL light values (Sampling with probability $p_l = 1$)
  - Sample heavy values with $p_h = 1/3$, and store ALL their tuples if sampled.

Why Use Weighted Distinct Sampling?

- In distinct count estimation, heavy hitters are not more important.
  - Any distinct value can only contribute 1 to the distinct count post filter D^{ϕ}.
- However, heavy hitters are harder to estimate.
  - For light hitters, we may store all its tuples to remove the bias.
  - This is not possible for heavy hitters.

Weighted Distinct Sampling: Algorithm

- Parameters: vectors $\{p_i\}, \{\tau_i\}$ defined for $i \in \mathrm{dom}(A)$
- Algorithm: Sample each distinct value i with probability p_i.
  If sampled, augment it with τ_i of its tuples.
- Estimation: Let n_i^ϕ denote the number of tuples that passes ϕ among the τ_i sampled tuples. $n_i^\phi = 0$ if i itself was not sampled at all. Use the following estimator.

$$\widehat{D^\phi} = \sum_{i \in \mathrm{dom}(A)} \frac{I\left[n_i^\phi \geq 1\right]}{p_i}$$

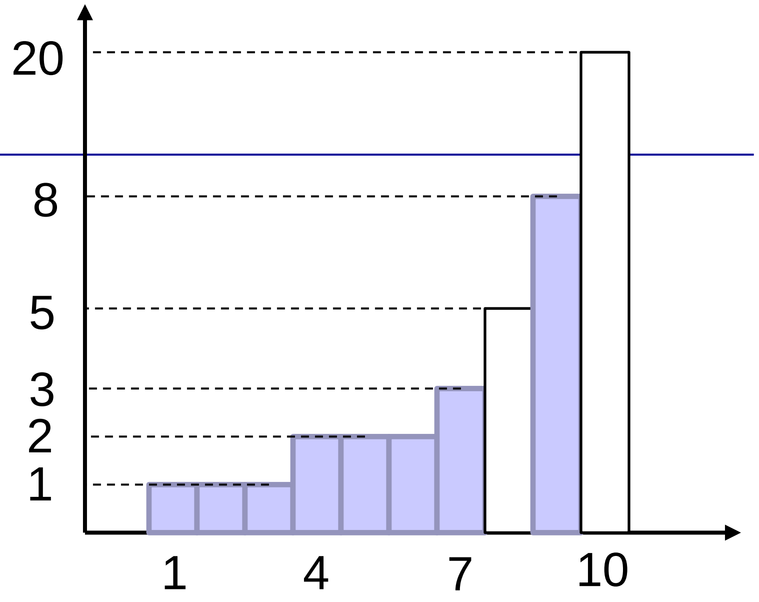
- When $p_i \equiv p$ and $\tau_i \equiv \tau$, it degenerates to uniform distinct sampling.
- What are the best parameters?
  - Solving an optimization problem.

Near Optimal Solution

- N_i: Frequency if items i.
- In general, $p_i \propto \frac{1}{\sqrt{N_i}}$, and $\tau_i = N_i$.
  - When N_i is too small, we set $p_i = 1$.
  - When N_i is too large, we never sample the value.
- Intuition
  - Heavy hitters are harder to estimate, so the sampling probability p_i decreases wrt N_i
  - Bias is more important than variance, so we keep all tuples from a value if it is sampled.
  - The cost of sample budget for i is proportional to $\sqrt{N_i}$, so for large N_i, costs outweigh benefits, and we never sample them.

$$
\begin{aligned}
\underset{\boldsymbol{p},\boldsymbol{\tau}}{\text{minimize}}\ \max_{\phi}\ & \mathrm{MSE}(\boldsymbol{p},\boldsymbol{\tau},\phi) \\
\text{subject to}\quad & 0 \prec \boldsymbol{p} \preccurlyeq 1, \\
& 0 \preccurlyeq \boldsymbol{\tau} \preccurlyeq \boldsymbol{N}, \\
& \boldsymbol{p} \cdot \boldsymbol{\tau} \leq n,
\end{aligned}
$$

