CIS 6930 – Approximate Query Processing
Paper Presentation – Spring – 2004 - Instructor: Dr. Alin Dobra

Overcoming Limitations of Sampling for Aggregation Queries

Authors: Surajit Chaudhuri,
Gautam Das,
Mayur Datar,
Rajeev Motwani, and
Vivek R. Narasayya

ICDE 2001

Presented by: Andréa Matsunaga (ammatsun@ufl.edu)

Outline

- Introduction
  - The need for Approximate Query Processing
- Issues with uniform sampling
- Solutions
  - Outlier-indexes
  - Exploiting workload information
- Experimental results

Introduction

- Data analysis over large data is hard
- Data analytics often do not need exact answers
  - "ballpark" estimates are enough
- Examples
  - On Line Analytical Processing (OLAP)/Decision Support
    - E.g. what is the percent increase in the sales of Windows XP over last year in California?
  - Data Mining
    - Building models (e.g. decision trees) does not require precise counts
- Focus on *Aggregate* queries

Issues

- Limitations of uniform sampling in answering *Aggregation* queries:
  - Data skew (large data variance)
    - Outlier-indexes
  - Low selectivity and small groups
    - Exploiting workload information

By: Andréa Matsunaga

Data Skew Effect Example

Relation R
(N=10000 tuples)

K	C
	1
	.
	.
	.
	.
	.
	.
	.
	.
	1
	1000

99% (the rows with C = 1)
1% (the row with C = 1000)

Sum(C) = 109,900

1% uniform sample
(100 tuples)
Extrapolate
(multiply by 100)

Severe underestimate
if outlier not in sample

No tuple with 1000:
Est(SUM(C))=10,000

$$P = \frac{\binom{R\text{-}9900}{1}\binom{R\text{-}100}{99}}{\binom{N}{100}} = 0.37$$

1 tuple with 1000:
Est(SUM(C))=109,900

Severe overestimate
if outlier not in sample

2 or more tuples with 1000:
Est(SUM(C))=209,800
Est(SUM(C))=309,700
.
.

Probability of 0.63 to get large error in estimate!!!

Theorem 1

- R = Relation of size N
- $\{y_1, y_2, \ldots, y_N\}$ = Set of values associated with the tuples in the relation
- U = uniform sample of y_i's of size n
- $Y_e = \left(\frac{N}{n}\right)\sum_{y_i \in U} y_i$ = Unbiased estimator of the actual sum $Y = \sum_{i=1}^{N} y_i$
- with standard error:

$$\varepsilon = \frac{NS}{\sqrt{n}} \cdot \sqrt{1 - \frac{n}{N}}$$

- where S = standard deviation

$$S = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \overline{Y})^2}{N-1}}$$

By: Andréa Matsunaga

Theorem 1 - Proof

$$Y_e = \left(\frac{N}{n}\right)\sum_{y_i \in U} y_i \qquad Y = \sum_{i=1}^{N} y_i$$

Properties of expectation:

- $E(a) = a$

Properties of variance:

- $Var(aX) = a^2 Var(X)$
- $Var(\sum_i X_i) = \sum_i Var(X_i)$

(For independent random variables)

$$E(Y_e) = E\left(\left(\frac{N}{n}\right)\sum_{y_i \in U} y_i\right) = E\left(\left(\frac{N}{n}\right)\sum_{i=1}^{N} y_i \cdot P_U(i)\right) = E\left(\left(\frac{N}{n}\right)\sum_{i=1}^{N} y_i \cdot \frac{n}{N}\right) = Y$$

$$Var(Y_e) = Var\left(\left(\frac{N}{n}\right)\sum_{y_i \in U} y_i\right) = \left(\frac{N^2}{n^2}\right)\sum_{y_i \in U} Var(y_i) = \left(\frac{N^2}{n^2}\right) \cdot n \cdot S$$

