TẠP CHÍ KHOA HỌC, Đại học Huế, Tập 74B, Số 5, (2012), 157-165

TRUY VẤN DỮ LIỆU DỰA TRÊN CÂY CHỮ KÝ CỦA KHỐI VĂN BẢN

Văn Thế Thành, Trần Minh Bảo

Trường Đại học Công nghiệp thực phẩm TP. HCM

Tóm tắt. Bài báo xây dựng mô hình cấu trúc dữ liệu lưu trữ tập tin chữ ký của văn bản dưới dạng các khối văn bản, mỗi khối văn bản được mã hóa và xây dựng dưới dạng một cấu trúc cây chữ ký, từ đó xây dựng ứng dụng mô phỏng việc truy vấn dữ liệu trên cây chữ ký khối văn bản, đồng thời thực hiện việc đánh dấu dữ liệu đã được truy vấn lên văn bản gốc. Bài báo thực hiện việc mô phỏng thực nghiệm phương pháp truy vấn trên các văn bản có hơn 20.000 từ, qua đó bài báo đưa ra việc đánh giá chi phí của phương pháp thông qua thực nghiệm dựa trên mô hình cấu trúc dữ liệu đã đưa ra.

1. Giới thiệu

Việc thực thi truy vấn trực tiếp trên cơ sở dữ liệu có thể rất tốn kém chi phí trong việc duyệt trên một số lượng lớn các mẫu tin trong cơ sở dữ liệu. Do đó, ta cần mô tả lại hệ thống cơ sở dữ liệu dựa trên một cấu trúc tham chiếu có không gian tìm kiếm nhỏ hơn để từ đó giảm thời gian tìm kiếm trong quá trình truy vấn dữ liệu và đồng thời cấu trúc tham chiếu trung gian này có thể truy vấn ngược lại cơ sở dữ liệu thực sự.

Để giảm không gian truy vấn dữ liệu, trong bài báo này sẽ tiếp cận phương pháp tạo chữ ký cho các đối tượng dữ liệu, từ đó xây dựng các cấu trúc dữ liệu để tham chiếu đến cơ sở dữ liệu thực sự. Chữ ký nhỏ hơn rất nhiều so với đối tượng dữ liệu thực sự (khoảng từ 10% – 20% so với đối tượng dữ liệu [4]). Chữ ký của các đối tượng dữ liệu sẽ được lưu trữ trong tập tin chữ ký và qua đó thực hiện phép truy vấn các đối tượng dữ liệu dựa trên tập tin chữ ký này. Ngoài ra, để việc tìm kiếm hiệu quả hơn, cần xây dựng cấu trúc dữ liệu lưu trữ tập tin chữ ký, cấu trúc lưu trữ tập tin chữ ký này có thể dưới dạng các tập tin chữ ký tuần tự, các tập tin chữ ký phân mảnh, cấu trúc cây chữ ký, cấu trúc dạng đồ thị chữ ký,… quá trình tạo ra các cấu trúc lưu trữ tập tin chữ ký sẽ làm giảm không gian tìm kiếm và tối ưu quá trình truy vấn dữ liệu.

Các phương pháp tạo cấu trúc dữ liệu lưu trữ chữ ký để truy vấn dữ liệu đã công bố như: truy vấn dữ liệu đối tượng dựa trên cây chữ ký SD-Tree [1], xây dựng cấu trúc cây chữ ký để giảm không gian tìm kiếm dữ liệu [2, 6], truy vấn dữ liệu trên tập tin văn bản bằng phương pháp tạo tập tin chữ ký tuần tự và tập tin chữ ký phân mảnh [3, 7], tạo chỉ mục truy vấn cho các tập tin văn bản [4, 5, 9], truy vấn cơ sở dữ liệu đối tượng dựa trên các cấu trúc tập tin chữ ký [4, 5, 8].