Weighted Distinct Sampling: Example



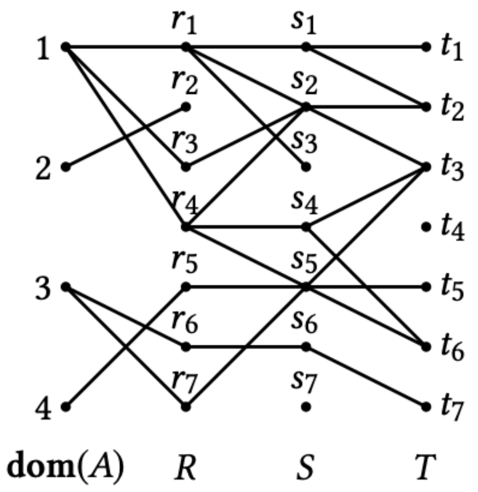
- Consider a frequency distribution as below.
  - $N_1, N_2, N_3 = 1, N_4, N_5, N_6 = 2$
  - $N_7 = 3, N_8 = 5, N_9 = 8, N_{10} = 20$
- Say our sample budget is $n = 20$, then
  - For $i = 1, \dots 6, p_i = 1$, we deterministically keep them in the sample. (cost = 9)
  - $p_7 = 0.93, p_8 = 0.72, p_9 = 0.57$ is inversely proportional to $\sqrt{N_i}$. Once sampled, all their tuples will be maintained. (cost = 0.93*3+0.72*5+0.57*8=11)
  - N_{10} is too large, so we never sample value 10. (cost = 0)
- Estimation: Suppose our current sample is $A_s = \{1,2,3,4,5,6,7,9\}$, and the filter passes a tuple for each $i = 1, \dots 10$. Our estimator is

$$\widehat{D^\phi} = 6 + 0.93^{-1} + 0.57^{-1} = 8.82$$

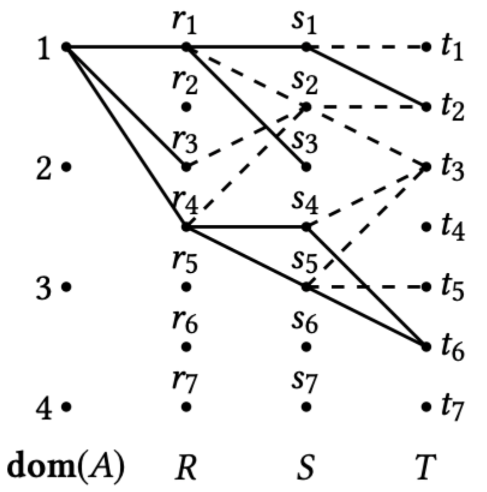
when actual $D^\phi = 10$.

Weighted Distinct Sampling for SPJ queries

- Direct Extension: Join-and-Run
- More efficient approach: using random walks
- View the join as a graph
  - Nodes: distinct values + tuples
  - Edges: value ∈ tuple + between joining tuples
  - Example: $R(A, \dots) \bowtie S \bowtie T$
    - Each length 3 path from $i \to t_j$ is s join result
- Start by running WDS on R
  - Scale $\boldsymbol{\tau}$ up by a constant as joins can expand tuples
  - For each sampled value, perform a BFS in the graph while being careful not to break τ.
- Estimation time: WDS + Bias Correction



