Phân mảnh dọc

- Đã được nghiên cứu trong ngữ cảnh tập trung
  - Phương pháp thiết kế
  - Phân cụm vật lý
- Khó hiểu hơn phân mảnh ngang, vì có nhiều lựa chọn phân mảnh hơn.

  Hai cách tiếp cận:
  - Phân cụm thuộc tính
    - Nhóm các thuộc tính được phân thành các mảnh
  - Phân tách
    - Tách quan hệ thành các mảnh

Phân mảnh dọc

- Các mảnh chồng chéo
  - Trong ngữ cảnh phân cụm thuộc tính
- Các mảnh không chồng chéo
  - Phân tách

Không coi các thuộc tính khóa nhân bản là chồng chéo.

Ưu điểm:

Dễ dàng thực thi các phụ thuộc hàm hơn (để kiểm tra tính toàn vẹn,…)

VF – Thông tin yêu cầu

- Thông tin về ứng dụng
  - Độ liên quan thuộc tính (Attribute affinities)
    - Là độ đo cho biết mức độ liên quan chặt chẽ của các thuộc tính.
    - Khái niệm này là do trong phân mảnh dọc, mỗi mảnh sẽ bao gồm các thuộc tính thường được truy nhập chung với nhau.
  - Giá trị sử dụng thuộc tính
    - Cho một tập truy vấn $Q = \{q_1, q_2, \ldots, q_q\}$ sẽ chạy trên quan hệ $R[A_1, A_2, \ldots, A_n]$,

$$use(q_i, A_j) = \begin{cases} 1 \text{ nếu thuộc tính } A_j \text{ được tham chiếu bởi truy vấn } q_i \\ 0 \text{ ngược lại} \end{cases}$$

$use(q_i, \bullet)$ có thể được định nghĩa tương ứng

VF – Định nghĩa $use(q_i, A_j)$

Xét 4 truy vấn sau cho quan hệ PROJ

Tìm tên và ngân sách của tất cả các dự án

q_1: **SELECT** PNAME,BUDGET
FROM PROJ

Tìm tên và ngân sách của dự án khi biết mã dự án

q_2: **SELECT** PNAME,BUDGET
FROM PROJ
WHERE PNO=Value

Tìm mã các dự án khi biết địa điểm

q_3: **SELECT** PNO
FROM PROJ
WHERE LOC=Value

Tìm tổng ngân sách của tất cả các dự án

q_4: **SELECT** **SUM**(BUDGET)
FROM PROJ

VF – Định nghĩa *use*(q_i,A_j)

Xét 4 truy vấn sau cho quan hệ PROJ

q_1: **SELECT** PNAME,BUDGET
FROM PROJ

q_2: **SELECT** PNAME,BUDGET
FROM PROJ
WHERE PNO=Value

q_3: **SELECT** PNO
FROM PROJ
WHERE LOC=Value

q_4: **SELECT** **SUM**(BUDGET)
FROM PROJ

	PNO	PNAME	BUDGET	LOC
q_1	0	1	1	0
q_2	1	1	1	0
q_3	1	0	0	1
q_4	0	0	1	0

55

VF – Độ liên quan *aff*(A_i,A_j)

Độ liên quan thuộc tính giữa hai thuộc tính A_i và A_j của một quan hệ $R[A_1, A_2, \ldots, A_n]$ đối với tập truy vấn $Q = (q_1, q_2, \ldots, q_q)$ được định nghĩa như sau:

$$aff(A_i, A_j) = \sum_{k|use(q_k,A_i)=1 \wedge use(q_k,A_j)=1} \sum_{\forall S_l} ref_l(q_k) acc_l(q_k)$$

56

VF – Độ liên quan *aff*(A_i,A_j)

Giải nghĩa tương đương:

$$aff(A_i, A_j) = \sum_{\text{tất cả các truy vấn truy nhập } A_i \text{ và } A_j} (\text{truy nhập truy vấn})$$

$$\text{Truy nhập truy vấn} = \sum_{\text{tất cả các trạm}} \text{Tần số truy nhập của một truy vấn} * \frac{\text{Truy nhập}}{\text{Thực thi}}$$

57

VF – Tính *aff*(A_i, A_j) – Ví dụ

Giả thiết mỗi truy vấn trong ví dụ trước truy nhập vào các thuộc tính một lần trong mỗi lần thực thi.

