Distributed Databases

by Farnoush Banaei-Kashani

Excerpt from “Principles of Distributed Database Systems” by M. Tamer Özsu and Patrick Valduriez

Distributed DBMS – Reality

[Diagram showing distributed DBMS architecture]
Distributed Database – User View

Topics

- Introduction
- Background
- Distributed DBMS Architecture
  - Distributed Database Design
- Semantic Data Control
- Distributed Query Processing
- Distributed Transaction Management
- Parallel Database Systems
- Distributed Object DBMS
- Database Interoperability
- Current Issues
Outline

- Design what?
- Fragmentation

Design Approaches

- **Top-down**
  - mostly in designing systems from scratch
  - mostly in homogeneous systems
- **Bottom-up**
  - when the databases already exist at a number of sites
Top-Down Design

Distribution Design Issues

1. Why fragment at all?
2. How to fragment?
3. How much to fragment?
4. How to test correctness?
5. How to allocate?
6. Information requirements?
Fragmentation

- Can't we just distribute relations?
- What is a reasonable unit of distribution?
  - relation
    - views are subsets of relations; partition to achieve locality
    - extra communication
  - fragments of relations (sub-relations)
    - concurrent execution of a number of transactions that access different portions of a relation
    - views that cannot be defined on a single fragment will require extra processing
    - semantic data control (especially integrity enforcement) more difficult

Fragmentation Alternatives - Horizontal

**PROJ₁**: projects with budgets less than $200,000
**PROJ₂**: projects with budgets greater than or equal to $200,000

---

<table>
<thead>
<tr>
<th>PNO</th>
<th>PNAME</th>
<th>BUDGET</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Instrumentation</td>
<td>150000</td>
<td>Montreal</td>
</tr>
<tr>
<td>P2</td>
<td>Database Develop.</td>
<td>135000</td>
<td>New York</td>
</tr>
<tr>
<td>P3</td>
<td>CAD/CAM</td>
<td>150000</td>
<td>New York</td>
</tr>
<tr>
<td>P4</td>
<td>Maintenance</td>
<td>310000</td>
<td>Paris</td>
</tr>
<tr>
<td>P5</td>
<td>CAD/CAM</td>
<td>500000</td>
<td>Boston</td>
</tr>
</tbody>
</table>
Fragmentation Alternatives - Vertical

PROJ₁: information about project budgets
PROJ₂: information about project names and locations

<table>
<thead>
<tr>
<th>PNO</th>
<th>PROJ₁</th>
<th>BUDGET</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Instrumentation</td>
<td>150000</td>
</tr>
<tr>
<td>P2</td>
<td>Database Develop</td>
<td>135000</td>
</tr>
<tr>
<td>P3</td>
<td>CAD/CAM</td>
<td>250000</td>
</tr>
<tr>
<td>P4</td>
<td>Maintenance</td>
<td>310000</td>
</tr>
<tr>
<td>P5</td>
<td>CAD/CAM</td>
<td>500000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PNO</th>
<th>PROJ₂</th>
<th>NAME</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Instrumentation</td>
<td>Montreal</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>Database Develop</td>
<td>New York</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>CAD/CAM</td>
<td>New York</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>Maintenance</td>
<td>Paris</td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>CAD/CAM</td>
<td>Boston</td>
<td></td>
</tr>
</tbody>
</table>

Degree of Fragmentation

finite number of alternatives

Finding the suitable level of partitioning within this range
Correctness of Fragmentation

- Completeness
  - Decomposition of relation $R$ into fragments $R_1, R_2, ..., R_n$ is complete if and only if each data item in $R$ can also be found in some $R_i$.

- Reconstruction
  - If relation $R$ is decomposed into fragments $R_1, R_2, ..., R_n$, then there should exist some relational operator $\forall$ such that $R = \bigvee_{1 \leq i \leq n} R_i$.

- Disjointness
  - If relation $R$ is decomposed into fragments $R_1, R_2, ..., R_n$, and data item $d_i$ is in $R_j$, then $d_i$ should not be in any other fragment $R_h$ ($h \neq j$).