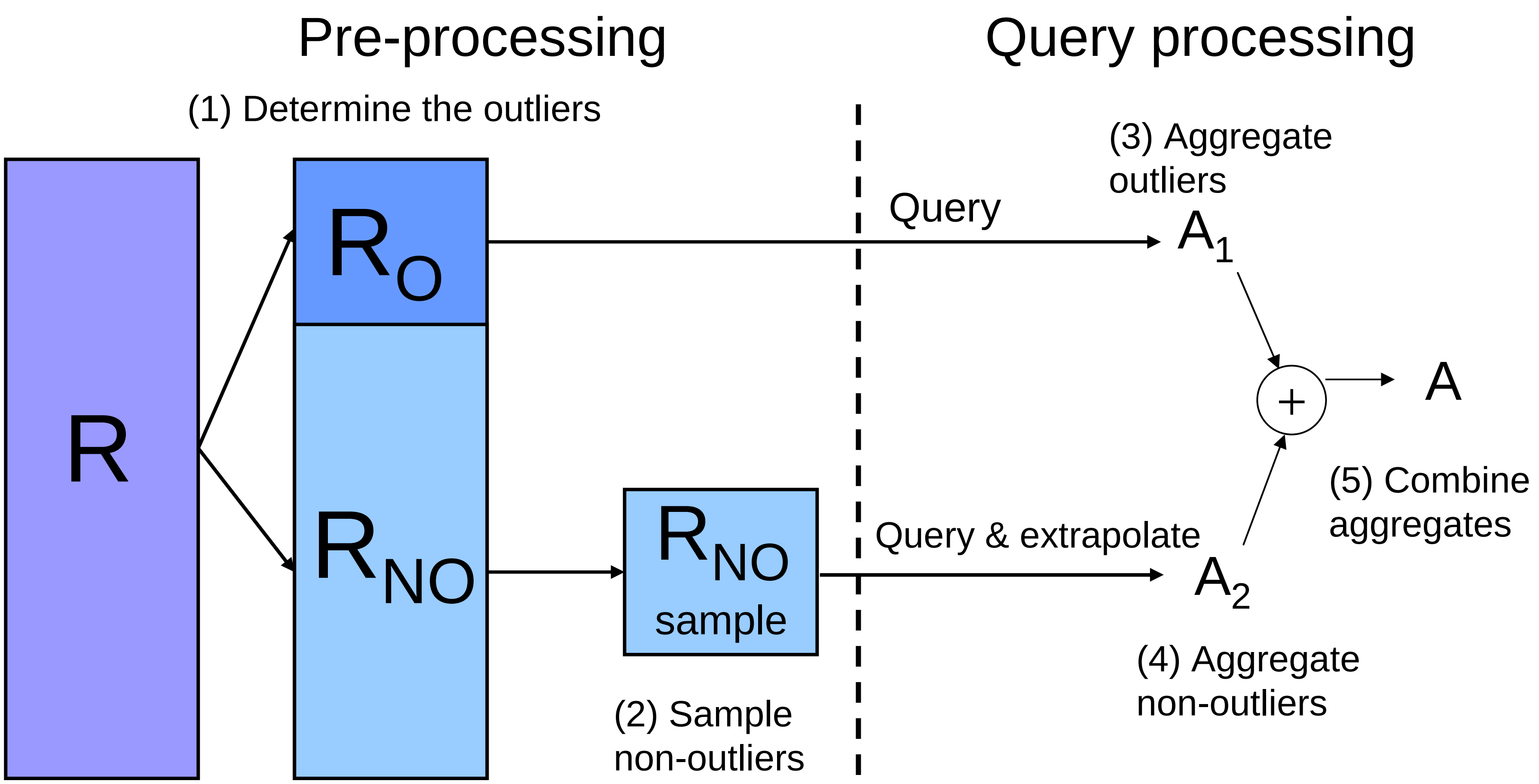
$$\varepsilon = \sqrt{Var(Y_e)} = \frac{NS}{\sqrt{n}} \qquad Var(y_i) = S = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \overline{Y})^2}{N-1}}$$

By: Andréa Matsunaga

Solution 1: Outlier Indexing

- To handle data skew in a aggregation query
- The idea:
  - Separate the outliers (R_O) from the rest of the data or non-outliers (R_{NO}) into an *outlier index*
  - Keep a *uniform random sample* of the remaining data
  - Use *outlier index* as well as *random sample* to answer queries

Outlier Indexing Implementation



Note: Since DB content change over time, selection of outliers indexes and samples should be refreshed appropriately.