Bài báo sẽ tập trung vào việc xây dựng cấu trúc lưu trữ tập tin chữ ký với đối

tượng dữ liệu là tập tin văn bản, đồng thời xây dựng ứng dụng mô phỏng phương pháp. Nội dung của bài báo được tổ chức như sau: phần đầu tiên sẽ giới thiệu khái quát về phương pháp tạo ra chữ ký cho các đối tượng dữ liệu và giới thiệu các phương pháp đã được công bố về việc xây dựng cấu trúc dữ liệu lưu trữ tập tin chữ ký; phần thứ hai của bài báo nhằm giới thiệu các khái niệm cơ bản về chữ ký của đối tượng dữ liệu; phần thứ ba sẽ mô tả cấu trúc cây chữ ký cho tập tin chữ ký và giới thiệu việc tìm kiếm trên cây chữ ký; phần thứ tư sẽ xây dựng cấu trúc lưu trữ tập tin chữ ký ứng với dữ liệu là đối tượng văn bản, sau đó xây dựng ứng dụng nhằm mô phỏng quá trình xử lý truy vấn dữ liệu văn bản và đánh giá việc thực nghiệm của quá trình truy vấn và đưa ra hướng phát triển.

2. Chữ ký

Chữ ký là các vector bit được xây dựng bằng phép băm mã hóa các đối tượng dữ liệu. Chữ ký là sự trừu tượng hóa của dữ liệu gốc với kích thước nhỏ hơn các đối tượng dữ liệu và được duyệt nhanh hơn quá trình duyệt các đối tượng. Việc mã hóa này sẽ tạo ra *k* bit 1 trong dãy bit [1..*m*], với *m* là chiều dài của chữ ký. Một mẫu nhị phân với *k* bit 1 và *(m-k)* bit 0 được gọi là một chữ ký.

Các đối tượng dữ liệu và các giá trị câu truy vấn được mã hóa trên cùng một thuật toán mã hóa chữ ký. Khi các bit trong chữ ký đối tượng dữ liệu hoàn toàn phủ các bit trong chữ ký câu truy vấn, thì đối tượng dữ liệu này là một ứng viên thỏa mãn câu truy vấn.

Chữ ký s_i của đối tượng *O* là một tổ hợp các dãy bit được tạo ra từ các giá trị của các thuộc tính trong đối tượng *O*. Chữ ký của đối tượng được lưu trữ trong một tập tin chữ ký.

Chữ ký s_q của câu truy vấn được đối sánh với các chữ ký trong tập tin chữ ký. Có ba kết quả của phép đối sánh:

(1) Đối tượng phù hợp với câu truy vấn, có nghĩa là mọi bit trong s_q được phủ bởi các *bit* trong chữ ký s_i của đối tượng dữ liệu (nghĩa là $s_q \wedge s_i = s_q$);

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

(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 câu 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.

Chữ ký của khối văn bản hoặc của một đối tượng có thể được tạo ra bằng cách tổ hợp tất cả các chữ ký thành phần, tập hợp các chữ ký sẽ tạo ra tập tin chữ ký. Tập tin chữ ký tuần tự gồm các chữ ký lưu trữ tuần tự. Tập tin chữ ký phân mảnh lưu trữ các bit của chữ ký theo từng cột.

Đối tượng dữ liệu:	John	12345678	professor

Chữ ký của thuộc tính:

John	010 000 100 110
12345678	100 010 010 100
professor	110 100 011 000

Chữ ký của đối tượng: (∨) 110 110 111 110

Hình 1. *Một ví dụ về chữ ký của đối tượng [2]*

3. Cây chữ ký

3.1. Cây chữ ký

Cây chữ ký T ứng với một tập tin chữ ký $S = s_1.s_2. \ldots .s_n$ với $s_i \neq s_j$ $(i \neq j)$ và $|s_k|= m$, $k = 1,\ldots, n$ là một cây nhị phân sao cho: [2, 6]

1. Với một nút trong của T, nhánh cây bên trái luôn được gán nhãn là 0 và nhánh cây bên phải luôn được gán nhãn là 1.

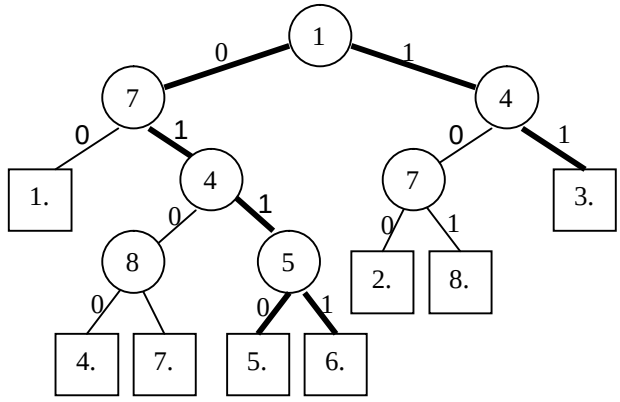
2. T có n nút lá được gán nhãn là 1, 2, …, n được sử dụng như các con trỏ để trỏ đến n vị trí khác nhau của $s_1, s_2,\ldots, s_n$ trong tập tin chữ ký S.