Allocation Alternatives

- Non-replicated
  - partitioned: each fragment resides at only one site

- Replicated
  - fully replicated: each fragment at each site
  - partially replicated: each fragment at some of the sites

- Rule of thumb:
  
  If $\frac{\text{read - only queries}}{\text{update queries}} \geq 1$ replication is advantageous,
  
  otherwise replication may cause problems
Information Requirements

- Four categories:
  - Database information
  - Application information
  - Communication network information
  - Computer system information

Outline

- Design what?
- Fragmentation
Types of Fragmentation

- Horizontal Fragmentation (HF)
  - Primary Horizontal Fragmentation (PHF)
  - Derived Horizontal Fragmentation (DHF)
- Vertical Fragmentation (VF)
- Hybrid Fragmentation (HF)

VF – Information Requirements

- Application Information
  - Attribute affinities
    - a measure that indicates how closely related the attributes are
    - This is obtained from more primitive usage data
  - Attribute usage values
    - Given a set of queries \( Q = \{q_1, q_2, ..., q_r\} \) that will run on the relation \( R[A_1, A_2, ..., A_n]\),
    \[
    uae(q_i, A_j) = \begin{cases} 
    1 & \text{if attribute } A_j \text{ is referenced by query } q_i \\
    0 & \text{otherwise}
    \end{cases}
    \]
VF – Information Requirements

Consider the following 4 queries for relation PROJ

- $q_1$: SELECT BUDGET FROM PROJ WHERE PNO=Value
- $q_2$: SELECT PNAME, BUDGET FROM PROJ
- $q_3$: SELECT PNAME FROM PROJ WHERE LOC=Value
- $q_4$: SELECT SUM(BUDGET) FROM PROJ WHERE LOC=Value

Let $A_1 = \text{PNO}$, $A_2 = \text{PNAME}$, $A_3 = \text{BUDGET}$, $A_4 = \text{LOC}$

\[
\begin{array}{cccc}
A_1 & A_2 & A_3 & A_4 \\
q_1 & 1 & 0 & 1 & 0 \\
q_2 & 0 & 1 & 1 & 0 \\
q_3 & 0 & 1 & 0 & 1 \\
q_4 & 0 & 0 & 1 & 1 \\
\end{array}
\]
Algorithm: 1. Affinity Measure \( \text{aff}(A_i, A_j) \)

The attribute affinity measure between two attributes \( A_i \) and \( A_j \) of the relation \( R[A_1, A_2, \ldots, A_n] \) with respect to the set of applications/queries \( Q = \{q_1, q_2, \ldots, q_q\} \) is defined as follows:

\[
\text{aff}(A_i, A_j) = \sum_{\text{site}(q_k, A_i) = 1} \sum_{\text{site}(q_k, A_j) = 1} \sum_{S_l} \text{ref}_l(q_k) \times \text{acc}_l(q_k)
\]

where \( S_l \) is the \( l \)-th site of the distributed database, \( \text{ref}_l(q_k) \) is the number of access to attributes \( (A_i, A_j) \) for each execution of the query \( q_k \) at site \( S_l \) and \( \text{acc}_l(q_k) \) is the query access frequency measure.

Algorithm: 2. Affinity Matrix \( AA \)

Assume each query in the previous example accesses the attributes once during each execution.

Also assume the access frequencies:

<table>
<thead>
<tr>
<th>( q_1 )</th>
<th>( q_2 )</th>
<th>( q_3 )</th>
<th>( q_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>20</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Then

\[
\text{aff}(A_1, A_3) = 15 \times 1 + 20 \times 1 + 10 \times 1 = 45
\]

and the attribute affinity matrix \( AA \) is:

\[
\begin{bmatrix}
  A_1 & A_2 & A_3 & A_4 \\
  A_1 & 45 & 0 & 45 & 0 \\
  A_2 & 0 & 80 & 5 & 75 \\
  A_3 & 45 & 5 & 53 & 3 \\
  A_4 & 0 & 75 & 3 & 78
\end{bmatrix}
\]
Algorithm: 3. Reorder AA

- Take the attribute affinity matrix $AA$ and reorganize the attribute orders to form clusters where the attributes in each cluster demonstrate high affinity to one another.