Outlier Selection: Definition 1

- For any sub-relation R' ($R' \subset R$)

- $\varepsilon(R')$ = standard error in estimating the sum of values in R' (uniform sampling followed by extrapolation)

- An optimal outlier-index $R_O(R,C,\tau)$ is defined as a sub-relation $R_O \subset R$:
  - $| R_O | \leq \tau$
  - $\varepsilon(R \backslash R_O) = \min_{R' \subset R,\ |R'| \leq \tau}\{\varepsilon(R \backslash R')\}$

By: Andréa Matsunaga

Outlier Selection: Theorem 2

- Consider a multiset $R = \{y_1, y_2, \ldots, y_N\}$ where the y_i's are in sorted order.

- Let $R_O \subset R$ be the subset such that:
  - $| R_O | \leq \tau$
  - $S(R \backslash RO) = \min_{R' \subset R, |R'| \leq \tau}\{S(R \backslash R')\}$

- Then exists some $0 \leq \tau' \leq \tau$ such that
  $R_O = \{y_i \mid 1 \leq i \leq \tau'\} \cup \{ y_i \mid (N+\tau'+1-\tau) \leq i \leq N\}$

Outlier Selection: Algorithm

1) Read the values in column C of the relation R. Let $\{y_1, y_2, \ldots, y_N\}$ be the sorted order of the values appearing in C (each value corresponds to a tuple).

2) For i = 1 to τ+1, compute
$E(i) = S(\{y_i, y_{i+1}, \ldots, y_{N-\tau+i-1}\})$.

3) Let i' be the value of i where E(i) takes its minimum value. Then the outlier-index is the tuples that correspond to the set of values

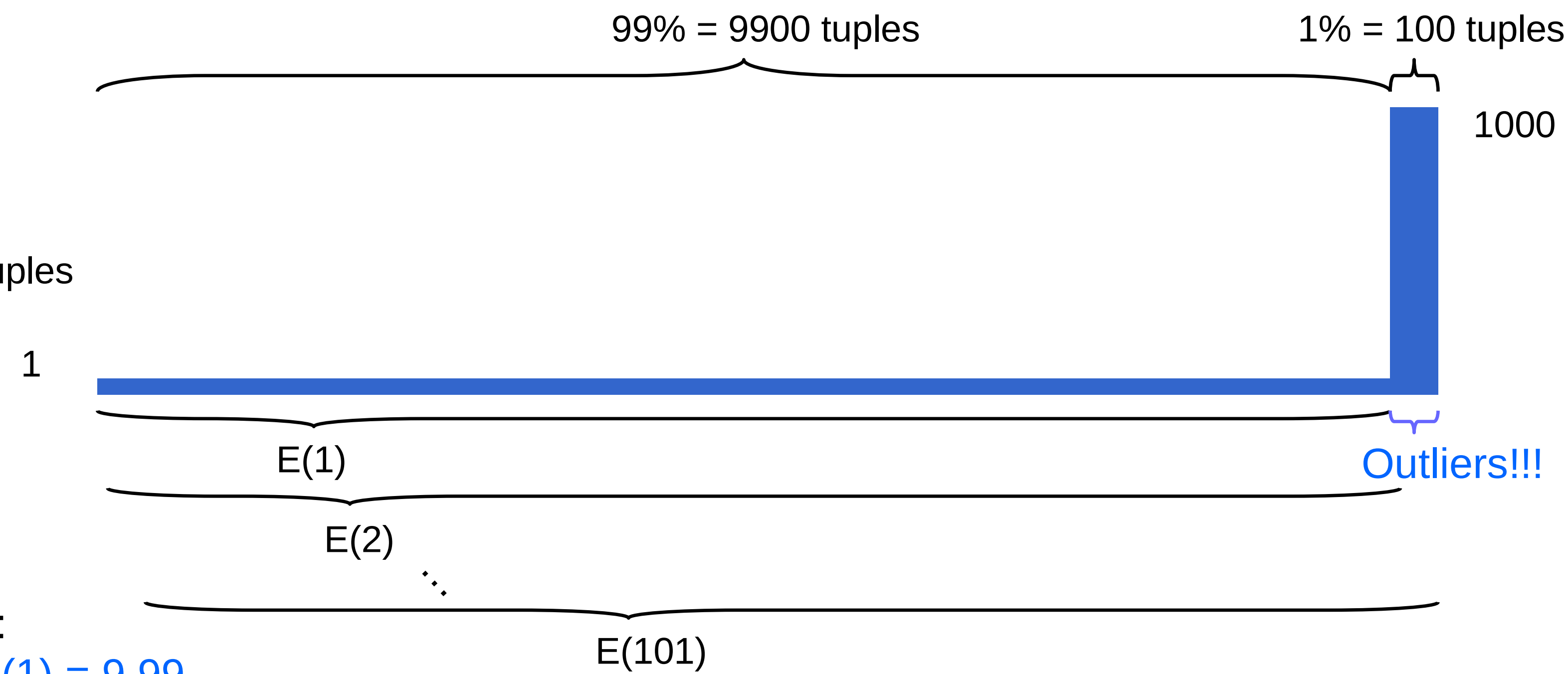
$$\{y_j \mid 1 \leq j \leq \tau'\} \cup \{y_j \mid (N+\tau'+1-\tau) \leq j \leq N\} \text{ where } \tau' = i'-1$$

