PHÂN MẢNH VÀ CẤP PHÁT DỮ LIỆU TRONG CƠ SỞ DỮ LIỆU HƯỚNG ĐỐI TƯỢNG PHÂN TÁN

Lê Thu Trang[1*], Lê Bích Liên[2], Nguyễn Tuấn Anh[3]
[1]Trường Đại học Công nghệ Thông tin và Truyền thông – ĐH Thái nguyên
[2]Trường Đại học Sư phạm – ĐH Thái Nguyên, [3]Đại Học Thái Nguyên

TÓM TẮT

Trong thiết kế phân tán, phân mảnh và cấp phát là một vấn đề quan trọng. Cơ sở dữ liệu hướng đối tượng phân tán khi thiết kế còn phát sinh thêm một số vấn đề phức tạp khác. Các vấn đề phức tạp này bắt nguồn từ các đặc điểm của mô hình hướng đối tượng, đó là tính đóng gói, kế thừa, sự phân cấp lớp, sự có mặt của các thuộc tính và phương thức phức hợp. Bài bào này trình bày về thuật toán cấp phát lớp trong cơ sở dữ liệu hướng đối tượng phân tán.

Từ khóa: *phân tán, cơ sở dữ liệu hướng đối tượng phân tán, phân mảnh, cấp phát dữ liệu*

ĐẶT VẤN ĐỀ

Sự phát triển của các ứng dụng dữ liệu chuyên sâu đã vượt qua khả năng xửa lý của hệ thống quản trị cơ sở dữ liệu quan hệ. Có thể liệt kê một số lĩnh vực chuyên môn sâu của cơ sở dữ liệu như Multimedia, CAD/CAM và các hệ thống tài chính phức tạp. Các hạn chế của cơ sở dữ liệu quan hệ đã thúc đẩy sự phát triển của hệ thống cơ sở dữ liệu hướng đối tượng (OODBS – Object Oriented Database System). OODBS được xây dựng dựa trên mô hình cơ sở dữ liệu hướng đối tượng (OODB), mỗi đối tượng được lưu trữ không chỉ dữ liệu mà còn thao tác trên chúng. Các nghiên cứu cho thấy OODB sẽ tiếp tục phát triển và cung cấp các khả năng nổi trội trong việc xử lý dữ liệu phức tạp.

Để đáp ứng nhu cầu của doanh nghiệp lớn với sự phân bố nhiều trạm ở các vị trí địa lý khác nhau, OODB được phát triển trên môi trường mạng tạo thành mô hình cơ sở dữ liệu hướng đối tượng phân tán (DOODB – Distributed Object Oriented Database System).

Cơ sở dữ liệu phân tán cần có phương án thiết kế tốt nhằm cải thiện hiệu năng của hệ thống. Hai vấn đề trong thiết kế trong cơ sở dữ liệu phân tán là phân mảnh (fragment) và cấp phát (allocation). Với các đặc điểm của OODB như đóng gói, kế thừa, phân cấp thì các kĩ thuật phân mảnh và cấp phát sẽ gặp khó khăn hơn nhiều. Bài toán cấp phát dữ liệu đã được chứng minh là bài toàn NP đầy đủ, trong nghiên cứu này tôi đề cập tới một thuật toán cấp phát lớp trong OODB.

CÁC NGHIÊN CỨU LIÊN QUAN

Phân mảnh được chia làm 3 loại: phân mảnh ngang, phân mảnh dọc và phân mảnh hỗn hợp. Phân mảnh dọc nhằm chia một quan hệ thành tập các quan hệ nhỏ hơn, phân mảnh ngang nhằm chia các bộ dữ liệu thành các quan hệ, phân mảnh hỗn hợp là kết hợp cả phân mảnh ngang và phân mảnh dọc. Phân mảnh trong cơ sở dữ liệu quan hệ đã được đề cập trong rất nhiều nghiên cứu, và cũng có nhiều công trình liên quan đến cấp phát trong cơ sở dữ liệu [4], [8], [9].