$$\begin{array}{cccc}
A_1 & A_2 & A_3 & A_4 \\
A_1 & 45 & 0 & 45 & 0 \\
A_2 & 0 & 80 & 5 & 75 \\
A_3 & 45 & 5 & 53 & 3 \\
A_4 & 0 & 75 & 3 & 78 \\
\end{array} \quad \rightarrow \quad \begin{array}{cccc}
A_1 & A_2 & A_3 & A_4 \\
A_1 & 45 & 45 & 0 & 0 \\
A_2 & 45 & 53 & 5 & 3 \\
A_3 & 0 & 5 & 80 & 75 \\
A_4 & 0 & 3 & 75 & 78 \\
\end{array}$$

Bond Energy Algorithm (BEA)

- **Bond Energy Algorithm** (BEA) has been used for clustering of entities. BEA finds an ordering of entities (in our case attributes) such that the global affinity measure

$$AM = \sum_i \sum_j \text{(affinity of } A_i \text{ and } A_j \text{ with their neighbors)}$$

is maximized.
Affinity Measure

\[ AM = \sum_{i=1}^{n} \sum_{j=1}^{n} \text{aff}(A_i, A_j) \left[ \text{aff}(A_i, A_{j-1}) + \text{aff}(A_j, A_{j+1}) + \text{aff}(A_{i-1}, A_j) + \text{aff}(A_{i+1}, A_j) \right] \]

Boundary conditions:

\[ \text{aff}(A_b, A_j) = \text{aff}(A_i, A_b) = \text{aff}(A_{n+1}, A_j) = \text{aff}(A_i, A_{n+1}) = 0 \]

And since the \( AA \) matrix is symmetric, we revise the definition of affinity measure to:

\[ AM = \sum_{i=1}^{n} \sum_{j=1}^{n} \text{aff}(A_i, A_j) \left[ \text{aff}(A_i, A_{j-1}) + \text{aff}(A_j, A_{j+1}) \right] \]

---

BEA Algorithm

**Input:** The \( AA \) matrix

**Output:** The clustered affinity matrix \( CA \) which is a perturbation of \( AA \)

1. **Initialization:** Place and fix one of the columns of \( AA \) in \( CA \).

2. **Iteration:** Place the remaining \( n-i \) columns in the remaining \( n-i \) positions in the \( CA \) matrix. For each column, choose the placement that makes the most contribution to the global affinity measure.

3. **Row order:** Order the rows according to the column ordering.
### Bond Energy Algorithm (BEA)

**Bond Matrix (AA)**:

\[
\begin{bmatrix}
A_1 & A_2 & A_3 & A_4 & A_5 \\
45 & 0 & 45 & 0 & 0 \\
0 & 80 & 5 & 75 & 0 \\
45 & 5 & 53 & 3 & 0 \\
75 & 3 & 78 & 0 & 0 \\
\end{bmatrix}
\]

**Cohesion Matrix (CA)**:

\[
\begin{bmatrix}
A_1 & A_2 & A_3 & A_4 & A_5 \\
45 & 45 & 0 & 0 & 0 \\
45 & 53 & 5 & 3 & 0 \\
0 & 5 & 80 & 75 & 0 \\
0 & 3 & 75 & 78 & 0 \\
\end{bmatrix}
\]

At each step \(i\):

\[
\begin{bmatrix}
A_1 & A_2 & \ldots & A_{i-1} & A_i \\
\end{bmatrix}_{AM_{old}} \rightarrow \begin{bmatrix}
A_1 & A_2 & \ldots & A_{i-1} & A_i \\
\end{bmatrix}_{AM_{new}}
\]

\[\text{cont}(A_i, A_{i+1}) = AM_{new} - AM_{old}\]

---

**Bond Energy Algorithm (BEA)**

Define \(\text{bond}(A_i, A_j)\) :

\[
\text{bond}(A_i, A_j) = \sum_{r=1}^{n} \text{aff}(A_i, A_r) \times \text{aff}(A_r, A_j)
\]

But:

\[
AM = \sum_{i=1}^{n} \sum_{j=1}^{n} \text{aff}(A_i, A_j) \left[ \text{aff}(A_i, A_{j-1}) + \text{aff}(A_i, A_{j+1}) \right]
\]

\[\Rightarrow AM = \sum_{j=1}^{n} \left[ \text{bond}(A_j, A_{j-1}) + \text{bond}(A_j, A_{j+1}) \right]\]