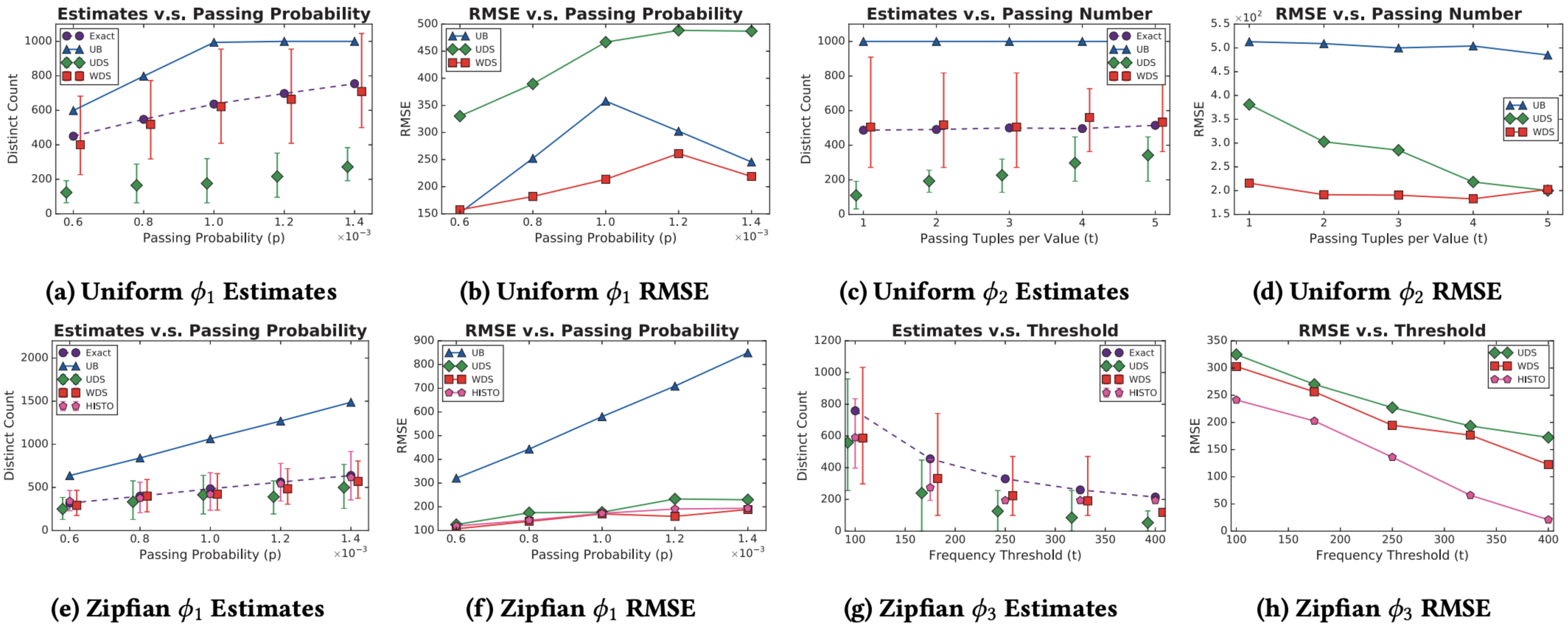
(a) Adding conceptual column.



(b) Distinct sampling from join.

Figure 1: Sampling by random walks.

Experiment Results (SP, Synthetic)

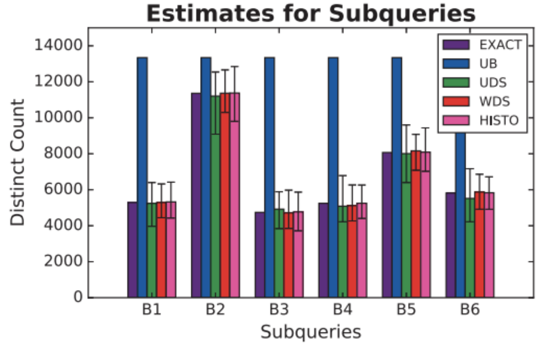


(a) Uniform ϕ_1 Estimates **(b) Uniform ϕ_1 RMSE** **(c) Uniform ϕ_2 Estimates** **(d) Uniform ϕ_2 RMSE**

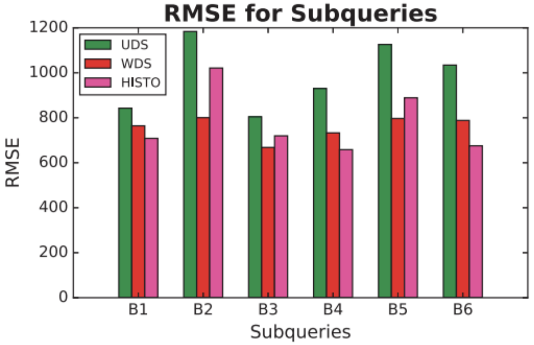
(e) Zipfian ϕ_1 Estimates **(f) Zipfian ϕ_1 RMSE** **(g) Zipfian ϕ_3 Estimates** **(h) Zipfian ϕ_3 RMSE**

Figure 2: Performance Evaluation for Synthetic Datasets

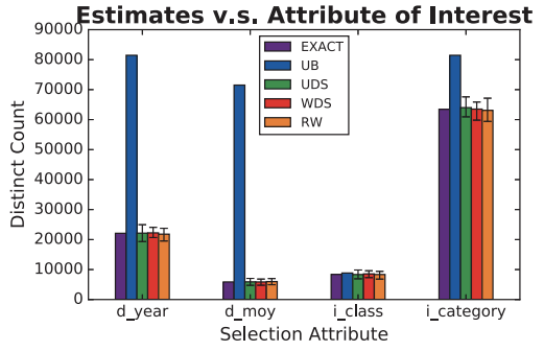
Experiment Results (SPJ, Benchmark & Real)