Đồng thời, giả thiết tần số truy nhập:

	S_1	S_2	S_3
q_1	15	20	10
q_2	5	0	0
q_3	25	25	25
q_4	3	0	0

$acc_1(q_1) = 15$ $\quad$ $acc_1(q_2) = 5$
$acc_1(q_3) = 25$ $\quad$ $acc_1(q_4) = 3$
$acc_2(q_1) = 20$ $\quad$ $acc_2(q_2) = 0$
$acc_2(q_3) = 25$ $\quad$ $acc_3(q_4) = 0$
$acc_3(q_1) = 10$ $\quad$ $acc_3(q_2) = 0$
$acc_3(q_3) = 25$ $\quad$ $acc_2(q_4) = 0$

58

VF – Tính *aff*(A_i, A_j) – Ví dụ

Mối liên hệ giữa các thuộc tính PNO và BUDGET được đo như sau:

$$aff(\text{PNO}, \text{BUDGET}) = \sum_{k=1}^{1}\sum_{l=1}^{3} acc_l(q_k) = acc_1(q_1) + acc_2(q_1) + acc_3(q_1) = 45$$

15*1 + 20*1 + 10*1

và ma trận liên quan thuộc tính *AA* là:

	PNO	PNAME	BUDGET	LOC
PNO	–	0	45	0
PNAME	0	–	5	75
BUDGET	45	5	–	3
LOC	0	75	3	–

VF – Thuật toán phân cụm

- Lấy ma trận liên quan thuộc tính *AA* và sắp xếp lại thứ tự thuộc tính để tạo thành các cụm trong đó các thuộc tính tại mỗi cụm thể hiện mối liên quan cao với nhau.
- Thuật toán Bond Energy (BEA) đã được sử dụng để phân cụm các thực thể. BEA tìm một thứ tự các thực thể (trong trường hợp ở đây là các thuộc tính) sao cho độ liên quan toàn cục là lớn nhất.

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

➔ Nghĩa là:

$$AM = \sum_{i}\sum_{j} \text{(liên quan } A_i \text{ và } A_j \text{ với các hàng xóm của nó)}$$

Thuật toán Bond Energy

Đầu vào: Ma trận *AA*

Đầu ra: Ma trận liên quan phân cụm *CA* là một biến thể của *AA*.

1. *Khởi tạo*: Đặt và cố định một trong các cột của *AA* trong *CA*.
2. *Vòng lặp*: Đặt các cột *n-i* còn lại vào các vị trí *i*+1 còn lại trong ma trận *CA*. Với mỗi cột, hãy chọn vị trí có đóng góp nhiều nhất cho độ liên quan toàn cục.
3. *Thứ tự hàng*: Sắp xếp các hàng theo thứ tự cột.

Thuật toán Bond Energy

Vị trí "tốt nhất" là gì? Xác định sự đóng góp của một vị trí:

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

trong đó,

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

BEA – Ví dụ

Xét ma trận *AA* sau đây và ma trận *CA* tương ứng trong đó đặt PNO và PNAME. Đặt BUDGET:

	PNO	PNAME	BUDGET	LOC
PNO	–	0	45	0
PNAME	0	–	5	75
BUDGET	45	5	–	3
LOC	0	75	3	–

	PNO	PNAME
PNO	45	0
PNAME	0	80
BUDGET	45	5
LOC	0	75

Trình tự (0-3-1) :

cont(A_0,BUDGET,PNO) = 2*bond*(A_0, BUDGET)+2*bond*(BUDGET, PNO) –2*bond*(A_0 , PNO)
= 8820