3. Mỗi một nút trong được kết hợp với một số (gọi là bước nhảy bit) để bỏ qua trong quá trình tìm kiếm.

4. Đặt $i_1, \ldots, i_h$ là các số được kết hợp với các nút trên đường dẫn từ nút gốc đến nút lá có nhãn là i (mỗi nút lá trỏ đến chữ ký thứ i trong S). Đặt $p_1,\ldots, p_h$ là dãy các nhãn của các nhánh trên đường dẫn. Thì khi đó $(j_1, p_1), \ldots, (j_h, p_h)$ sẽ tạo ra một định danh chữ ký cho s_i, ký hiệu là $s_i(j_1, \ldots, j_h)$.

3.2. Tìm kiếm trên cây chữ ký

s_1. 011 001 000 101
s_2. 111 011 001 111
s_3. 111 101 010 111
s_4. 011 001 101 111
s_5. 011 101 110 101
s_6. 011 111 110 101
s_7. 011 001 111 111
s_8. 111 011 111 111



Hình 2. *Cây chữ ký [6]*

Đặt s_q là nút tìm được trong quá trình duyệt cây truy vấn $Q_{(s, h)}$. Vị trí thứ i của s_q ký hiệu là $s_q(i)$, trong quá trình duyệt cây truy vấn, được thực hiện như sau:

(i) Đặt v là nút tìm được, đặt $s_q(i)$ là vị trí để kiểm tra.

(ii) Nếu $s_q(i) = 1$, ta duyệt qua cây con bên phải của v.

(iii) Nếu $s_q(i) = 0$, ta duyệt cả cây con bên trái và bên phải của v.

Xét tập tin chữ ký và cây chữ ký trên, giả sử rằng s_q = 000 100 100 000, lúc đó chỉ một phần của cây được tìm kiếm. Để tìm ra nút lá, thì chữ ký của nút lá sẽ được kiểm tra tương ứng với s_q. Rõ ràng rằng cách tìm kiếm này hiệu quả hơn việc tìm kiếm tuần tự vì chỉ cần kiểm tra 3 chữ ký, trong khi đó phép duyệt tập tin chữ ký sẽ kiểm tra 8 chữ ký.

4. Truy vấn dữ liệu trên văn bản

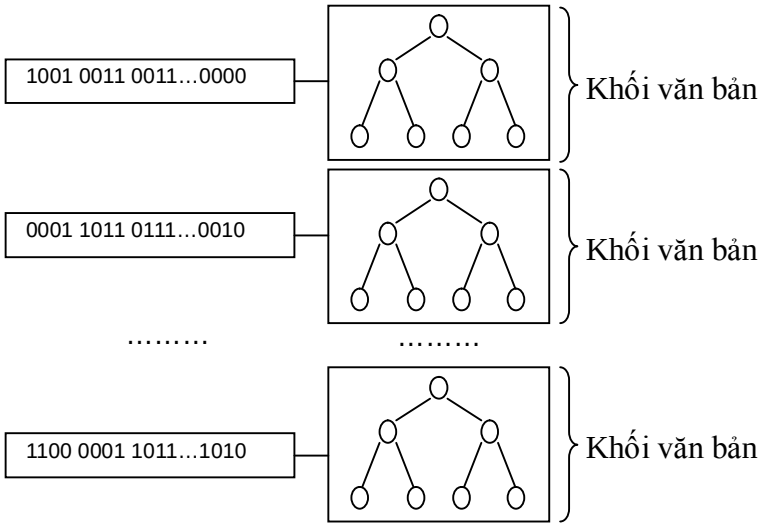
4.1. Xây dựng cấu trúc lưu trữ chữ ký văn bản

Cấu trúc cây chữ ký được lưu trữ hoàn toàn bên trong bộ nhớ chính, trong trường hợp này, việc chèn và xóa một chữ ký trên cây được thực hiện dễ dàng. Tuy nhiên, trong cơ sở dữ liệu các tập tin thường rất lớn, vì vậy cây chữ ký sẽ không thể lưu trữ trên bộ nhớ chính mà phải được lưu trữ trên bộ nhớ ngoài.

