Kỷ yếu Hội nghị Quốc gia lần thứ VIII về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR); Hà Nội, ngày 9-10/7/2015
DOI: 10.15625/vap.2015.000212

TRUY VẤN HƯỚNG ĐỐI TƯỢNG DỰA TRÊN ĐỒ THỊ CHỮ KÝ

Trần Minh Bảo[1], Trương Công Tuấn[2]
[1, 2]Trường Đại học Khoa học, Đại học Huế
tmbaovn@gmail.com, tctuan_it_dept@yahoo.com

TÓM TẮT *- Các kỹ thuật lập chỉ mục luôn luôn là một vấn đề quan trọng trong việc tìm kiếm thông tin hiệu quả từ cơ sở dữ liệu (CSDL). Đối với CSDL hướng đối tượng, việc sử dụng các tập tin chữ ký như là một phương pháp chỉ mục đã được thừa nhận là một tiếp cận phổ biến trong việc xử lý truy vấn trên CSDL hướng đối tượng. Các đối tượng của lớp được mã hóa thành các chữ ký đối tượng bằng cách dùng hàm băm và được lưu trữ trong một tập tin chữ ký. Tuy nhiên, mỗi khi xử lý truy vấn thì toàn bộ tập tin chữ ký phải được quét. Vì vậy phương pháp này đòi hỏi chi phí xử lý cao. Để khắc phục nhược điểm này, chúng tôi đề xuất một mô hình cấu trúc đồ thị được xây dựng trên chữ ký của các đối tượng trong một CSDL hướng đối tượng, gọi là đồ thị chữ ký. Đồng thời đề xuất cách thức xây dựng đồ thị chữ ký và thuật toán truy vấn trên đồ thị chữ ký cùng với phần mô phỏng ứng dụng của phương pháp này.*

Từ khóa *- Truy vấn hướng đối tượng, chữ ký đối tượng, tập tin chữ ký, đồ thị chữ ký.*

I. MỞ ĐẦU

Việc nghiên cứu các kỹ thuật lập chỉ mục luôn luôn là một vấn đề quan trọng trong việc tìm kiếm thông tin hiệu quả từ cơ sở dữ liệu (CSDL). Đối với CSDL hướng đối tượng, truy vấn trực tiếp trên các đối tượng có chi phí thời gian thực hiện lớn. Đã có nhiều kỹ thuật chỉ mục CSDL nhằm xử lý truy vấn trên CSDL hướng đối tượng, trong đó phương pháp tập tin chữ ký được thừa nhận rộng rãi và là một tiếp cận khá hiệu quả trong việc xử lý truy vấn trên CSDL hướng đối tượng. Đối với phương pháp này, các đối tượng của lớp được mã hóa thành các chữ ký đối tượng bằng cách dùng hàm băm và được lưu trữ trong một tập tin chữ ký. Tuy nhiên, việc truy vấn trên tập tin chữ ký lại có nhược điểm là tốn kém chi phí do phải quét toàn bộ tập tin. Một số các phương pháp chỉ mục khác nhằm khắc phục điều này và có thể tìm thấy trong nhiều công trình nghiên cứu [1] [2] [3] [8] [9].

Trong bài báo này, chúng tôi đề xuất việc tổ chức tập tin chữ ký của một lớp trong CSDL hướng đối tượng trong một đồ thị chữ ký và xây dựng các thuật toán để tạo đồ thị chữ ký và truy vấn đối tượng trên đồ thị chữ ký. Việc sử dụng đồ thị chữ ký có không gian tìm kiếm nhỏ hơn để từ đó giảm thời gian truy vấn dữ liệu.

Bài báo này được tổ chức như sau: Phần đầu của bài báo sẽ giới thiệu khái quát về chữ ký đối tượng, tập tin chữ ký, chữ ký truy vấn và cấu trúc đồ thị chữ ký, sau đó bài báo thực hiện việc xây dựng đồ thị chữ ký và thuật toán truy vấn đối tượng trên đồ thị chữ ký, đồng thời xây dựng mô hình ứng dụng, phần cuối của bài báo là kết luận.