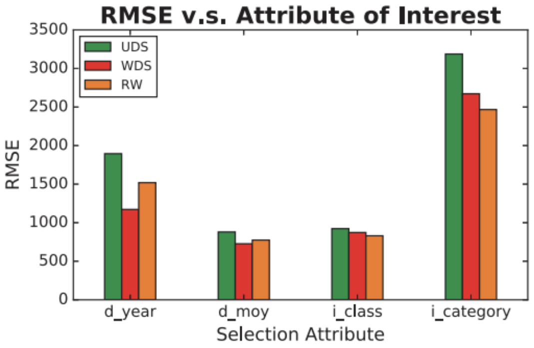
(a) Query 28 Estimates



(b) Query 28 RMSE

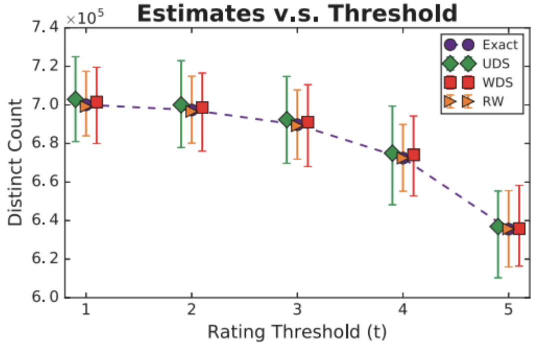


(c) Query 54 Estimates

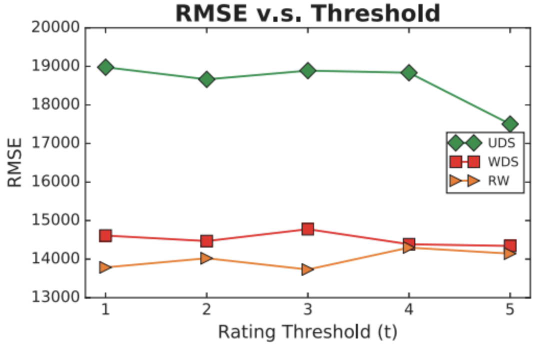


(d) Query 54 RMSE

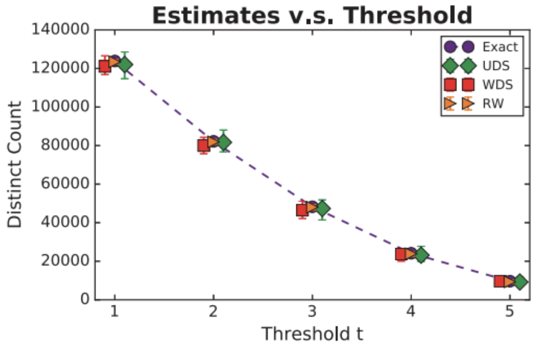
Figure 3: Performance Evaluation for TPC-DS Benchmark



