Vertical Fragmentation

Vertical Fragmentation

- More difficult than horizontal, because more alternatives exist.
- Two approaches :
  - Grouping
    - attributes to fragments
  - Splitting
    - relation to fragments

Vertical Fragmentation

- Overlapping fragments
  - → grouping
- Non-overlapping fragments
  - → splitting
- We do not consider the replicated key attributes to be overlapping.
- Advantage:
  - Easier to enforce functional dependencies (for integrity checking etc.)

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 $Q = \{q_1, q_2, \ldots, q_q\}$ that will run on the relation $R[A_1, A_2, \ldots, A_n]$,

$$use(q_i, A_j) = \begin{cases} 1 \text{ if attribute } A_j \text{ is referenced by query } q_i \\ 0 \text{ otherwise} \end{cases}$$

$use(q_i, \bullet)$ can be defined accordingly

VF – Definition of *use*(q_i,A_j)

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= PNO, A_2= PNAME, A_3= BUDGET, A_4= LOC

$$\begin{array}{c c} & \begin{array}{cccc} A_1 & A_2 & A_3 & A_4 \end{array} \\ \begin{array}{c} q_1 \\ q_2 \\ q_3 \\ q_4 \end{array} & \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

VF – Affinity Measure *aff*(A_i,A_j)

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

$$aff(A_i, A_j) = \sum_{\text{all queries that access } A_i \text{ and } A_j} (\text{query access})$$

$$\text{query access} = \sum_{\text{all sites}} \text{access frequency of a query} * \frac{\text{access}}{\text{execution}}$$

VF – Calculation of *aff*(A_i, A_j)

$$\begin{array}{c|c} & \begin{array}{cccc} A_1 & A_2 & A_3 & A_4 \end{array} \\ \begin{array}{c} q_1 \\ q_2 \\ q_3 \\ q_4 \end{array} & \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

$$\begin{array}{c|c} & \begin{array}{ccc} S_1 & S_2 & S_3 \end{array} \\ \begin{array}{c} q_1 \\ q_2 \\ q_3 \\ q_4 \end{array} & \begin{bmatrix} 15 & 20 & 10 \\ 5 & 0 & 0 \\ 25 & 25 & 25 \\ 3 & 0 & 0 \end{bmatrix} \end{array}$$

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

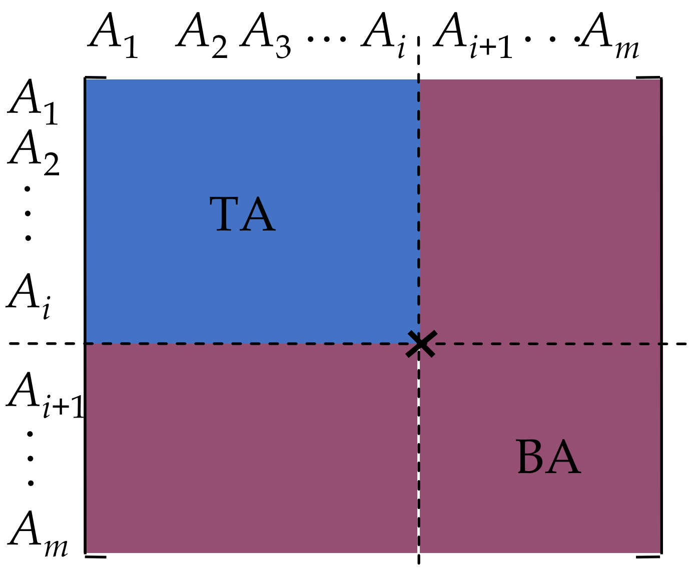
aff(A_1, A_1)= 15*1 + 20*1+10*1 =45
aff(A_1, A_2)= 0
aff(A_1, A_3)= 15*1 + 20*1+10*1=45
aff(A_1, A_4)= 0

aff(A_2, A_2)= 5+75 =80
aff(A_2, A_3)= 5
aff(A_2, A_4)= 25*1 + 25*1+25*1=75

aff(A_3, A_4)= 3
aff(A_4, A_4)= 75+3=78

VF – Algorithm

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



VF – Clustering Algorithm

- 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.
- 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 is maximized.