- The algorithm depends on computing standard deviations
- Standard deviations computed in O(1) time for insertions and deletions (e.g. E(i+1) can be computed from E(i), y_i and $y_{N-\tau+1}$).

Outlier Selection: Example

Relation R

$\bar{Y}$ = 10.99
Y = 109,900
N = 10,000 tuples

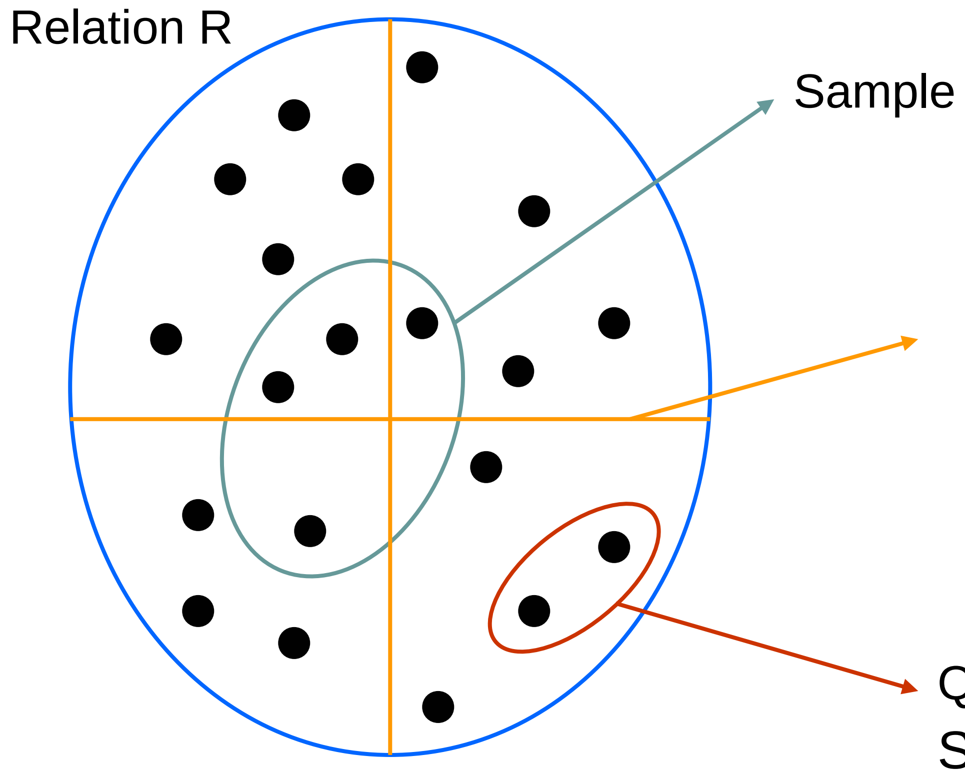


For τ = 100:

E(1) = 9.99
E(2) = 14.09
E(3) = 17.25
.
.
E(101) = 99.9

```
CREATE VIEW c_otl_idx AS
        SELECT * from R
        WHERE (C > 1000)
```

Low Selectivity and Small Groups Effect Example



Query with group-by's
Sample may not contain even a **single** row that belongs to the sub-relation

Query with low selectivity
Sample may not contain even a **single** row selected by the query

Solution 2: Exploiting Workload Information

- To handle low selectivity and small groups
- The idea:
  - Use *weighted sampling*
  - Sample *more* from subsets of data that are small in size but are important (have high usage).
  - Exploit DB access pattern *locality*.
  - Using *pre-computed* samples.

Exploiting Workload Information

- Steps:
  1) **Workload Collection**: obtain a workload consisting of representative queries against the DB (e.g. Microsoft SQL Server Profiler).

  2) **Trace Query Patterns**: analyze workload to obtain parsed information (e.g. the set of selection conditions that are posed).

  3) **Trace Tuple Usage**: The execution of the workload reveals additional information on usage of specific tuples (e.g. frequency of access to each tuple). Since tracking this information at the level of tuples can be expensive, it can be kept at coarser granularity (e.g. on page-level). For the experiments, assumed that a tuple t_i has weight w_i if the tuple t_i is required to answer w_i queries in the workload).

  4) **Weighted Sampling**: Perform sampling by taking into account weights of tuples in step 3. The probability to accept the sample is $p_i = n \cdot w_i'$, where:

$$w_i' = w_i / \sum_{j=1}^{N} w_j$$

  Need to store the normalized weight w_i' together with the tuple since its inverse (multiplication factor) will be used to answer the aggregate query.

By: Andréa Matsunaga

Exploiting Workload Information

- When weighted sampling based on workload information works well?
  - Access pattern of queries are local
  - We have a workload that is a good representative of future queries.

Experimental Setup

- **Platform**: Dell Precision 610 system with a Pentium III Xeon 450 MHz processor with 128 MB RAM and an external 23GB hard drive.
- **Databases**: 100MB TPC-R databases. TPC-R benchmark modified to vary the degree of *skew* determined by the Zipfian parameter $\mathbf{z}^5$ distribution, since original data is generated from a *uniform* distribution.
- **Workloads**: random query generation program with *sum* aggregate function.
- **Parameters**: (a) skew of the data (**z**) was varied over 1, 1.5, 2, 2.5, and 3 (b) the sampling fraction (**f**) was varied over a wide range from 1% to 100%, (c) the storage for the outlier-index was varied over 1%, 5%, 10%, and 20%; and (d) average over 3 runs.
- **Techniques**:USAMP: uniform sampling
  WSAMP: weighted sampling
  WSAMP+OTLIDX: weighted sampling + outlier-indexing