Do the math:

\[
\text{cont}(A_i, A_{i+1}) = AM_{new} - AM_{old}
\]

\[= 2\text{bond}(A_i, A_{i+1}) + 2\text{bond}(A_i, A_{i+1}) - 2\text{bond}(A_i, A_{i+1})
\]
BEA Example

Consider the following $AA$ matrix and the corresponding $CA$ matrix where $A_1$ and $A_2$ have been placed. Place $A_3$:

$$
\begin{bmatrix}
A_1 & A_2 & A_3 & A_4 \\
A_1 & 45 & 0 & 45 & 0 \\
A_2 & 0 & 80 & 5 & 75 \\
A_3 & 45 & 5 & 53 & 3 \\
A_4 & 0 & 75 & 3 & 78
\end{bmatrix}
$$

$$
\begin{bmatrix}
A_1 & A_2 \\
A_1 & 45 & 0 \\
A_2 & 0 & 80 \\
A_3 & 45 & 5 \\
A_4 & 0 & 75
\end{bmatrix}
$$

Ordering (0-3-1):

$$\text{cont}(A_0, A_3, A_1) = 2\text{bond}(A_0, A_3) + 2\text{bond}(A_3, A_1) - 2\text{bond}(A_0, A_1) = 2^* 0 + 2^* 4410 - 2^* 0 = 8820$$

Ordering (1-3-2):

$$\text{cont}(A_1, A_3, A_2) = 2\text{bond}(A_1, A_3) + 2\text{bond}(A_3, A_2) - 2\text{bond}(A_1, A_2) = 2^* 4410 + 2^* 890 - 2^* 225 = 10150$$

Ordering (2-3-4):

$$\text{cont}(A_2, A_3, A_4) = 1780$$

Therefore, the $CA$ matrix has to form

$$
\begin{bmatrix}
A_1 & A_3 & A_2 \\
45 & 45 & 0 \\
0 & 5 & 80 \\
45 & 53 & 5 \\
0 & 3 & 75
\end{bmatrix}
$$
BEA Example

When \( A_i \) is placed, the final form of the \( CA \) matrix (after row organization) is

\[
\begin{array}{cccc}
A_1 & A_2 & A_3 & A_4 \\
45 & 45 & 0 & 0 \\
45 & 53 & 5 & 3 \\
0 & 5 & 80 & 75 \\
0 & 3 & 75 & 78 \\
\end{array}
\]

Algorithm: 4. Recognizing Clusters in \( CA \)

How can you divide a set of clustered attributes \( \{A_1, A_2, ..., A_n\} \) into two (or more) sets \( \{A_1, A_2, ..., A_i\} \) and \( \{A_i, ..., A_n\} \) such that there are no (or minimal) applications that access both (or more than one) of the sets.
Clustering Measure

Define

\[ TQ = \text{set of applications that access only } TA \]
\[ BQ = \text{set of applications that access only } BA \]
\[ OQ = \text{set of applications that access both } TA \text{ and } BA \]

and

\[ C_{xx} = \sum_{i=1}^{m} \sum_{j=1}^{n} ref(q_i) \times acc(q_j) \]

\[ CTQ = \text{total number of accesses to attributes by applications that access only } TA \]
\[ CBQ = \text{total number of accesses to attributes by applications that access only } BA \]
\[ COQ = \text{total number of accesses to attributes by applications that access both } TA \text{ and } BA \]

Then find the point along the diagonal that maximizes

\[ CTQ = CBQ - COQ^2 \]

Clustering Problems

Two problems:

1. Cluster forming in the middle of the CA matrix
   - Shift a row up and a column left and apply the algorithm to find the “best” partitioning point
   - Do this for all possible shifts
   - Cost \( O(m^2) \)

2. More than two clusters
   - \( m \)-way partitioning
   - try 1, 2, ..., \( m-1 \) split points along diagonal and try to find the best point for each of these
   - Cost \( O(2^m) \)
VF – Correctness

A relation \( R \), defined over attribute set \( A \) and key \( K \), generates the vertical partitioning \( F_R = \{ R_1, R_2, ..., R_i \} \).