II. MỘT SỐ KHÁI NIỆM CƠ SỞ

Phần này chỉ trình bày một số khái niệm cơ sở liên quan đến chữ ký đối tượng và tập tin chữ ký. Chi tiết đầy đủ hơn có thể xem trong [1] [2].

A. Chữ ký đối tượng, tập tin chữ ký

Trong một CSDL hướng đối tượng, mỗi đối tượng được biểu diễn bởi một bộ giá trị của các thuộc tính. Chữ ký của một giá trị thuộc tính là một chuỗi các bit được mã hóa bằng hàm băm. Chữ ký đối tượng được xây dựng bằng phép toán logic OR cho tất cả các chữ ký của các giá trị thuộc tính của đối tượng. Sau đây là một ví dụ về chữ ký của đối tượng:

Ví dụ 2.1. Xét một đối tượng có các giá trị thuộc tính lần lượt là “Dung”, “12345678”, “giao su”. Giả sử chữ ký của các đối tượng này là:

010 000 100 110

100 010 010 100

110 100 011 000

Lúc đó chữ ký của đối tượng là 110 110 111 110, nhận được từ chữ ký các giá trị thuộc tính bằng phép toán logic OR.

Các chữ ký đối tượng của một lớp được lưu trữ trong một tập tin, gọi là tập tin chữ ký đối tượng.

B. Chữ ký truy vấn

Một truy vấn đối tượng sẽ được mã hóa thành một chữ ký truy vấn theo cùng hàm băm đã thực hiện đối với các đối tượng. Khi có một truy vấn cần thực hiện, các chữ ký đối tượng sẽ được quét và những đối tượng không phù hợp sẽ bị loại trừ. Lúc đó chữ ký truy vấn được so sánh đối với mọi chữ ký đối tượng trong tập tin chữ ký. Có ba trường hợp có thể xảy ra:

(1) Đối tượng phù hợp với truy vấn, nghĩa là đối với mọi bit trong chữ ký truy vấn s_q, bit tương ứng trong chữ ký đối tượng s cũng chính là nó, tức là $s_q \wedge s = s_q$, đối tượng thực sự chứa từ truy vấn;

(2) Đối tượng không phù hợp với câu truy vấn, nghĩa là $s_q \wedge s \neq s_q$;

(3) Chữ ký được đối sánh và cho ra một kết quả phù hợp nhưng đối tượng không phù hợp với điều kiện tìm kiếm trong truy vấn. Để loại ra trường hợp này, các đối tượng phải được kiểm tra sau khi các chữ ký đối tượng được đối sánh phù hợp.

Ví dụ 2.2. Ví dụ này minh họa việc truy vấn đối với chữ ký đối tượng ở ví dụ 1.1:

Truy vấn	**Chữ ký truy vấn**	**Kết quả**
Dung	010 000 100 110	thành công
Binh	011 000 100 100	không thành công
Duong	110 100 100 000	thành công nhưng không phù hợp

Nhận xét: Việc so sánh chữ ký truy vấn s_q với chữ ký đối tượng s là loại đối sánh không chính xác. Nghĩa là, chữ ký truy vấn s_q phù hợp chữ ký s nếu mọi bit 1 trong s_q, các bit tương ứng trong s cũng là bit 1. Tuy nhiên, đối với mọi bit 0 trong s_q thì không quan trọng bit tương ứng trong s là 0 hay 1.

