Database Supports for Efficient Frequent Pattern Mining

Ruoming Jin
Kent State University

Joint work with Dave Furhy (KSU), Scott McCallen (KSU), Dong Wang (KSU), Yuri Breitbart (KSU), and Gagan Agrawal (OSU)
Motivation

• Data mining is an iterative process
  – Mining at different support level
  – Mining with different dimensions
  – Mining with different constraints
  – Comparative mining

• Standard data mining operators being implemented in modern database system
  – Oracle, SQL server, DB2, ...

• Need fundamental techniques to speedup the mining process!
Frequent Itemset Mining (FIM)

- One of the most well-studied area in KDD; one of the most widely used data mining techniques; one of the most costly data mining operators
- Tens (or maybe well over one hundred) of algorithms have been developed
  - Among them, Apriori and FP-Tree
- Frequent Pattern Mining (FPM)
  - Sequences, Trees, Graphs, Geometric structures, ...
- However, FIM/FPM can still be very time consuming!
Let’s first have a quick review

<table>
<thead>
<tr>
<th>TID</th>
<th>Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>{ A, B, E }</td>
</tr>
<tr>
<td>200</td>
<td>{ B, D }</td>
</tr>
<tr>
<td>300</td>
<td>{ A, B, E }</td>
</tr>
<tr>
<td>400</td>
<td>{ A, C }</td>
</tr>
<tr>
<td>500</td>
<td>{ B, C }</td>
</tr>
<tr>
<td>600</td>
<td>{ A, C }</td>
</tr>
<tr>
<td>700</td>
<td>{ A, B }</td>
</tr>
<tr>
<td>800</td>
<td>{ A, B, C, E }</td>
</tr>
<tr>
<td>900</td>
<td>{ A, B, C }</td>
</tr>
<tr>
<td>1000</td>
<td>{ A, C, E }</td>
</tr>
</tbody>
</table>

- Desired frequency 50%
  - \{A\}, \{B\}, \{C\}, \{A,B\}, \{A,C\}

- Down-closure (apriori) property
  - If an itemset is frequent, all of its subset must also be frequent
Roadmap

• Techniques for Frequent Itemset Mining on Multiple Databases

• Cardinality Estimation for Frequent Itemsets
Why we care about mining multiple datasets?

• Multiple datasets are everywhere
  - Data warehouse
  - Data collected at different places, at different time
  - A large dataset can be logically partitioned into several small datasets

• Comparing the patterns from different datasets is very important

• Combining mining results from each individual dataset is not good enough
Motivating Examples

• Mining the Data Warehouse for a Nationwide Store:
  - Three branches in OH, MI, CA
  - One week’s retail transactions

• Queries
  - Find the itemsets that are frequent with support level 0.1% in each of the stores
  - Find the itemsets that are frequent with support level 0.05% in both the stores in midwest, but are very infrequent (support less that 0.01%) in the west coast store
Finding Signature Itemsets for Network Intrusion

- **TCP-dump dataset**
  - Split the available data into several sub-datasets, corresponding to different intrusion types

- **Queries**
  - Find the itemsets that are frequent with a support level 80% in either of the intrusion datasets, but are very infrequent (support less than 50%) in the normal dataset.
  - Find the itemsets that are frequent with a support level 85% in one of the intrusion datasets, but are very infrequent (support less than 65%) in all other datasets.
So, how to answer these queries?

• Imagine we have only two transaction datasets, A and B

• A simple query Q1
  - Find the itemsets that are frequent in A and B with support level 0.1 and 0.3, respectively, or the itemsets that are frequent in A and B with support level 0.3 and 0.1, respectively.

• We have the following options to evaluate this query
  - Option 1
    • Finding frequent itemsets in A with support level 0.1
    • Finding frequent itemsets in B with support level 0.3
    • Finding frequent itemsets in A with support level 0.3
    • Finding frequent itemsets in B with support level 0.1
How to? (con’t)

- Option 2
  - Finding frequent itemsets in A with support 0.1
  - Finding frequent itemsets in B with support 0.1

- Option 3
  - Finding frequent itemsets in A (or B) with support 0.1
    - Among them, finding itemsets that are also frequent in B (or A) with support 0.1

- Option 4
  - Finding frequent itemsets in A with support 0.3
    - Among them, finding itemsets that are also frequent in B with support 0.1
  - Finding frequent itemsets in B with support 0.3
    - Among them, finding itemsets that are also frequent in A with support 0.1

- ...