$$AM = \sum_{i} \sum_{j} \text{(affinity of } A_i \text{ and } A_j \text{ with their neighbors)}$$

VF – ALgorithm

Define

TQ = set of applications that access only TA

BQ = set of applications that access only BA

OQ = set of applications that access both TA and BA

and

CTQ = total number of accesses to attributes by applications that access only TA

CBQ = total number of accesses to attributes by applications that access only BA

COQ = total number of accesses to attributes by applications that access both TA and BA

Then find the point along the diagonal that maximizes

$$CTQ * CBQ - COQ^2$$

VF – Algorithm

Two problems :

- 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 $O(m^2)$
- 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 $O(2^m)$

Bond Energy Algorithm

Input: The *AA* matrix

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

❶ *Initialization*: Place and fix one of the columns of *AA* in *CA*.

❷ *Iteration*: Place the remaining *n-i* columns in the remaining *i*+1 positions in the *CA* matrix. For each column, choose the placement that makes the most contribution to the global affinity measure.

❸ *Row order*: Order the rows according to the column ordering.

Bond Energy Algorithm

"Best" placement? Define contribution of a placement:

$$cont(A_i, A_k, A_j) = 2bond(A_i, A_k)+2bond(A_k, A_l) -2bond(A_i, A_j)$$

where

$$bond(A_x,A_y) = \sum_{z=1}^{n} aff(A_z,A_x)aff(A_z,A_y)$$

BEA – Example

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

$$AA = \begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \\ A_4 \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & A_3 & A_4 \end{array} \\ \begin{bmatrix} 45 & 0 & 5 & 0 \\ 0 & 80 & 5 & 75 \\ 45 & 5 & 53 & 3 \\ 0 & 75 & 3 & 78 \end{bmatrix} \end{array} \qquad CA = \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & & \end{array} \\ \begin{bmatrix} 45 & 0 & & \\ 0 & 80 & & \\ 45 & 5 & & \\ 0 & 75 & & \end{bmatrix} \end{array}$$

$2bond(A_1, A_3) = 45 * 5 + 0 * 5 + 45 * 53 + 0 * 3 = 4410$

$2bond(A_2, A_3)$

$= 0 * 5 + 80 * 5 + 5 * 53 + 75 * 3 = 890$

BEA – Example

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

$$AA = \begin{array}{c|cccc|} & A_1 & A_2 & A_3 & A_4 \\ A_1 & 45 & 0 & 5 & 0 \\ A_2 & 0 & 80 & 5 & 75 \\ A_3 & 45 & 5 & 53 & 3 \\ A_4 & 0 & 75 & 3 & 78 \end{array} \qquad CA = \begin{array}{|cccc|} A_1 & A_2 & & \\ 45 & 0 & & \\ 0 & 80 & & \\ 45 & 5 & & \\ 0 & 75 & & \end{array}$$

Ordering (0-3-1) :

$cont(A_0,A_3,A_1) = 2bond(A_0, A_3)+2bond(A_3, A_1)–2bond(A_0, A_1)$

$= 2^* 0 + 2^* 4410 – 2^*0 = 8820$

Ordering (1-3-2) :

$cont(A_1,A_3,A_2) = 2bond(A_1, A_3)+2bond(A_3, A_2)–2bond(A_1,A_2)$

$= 2^* 4410 + 2^* 890 – 2^*225 = 10150$

Ordering (2-3-4) :

$cont (A_2,A_3,A_4) = 1780$

BEA – Example

- Therefore, the CA matrix has the form

$$\begin{array}{ccc} A_1 & A_3 & A_2 \end{array}$$

$$\begin{bmatrix} 45 & 5 & 0 \\ 0 & 5 & 80 \\ 45 & 53 & 5 \\ 0 & 3 & 75 \end{bmatrix}$$

- When A_4 is placed, the final form of the CA matrix (after row organization) is

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

BEA – Example

- After adding A3, CA matrix has the form

$$AA = \begin{array}{cc} & \begin{array}{cccc} A_1 & A_2 & A_3 & A_4 \end{array} \\ \begin{array}{c} A_1 \\ A_2 \\ A_3 \\ A_4 \end{array} & \begin{bmatrix} 45 & 0 & 5 & 0 \\ 0 & 80 & 5 & 75 \\ 45 & 5 & 53 & 3 \\ 0 & 75 & 3 & 78 \end{bmatrix} \end{array} \qquad CA = \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & & \end{array} \\ \begin{bmatrix} 45 & 0 & & \\ 0 & 80 & & \\ 45 & 5 & & \\ 0 & 75 & & \end{bmatrix} \end{array}$$

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

Now arranging the rows:

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

BEA – Example

- The final step is to divide the set of attributes into two sets such that the queries that access both sets are minimized.