III. ĐỒ THỊ CHỮ KÝ VÀ THUẬT TOÁN

A. Đồ thị chữ ký

Để tìm một chữ ký đối tượng trong tập tin chữ ký có phù hợp với chữ ký truy vấn, ta cần phải quét một tập tin chữ ký đó. Nếu tập tin là lớn thì thời gian để tìm kiếm trên một tập tin như vậy là đáng kể. Ý tưởng trước tiên để cải thiện quá trình này là sắp xếp tập tin chữ ký và sau đó sử dụng việc tìm kiếm nhị phân. Tuy nhiên, điều này không thực hiện được do việc đối sánh trên tập tin chữ ký là một đối sánh không chính xác. Ta xem ví dụ sau đây:

Ví dụ 3.1. Xem tập tin chữ ký đã được sắp xếp gồm 3 chữ ký đối tượng:

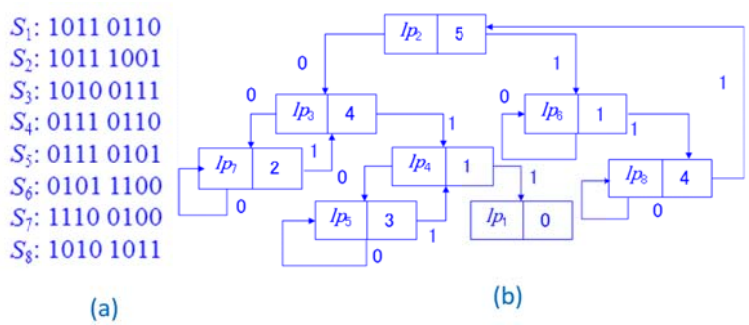
010 000 100 110
100 010 010 100
010 100 011 000

Giả sử chữ ký truy vấn s_q là 000 010 010 100. Nó phù hợp với chữ ký 100 010 010 100. Tuy nhiên, nếu ta sử dụng tìm kiếm nhị phân thì chữ ký 100 010 010 100 không thể tìm thấy. Mặt khác, trong tập tin chữ ký có thể sẽ xuất hiện nhiều chữ ký giống nhau ứng với các đối tượng có nội dung giống nhau, quá trình truy vấn cần tìm ra tất cả vị trí xuất hiện của đối tượng phù hợp. Vì lý do này, ta sẽ tổ chức tập tin chữ ký thành một đồ thị, gọi là đồ thị chữ ký nhằm lưu trữ được danh sách các chữ ký và cho phép truy ngược lại vị trí của dữ liệu tương ứng. Ta có định nghĩa sau:

Định nghĩa 3.1. Đồ thị chữ ký G_s đối với tập tin chữ ký S là một đồ thị có hướng $G_s = (V, E)$ sao cho:

1. Mỗi nút $u \in V$ có dạng $< lp(u), skip(u) >$, với $lp(u)$ là danh sách các con trỏ tham chiếu đến các vị trí của chữ ký $sig(u)$ trong tập tin chữ ký S và $skip(u)$ là một số nguyên i không âm, gọi là bước nhảy nhảy bit. Nếu $i > 0$, bit thứ i của chữ ký truy vấn s_q sẽ được kiểm tra khi tìm kiếm. Nếu $i = 0$, $sig(u)$ sẽ được so sánh với s_q.
2. Đặt $e = (u, v) \in E$. Lúc đó e được gán nhãn là 0 hoặc 1 và $skip(u) > 0$. Đặt $skip(u) = i$. Nếu e được gán nhãn là 0 và $i > 0$ thì bit thứ i của chữ ký được trỏ bởi $lp(v)$ là 0. Nếu e được gán nhãn là 1 và $i > 0$ thì bit thứ i của chữ ký được trỏ $lp(v)$ là 1. Một nút v với $skip(u) = 0$ thì không có nút con.