Đối với dữ liệu văn bản, chúng sẽ được lưu trữ và thực thi trên bộ nhớ ngoài. Tập tin văn bản sẽ được chia thành dãy gồm các khối văn bản và được tổ chức liên tục trên tập tin văn bản. Ứng với mỗi khối văn bản sẽ được xây dựng thành một cấu trúc cây chữ ký tìm kiếm, đồng thời mỗi khối văn bản này sẽ tạo ra một chữ ký đối tượng văn bản.

Chữ ký của mỗi khối văn bản được xây dựng trong mô hình này có chiều dài 64 bit, đó là sự tổ hợp các đối tượng thành phần trong khối văn bản.

Toàn bộ văn bản sẽ được phân hoạch dưới dạng cấu trúc một bảng băm gồm các chữ ký của khối dữ liệu văn bản để thực hiện quá trình truy vấn.

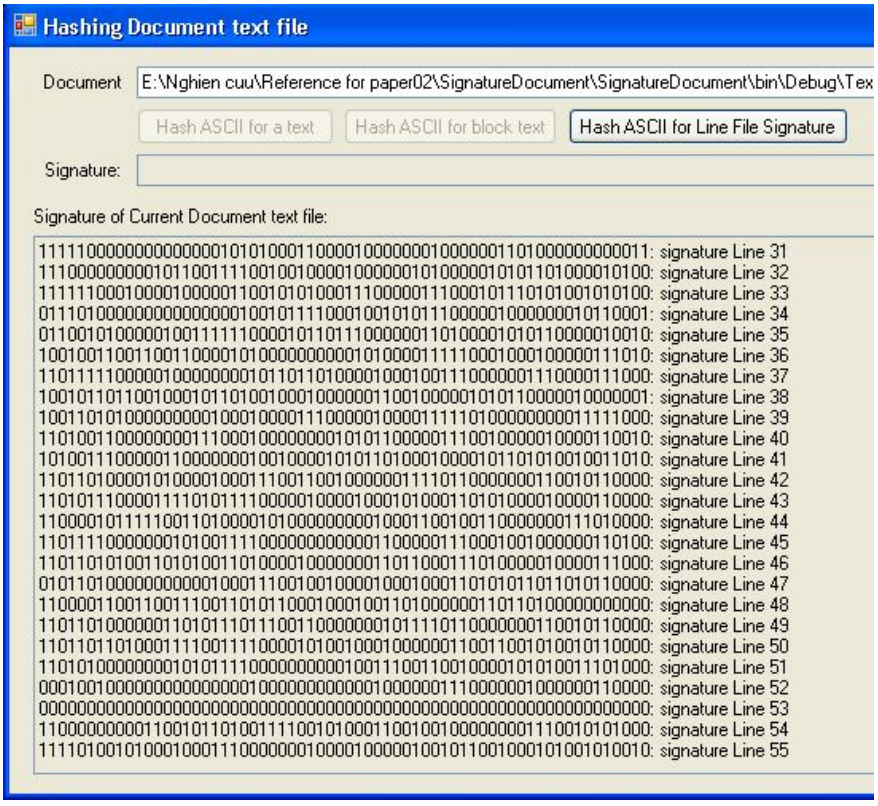


Hình 3. *Cấu trúc lưu trữ chữ ký cho tập tin văn bản*

Một cây chữ ký trong một khối văn bản có dạng: $b_n(r_n; a_1, ..., a_m)$, với b_n mô tả một cây chữ ký của khối văn bản thứ n, r_n là nút gốc của cây b_n và $a_1, ...,a_m$ là các nút lá của cây b_n. Mỗi một nút u trong cây b_n có dạng: $<v(u), l(u), r(u)>$, với $v(u), l(u), r(u)$ lần lượt là: giá trị khóa, liên kết trái và liên kết phải của nút u. Với mỗi nút lá a_i của b_n có dạng: $<v(a_i), lp(a_i), rp(a_i)>$, với $v(a_i)$ là giá trị của a_i,và $lp(a_i), rp(a_i)$ là các con trỏ sẽ trỏ đến các chữ ký của dữ liệu trong khối văn bản. Kích thước $|b|$ của cây nhị phân b là số lượng các nút trong cây b của một khối văn bản thõa $|b| \leq 2^k$, với k là một số nguyên.

