On Random Sampling over Joins *

Surajit Chaudhuri (M$ research)
Rajeev Motwani (Stanford)
Vivek Narasayya (M$ research)

Presented by
Manas Somaiya (UF CISE)

* SIGMOD 1999
Outline of the talk

- What the paper is about?
- What shall we focus on?
  - Semantics of a Sample
  - Algorithms for Sequential and Weighted Sampling
  - The Join Sampling Problem
  - Strategies for Join Sampling
What the paper is about?

- **Sampling** as a primitive operation
- Difficulty in **commuting** the sample operation
  - Specifically the **join** operator
- Theoretical explanation
- Limits on the efficiency
- Develop new insights and techniques
  - Where the **negative** results do not apply
- Experimental evaluation
Why Sample?

- Provide a sample of the result or approximate answer
- Query optimization
- Privacy preserving data mining
- Advantages:
  - Fast answers to queries
  - Faster post-processing of query results
  - Query optimization decisions
  - Privacy protection in statistical databases
Sample as a primitive relational operator

- $\text{SAMPLE}(R, \phi)$
  - Produces a uniform random sample $S$
  - $S$ is a $\phi$-fraction of a relation $R$
- Well studied problem and efficient strategies if $R$ is a base relation
- Ineffective if $R$ is produced by a query $Q$
- Grossly inefficient to evaluate $Q$
  - Compute the entire relation $R$
  - Throw away most of it when sampling
Sample as a primitive relational operator ...

- Solution: Partially evaluate $Q$
  - Generate only the sample of $R$
- Push the sampling operator down
  - To minimize cost of query evaluation
- Commute the sample operation
  - Can freely interchange with selection
  - Projection:
    - Duplicate removal issue
    - Skews the probability distribution
- Join:
  - Does not commute
Semantics of a Sample

- Relation \( R \) with \( n \) tuples
- \( f \) - fraction sample \( S \)
- Sampling with Replacement (WR)
  - Sample \( f \times n \) tuples
  - Uniformly and independently from \( R \)
  - Sample is a bag of \( f \times n \) tuples from \( R \)
    - A tuple could be sampled multiple times
- Sampling without Replacement (WoR)
  - Sample \( f \times n \) distinct tuples from \( R \)
  - Each successive sample is chosen uniformly
    - From the set of tuples not already sampled
  - Sample is a set of \( f \times n \) distinct tuples from \( R \)
Semantics of a sample ...

- Independent Coin Flips (CF)
  - For each tuple in \( R \)
    - Choose it for the sample with probability \( f \)
    - Independent of other tuples
  - Same as flipping a coin with bias \( f \) for each tuple
  - Sample is a set of \( \chi \) distinct tuples from \( R \)
    - \( \chi \) is a random variable
      - Binomial distribution \( B(n, f) \)
      - Expectation \( f^*n \)
Semantics of a sample ...

- Possible to convert from one type of sample semantics to another
- Can not obtain a CF sample from either a WR or WoR sample
  - Non-zero probability of sampling entire relation in the CF semantics
Algorithms for Sequential and Weighted Sampling

- Can we perform sampling on a relation as it is streaming by?
  - Critical for efficiency when the relation is materialized on the disk
  - More important in a pipeline (query tree)
    - Do not want to materialize it on the disk

- Called *sequential* sampling
- Un-weighted sampling
  - Each element is sampled uniformly at random
- Weighted sampling
  - Each element is sampled with a probability
    - Proportional to its weight
    - For some pre-specified set of weights
Algorithms for Sequential and Weighted Sampling ...

- Easy to get Sequential WoR sampling
  - With CF semantics
    - Flip a coin for each tuple as it goes by
      - With probability $f$ for heads
    - Add it to sample is flip turns out to be a head
  - With standard reservoir sampling
Un-weighted Sequential WR Sampling

**Black-Box U1:** Given relation $R$ with $n$ tuples, generate an un-weighted WR sample of size $r$

1. $x \leftarrow r$; $i \leftarrow 0$
2. While tuples are streaming by and $x > 0$ do begin
   a) Get next tuple $t$
   b) Generate random variable $X$ from $B(x, 1/(n-i))$
   c) Output $X$ copies of $t$
   d) $x \leftarrow x - X$
   e) $i \leftarrow i + 1$
End
Un-weighted Sequential WR Sampling …

**Black-Box U2**: Given relation $R$ with $n$ tuples, generate an un-weighted WR sample of size $r$

1. $N \leftarrow 0$
2. Initialize reservoir array $A[1 \ldots r]$ with dummy values
3. While tuples are streaming by do begin
   a) Get next tuple $t$
   b) $N \leftarrow N + 1$
   c) For $j = 1$ to $r$ do set $A[j]$ to $t$ with probability $1/N$
End
Weighted Sequential Sampling

- Relation \( r \) with a total of \( n \) tuples
  - Each tuple \( t \) has a specified weight \( w(t) \)
- A weighted WR sample is obtained by repeating \( f*n \) times
  - Choose a tuple from \( R \) at random
    - Any tuple \( t \) is chosen with probability proportional to \( w(t) \)
- Assume \( w(t) \) are non-negative integers
- Weighted WR sample from \( R \)
  - Is the same as an un-weighted WR sample
    - Modify relation \( R \) to a relation \( R^w \)
    - \( w(t) \) copies of each tuple \( t \in R \)
**Weighted Sequential Sampling** ...

**Black-Box WR1:** Given relation $R$ with $n$ tuples, generate a weighted WR sample of size $r$

1. $x \leftarrow r$ ; $D \leftarrow 0$
2. $W \leftarrow \sum_{t \in R} w(t)$
3. While tuples are streaming by and $x > 0$ do begin
   a) Get next tuple $t$ with weight $w(t)$
   b) Choose random variable $X$ from $B(x, w(t)/(W-D))$
   c) Output $X$ copies of $t$
   d) $x \leftarrow x - X$
   e) $D \leftarrow D + w(t)$
End
Weighted Sequential Sampling ...

Black-Box WR2: Given relation $R$ with $n$ tuples, generate an weighted WR sample of size $r$

1. $W \leftarrow 0$
2. Initialize reservoir array $A[1 \ldots r]$ with dummy values
3. While tuples are streaming by do begin
   a) Get next tuple $t$ with weight $w(t)$
   b) $W \leftarrow W + w(t)$
   c) For $j = 1$ to $r$ do set $A[j]$ to $t$ with probability $w(t)/W$

End