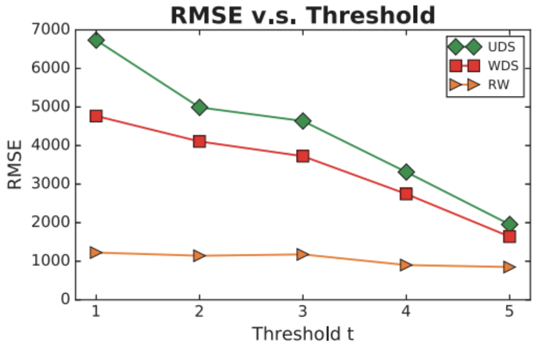
(a) IMDb Query 1 Estimates



(b) IMDb Query 1 RMSE



(c) IMDb Query 2 Estimates



(d) IMDb Query 2 RMSE

Figure 4: Performances Evaluation of IMDb Data

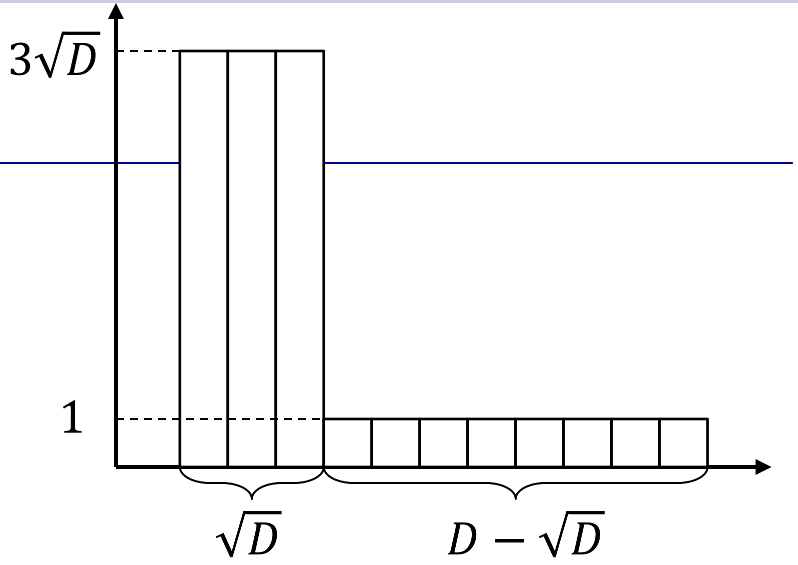
Conclusions and Future Directions

- We introduced Weighed Distinct Sampling for cardinality estimation of SP(J) queries.
- Implemented in AnalyticDB, product of Alibaba Cloud
- Future Directions
  - Dynamic Maintenance
  - Special Predicates (e.g. ranges)

Thank you!

BACK-UP SLIDES

Uniform Distinct Sampling: Hard Case



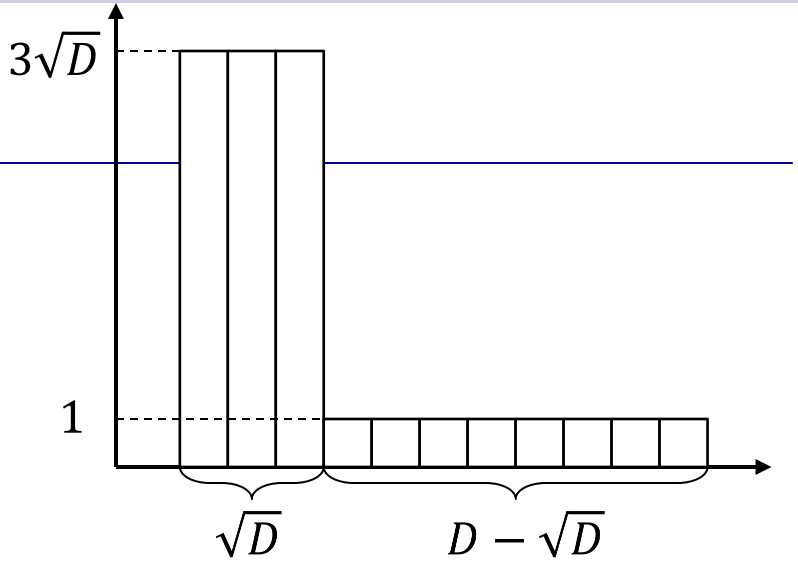
- There are $\sqrt{D}$ heavy hitters, each having $3\sqrt{D}$ tuples
- Remaining $D-\sqrt{D}$ values are light, each having 1 tuple
  - There are $3D+D-\sqrt{D}\approx 4D$ tuples in total
- Suppose we allow a sample budget of $2D$, sampling half the database!
- Intuition: If τ is large, then p must be small, so variance is large. Otherwise τ is small, and bias is large.

- If $\tau > 2\sqrt{D}$, then $p\le 3/4$. Otherwise the expected sample size is at last

$$\frac{3}{4}\left(\sqrt{D}\cdot 2\sqrt{D}+D-o(D)\right)=\frac{9}{4}D-o(D)>2D$$

  - Since $p\le\frac{3}{4}$, the variance is $\Omega\left(\frac{D}{p}\right)=\Omega(D)$