- **Completeness**
  - The following should be true for \( A \):
    \[ A = \bigcup A_{R_i} \]

- **Reconstruction**
  - Reconstruction can be achieved by
    \[ R \rightarrow\bowtie^R R_i \quad \forall R_i \in F_R \]

- **Disjointness**
  - TID’s are not considered to be overlapping since they are maintained by the system
  - Duplicated keys are not considered to be overlapping

---

Types of Fragmentation

- **Horizontal Fragmentation (HF)**
  - Primary Horizontal Fragmentation (PHF)
  - Derived Horizontal Fragmentation (DHF)

- **Vertical Fragmentation (VF)**

- **Hybrid Fragmentation (HF)**
Our Running Example

<table>
<thead>
<tr>
<th>EMP</th>
<th>ASG</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>E1</td>
</tr>
<tr>
<td>E2</td>
<td>E2</td>
</tr>
<tr>
<td>E3</td>
<td>E3</td>
</tr>
<tr>
<td>E4</td>
<td>E4</td>
</tr>
<tr>
<td>E5</td>
<td>E5</td>
</tr>
<tr>
<td>E6</td>
<td>E6</td>
</tr>
<tr>
<td>E7</td>
<td>E7</td>
</tr>
<tr>
<td>E8</td>
<td>E8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ENO</th>
<th>ENAME</th>
<th>TITLE</th>
<th>ENO</th>
<th>PNO</th>
<th>RESP</th>
<th>DUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>J. Doe</td>
<td>Elect. Eng.</td>
<td>E1</td>
<td>P1</td>
<td>Manager</td>
<td>12</td>
</tr>
<tr>
<td>E2</td>
<td>M. Smith</td>
<td>Syst. Anal.</td>
<td>E2</td>
<td>P1</td>
<td>Analyst</td>
<td>24</td>
</tr>
<tr>
<td>E4</td>
<td>J. Miller</td>
<td>Programmer</td>
<td>E4</td>
<td>P3</td>
<td>Consultant</td>
<td>10</td>
</tr>
<tr>
<td>E5</td>
<td>B. Casey</td>
<td>Syst. Anal.</td>
<td>E5</td>
<td>P4</td>
<td>Engineer</td>
<td>48</td>
</tr>
<tr>
<td>E7</td>
<td>R. Davis</td>
<td>Mech. Eng.</td>
<td>E7</td>
<td>P3</td>
<td>Manager</td>
<td>24</td>
</tr>
<tr>
<td>E8</td>
<td>J. Jones</td>
<td>Syst. Anal.</td>
<td>E8</td>
<td>P4</td>
<td>Manager</td>
<td>48</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROJ</th>
<th>PAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNO</td>
<td>PNAME</td>
</tr>
<tr>
<td>P1</td>
<td>Instrumentation</td>
</tr>
<tr>
<td>P2</td>
<td>Database Develop</td>
</tr>
<tr>
<td>P3</td>
<td>CAD/CAM</td>
</tr>
<tr>
<td>P4</td>
<td>Maintenance</td>
</tr>
</tbody>
</table>