Ví dụ 3.2. Hình vẽ sau minh họa một tập tin chữ ký và đồ thị chữ ký của nó:



Hình 1. Minh họa tập tin chữ ký và đồ thị chữ ký

B. Thuật toán

1. Thuật toán tạo đồ thị chữ ký

Thuật toán 1: Thuật toán xây dựng đồ thị chữ ký của một lớp các đối tượng:

Vào: Lớp đối tượng

Ra: Đồ thị chữ ký

Phương pháp:

Begin
objs = {o_j | $o_j \in$ class, j = 1,…, n}
lp(root) = null; sig_1 = Hashing(o_1);
lp(root) = lp(root) ∪ sig_1;
root = <1, lp(root)>;
j = 2;
Bước 1. Tạo chữ ký sig_j = Hashing(o_j);
v ← root;
Qua bước 2;
Bước 2. if (v không đánh dấu và skip(root) ≠ 0) then Qua bước 3;
else{i ← skip(v) đánh dấu v;
if (sig_v[i] = 1) then v = v→right;
else v = v→left;
Quay lại bước 2; }
Bước 3.
if (sig_j = sig_v) then
{ lp(v) = lp(v) ∪ sig_j;
j ← j + 1; Quay lại bước 1; }
else{o ← v; s=<sig_j, lp(s)>; Qua bước 4;}
Bước 4.
Gọi k+1 là vị trí khác nhau đầu tiên giữa sig_j và sig_v;
Thay thế nút v trở thành:
<k+1, left(v)=null, right(v)= null>;
if (sig_j[k+1] = 1) then
{v → right = s; v → left = o;}
else {v → right = o; v → left = s;}
j ← j + 1; Quay lại bước 1;

End.

Trong thuật toán tạo đồ thị chữ ký G_s, vì mỗi lớp là hữu hạn có n đối tượng, đặt:

$$objs = \{o_j | o_j \in class, j = 1, \dots, n\}$$

Do đó, sẽ tạo được tập hữu hạn có n chữ ký đối tượng:

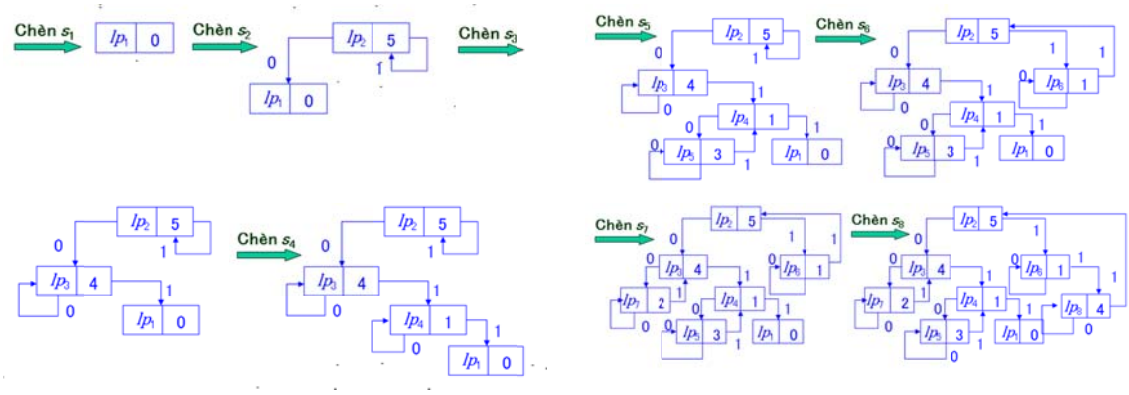
$$Sig = \{sig_j | sig_j = Hashing(o_j), j = 1, \dots, n\}$$

Với mỗi giá trị j, thuật toán sẽ duyệt từ nút gốc của đồ thị G_s để đi tìm nút phù hợp, quá trình này là hữu hạn vì số nút tạo ra trong đồ thị G_s là hữu hạn. Vì vậy thuật toán sẽ tìm được nút phù hợp để đưa chữ ký sig_j vào hoặc tạo ra một nút mới. Do đó, sau hữu hạn bước ứng với giá trị của $j = 1, \dots, n$ thì thuật toán cho kết quả là một đồ thị chữ ký G_s với các nút u có dạng $< lp(u), skip(u) >$.