Depending on the characteristics of datasets A and B, and the support levels, each option can have very different total mining cost!
Challenges

• Goal
  – Develop a systematic approach to find efficient options (query plans) to answer these queries

• The key issues
  – How to formally define the search space of all possible options for a given query?
    • How to formally describe a mining query across multiple datasets?
    • What are the basic mining operators that can be used in the evaluation?
  – How to identify the efficient query plans?
    • The cost of basic mining operators may not be available
Our Contributions

- SQL-Extension to describe mining queries across multiple datasets
- Basic algebra and new mining operators for query evaluation
- M-Table to explore the possible query plans
- Algorithms for generating efficient query plans
SQL-Extension (1) - Virtual Frequency Table (F-Table)

- Multiple transaction datasets A1, ..., Am
- Item={1, 2, ..., n}
- F-Table scheme
  - Frequency (I, A1, ..., Am)
- Examples:
  - A1, A2
  - Item={1, 2, 3}
- Why virtual table?
  - |Item|=1000

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,3}</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
<td>0.3</td>
</tr>
<tr>
<td>{1,2,3}</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>
SQL Extension (2) - Querying the F-Table

```
SELECT F.I, F.A1, F.A2
FROM Frequency (I, A1, A2) F
WHERE F.A1 ≥ 0.5
AND F.A2 ≥ 0.4

SELECT F.I, F.A1, F.A2
FROM Frequency (I, A1, A2) F
WHERE F.A1 ≥ 0.5
AND F.A2 < 0.4
```

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,3}</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
<td>0.3</td>
</tr>
<tr>
<td>{1,2,3}</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>
### Basic Algebra (1) - Single Frequent Itemsets Mining Operator

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,3}</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
<td>0.3</td>
</tr>
<tr>
<td>{1,2,3}</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

**F-Table F(I,A1,A2)**

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
</tr>
</tbody>
</table>

**SF(A1,0.5)**

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
</tr>
</tbody>
</table>

**SF(A2,0.4)**

<table>
<thead>
<tr>
<th>I</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.7</td>
</tr>
<tr>
<td>{2}</td>
<td>0.6</td>
</tr>
<tr>
<td>{3}</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,3}</td>
<td>0.4</td>
</tr>
</tbody>
</table>
### Basic Algebra (2) - Operations

#### Intersection (∩)

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

SF(A1,0.5) ∩ SF(A2,0.4)

#### Union (⊔)

<table>
<thead>
<tr>
<th>I</th>
<th>A1</th>
<th>A2</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1}</td>
<td>0.6</td>
<td>0.7</td>
</tr>
<tr>
<td>{2}</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>{3}</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,2}</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>{1,3}</td>
<td></td>
<td>0.4</td>
</tr>
<tr>
<td>{2,3}</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

SF(A1,0.5) ⊔ SF(A2,0.4)

(Number of elements) (NULL 0°)
Mapping SQL Queries to Algebra

SELECT F.I, F.A, F.B, F.C, F.D
FROM Frequency(I,A,B,C,D) F
WHERE (F.A ≥ 0.1 AND F.B ≥ 0.1 AND F.D ≥ 0.05)
    OR (F.C ≥ 0.1 AND F.D ≥ 0.1 AND
        (F.A ≥ 0.05 OR F.B ≥ 0.05))

Condition = (A ≥ 0.1 ∧ B ≥ 0.1 ∧ D ≥ 0.05) ∨
             (C ≥ 0.1 ∧ D ≥ 0.1 ∧ A ≥ 0.05) ∨
             (C ≥ 0.1 ∧ D ≥ 0.1 ∧ B ≥ 0.05)

(SF(A,0.1) ∩ SF(B,0.1) ∩ SF(D, 0.05)) ∪
(SF(A,0.05) ∩ SF(C,0.1) ∩ SF(D,0.1)) ∪
(SF(B,0.05) ∩ SF(C,0.1) ∩ SF(D,0.1))
Basic Optimization Tools

- **New Mining Operators**
  - Frequent itemsets mining operator with constraints \( CF(A_j, \alpha, X) \)
    - \( SF(A, 0.1) \cap SF(B, 0.1) \)
    - \( SF(A, 0.1), CF(B, 0.1, SF^I(A, 0.1)) \)
    - \( SF(B, 0.1), CF(A, 0.1, SF^I(B, 0.1)) \)
  - Group frequent itemset mining operator
    - \( GF(<A, 0.1>, <B, 0.1>) \)