PHF – Information Requirements

- Database Information
  - relationship

  ```plaintext
  EMP \[\rightarrow\] L_1 \[\rightarrow\] PROJ
  EMP \[\rightarrow\] L_2 \[\rightarrow\] ASG
  PAY \[\rightarrow\] L_3 \[\rightarrow\] ASG

<table>
<thead>
<tr>
<th>ENO</th>
<th>ENAME</th>
<th>TITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>J. Doe</td>
<td>Elect. Eng.</td>
</tr>
<tr>
<td>E2</td>
<td>M. Smith</td>
<td>Syst. Anal.</td>
</tr>
<tr>
<td>E4</td>
<td>J. Miller</td>
<td>Programmer</td>
</tr>
<tr>
<td>E5</td>
<td>B. Casey</td>
<td>Syst. Anal.</td>
</tr>
<tr>
<td>E8</td>
<td>J. Jones</td>
<td>Syst. Anal.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ENO</th>
<th>PNO</th>
<th>RESP</th>
<th>DUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>P1</td>
<td>Manager</td>
<td>12</td>
</tr>
<tr>
<td>E2</td>
<td>P1</td>
<td>Analyst</td>
<td>24</td>
</tr>
<tr>
<td>E3</td>
<td>P2</td>
<td>Analyst</td>
<td>6</td>
</tr>
<tr>
<td>E4</td>
<td>P3</td>
<td>Consultant</td>
<td>10</td>
</tr>
<tr>
<td>E5</td>
<td>P4</td>
<td>Engineer</td>
<td>48</td>
</tr>
<tr>
<td>E6</td>
<td>P2</td>
<td>Programmer</td>
<td>18</td>
</tr>
<tr>
<td>E7</td>
<td>P3</td>
<td>Manager</td>
<td>24</td>
</tr>
<tr>
<td>E8</td>
<td>P4</td>
<td>Manager</td>
<td>48</td>
</tr>
</tbody>
</table>

  PROJ
  PROJ
  PROJ
  PROJ

<table>
<thead>
<tr>
<th>PNO</th>
<th>PNAME</th>
<th>BUDGET</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Instrumentation</td>
<td>150000</td>
<td>Montreal</td>
</tr>
<tr>
<td>P2</td>
<td>Database Develop</td>
<td>135000</td>
<td>New York</td>
</tr>
<tr>
<td>P3</td>
<td>CAD/CAM</td>
<td>250000</td>
<td>New York</td>
</tr>
<tr>
<td>P4</td>
<td>Maintenance</td>
<td>310000</td>
<td>Paris</td>
</tr>
</tbody>
</table>

  TITLE | SAL
  Elect. Eng. | 40000
  Syst. Anal. | 34000
  Mech. Eng. | 27000
  Programmer | 24000
```

- cardinality of each relation: \(\text{card}(R)\)
PHF – Information Requirements

- Application Information
  - **simple predicates**: Given \( R[A_1, A_2, ..., A_n] \), a simple predicate \( p_j \) is
    \[
    p_j : A_i \theta \text{ Value}
    \]
    where \( \theta \in \{=, <, \leq, >, \geq, \neq \} \), \( Values \) \( D_i \) and \( D_i \) is the domain of \( A_i \).
    For relation \( R \) we define \( Pr = \{ p_1, p_2, ..., p_m \} \).
    Example:
    - \( PNAME = "Maintenance" \)
    - \( BUDGET \leq 200000 \)
  - **minterm predicates**: Given \( R \) and \( Pr = \{ p_1, p_2, ..., p_m \} \)
    define \( M = \{ m_1, m_2, ..., m_r \} \) as
    \[
    M = \{ m_i | m_i = \bigwedge_{p_j \in Pr} p_j^* \}, \ 1 \leq j \leq m, \ 1 \leq i \leq r
    \]
    where \( p_j^* = p_j \) or \( p_j^* = \neg(p_j) \).

---

Example

\( m_1: PNAME="Maintenance" \land BUDGET\leq200000 \)

\( m_2: \neg(PNAME="Maintenance") \land BUDGET\leq200000 \)

\( m_3: PNAME="Maintenance" \land \neg(BUDGET\leq200000) \)

\( m_4: \neg(PNAME="Maintenance") \land \neg(BUDGET\leq200000) \)
Primary Horizontal Fragmentation

Definition:

\[ R_j = \sigma_{F_j}(R), \ 1 \leq j \leq w \]

where \( F_j \) is a selection formula, which is (preferably) a minterm predicate.

Therefore,

A horizontal fragment \( R_j \) of relation \( R \) consists of all the tuples of \( R \) which satisfy a minterm predicate \( m_i \).

Given a set of minterm predicates \( M \), there are as many horizontal fragments of relation \( R \) as there are minterm predicates.

Set of horizontal fragments also referred to as minterm fragments.

Selecting Simple Predicates

Given: A relation \( R \), the set of simple predicates \( Pr \)

Output: The set of fragments of \( R = \{R_1, R_2, ..., R_w\} \) which obey the fragmentation rules.

Preliminaries:

- \( Pr \) should be complete
- \( Pr \) should be minimal
Completeness of Simple Predicates

A set of simple predicates $Pr$ is said to be complete if and only if the accesses to the tuples of the minterm fragments defined on $Pr$ requires that two tuples of the same minterm fragment have the same probability of being accessed by any application.