Gọi n là số đối tượng trong một lớp, khi đó $n = |objs|$. Mỗi lần duyệt đồ thị chữ ký sẽ duyệt theo nhánh con bên trái hoặc nhánh bên phải, nên sẽ có $2^{\sum_{i=0}^{k} 2^i}$ lần duyệt, với $k = 0, 1, \dots, [\log_2 n]$. Vì có n chữ ký lần lượt đưa vào đồ thị

chữ ký G_s, nên số lần duyệt đồ thị tối đa sẽ là ≈ n. Tuy nhiên, trong đồ thị chữ ký G_s sẽ lần lượt kiểm tra số bit trên mỗi chữ ký trong quá trình duyệt đồ thị, gọi m là chiều dài của mỗi chữ ký, chi phí của thuật toán tạo ra đồ thị chữ ký G_s sẽ là $n.(2.m)$. Do đó, độ phức tạp trong trường hợp này sẽ là $O(n.m)$.

Ví dụ 3.3. Một ví dụ về tạo đồ thị chữ ký dựa trên tập tin chữ ký ở ví dụ 3.2 được minh họa như sau:



Hình 2. Tạo đồ thị chữ ký

2. Thuật toán truy vấn đối tượng trên đồ thị chữ ký

Sau khi đã tạo ra đồ thị chữ ký G_s, quá trình truy vấn hướng đối tượng ứng với yêu cầu truy vấn sẽ được thực hiện. Dữ liệu cần truy vấn sẽ được băm thành dạng chữ ký theo cùng phương pháp băm chữ ký trên đồ thị G_s, sau đó sẽ tiến hành tìm kiếm trên đồ thị chữ ký G_s. Kết quả của quá trình tìm kiếm này là một danh sách các con trỏ liên kết đến vị trí chữ ký trong tập tin chữ ký của cơ sở dữ liệu hướng đối tượng tương ứng.

Thuật toán 2: Thuật toán truy vấn chữ ký sig trên đồ thị G_s được thực hiện như sau:

Vào: Chữ ký sig và đồ thị G_s

Ra: Tập các con trỏ lp liên kết đến các chữ ký giống nhau nhưng có vị trí xuất hiện khác nhau trong tập tin chữ ký.

Phương pháp:

Begin
lp = ∅; v ← root; S = {v};
Bước 1. if (S = ∅) then return lp;
else Chọn v_j∈ S; S = S \ {v_j};
if (v_j được đánh dấu) then Qua bước 2;
else { i ← skip(v_j);
if sig[i] = 0 then
S = S ∪ { v_j →right, v_j→left};
else S = S ∪ {v_j→left};
Quay lại bước 1;
}
Bước 2. if sig được phủ bởi sig(v_j) then
lp = lp ∪ lp(v_j);
Quay lại bước 1;
End.

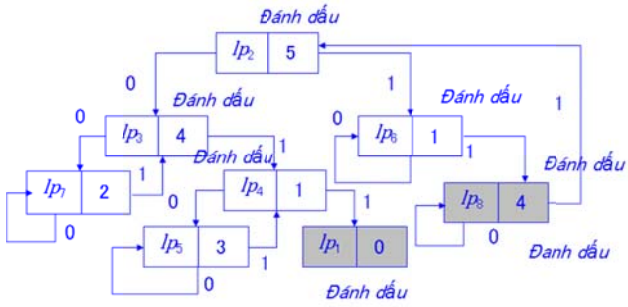
Đối với thuật toán 2, vì G_s đã tạo ra trong thuật toán 1 là hữu hạn nên tập S là một tập hữu hạn chứa các phần tử v_j sẽ được duyệt ở các bước tiếp theo.

Khi duyệt một nút $v_j \in S$, thì v_j sẽ được loại ra khỏi tập S. Do đó việc duyệt đồ thị sẽ không quay lại một nút sẽ đi qua. Thuật toán sẽ đối sánh chữ ký truy vấn và chữ ký tại các nút. Quá trình đối sánh được thực hiện trên một số hữu hạn các nút của đồ thị G_s. Vì vậy sau hữu hạn bước thuật toán sẽ cho ra được kết quả là một danh sách lp gồm các con trỏ tham chiếu đến các vị trí của chữ ký truy vấn trong tập tin chữ ký.