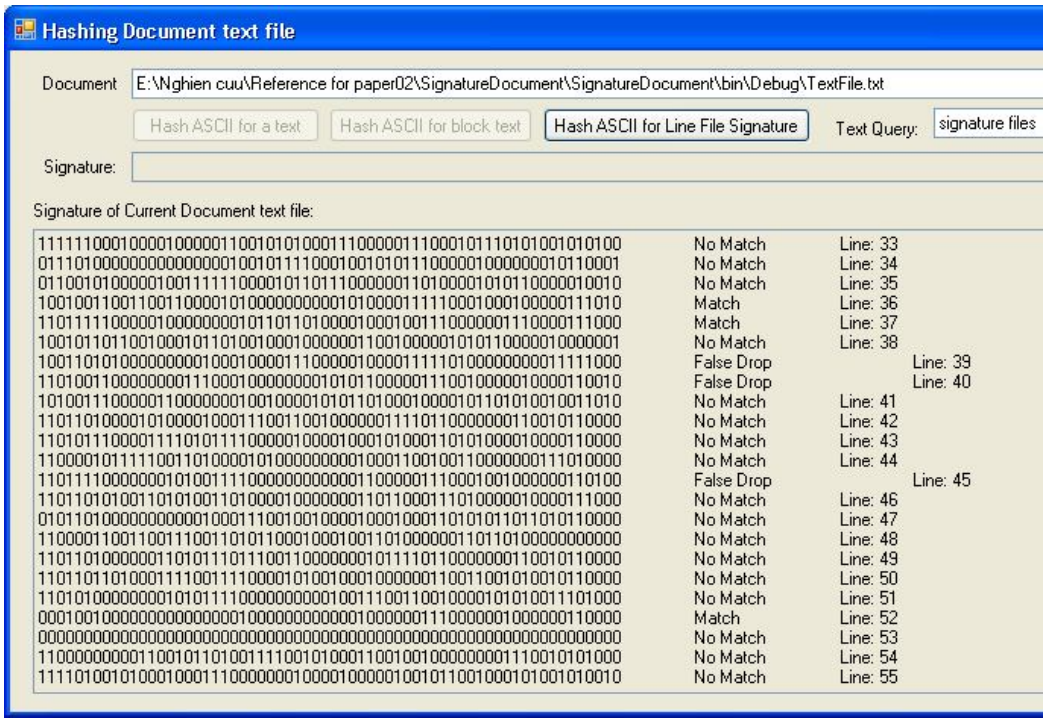
4.2. Xử lý truy vấn dữ liệu văn bản

Bước 1. Xử lý dữ liệu ban đầu: Phân đoạn văn bản thành các khối văn bản. Sau đó, tạo cây chữ ký cho mỗi khối văn bản, đồng thời tạo chữ ký cho các khối văn bản và tạo ra các định danh cho khối của văn bản để truy ngược lại văn bản gốc ban đầu.



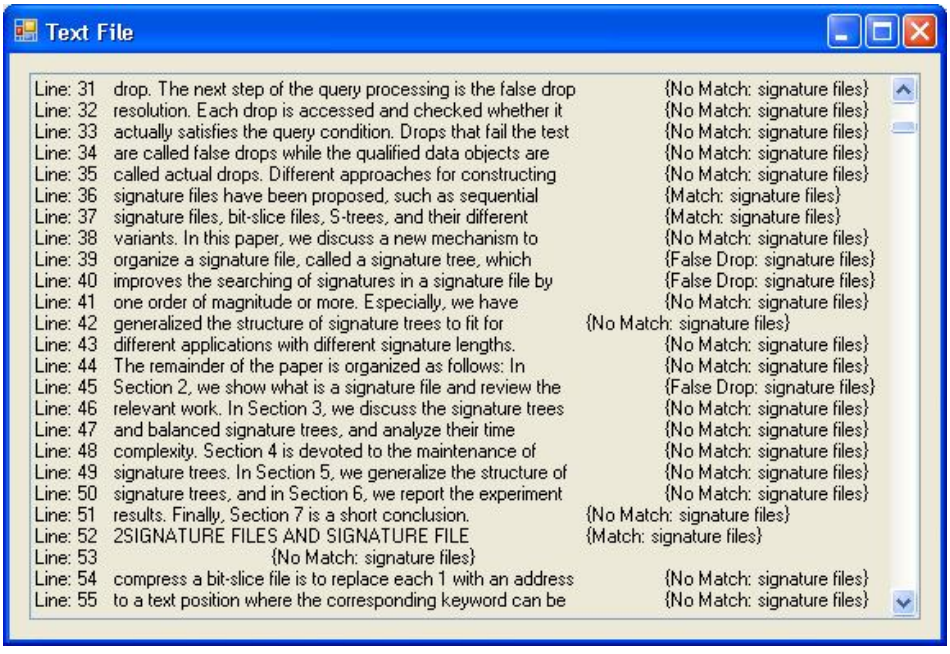
Hình 4. *Tạo chữ ký cho các khối văn bản*