Example:
- Assume $PROJ[PNO, PNAME, BUDGET, LOC]$ has two applications defined on it.
- Find the budgets of projects at each location. (1)
- Find projects with budgets less than $200000$. (2)

According to (1),

$$Pr = \{\text{LOC} = \text{"Montreal"}, \text{LOC} = \text{"New York"}, \text{LOC} = \text{"Paris"}\}$$

which is not complete with respect to (2).

Modify

$$Pr = \{\text{LOC} = \text{"Montreal"}, \text{LOC} = \text{"New York"}, \text{LOC} = \text{"Paris"}, \ BUDGET \leq 200000, \ BUDGET > 200000\}$$

which is complete.
Minimality of Simple Predicates

- If a predicate influences how fragmentation is performed, (i.e., causes a fragment \( f \) to be further fragmented into, say, \( f_i \) and \( f_j \)) then there should be at least one application that accesses \( f_i \) and \( f_j \) differently.
- In other words, the simple predicate should be \textit{relevant} in determining a fragmentation.

Example:

\[
Pr = \{\text{LOC}="\text{Montreal}", \text{LOC}="\text{New York}", \text{LOC}="\text{Paris}", \text{BUDGET} \leq 200000, \text{BUDGET} > 200000\}
\]

is minimal (in addition to being complete). However, if we add

\[
\text{PNAME} = "\text{Instrumentation}\"
\]

then \( Pr \) is not minimal.
COM_MIN Algorithm

**Given:** a relation $R$ and a set of simple predicates $Pr$

**Output:** a complete and minimal set of simple predicates $Pr'$ for $Pr$

**Rule 1:** a relation or fragment is partitioned into at least two parts which are accessed differently by at least one application.

---

1. **Initialization:**
   - find a $p_i \in Pr$ such that $p_i$ partitions $R$ according to Rule 1
   - set $Pr' = p_i$; $Pr \leftarrow Pr - p_i$; $F \leftarrow f_i$

2. **Iteratively add predicates to $Pr'$ until it is complete:**
   - find a $p_i \in Pr$ such that $p_i$ partitions some $f_b$ defined according to minterm predicate over $Pr'$ according to Rule 1
   - set $Pr' = Pr' \cup p_i$; $Pr \leftarrow Pr - p_i$; $F \leftarrow F \cup f_i$
   - if $\exists p_b \in Pr'$ which is nonrelevant then
     - $Pr' \leftarrow Pr' - p_b$
     - $F \leftarrow F - f_b$
PHORIZANTAL Algorithm

Makes use of COM_MIN to perform fragmentation.

Input: a relation $R$ and a set of simple predicates $Pr$

Output: a set of minterm predicates $M$ according to which relation $R$ is to be fragmented

1. $Pr' \leftarrow \text{COM_MIN}(R,Pr)$
2. determine the set $M$ of minterm predicates
3. determine the set $I$ of implications among $p_i \in Pr$
4. eliminate the contradictory minterms from $M$

Examples

- Two candidate relations: PAY and PROJ.
- Fragmentation of relation PAY
  - Application: Check the salary info and determine raise.
  - Employee records kept at two sites application run at two sites
  - Simple predicates
    - $p_1 : \text{SAL} \leq 30000$
    - $p_2 : \text{SAL} > 30000$
  - Minterm predicates
    - $m_1 : (\text{SAL} \leq 30000)$
    - $m_2 : \neg(\text{SAL} \leq 30000) = (\text{SAL} > 30000)$
Examples

<table>
<thead>
<tr>
<th>PAY₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
</tr>
<tr>
<td>Mech. Eng.</td>
</tr>
<tr>
<td>Programmer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PAY₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
</tr>
<tr>
<td>Elect. Eng.</td>
</tr>
<tr>
<td>Syst. Anal.</td>
</tr>
</tbody>
</table>

Examples

- **Fragmentation of relation PROJ**
  - Applications:
    - Find the name and budget of projects given their no.
      - Issued at three sites
    - Access project information according to budget
      - one site accesses ≤200000 other accesses >200000
  - Simple predicates
    - For application (1)
      - \( p₁ \) : LOC = “Montreal”
      - \( p₂ \) : LOC = “New York”
      - \( p₃ \) : LOC = “Paris”
    - For application (2)
      - \( p₄ \) : BUDGET ≤ 200000
      - \( p₅ \) : BUDGET > 200000