Trong thuật toán 2, gọi n là số nút đã được tạo ra trong G_s, mỗi lần duyệt đồ thị có thể đi theo hai nhánh của đồ thị G_s nên số lần duyệt đồ thị sẽ là 2^k, với $k = 0, 1, 2, \ldots, [\log_2 n]$. Khi đó, chi phí quá trình duyệt đồ thị để tìm kiếm tối

đa sẽ là $\log_2 n$. Trong mỗi lần duyệt đồ thị sẽ kiểm tra chữ ký tại các nút để thực hiện bước nhảy bit và thực hiện đối sánh chữ ký tại các nút, giả sử chiều dài của mỗi chữ ký là m, chí phí quá trình tìm kiếm trên đồ thị G_s sẽ là $2m\log_2 n$. Do đó, độ lớn chi phí của thuật toán sẽ là $O(m.\log n)$.

Một ví dụ về tìm kiếm trên đồ thị chữ ký dựa trên hình 2 được minh họa như sau:

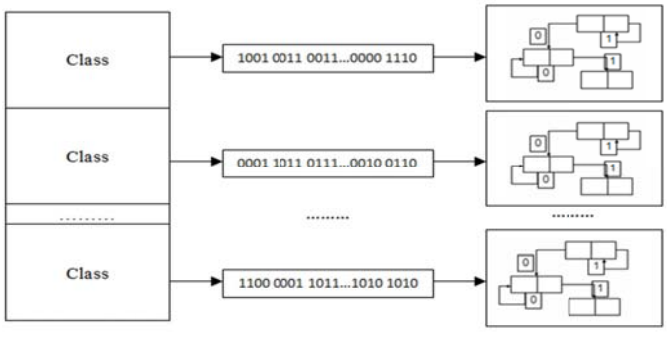


Hình 3. Tìm kiếm trên đồ thị chữ ký

Xét tập tin chữ ký và đồ thị chữ ký trên, giả sử rằng $s_q = 1011011$ là chữ ký truy vấn, lúc đó chỉ một phần của đồ thị được tìm kiếm. Để tìm ra nút v, v được đánh dấu hoặc $\text{skip}(v) = 0$ thì chữ ký của nút v sẽ được kiểm tra tương ứng với s_q. Rõ ràng cách tìm kiếm này hiệu quả hơn việc tìm kiếm tuần tự vì chỉ cần kiểm tra 2 chữ ký, trong khi đó phép duyệt tập tin chữ ký sẽ kiểm tra 8 chữ ký.

IV. MÔ HÌNH ỨNG DỤNG

Cấu trúc đồ thị chữ ký được lưu trữ hoàn toàn bên trong bộ nhớ chính, trong trường hợp này, việc chèn và xóa một chữ ký trên đồ thị được thực hiện dễ dàng. Tuy nhiên, trong cơ sở dữ liệu các tập tin thường rất lớn, vì vậy đồ thị chữ ký sẽ không thể lưu trữ trên bộ nhớ chính mà phải được lưu trữ trên bộ nhớ ngoài. Đối với cơ sở dữ liệu hướng đối tượng, chúng sẽ được lưu trữ và thực thi trên bộ nhớ ngoài. Cơ sở dữ liệu hướng đối tượng có nhiều lớp, mỗi lớp có nhiều đối tượng. Ứng với mỗi lớp sẽ được xây dựng thành một cấu trúc đồ thị chữ ký tìm kiếm, đồng thời mỗi đối tượng này sẽ tạo ra một chữ ký đối tượng. Chữ ký của mỗi đối tượng được xây dựng trong mô hình này có chiều dài 64 bit, đó là sự tổ hợp các thuộc tính trong một đối tượng. Toàn bộ cơ sở dữ liệu hướng đối tượng sẽ được phân hoạch dưới dạng cấu trúc một bảng băm gồm các chữ ký của đối tượng để thực hiện quá trình truy vấn.