Uniform Distinct Sampling: Hard Case

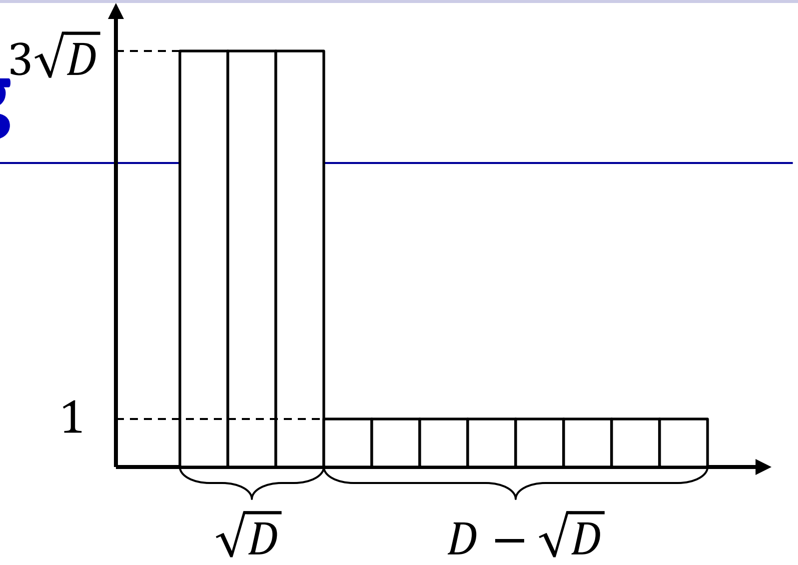


- If $\tau \leq 2\sqrt{D}$, for simplicity we just consider $p = 1$.
- Consider ϕ_x that 1) blocks all light value, and

  2) passes x tuples for any heavy value

  - When x varies from 1 to $3\sqrt{D}$, $D^{\phi_x} \equiv \sqrt{D}$
  - Our estimator is $\widehat{D^{\phi_x}} = \left|\pi_A \sigma_{\phi_x} R_s\right| = \sum_{i=1}^{\sqrt{D}} I\left[n_i^{\phi_x} \geq 1\right]$, where $n_i^{\phi_x}$ is the number of passing tuples sampled for value i. Its expectation is $\mathrm{E}\left[\widehat{D^{\phi_x}}\right] = p(x) \cdot \sqrt{D}$.
  - $p(x)$ is the probability of sampling at least one passing tuple for any value.
    - If $x = 3\sqrt{D}$, we must sampled passing tuples, thus $p(x) = 1$
    - If $x = 1$, $p(x) = \tau/3\sqrt{D} \leq 2/3$.
  - The gap of the estimator is $\Omega\left(\sqrt{D}\right)$ when the actual D^{ϕ_x} is fixed. So the bias is $\Omega\left(\sqrt{D}\right)$

Uniform vs. Weighted Distinct Sampling



- Uniform Distinct Sampling:
  - If $\tau > 2\sqrt{D}$, variance is $\Omega(D)$
  - If $\tau \leq 2\sqrt{D}$, bias is $\Omega\left(\sqrt{D}\right)$
  - Either way, $\mathrm{MSE} = \mathrm{Bias}^2 + \mathrm{Var} = \Omega(D)$
- Can we do better?
- For this specific case:
  - Keep all light values (Sampling with probability $p_l = 1$)
  - Sample heavy values with $p_h = 1/3$, and take ALL their tuples if sampled.
  - Expected sample size is $p_h\sqrt{D} \cdot 3\sqrt{D} + p_l\left(D - \sqrt{D}\right) \cdot 1 < 2D$
  - There is no bias, and the variance (from heavy values) is $O\left(\sqrt{D}/p_h\right) = O\left(\sqrt{D}\right)$.

WDS for SPJ: Estimation

- We are no longer able to store ALL join results for a distinct value (they are huge!)
- So we want to reduce the bias.
- At estimation time, we check each distinct value in our sample:
  - If none of its join results passed the filter, or if it failed to extend to any join result at all, we regard that it does not appear in the original (post-filter) join result, and estimate 0.
  - If ≥ 2 of its join results passed the filter, we assume there are many candidates, so we regard the probability of sampling a passing join result is high, and estimate 1.
  - If there is a single passing join result, we have have sampled it due to luck. And we want to estimate the probability of sampling a passing tuple.
    - Lower bounded by p_t, the probability of sampling this exact tuple; Upper bounded by 1, so we use a scaled up estimator $\frac{1}{\sqrt{p_t}}$, and p_t can be calculated in random walks.
- Finally, scale it up by the inverse of p_i.