<table>
<thead>
<tr>
<th>PROJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNO</td>
</tr>
<tr>
<td>P1</td>
</tr>
<tr>
<td>P2</td>
</tr>
<tr>
<td>P3</td>
</tr>
<tr>
<td>P4</td>
</tr>
</tbody>
</table>
Examples

- Fragmentation of relation PROJ continued
  - Minterm fragments left after elimination
    - $m_1$: ($\text{LOC} = \text{"Montreal"} \land (\text{BUDGET} \leq 200000)$)
    - $m_2$: ($\text{LOC} = \text{"Montreal"} \land (\text{BUDGET} > 200000)$)
    - $m_3$: ($\text{LOC} = \text{"New York"} \land (\text{BUDGET} \leq 200000)$)
    - $m_4$: ($\text{LOC} = \text{"New York"} \land (\text{BUDGET} > 200000)$)
    - $m_5$: ($\text{LOC} = \text{"Paris"} \land (\text{BUDGET} \leq 200000)$)
    - $m_6$: ($\text{LOC} = \text{"Paris"} \land (\text{BUDGET} > 200000)$)

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{PROJ}_1 & \textbf{PROJ}_2 \\
\hline
\textbf{PNO} & \textbf{PNAME} & \textbf{BUDGET} & \textbf{LOC} & \textbf{PNO} & \textbf{PNAME} & \textbf{BUDGET} & \textbf{LOC} \\
\hline
P1 & Instrumentation & 150000 & Montreal & P2 & Database Develop. & 135000 & New York \\
\hline
\textbf{PROJ}_4 & \textbf{PROJ}_5 \\
\hline
\textbf{PNO} & \textbf{PNAME} & \textbf{BUDGET} & \textbf{LOC} & \textbf{PNO} & \textbf{PNAME} & \textbf{BUDGET} & \textbf{LOC} \\
\hline
P3 & CAD/CAM & 250000 & New York & P4 & Maintenance & 310000 & Paris \\
\hline
\end{tabular}
Correctness

- **Completeness**
  - Since $Pr'$ is complete and minimal, the selection predicates are complete

- **Reconstruction**
  - If relation $R$ is fragmented into $F_R = \{R_1, R_2, ..., R_k\}$
    \[ R = \bigcup_{R_i \in F_R} R_i \]

- **Disjointness**
  - Minterm predicates that form the basis of fragmentation should be mutually exclusive.

---

Derived Horizontal Fragmentation

- Defined on a member relation of a link according to a selection operation specified on its owner.
  - Each link is an equijoin.
  - Equijoin can be implemented by means of semijoins.

![Diagram of equijoin with relation PAY, TITLE, SAL, EMP, L1, PROJ, ENQ, NAME, TITLE, PNO, PHNAME, BUDGET, LOC, ASG, ENQ, PNO, RESP, DUR, L2, L3]
Definition

Given a link $L$ where $owner(L) = S$ and $member(L) = R$, the derived horizontal fragments of $R$ are defined as

$$R_i = R \bowtie S_i, \quad 1 \leq i \leq w$$

where $w$ is the maximum number of fragments that will be defined on $R$ and

$$S_i = \sigma_{F_i}(S)$$

where $F_i$ is the formula according to which the primary horizontal fragment $S_i$ is defined.

Example

Given link $L_1$ where $owner(L_1) = PAY$ and $member(L_1) = EMP$

$EMP_1 = EMP \bowtie PAY_1$

$EMP_2 = EMP \bowtie PAY_2$

where

$$PAY_1 = \sigma_{SAL=30000}(PAY)$$
$$PAY_2 = \sigma_{SAL=30000}(PAY)$$

<table>
<thead>
<tr>
<th>EMP_1</th>
<th>EMP_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENO</td>
<td>ENAME</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Types of Fragmentation

- Horizontal Fragmentation (HF)
  - Primary Horizontal Fragmentation (PHF)
  - Derived Horizontal Fragmentation (DHF)
- Vertical Fragmentation (VF)
- Hybrid Fragmentation (HF)

Hybrid Fragmentation

Diagram showing the hybrid fragmentation process.