- The appropriate split point on the diagonal must be figured out.
- This is an optimization problem. The optimal points on the diagonal must be determined.

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

VF – Correctness

A relation R, defined over attribute set A and key K, generates the vertical partitioning $F_R = \{R_1, R_2, \ldots, R_r\}$.

- Completeness
  - The following should be true for A:

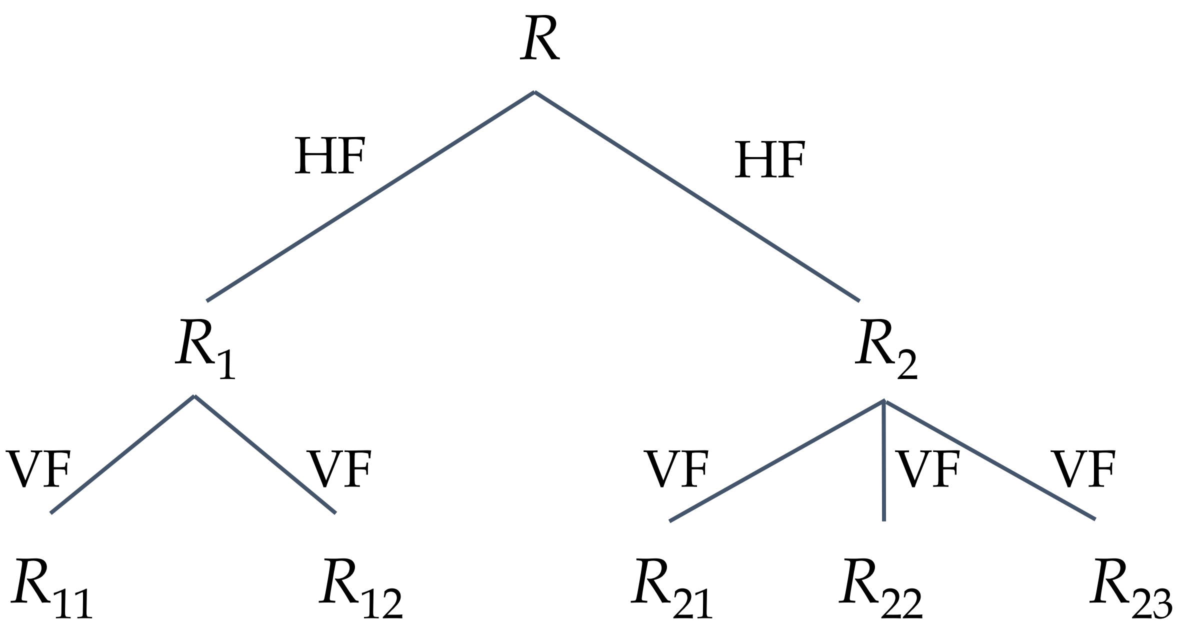
$$A = \bigcup A_{R_i}$$

- Reconstruction
  - Reconstruction can be achieved by

$$R = \bowtie_K R_i,\ \forall R_i \in F_R$$

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

Hybrid Fragmentation



Fragment Allocation

- Problem Statement

  Given

  $F = \{F_1, F_2, \ldots, F_n\}$ fragments

  $S = \{S_1, S_2, \ldots, S_m\}$ network sites

  $Q = \{q_1, q_2, \ldots, q_q\}$ applications

  Find the "optimal" distribution of F to S.

- Optimality
  - Minimal cost
    - Communication + storage + processing (read & update)
    - Cost in terms of time (usually)
  - Performance

    Response time and/or throughput
  - Constraints
    - Per site constraints (storage & processing)

Reference

M. Tamer Ozsu and Patrick Valduriez, "Principles of Distributed Database Systems,“ Prentice Hall