Experimental Results

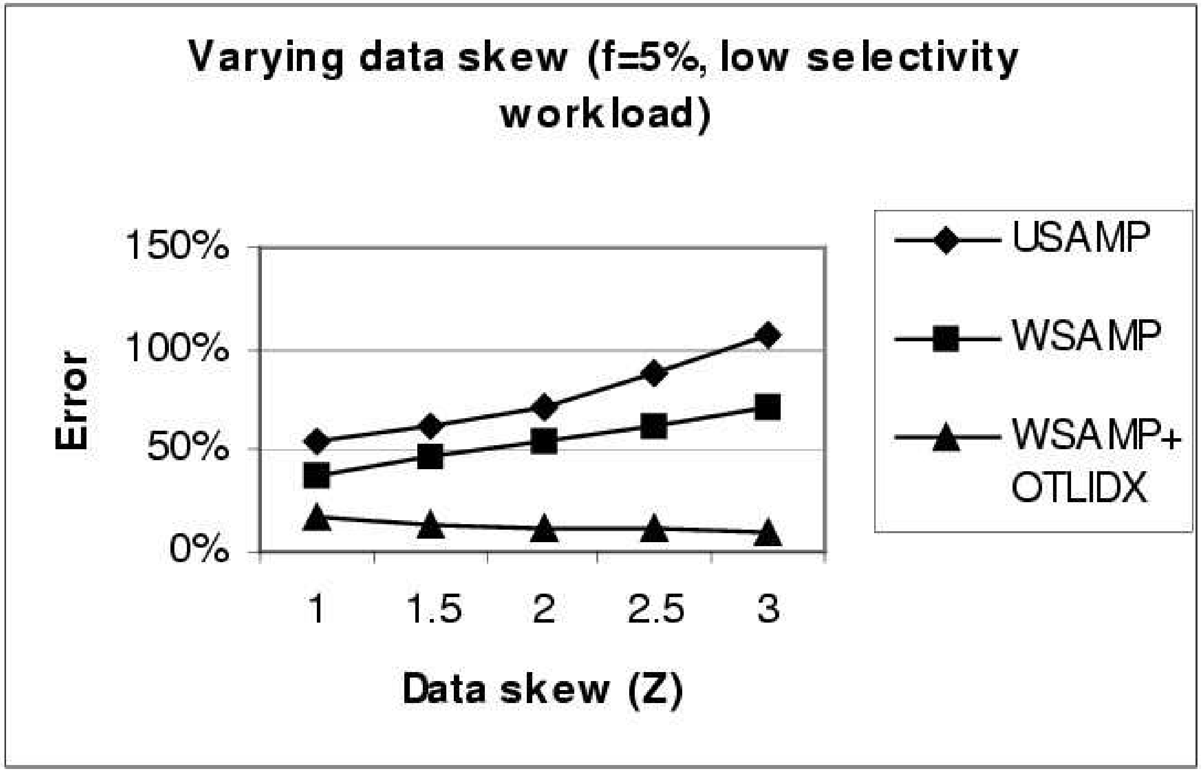


Figure 1. Error versus data skew

Experimental Results

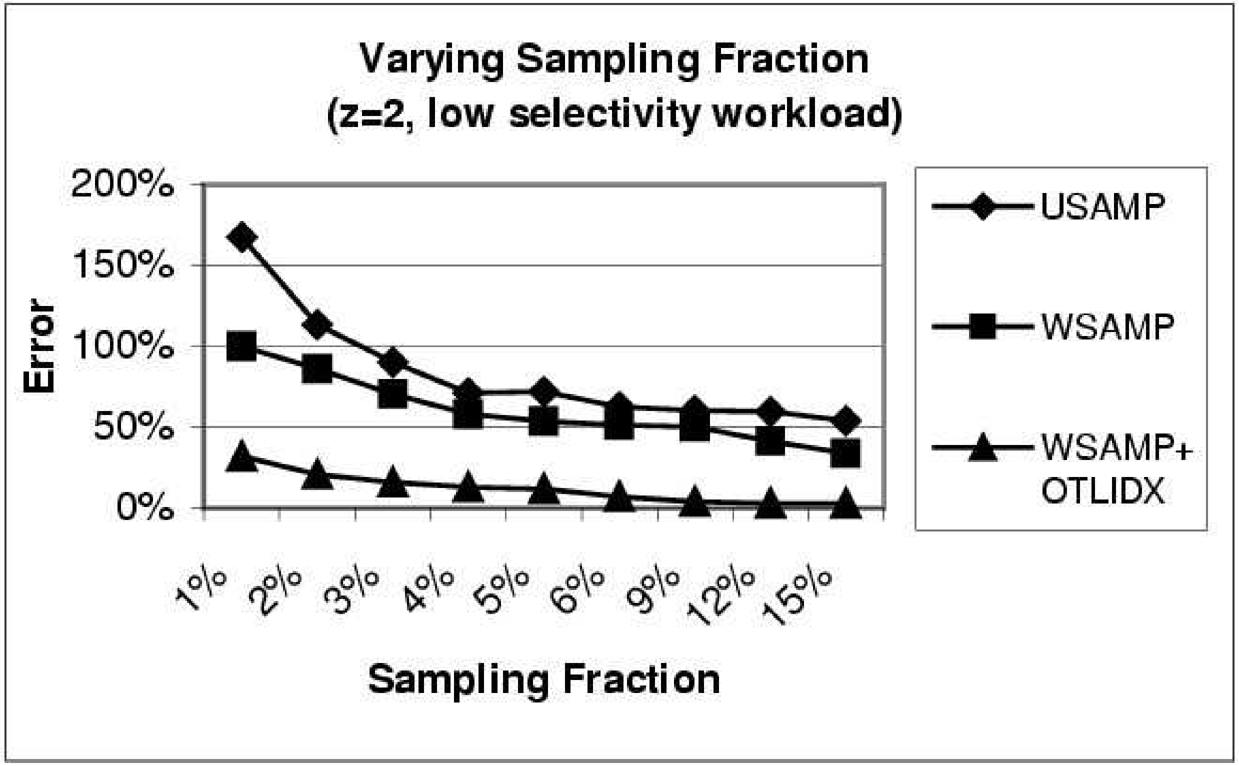