- **Containing Relationship**
  - \( SF(A, 0.3) \subseteq SF(A, 0.1) \)
  - \( CF(B, 0.1, SF^I(A, 0.3)) \subseteq CF(B, 0.1, SF^I(A, 0.1)) \)
  - \( GF(<A, 0.1>, <B, 0.3>) \subseteq GF(<A, 0.1>, <B, 0.1>) \)
Alternative Query Plans

```
SELECT F.I, F.A, F.B
FROM Frequency(I,A,B) F
WHERE (F.A ≥ 0.1 AND F.B ≥ 0.3)
    OR (F.A ≥ 0.3 OR F.B ≥ 0.1)
```

Query Q1:
Find the itemsets that are frequent in A and B with support level 0.1 and 0.3, respectively, or the itemsets that are frequent in A and B with support level 0.3 and 0.1, respectively.
Alternative Query Plans

SELECT F.I, F.A, F.B
FROM Frequency(I,A,B) F
WHERE (F.A ≥ 0.1 AND F.B ≥ 0.3) OR (F.A ≥ 0.3 OR F.B ≥ 0.1)

• Query Plan 1:
  – SF(A,0.1), SF(B,0.3), SF(A,0.3), SF(B,0.1)
• Query Plan 2: (Using Containing Relationship)
  – SF(A,0.1), SF(B,0.1)
• Query Plan 3: (Using CF)
  – SF(A,0.1), CF(B,0.1, SF(A,0.1))
• Query Plan 4: (Using CF)
  – SF(B,0.1), CF(A,0.1, SF(A,0.1))
• Query Plan 5: (Using GF)
  – GF(<A,0.1>, <B,0.3>), GF(<A,0.3>, <B,0.1>)

(SF(A,0.1) △ SF(B,0.3)) ⊆ SF(A,0.1) △ SF(B,0.1)
(SF(A,0.3) △ SF(B,0.1)) ⊆ SF(A,0.1) △ SF(B,0.1)

How can we generate efficient query plans systematically by using these basic tools?
M-Table Representation

\[(SF(A,0.1) \cap SF(B,0.1) \cap SF(D, 0.05)) \cup (SF(A,0.05) \cap SF(C,0.1) \cap SF(D,0.1)) \cup (SF(B,0.05) \cap SF(C,0.1) \cap SF(D,0.1))\]

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
M-Table Representation

\[(\text{SF}(A,0.1) \cap \text{SF}(B,0.1) \cap \text{SF}(D,0.05)) \cup (\text{SF}(A,0.05) \cap \text{SF}(C,0.1) \cap \text{SF}(D,0.1)) \cup (\text{SF}(B,0.05) \cap \text{SF}(C,0.1) \cap \text{SF}(D,0.1))\]

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>
### Coloring the M-Table

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td><strong>D</strong></td>
<td>0.05</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

- **SF** and **GF** operators are order-independent,
- **CF** operators are order-dependent!

---

2008-3-18
Two-Phase Heuristic Query Plan Generation (ICDE’06)

- **Phase 1**
  - Using SF operators so that each column has at least one cell being colored
  - GF operators can be used in this stage

- **Phase 2**
  - Use the CF operators to color all other non-empty cells in the table

- **Minimizing the cost of each stage**
  - Cost functions are not available
  - CF operator are order-dependent
  - Both phases rely on heuristics
Phase 1 (Using only SF, Algorithm CF-1)

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td>0.05</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Possible query plans (only considering support level):

- SF(A,0.1), SF(C,0.1)
- SF(A,0.1), SF(D,0.1)
- SF(B,0.1), SF(C,0.1)
- SF(B,0.1), SF(D,0.1)

We can enumerate the query plans for Phase 1, and base on the heuristics for the cost function to pick up a minimal one.
Phase 2 (Algorithm CF-1)

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

For each row, find the lowest support level among the non-colored cells;
On each row, we invoke the CF operator with the lowest support level;
The invocation order of CF operators is in decreasing order of the support levels.