Trình tự (1-3-2) :

cont(PNO,BUDGET,PNAME) = 10150

Trình tự (2-3-4) :

cont (PNAME,BUDGET,LOC) = 1780

BEA – Ví dụ

- Do đó, ma trận *CA* có dạng

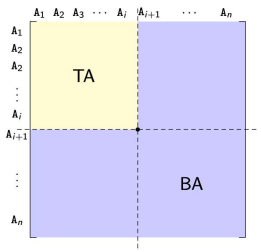
	PNO	BUDGET	PNAME
PNO	45	45	0
PNAME	0	5	80
BUDGET	45	53	5
LOC	0	3	75

- Khi LOC được đặt, ma trận *CA* cuối cùng là:

	PNO	BUDGET	PNAME	LOC
PNO	45	45	0	0
BUDGET	45	53	5	3
PNAME	0	5	80	75
LOC	0	3	75	78

VF – Thuật toán phân mảnh dọc

Bằng cách nào có thể phân chia một tập $\{A_1, A_2, \ldots, A_n\}$ thành hai (hoặc nhiều hơn) nhóm (cụm) thuộc tính $\{A_1, A_2, \ldots, A_i\}$ và $\{A_i, \ldots, A_n\}$ sao cho không có (hoặc có ít nhất) ứng dụng nào có thể truy nhập cả hai (hoặc nhiều hơn một) tập.



VF – Thuật toán phân mảnh dọc

Định nghĩa

TQ = tập các ứng dụng chỉ truy nhập *TA*
BQ = tập các ứng dụng chỉ truy nhập *BA*
OQ = tập các ứng dụng truy nhập cả *TA* và *BA*

và

CTQ = tổng số truy nhập vào thuộc tính của các ứng dụng chỉ truy nhập *TA*
CBQ = tổng số truy nhập vào thuộc tính của các ứng dụng chỉ truy nhập *BA*
COQ = tổng số truy nhập vào thuộc tính của các ứng dụng truy nhập cả *TA* và *BA*

Sau đó tìm điểm dọc theo đường chéo tối đa hóa

$$CTQ * CBQ - COQ^2$$

VF – Thuật toán phân mảnh dọc

Hai vấn đề:

- Cụm hình thành ở giữa ma trận *CA*
  - Chuyển một hàng lên trên, một cột sang trái và áp dụng thuật toán tìm điểm phân vùng tốt nhất.
  - Thực hiện cho tất cả các trường hợp có thể
  - Chi phí $O(m^2)$
- Nhiều hơn 2 cụm
  - Phân vùng *m*-way
  - Thử 1, 2, …, *m*–1 các điểm chia dọc theo đường chéo và cố gắng tìm điểm chia tốt nhất cho mỗi vùng này.
  - Chi phí $O(2^m)$

VF – Tính đúng đắn

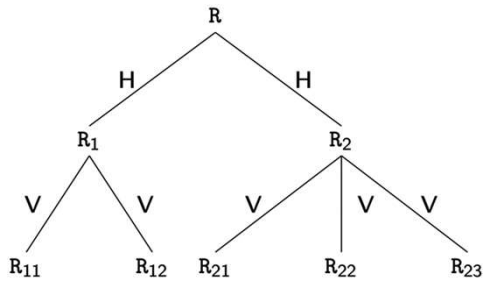
Một quan hệ *R*, được xác định trên tập thuộc tính *A* và khóa *K*, tạo ra phân mảnh dọc $F_R = \{R_1, R_2, \ldots, R_n\}$.

- Tính đầy đủ

  $A = \cup A_{R_i}$
- Tính phục hồi
  - Phục hồi quan hệ có thể được thực hiện bằng cách

    $R = \bowtie_K R_i, \forall R_i \in F_R$
- Tính tách biệt
  - TID (Mã bộ/bản ghi) không bị coi là chồng chéo do chúng được hệ thống duy trì
  - Các khóa nhân bản không bị coi là chồng chéo

HF - Phân mảnh lai



Tính phục hồi của HF