Trong OODB, mục tiêu của phân mảnh dọc là chia các lớp thành các lớp nhỏ hơn, còn phân mảnh ngang là chia bộ các đối tượng của lớp thành các mảnh. Dữ liệu trong OODB bao gồm các đối tượng được đóng gói, mỗi đối tượng bao gồm các thuộc tính và các phương thức. Các đối tượng được tạo ra từ các lớp. Một lớp trong quan hệ thứ tự được biểu diễn bởi C = (K, A, M, I) trong đó K là tập các định danh, A là tập các thuộc tính, M là tập các phương thức, I là tập các đối tượng được định nghĩa bởi A và M. Phân mảnh dọc của C là C_v ={K, A', M', I} trong đó A'$\subset A$, M'$\subset M$. Phân mảnh ngang của C là C_h={K, A, M, I'} trong đó $I' \subset I$. Một số nghiên cứu

* *Tel: 0983 754948, Email: trangtip@gmail.com*

đã thực hiện với việc phân mảnh trong OODB [3], [7], [11], Cấp phát là định vị các mảnh f_i vào các trạm s_j của mạng truyền thông. Các nghiên cứu tập trung tìm ra các thuật toán cấp phát nhằm giảm chi phí. Chỉ có một số rất ít các nghiên cứu đề cập đến vấn đề cấp phát các thuộc tính và các phương thức. Trong [6] K. Barker and S. Bhar đã đưa ra các khái niệm cơ bản để thiết lập cho bài toán cấp phát lớp, trong đó các tác giả cũng đề nghị một hướng tiếp cận đồ thị để giải quyết bài toán. Trong [1] và [10] đề cập đến thuật toán di truyền để chọn ra phương án cấp phát gần tối ưu. Các giải thuật heuristic cho bài toán cấp phát lớp trong OODB được đề cập trong [2], [5]

BÀI TOÁN CẤP PHÁT LỚP

Mô tả bài toán

Trong giai đoạn cấp phát, các mảnh lớp được định vị vào các trạm trong mạng liên kết, bài toán đặt ra là tìm một phương án cấp phát tối ưu. Phương án này với tiêu chí là chi phí cấp phát là nhỏ nhất. Chi phí cấp phát là tổng các chi phí thành phần: chi phí lưu trữ dữ liệu, chi phí xử lý vấn tin, chi phí để truyền dữ liệu giữa các trạm.

Các thông tin cần thiết để thiết lập công thức tính chi phí cấp phát bao gồm: thông tin về dữ liệu, các truy vấn, thông tin mạng gồm các trạm, khả năng lưu trữ của từng trạm và hiệu năng hoạt động của mỗi trạm

Mô hình được thiết lập như sau:

- Mạng kết nối gồm m trạm $S= \{s_1, s_2, \dots s_m\}$

- Tập các mảnh $F= \{f_1, f_2, \dots f_n\}$, giả thiết số lượng các mảnh nhiều hơn số trạm rất nhiều.

- Tập các truy vấn $Q = \{q_1, q_2, \dots q_h\}$

Mục tiêu của bài toán là xác định ánh xạ từ F vào S sao cho tổng chi phí là nhỏ nhất.

Thông tin về dữ liệu

Để đơn giản, tất cả các thuộc tính và phương thức của tất cả các lớp được đánh số và chỉ mang một chỉ số. Ma trận sử dụng thuộc tnhs của các phương thức MAU (Method Attribute Usage) biểu diễn việc sử dụng các thuộc tính bởi các phương thức. Trong ma trận MAU, các hàng thể hiện các phương thức, các cột thể hiện các thuộc tính, giá trị 1 chỉ ra phương thức truy cập thuộc tính tương ứng, ngược lại là 0. Bảng 1 là một ví dụ về MAU.

Bảng 1. *Ma trận MAU*

	a_1	a_2	a_3	a_4	a_5
m_1	1	0	1	0	0
m_2	0	1	0	0	0
m_3	0	0	0	1	1

Ma trận sử dụng phương thức của các phương thức MMU (Method Method Usage) biểu diễn việc sử dụng các phương thức bởi các phương thức khác. Trong ma trận MMU, các hàng và các cột bao gồm các phương thức, giá trị 1 chỉ ra phương thức truy cập các thuộc tính tương ứng, ngược lại là 0. Bảng 2 là một ví dụ về MMU.