Bước 2. Đối sánh dữ liệu truy vấn với tập tin chữ ký: Ta thực hiện việc tạo ra chữ ký cho câu truy vấn, phương pháp tạo chữ ký cho câu truy vấn giống như cách tạo chữ ký cho văn bản. Sau đó thực hiện việc đối sánh chữ ký truy vấn trên tập tin chữ ký của các khối văn bản và xác định các khối văn bản có chứa chữ ký câu truy vấn. Sau khi xác định khối văn bản phù hợp với dữ liệu truy vấn, ta thực hiện việc xác định dữ liệu trên cây chữ ký của khối văn bản này để từ đó xác định kết quả truy vấn dữ liệu.



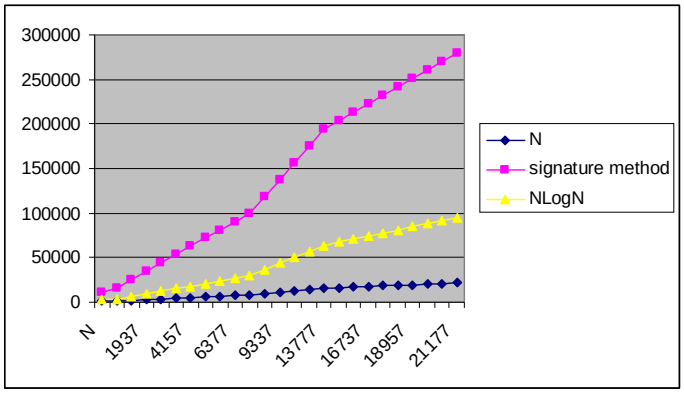
Hình 5. *Truy vấn trên tập tin chữ ký với khóa truy vấn: "signature files"*

Bước 3. Phát hiện các trường hợp "tìm kiếm nhầm lẫn": Sau khi xác định được kết quả trên các khối văn bản, ta lần lượt kiểm chứng dữ liệu trên cây chữ ký văn bản với các dữ liệu được liên kết để phát hiện ra các trường hợp mặc dầu thõa mãn yêu cầu đối sánh nhưng không thõa điều kiện tìm kiếm của câu truy vấn (truy vấn "*nhầm lẫn*"). Sau đó, tiến hành ghi nhận và đánh dấu lên tập tin văn bản gốc.

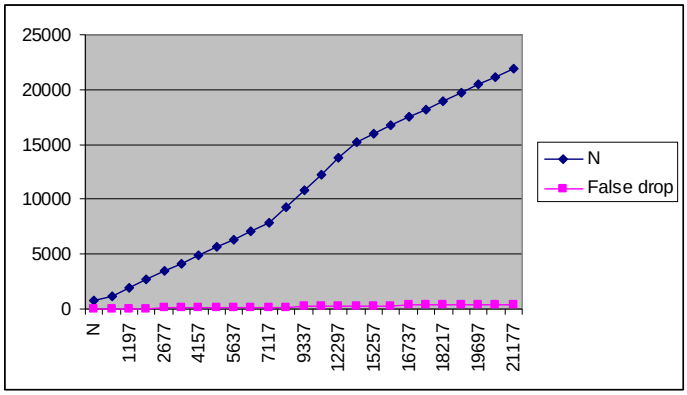


Hình 6. *Ghi nhận trên tập tin văn bản với khóa truy vấn: "signature files"*

4.3. Đánh giá thực nghiệm



Hình 7. *Đánh giá thực nghiệm truy vấn dữ liệu văn bản theo mô hình đã đưa ra*



Hình 8. *Đánh giá thực nghiệm các trường hợp truy vấn bị "nhầm lẫn"*

Phương pháp tạo ra tập tin gồm các khối chữ ký văn bản sẽ tốn nhiều chi phí cho việc tạo ra tập tin chữ ký. Hơn nữa, sẽ chi phí cho không gian lưu trữ các cây chữ ký của các khối chữ ký văn bản. Tuy nhiên, chi phí truy vấn chữ ký dựa trên cây chữ ký diễn ra tương đối nhanh, hơn nữa số trường hợp truy vấn bị *"nhầm lẫn"* tương đối ít, do đó chi phí kiểm nghiệm trường hợp *"nhầm lẫn"* không đáng kể so với khối lượng một văn bản lớn.