Figure 2. Error versus sampling fraction

Experimental Results

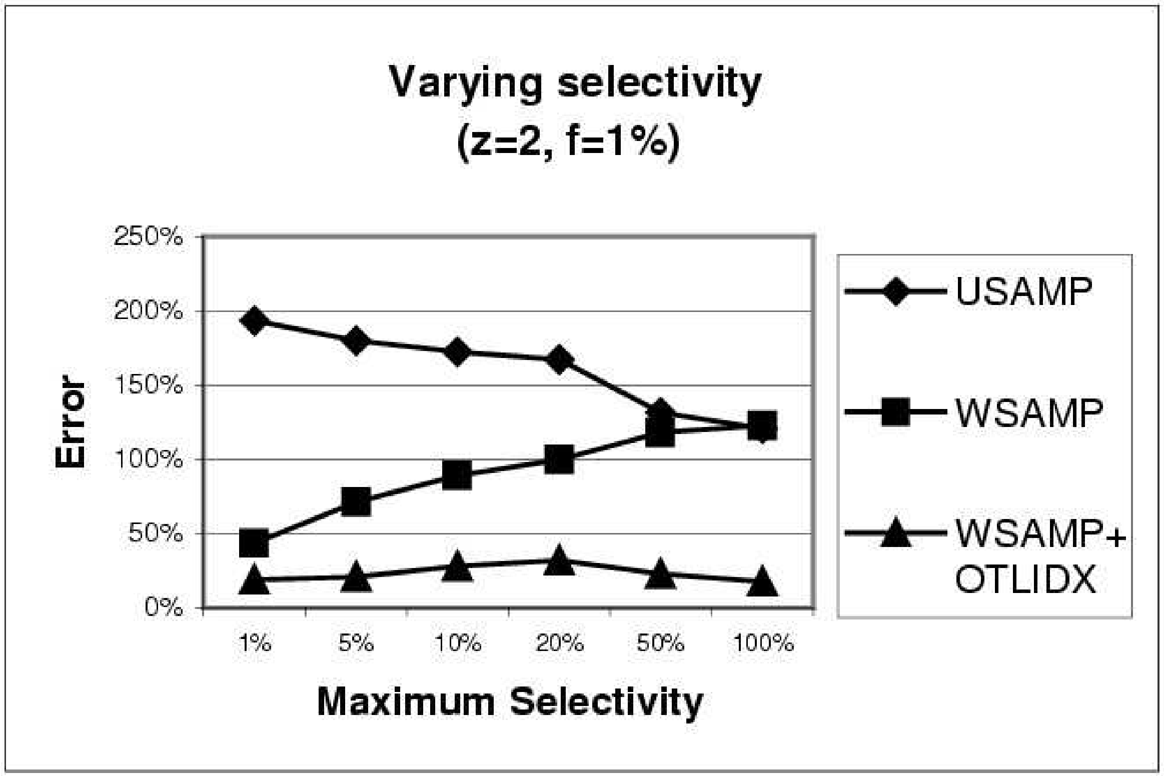


Figure 3. Error versus selectivity of queries

Questions?

Thank you!