CF(A,0.05, SF(C,0.1))
CF(B,0.05, (SF(A,0.1) ∪ SF(C,0.1)))
CF(D,0.05, (SF(A,0.1) ∩ SF(B,0.1)) ∪ SF(C,0.1))
CF(C,0, (SF(A,0.1) ∩ SF(B,0.1)))
Cost-Based Query Optimization (EDBT’08)

• Cost estimation for SF and CF
  – Factors:
    • The number of transactions: \( n \)
    • The average length of the transactions: \(|I|\)
    • The density of the datasets: \( d \) (entropy of correlations)
    • The support level: \( s \)
  – Formula:
    \[
    \log C = \beta_0 + \beta_1 s + \beta_2 d + \beta_3 \log n + \beta_4 \log |I| 
    \]
  – Regression to determine parameters
  – CF based on SF
Cost-based Query Plan Generation

• Query plan enumeration
  – Similar to enumerating Partially Ordered Sets (poset)

• Algorithm utilizing cost estimation
  – Dynamic Programming
  – Branch-and-Bound

2008-3-18
System Architecture for Mining Multiple Datasets

A1
Query Evaluation

A2
Knowledgeable Cache

Am
Query Plan Optimizer

Query queue

Multiple Query Optimization

Using past mining results to help answering new queries

2008-3-18
Summary of Experimental Evaluation

- **Datasets**
  - IPUMS
    - 1990-5% census micro-data
    - 50,000 records, NY, NJ, CA, WS, 57 attributes
  - DARPA's Intrusion Detection
    - DARPA data sets
    - Neptune, Smurf, Satan, and normal
  - IBM's Quest

- **Single query plan optimization**
  - Heuristic algorithm generate efficient query plans, which achieve more than an order of magnitude speedup, compared with the naïve evaluation
  - The cost-based algorithm reduces the mining cost of the query plan generated by the heuristic algorithm by an average of 20% per query (significantly improves 40% of the queries)

- The multiple query optimization and knowledgeable cache can buy us an additional speedup up to an average of 9 times, compared with using the single query optimization
Roadmap

- Techniques for mining multiple databases
- Cardinality Estimation for Frequent Itemsets
Why we care about # of Frequent Itemsets (FI)?

• Help reduce the number of execution of frequent itemset mining operators
• Intelligently choosing the right parameters (support level) and right dimensions (items)
• Scheduling of data mining operators
  – Mining multiple databases
  – Cardinality estimation
  – Cost estimation
Is this problem hard?

• Counting FI is \#P-complete
  – Reduced to the number of satisfying assignments of a monotone-2CNF formula

• Counting maximal FI is \#P-complete
  – Reduced to the problem of counting the number of maximal bipartite cliques in a bipartite graph

• We have to resort to approximation
Our contributions

• We perform the first theoretical investigation of the sampling estimator
  – Asymptotically unbiased, consistent, and biased

• We propose the first algorithm (sketch matrix estimator) to estimate #FI without using sampling
Sampling Estimator

• What is the sampling estimator?
  – Sample the entire datasets
  – Count the #FI (by Enumeration) on the sample

• A simple example
  – A total of 100 transactions in DB
    • 50 Transactions: \{1,2,3,4,5,6,7,8,9,10,11,12\}
    • 25 Transactions: \{1,2,3,4,5,6,7,8,9,10\}
    • 25 Transactions: \{1,2,3,4,5\}
  – Support (50%, 51%, 60%, 75%)
    • $2^{12}-1=4095$, $2^{10}-1=1023$,
Average #FI from 500 samples (sampling with replacement)
Sampling Tends to Overestimate!

1000 samples (sampling with replacement)

2008-3-18
Sampling Behavior

- Asymptotic behavior of $\hat{Z}$
  - $Z_1$: the number of itemsets whose support exactly equal to the minimal support level
  - $Z_2$: the number of itemsets whose support is higher than the minimal support level
  - $\lim E(\hat{Z}) = Z_2 + Z_1 / 2$
- Consistent only when $Z_1 = 0$
  - $\lim \Pr(|\hat{Z} - Z| > \varepsilon) = \lim \Pr(|\hat{Z} - Z_2| > \varepsilon) = 0$
- The reason for bias
  - Skewed distribution for FIM

2008-3-18
The Sampling Problems

- As database grows, sample size needs to grow as well
  - The sample size can become very large
  - Running time/Memory cost
- Running time is not only determined by the number of transactions
  - Certain computation is determined by the “complexity” of database
Running Time of Sampling

connect.dat

accidents.dat

2008-3-18
Basic Ideas of Sketch Matrix Estimator

• Data Summarization
  – Compress the entire dataset into a sketch matrix for estimation purpose
  – The size of the matrix << the size of the database

• Estimation Process
  – Treat each compressed subcolumn corresponding to each cell as an (independent) random variable with Binomial distribution
### Data Summarization