Bảng 2. *Ma trận MMU*

	m_1	m_2	m_3
m_1	1	0	1
m_2	0	1	0
m_3	0	0	0

Kích thước các mảnh lớp f_i:

$$\text{Size}(f_i) = \text{card}(f_i) * \text{length}(f_i)$$

Trong đó $\text{card}(f_i)$ là số phần tử của các mảnh f_i, $\text{length}(f_i)$ là số byte của các thuộc tính trong mảnh f_i. Bảng 3 là ví dụ về kích thước mảnh.

Bảng 3. *Mảng Size (F)*

Fragment	f_1	f_2	f_3	f_4
Size	300	550	400	100

Thông tin về truy vấn của người dùng

Khi đóng gói các đối tượng, các ứng dụng chỉ truy cập được các mảnh đối tượng thông qua các phương thức, $acc(q_i, m_i)$ là tần suất truy cập vào phương thức m_i của câu truy vấn q_i, Ví dụ được thể hiện qua bảng 4.

Bảng 4. *Ma trận QMF*

Query/Method	m_1	m_2	m_3
q_1	3	5	1
q_2	4	0	2
q_3	4	1	0

Ma trận QSF biểu diễn tần suất thực hiện các truy vấn tại các trạm.

Bảng 5. *Ma trận QSF*

Query/Site	s_1	s_2	s_3
q_1	2	1	0
q_2	1	0	3
q_3	0	3	1

Thông tin các trạm

Mỗi trạm có dung lượng lưu trữ như sau:

Bảng 6. *Dung lượng trên các trạm*

Site	s_1	s_2	s_3
Dung lượng	1000	1700	600

Điều kiện đặt ra là tổng kích thước các mảnh lưu trữ trên mỗi trạm không được vượt quá dung lượng của trạm đó

Thông tin về mạng

Ma trận chi phí giao tiếp giữa các trạm là n*n, mỗi phần tử là chi phí giao tiếp giữa 2 trạm, Bảng 7 là ví dụ về ma trận chi phí giao tiếp

Bảng 7. *Ma trận liên kết trạm SSC_0*

Site	s_1	s_2	s_3
s_1	0	5	10
s_2	5	0	3
s_3	10	3	0

Áp dụng thuật toán tìm đường đi nhỏ nhất giữa 2 đỉnh bất kì để đưa về ma trận chi phí giao tiếp SSC, ví dụ ma trận trong bảng 7 đưa về kết quả ma trận SSC trong bảng 8.

Bảng 8. *Ma trận SSC*

Site	S_1	S_2	S_3
S_1	0	5	8
S_2	5	0	3
S_3	8	3	0

Xây dựng hàm chi phí cấp phát

Sử dụng một số hàm

- Par(m_j) trả về tập các tham số được phân tích ra khi phương thức m_j được gọi.
- res(m_j) là dữ liệu trả về khi phương thức m_j được gọi.
- MIS(q_k) là tập các phương thức mà truy vấn q_k tham chiếu tới.
- MMR(m_j) là tập các phương thức được tham chiếu bởi m_j.

Tính chi phí tham chiếu

Chi phí giao tiếp về mặt dữ liệu giữa các mảnh IFDC (Interfragment data communication) được thiết lập như sau, trước hết là giữa phương thức m_j với thuộc tính a_k

$$IFDC(m_l, a_k) = \sum_{i|q_i \in Q} acc(q_i, m_l) * MAU(m_l, a_k) * size(a_k)$$

Trong đó acc(q_i, m_l) là tần suất truy cập phương thức m_l của truy vấn q_i, size(a_k) là kích thước của thuộc tính a_k. Như vậy, tính cho toàn bộ các thuộc tính a_k thuộc mảnh f_j nhận được chi phí giao dịch giữa mảnh và phương thức IFC (Interfragment communication).

$$IFC(m_l, f_j) = \sum_{k|a_i \in f_j} IFDC(m_l, a_k)$$

$$IFDC(m_l, m_k) = \sum_{i|q_i \in Q} (I_1 + I_2) * acc(q_i, m_l) * MMU(m_l, m_k)$$

Trong đó, I_l là tổng tất cả kích thước các tham số trong phương thức m_k, I_2 là tổng tất cả kích thước các kết quả trả về của phương thức m_k

$$I_1 = \sum_{i|p_i \in par(m_k)} size(p_i) \qquad I_2 = size(res(m_k))$$

Thiết lập chi phí tham chiếu giữa mảnh f_i và f_j được xác định bởi:

$$FFR(f_i, f_j) = \sum_{l|m_l \in f_j} (IFC(m_l, f_j) + \sum_{k|m_k \in f_j \wedge m_k \in MMR(m_l)} IFDC(m_l, m_k))$$

Trong đó, m_l là các phương thức thuộc f_j và m_k là các phương thức m_l tham chiếu đến.