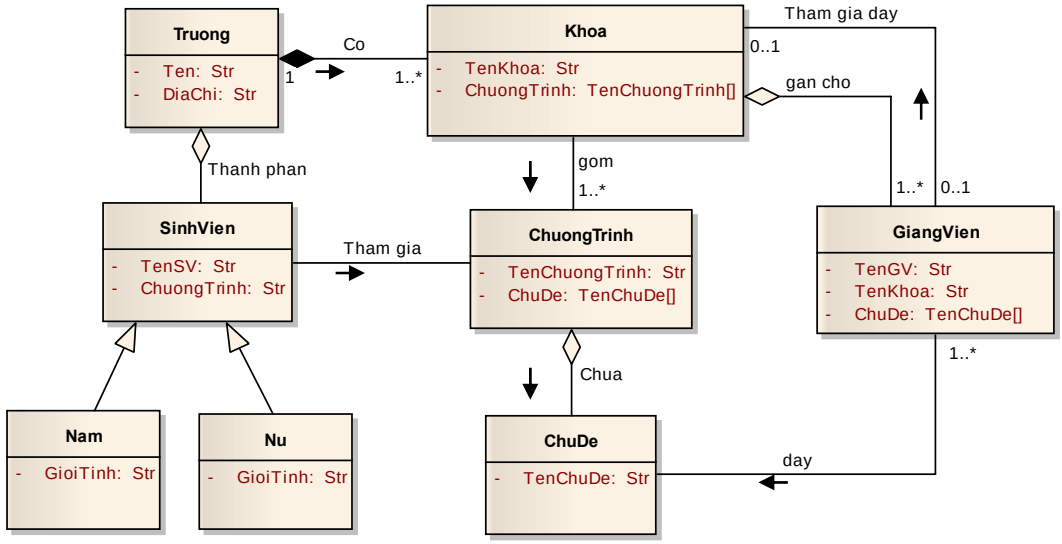
Hình 4. Mô hình cấu trúc lưu trữ đồ thị chữ ký cho cơ sở dữ liệu hướng đối tượng

A. Một ví dụ về mô hình cơ sở dữ liệu hướng đối tượng

Để thực nghiệm truy vấn hướng đối tượng trên CSDL hướng đối tượng. Một ví dụ mô hình CSDL hướng đối tượng được đưa ra ở hình 5 đồng thời cũng đưa ra những quan hệ trên các lớp đối tượng ở bảng 1. Dựa trên mô hình này để cài đặt CSDL hướng đối tượng ở mức vật lý.

Bảng 1. Quan hệ của các lớp

TT	Lớp 1	Lớp 2	Quan hệ
1	Truong	Khoa	Chứa trong
2	Truong	SinhVien	Kết tập
3	SinhVien	ChuongTrinh	Liên kết
4	Khoa	GiangVien	Liên kết
5	SinhVien	Nu	Khái quát quá
6	SinhVien	Nam	Khái quát quá
7	ChuongTrinh	ChuDe	Liên kết



Hình 5. Một ví dụ mô hình cơ sở dữ liệu hướng đối tượng

B. Xử lý truy vấn trên cơ sở dữ liệu hướng đối tượng

Để thực hiện việc truy vấn một đối tượng trên CSDL hướng đối tượng, đầu tiên phải chuyển đổi cơ sở dữ liệu hướng đối tượng thành cấu trúc dữ liệu như trên, ta thực hiện như sau:

Bước 1. Thuộc tính của đối tượng được băm thành chữ ký nhị phân và các thuộc tính tạo thành chữ ký đối tượng.

Bước 2. Các chữ ký đối tượng của cùng một lớp sẽ tạo thành đồ thị chữ ký.