#### Table

<table>
<thead>
<tr>
<th></th>
<th>b₁=3</th>
<th>b₂=4</th>
<th>b₃=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁=3</td>
<td>1 1 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>1 1 1</td>
<td>1 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>1 0 1</td>
<td>0 1 0</td>
<td>0 1 1</td>
</tr>
<tr>
<td></td>
<td>0 0 0</td>
<td>1 1 0</td>
<td>1 0 0</td>
</tr>
<tr>
<td></td>
<td>0 0 0</td>
<td>1 1 1</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>0 1 0</td>
<td>0 1 1</td>
<td>1 0 0</td>
</tr>
<tr>
<td></td>
<td>0 0 0</td>
<td>1 1 1</td>
<td>1 0 0</td>
</tr>
<tr>
<td>a₂=4</td>
<td>1 1 1</td>
<td>1 1 0</td>
<td>1 1 1</td>
</tr>
<tr>
<td></td>
<td>1 1 1</td>
<td>1 1 0</td>
<td>1 1 1</td>
</tr>
</tbody>
</table>

#### Matrix

<table>
<thead>
<tr>
<th></th>
<th>b₁</th>
<th>b₂</th>
<th>b₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>7/9</td>
<td>2/12</td>
<td>7/9</td>
</tr>
<tr>
<td>a₂</td>
<td>1/12</td>
<td>13/16</td>
<td>0</td>
</tr>
<tr>
<td>a₃</td>
<td>1</td>
<td>6/8</td>
<td>1</td>
</tr>
</tbody>
</table>
The Simple Case

<table>
<thead>
<tr>
<th>a₁</th>
<th>b₁</th>
<th>b₂</th>
<th>b₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>a₂</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>a₃</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

# of rows (transactions):

a₁ = a₂ = a₃ = 1000

# of columns (items):

b₁ = b₂ = b₃ = 100

The minimal support level = 10%

#FI = 3 \times (2^{100} - 1) + (2^{100} - 1)(2^{100} - 1)
1. Estimation Problem: How to do estimation based on the sketch matrix?

2. Optimization Problem: What is the good sketch matrix for the estimation?
Estimation Algorithm (1)

<table>
<thead>
<tr>
<th></th>
<th>b₁</th>
<th>b₂</th>
<th>b₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>0.9</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>a₂</td>
<td>1.0</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>a₃</td>
<td>0.9</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

# of rows (transactions): a₁ = a₂ = a₃ = 1000
# of columns (items): b₁ = b₂ = b₃ = 100

Estimating the # of Frequent Items

\[
\sum_{j=1}^{t} b_j \text{Pr}(\sum_{i=1}^{s} X_{ij} \geq \alpha |T|)
\]

\(X_{11} \sim \text{Bin}(1000, 0.9)\)
\(X_{21} \sim \text{Bin}(1000, 1.0)\)
\(X_{13} \sim \text{Bin}(1000, 0.9)\)

100 x Pr(\(X_{11} + X_{21} + X_{31} \geq 10% \times 3000\))
+ 100 x Pr(\(X_{12} + X_{22} + X_{32} \geq 10% \times 3000\))
+ 100 x Pr(\(X_{13} + X_{23} + X_{33} \geq 10% \times 3000\))
Estimation Algorithm (2)

<table>
<thead>
<tr>
<th></th>
<th>$b_1$</th>
<th>$b_2$</th>
<th>$b_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>0.9</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>$a_2$</td>
<td>1.0</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>$a_3$</td>
<td>0.9</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Estimating the # of Frequent Itemsets from the same columns

\[
\sum_{k=1}^{b_j} \text{Pr}(\sum_{i=1}^{s} X_{ij} \geq \alpha |T|) \]

# of rows (transactions):
$a_1 = a_2 = a_3 = 1000$

# of columns (items):
$b_1 = b_2 = b_3 = 100$

# of frequent 2-itemsets from the same columns

\[
(100 \times 99/2 \times \text{Pr}(X_{11} X'_{11} + X_{21} X'_{21} + X_{31} X'_{31} \geq 10\% \times 3000))
\]
\[
+(100 \times 99/2 \times \text{Pr}(X_{12} X'_{12} + X_{22} X'_{22} + X_{32} X'_{32} \geq 10\% \times 3000))
\]
\[
+(100 \times 99/2 \times \text{Pr}(X_{13} X'_{13} + X_{23} X'_{23} + X_{33} X'_{33} \geq 10\% \times 3000))
\]

$X[2]_{11} = X_{11} X'_{11} \sim \text{Bin}(1000, 0.9 \times 0.9)$

2008-3-18
Estimation Algorithm (3)