Như vậy, có thể thiết lập một ma trận tham chiếu giữa các mảnh FFR, ví dụ về một ma trận tham chiếu mảnh như bảng 9.

Bảng 9. *Ma trận FFR*

Fragment	F_1	F_2	F_3	F_4
F_1	2	1	0	5
F_2	4	0	0	1
F_3	3	1	0	0
F_4	0	2	3	0

Tham chiếu giữa mảnh và truy vấn ứng dụng xác định bằng hàm QFR (Query Fragment Reference).

$$QFR(q_k, f_i) = \sum_{l|m_l \in f_i \wedge m_l \in MIS(q_k)}^{h} (A_1 + A_2) * acc(q_k, m_l)$$

Trong đó A_1 là tổng tất cả kích thước các tham số trong phương thức m_l, A_2 là tổng tất cả kích thước các kết quả trả về của phương thức m_l

$$A_1 = \sum_{j|p_j \in par(m_l)} size(p_j) \qquad A_2 = size(res(m_l))$$

Như vậy, có thể thiết lập một ma trận tham chiếu giữa các truy vấn và mảnh QFR, ví dụ về một ma trận QFR như bảng 10.

Bảng 10. *Ma trận QFR*

Query/Fragment	f_1	f_2	f_3	f_4
q_1	2	1	0	5
q_2	0	0	2	1
q_3	3	1	0	0

Xác định phương án cấp phát

Kết hợp ma trận QFR và ma trận QSF để tính tham chiếu giữa mảnh và trạm.

Bảng 11. *Ma trận SFR_0*

S	Q	QSF	f_1	f_2	f_3	f_4
S1	q_1	2	2	1	0	5
	q_2	1	0	0	2	1
	q_3	0	3	1	0	0
S_2	q_1	0	2	1	0	5
	q_2	1	0	0	2	1
	q_3	3	3	1	0	0
S3	q_1	0	2	1	0	5
	q_2	3	0	0	2	1
	q_3	1	3	1	0	0

Lấy tần suất nhân với số truy cập tương ứng ta xây dựng được ma trận SFR biểu diễn sự tham chiếu giữa mảnh và trạm như sau:

Bảng 12. *Ma trận SFR*

Site/Fragment	f_1	f_2	f_3	f_4
s_1	4	2	2	11
s_2	11	3	0	5
s_3	3	1	6	3

Thực hiện nhân ma trận SFR với ma trận FFR, tiếp tục nhân với ma trận SSC, kết quả là ma trận SFC biểu diễn chi phí khi cấp phát các mảnh tại các trạm.

$SFR_1 = SFR^*FFR$

$SFC = SFR_1*SSC$

Với dữ liệu của SFR và FFR trong bảng 12 và bảng 9 có kết quả ma trận SFR_1 như sau:

Hình 13. *Ma trận SFR_1*

Site/Fragment	f_1	f_2	f_3	f_4
s_1	22	28	33	22
s_2	38	21	15	59
s_3	28	15	9	16

Kết hợp nhân dữ liệu SFC trong bảng 8 với SFRI có ma trận SFC như sau:

Bảng 14. *Ma trận SFC*

Site/Fragment	f_1	f_2	f_3	f_4
s_1	414	225	147	423
s_2	194	185	192	158
s_3	290	278	309	353

Tại mỗi cột của ma trận SFC, tìm giá trị lớn nhất, giả sử giá trị lớn nhất là SFC(i,j) thì mảnh f_j được cấp phát tại trạm S_i. Nếu kích thước của mảnh vượt quá dung lượng còn lại của trạm thì tìm trạm tương ứng với phần tử có giá trị lớn tiếp theo trong cột.

Nếu có 2 phần tử trong cột cùng có giá trị lớn nhất thì xây dựng thêm thuật toán tính độ ưu tiên để chọn, chẳng hạn đơn giản chọn trạm có dung lượng lớn hơn.