Bước 3. Tạo danh sách đồ thị chữ ký tương ứng với từng lớp.

Sau khi có cấu trúc dữ liệu để truy vấn, ta thực hiện quá trình truy vấn đối tượng trong cơ sở dữ liệu hướng đối tượng như sau:

Bước 1. Mã hoá từ khóa cần truy vấn thành chữ ký nhị phân.

Bước 2. Đối sánh chữ ký từ khóa để xác định thuộc lớp cần truy vấn.

Bước 3. Thực hiện truy vấn chữ ký từ khóa trên đồ thị chữ ký tương ứng với các lớp đã xác định.

V. KẾT LUẬN

Bài báo đã đề xuất việc xây dựng đồ thị chữ ký để lưu trữ chữ ký đối tượng của CSDL hướng đối tượng và xây dựng thuật toán truy vấn đối tượng trên đồ thị chữ ký. Đồng thời dựa trên cấu trúc đồ thị chữ ký đã tạo, bài báo đã đưa ra một mô hình ứng dụng. Phương pháp này có thể áp dụng để truy vấn các đối tượng dữ liệu lớn như đối tượng dữ liệu ảnh, các đối tượng multimedia, các đối tượng trong hệ thống thông tin địa lý,…

VI. TÀI LIỆU THAM KHẢO

[1] Yangjun Chen, Yibin Chen, “On the Signature Tree Construction and Analysis”, IEEE Trans. Knowl. Data Eng., 18(9), 2006, 1207-1224.

[2] Yangjun Chen, “Building Signature Trees into OODBs”, Journal of Information Science and Engineering, 20(2), 2004, 275-304.

[3] Yangjun Chen, Yibin Chen, “Signature File Hierarchies and Signature Graphs: a New Index Method for Object-Oriented Databases”, Proceedings of the 2004 ACM symposium on Applied computing, Nicosia, Cyprus, 14-17 March 2004, 724-728.

[4] D. Dervos, Y. Manolopulos and P. Linardis, “Comparison of signature file models with superimposed coding”, J. of Information Processing Letters 65 (1998) 101 - 106.

[5] C. Faloutsos, “Signature Files: Design and Performance Comparaison of Some Signature Extraction Methods”, ACM Sigmod Record, Volume 14, Issue 4, May 1985, pp. 63 – 82.

[6] D. L. Lee, Y. M. Kim, G. Patel, “Efficient Signature File Methods for Text Retrieval”, IEEE Trans. Knowl. Data Eng., 7(3), 1995, 423-435.

[7] W. C. Lee, D. L. Lee, “Signature File Methods for Indexing Object-Oriented Database systems”, Proceedings of the 2nd International Computer Science Conference, Hong Kong, 1992, 616-622.

[8] P. Mahatthanapiwat, "Flexible Searching for Graph Aggregation Hierarchy", Proceedings of the World Congress on Engineering, 2010, London, UK, June 30-July 2, 2010, 405-409.

[9] E. Tousidoua, P. Bozanis, Y. Manolopoulos, "Signature-based structures for objects with set-valued attributes", Elsevier Science, Information Systems, 27(2), 2002, 93-121.

OBJECT-ORIENTED QUERY PROCESSING BASED SIGNATURE BINARY GRAPH

Tran Minh Bao, Truong Cong Tuan

ABSTRACT *- Indexing is always an important issue in the efficient information retrieval from the databases. For object-oriented databases, the use of signature files as a method of indexing has been recognized as a common approach in searching on object-oriented databases. The objects of the class are encoded into the object signatures using hash functions and stored in a signature file. However, when a query arrives, the entire signature file must be scanned. So this method requires a high processing cost. To overcome this drawback, we propose a graph structure model constructed over signatures of objects for a class in an object-oriented database, called a signature graph. We also suggest an algorithm to build the signature graph and query algorithm on signature graph, as well as an application of this method.*