Estimating the # of Frequent Itemsets from the different columns

<table>
<thead>
<tr>
<th></th>
<th>b₁</th>
<th>b₂</th>
<th>b₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>a₁</td>
<td>0.9</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>a₂</td>
<td>1.0</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>a₃</td>
<td>0.9</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

# of rows (transactions):  
a₁ = a₂ = a₃ = 1000  
# of columns (items):  
b₁ = b₂ = b₃ = 100

# of frequent 3-itemsets, where two from the first column, and one from the second column

\[(100\times99/2)\times100\times\text{Pr}(X_{11}X'_{11}X_{12}+X_{21}X'_{21}X_{22}+X_{31}X'_{31}X_{32} \geq 10\% \times 3000)\]

\[X_{11}X'_{11}X_{12} \sim \text{Bin}(1000, 0.9\times0.9\times0.1)\]
Estimating the #FI

• Approximate Binomial using Gaussian
  \( \text{Bin}(n,p) \sim N(\mu=np, \sigma^2=np(1-p)) \)

• Applying the cut-off threshold
  Once we found out the particular type of itemsets have # of FIs less than 1, we will not count their supersets
Extensions

- Estimation of Frequent k-itemsets
- Size of the largest Frequent itemsets
- Frequent itemsets for a subset of items and/or for a subset of transactions
Do different sketch matrices matter?
Optimization Problem

• What criteria can help evaluate different sketch matrices in terms of the “goodness” of estimation?

• How to generate the “best” sketch matrix?
Variance Criteria

• A commonly used criteria for the accuracy of an estimator is the variance
• The direct variance is very hard to compute
• The variance of the sum of all the support for every possible itemset

\[ V = \text{Var} \left( \sum_{k_1=0}^{b_1} \cdots \sum_{k_t=0}^{b_t} (\binom{b_1}{k_1}) \times \cdots \times (\binom{b_t}{k_t}) X[k_1, \ldots, k_t]_{i[j_1, \ldots, j_t]} \right) \]

\[ = \sum_{i=1}^{s} \sum_{j=1}^{t} a_i \left( \prod_{j=1}^{t} \left(1 + 3d_{ij}\right)^{b_j} - \prod_{j=1}^{t} \left(1 + d_{ij}\right)^{2b_j} \right) \]
Biclustering algorithm

1. Initially, the transactions and items are randomly partitioned into $s$ and $t$ clusters, respectively.

2. For each transaction and each item, we try to move them to different clusters in order to maximally reduce the variance.

3. Repeat 2 until the improvement is very small or reach certain iterations.
Two-level Hierarchical bi-clustering

- Each block in the matrix will be further divided into smaller blocks
- In order to the estimation, the sub-column group are the same for all blocks in the same column
Experimental Results

![Graph showing experimental results for accidents.dat]

<table>
<thead>
<tr>
<th>Estimation</th>
<th>Support</th>
<th>Total Frequent Item Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-15-4-4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-15-10-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20-20-6-6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25-25-10-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apriori</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2008-3-18
Experimental Results

connect.dat

chess.dat

2008-3-18
Experimental Results

For the dataset mushroom.dat:
- Estimation 15–10–5–5
- Estimation 25–10–15–6
- Estimation 35–20–10–10
- Estimation 50–35–15–3
- Apriori

For the dataset retail.dat:
- Estimation 10–10–1–1
- Estimation 10–10–8–8
- Estimation 20–20–1–1
- Estimation 20–20–5–5
- Apriori

The graphs show the total frequent item sets against support for different estimation methods compared to the Apriori algorithm.
Running Time

accidents.dat

Time to calculate

Support

connect.dat

Time to calculate

Support
# frequent k-itemsets

connect.dat Support = 70

mushroom.dat Support = 12

2008-3-18
Size of Maximal Frequent Itemsets

accidents.dat

connect.dat

Estimation 15–15–4–4
Estimation 15–15–10–10
Estimation 20–20–6–6
Estimation 20–20–10–10
Apriori

Estimation 8–8–8–4
Estimation 20–15–8–8
Estimation 20–20–10–10
Estimation 20–20–15–15
Apriori

Maximal K–Itemsets
Support
Support
Conclusions

• A knowledge discovery and data mining management system (KDDMS)
  - Long-term goal for data mining
  - Interactive data mining
  - New techniques in database-type of environment to support efficient data mining
Thanks!!!

2008-3-18