Với ví dụ về ma trận SFC như trong bảng 14, ta xây dựng phương án cấp phát như sau:

- Trong cột f_1 giá trị lớn nhất tương ứng với hàng s_1,Vì vậy f_1 chọn cấp phát tại trạm s_1
- Tương tự f_2 được chọn cấp phát tại trạm s_2
- f_3 được chọn cấp phát tại trạm s_3 nhưng như vậy sẽ quá dung lượng S_3.

Giá trị lớn thứ 2 trong cột f_3 tương ứng với hàng s_2, vậy f_3 được chọn cấp phát tại trạm s_2

- f_4 được chọn cấp phát tại trạm s_1

Phương án lựa chọn như trong bảng 15.

Bảng 15. *Ma trận SFA*

Fragment	f_1	f_2	f_3	f_4
Site	s_1	s_3	s_2	s_1

THUẬT TOÁN CẤP PHÁT LỚP

Thuật toán chia thành các bước như sau:

Xác định ma trận chi phí giữa các trạm

Đầu vào: Ma trận chi phí SSC_0

Đầu ra: Ma trận chi phí được tối thiểu SSC

Thuật toán: Tìm đường đi ngắn nhất giữa 2 đỉnh trong đồ thị

Xây dựng ma trận tham chiếu giữa các mảnh và truy vấn

Đầu vào: Các ma trận MAU, MMR, QMF, $res(m_k)$, $res(m_k)$, $MIS(q_k)$, $MMR(m_k)$

Đầu ra: Ma trận FFR, QFR

Thuật toán:

```
// Tính IFDC (m1, ax)
For i = 1 to x do //method
 For j = 1 to y do // attribute
 For k = 1 to h do // query
 IFDC1 [i, j] + = QMF [k, i] *MMU [i, j] *size(ax)
// Tính IFC (mi, fj)
For i = 1 to x do //method
For j = 1 to n do //fragment
For (ak ϵ fj)
IFC [i , j] + = IFDC1 [i , j]:
// Tính IFDC (mi , mk):
For i = 1 to x do //method
For j = 1 to h do //query
{
```

```
    For   (p1 € par (mk))
        I1  +  =  size (p1);
      I2  =  size (res (mk))
       IFDC [i , j] + = (I1 + I2) * QMF [k , i]
*MMU [i , j]
           {
// Tính FFR (fi , fj)
For  i  = 1  to  n  do  // fragment
  For j  = 1  to  n  do  // fragment
   {
     FFR [ i , j]  + = IFC [1 , j]
       For  (k|mk € fj & mk € MMR (mi))
          FFR [ i , j]  + = IFDC [1 , k]
       }
    }
// Tính QFR (qk , fi)
For   k  =  1  to  h  do  // query
  For I   =  1  to  n   do   // fragment
    {
      For   (1|mi  €  fi  & mi   €   MIS (qk))
       {
         For   (p3  €   par (m1))
           A1  + = size  (p1);
         A2  =  size (res (mk))
         QFR { i , j}  +=  (A1 + A2)*QMF{ k
,1}
       }
}
```

Xác định chi phí cấp phát

Đầu vào: Các ma trận QSF, FFR, QFR, SFR

Đầu ra: Ma trận SFC

Thuật toán:

/* Xây dựng hàm nhân 2 ma trận: Ma trận A kích thước m*h, ma trận B kích thước h*n, ma trận kết quả C kích thước h*n */

```
Nhân ma trận (A , B)
for  i = 1 to  m do
  for  j  =  1 to n  do
     {
       C[i, j] = 0;
       for k = 1 to n do
          C[i, j] = C[i , j] + A[i , k] * B[k , j] ;
       end  k;
       }
   end  j;
end  i;
return  C:
end NhânMaTrân;
```

/ Gọi hàm nhân ma trận để thực hiện nhân ma trận SFR với FFR, lấy kết quả nhân với ma trận SSC, kết quả cuối cùng là ma trận SFC */*

SFR 1 = Nhân MaTrận (SFR , FFR);

SFC = Nhân MaTrận (SFR1, SSC);

Tìm phương án cấp phát

Đầu vào: SFC, Size(F), Capacity(S)

Đầu ra: Phương án cấp phát

Thuật toán:

Tìm giá trị lớn nhất trong các cột của ma trận SFC, xác định được trạm cấp phát tương ứng cho mảnh. Điều chỉnh nếu vượt qua dung lượng và sử dụng độ ưu tiên nếu có 2 giá trị lớn nhất tại một cột.

```
for  j  =  1  to  n  do
{
   for  i  = 1  to  m  do
     //Tìm i để SFC(i , j) đạt giá trị lớn nhất
       If  (tồn tại nhiều giá trị i)
              Chọn i mà s1 có dung lượng
lớn
       if (Capacity (si) > size (fj))
          {
              Cấp phát fj tại trạm Si;
              Capacity (s1)  = size (fj))
          }
}
```

ĐÁNH GIÁ VÀ KẾT LUẬN

Khi một mảnh f_j được cấp phát tại trạm S_k khi có chi phí cập nhật là cao nhất và như vậy chi phí giao tiếp là bé nhất. Điều này cũng đã được chứng minh trong [7].

Với phương án cấp phát đề xuất mỗi mảnh chỉ được cấp phát tại một trạm, nghĩa là không xảy ra trường hợp phải sao chép một mảnh và đặt tại các trạm khác nhau.

Trong thuật toán các phép tính để xác định độ phức tạp chính là phép tính nhân hai ma trận. Như vậy thuật toán có độ phức tạp là $O(h^*n^2)$, trong đó h là số các trạm và n là số các mảnh, Độ phức tạp này là có thể chấp nhận được.

TÀI LIỆU THAM KHẢO

1. A.Sarhan, "A New Allocation Technigue for Methods and Attributes in Distributed Object-Oriented Databases Using Genitic Algorithms," The Intrernational Arab Journal of Information Technology, vol.6,2009.

2. Bellatreche L, Karlapalem K, and Li Q, "Complex Methods and Class Allocation in Distributed Object Oriented Databases," in International Conference on Object-Oriented Databases, Paris, 1998.

3. Ezeife, C, I and Barker, K., "Vertical Class Fragmentation in a Distributed Object Based System," 1993.

4. H. I. Abdalla, "A New Data Re-Allocation for Distributed Databases," vol, 5 no, Internatinal of Database Theory and Application, 2012.

5. J.Graham, "Efficient Allocation in Distributed Object Oriented Databases," 2003.

6. K. Barker and S. Bhar, "Agraphical approach to allocation class fragments in distributed object-oriented base systems," no, Distributed and Parallel Databases, 2001.

7. Rajan John and Dr. V. Saravanna, "Vertical Partitioning in Object Oriented Databases Using Intelligent Agents," 2008.

8. Salvatore T. March, Sangkyu Rho, " Allocating data and operations to nodes in distributed database design," IEEE Transactions on Knowledge and Data Engineering vol. 7, no, IEEE Transactions on Knowledge and Data Engineering, 1995.

9. S-M, Lee, "Design of allocation algorithm in distributed database," in Procecding of Korean Information and Processing Society, 2003.

10. Soo-Mi Lee, Yan Ha, Hea-Sook Park, *Allocation of Classes in distributed object-oriented databases, *2009.

11. Soon-mi et al., "Attribute Partitioning Algorithm in DOODB," in International Conference on Parallel and Distributed Systems, 1997.

SUMMARY

FRAGMENTATION AND ALLOCATION IN DISTRIBUTED OBJECT -ORIENTED DATABASE

Le Thu Trang[1*], Le Bich Lien[2], Nguyen Tuan Anh[3]

[1]College of Information and Communication Technology – TNU,
[2]College of Education - TNU, [3]Thai Nguyen University

The two most important matters in distributed design are fragmentation and allocation. And the design in even generates more complexities. Such complexities are caused by characteristics of object-oriented model, which are encapsulation, inheritance, class-composite hierarchy, complex attributes and methods. This paper presents the class allocation algorithm in Distributed Object-Oriented Database.

Keywords: *distributed, Object-Oriented Database, fragmentation, allocation, distributed Database.*

Ngày nhận bài:15/10/2014; Ngày phản biện:30/10/2014; Ngày duyệt đăng: 25/11/2014
Phản biện khoa học: *TS. Nguyễn Văn Huân – Trường Đại học Công nghệ Thông tin & Truyền thông - ĐHTN*

* *Tel: 0983 754948, Email: trangtip@gmail.com*