Chi phí tìm kiếm trung bình trên cây chữ ký cân bằng sẽ là $O(\lambda.log_2 n)$, với n là số nút lá, λ là số đường đi tương ứng với số chữ ký được kiểm tra [6]. Việc tạo ra cây chữ ký cân bằng cho mỗi khối văn bản sẽ giảm chi phí tìm kiếm, tuy nhiên sẽ tốn thời gian tạo ra cây chữ ký cân bằng và trong quá trình thực thi phải tốn thêm chi phí cân bằng lại cây vì có thể thêm hoặc xóa dữ liệu trên cây trong quá trình thực thi.

5. Kết luận

Trong bài báo này đã xây dựng mô hình cấu trúc dữ liệu để lưu trữ tập tin chữ ký của văn bản nhằm phục vụ việc truy vấn dữ liệu văn bản; dựa trên mô hình cấu trúc này, bài báo đã xây dựng ứng dụng thực nghiệm và đánh giá phương pháp dựa trên các văn bản lớn. Mặc dầu việc tạo ra dữ liệu ban đầu tốn nhiều chi phí, tuy nhiên quá trình truy

vấn dữ liệu diễn ra tương đối nhanh, do đó có thể áp dụng phương pháp này trong trường hợp 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,… Một trong những hướng phát triển cho phương pháp lưu trữ này đó là áp dụng trong việc truy vấn trên đối tượng trong mô hình cơ sở dữ liệu hướng đối tượng.

TÀI LIỆU THAM KHẢO

[1]. Elizabeth Shanthi, R. Nadarajan, *Applying SD-Tree for Object-Oriented Query Processing*, Informatica 33, (2009), 177-187, .

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

[3]. Dik Lun Lee, Young Man Kim, Gaurav Patel, *Efficient Signature File Methods for Text Retrieval*, IEEE Transaction on Knowledge and Data Engineering, Vol. 7, No. 3, (1995), 423-435.

[4]. Walter W.Chang, Hans J. Schek, *A signature Access Method for the Starburst Database System*, Proceedings of the Fifteenth International Conference on Very Large Database, Amsterdam, (1989), 145-153.

[5]. Wang-chien Lee and Dik L. Lee, *Signature File Methods for Indexing Object-Oriented Database systems*, Proceedings of the 2nd International Computer Science Conference, Hong Kong, (1992), 616-622.

[6]. Yangjun Chen and Yibin Chen, *On the Signature Tree Construction and Analysis*, IEEE Transactions On Knowledge and Data Engineering, Vol. 18, No. 9, 2006.

[7]. Seyit Kocberber, Fazli Can, *Partial evaluation of queries for bit-sliced signature files*, ELSEVIER, Information Processing Letters, 60, (1996), 305-311.

[8]. Kjetil Norvag, *Signature caching in parallel object database systems*, ELSEVIER, Information and Software Technology, 44, (2002), 331-334.

[9]. Edi Winarko, John F. Roddick, *A Signature-Based Indexing Method for Efficient Content-Based Retrieval of Relative Temporal Patterns*, IEEE Transactions On Knowledge and Data Engineering, Vol. 20, No. 6, (2008), 825-835.

QUERY ABOUT *DATA* BASED ON THE SIGNATURE TREE OF TEXT BLOCK

Van The Thanh, Tran Minh Bao

HoChiMinh city Univetsity of Food Industry

Abstract. In this paper, we propose an approach to data structure models to store the signature file of the text as text blocks, each of which is encrypted and built in form of a signature tree structure, from which to build simulation application to query data on the signature tree of text block and simultaneously perform the marking of data on plain text. The papers represents the experimental simulation on the text file with more than 20,000 words through which it advocates the assessment on the cost of this method by the experiments based on the data structure model previously given.