PHƯƠNG PHÁP CHỈ MỤC TÀI LIỆU TRONG THƯ VIỆN SỐ

ĐỖ QUANG VINH

1. MỞ ĐẦU

Hầu hết mọi người quen thuộc với cách dùng một chỉ mục trong một cuốn sách. Sử dụng một chỉ mục có thể định vị các trang liên quan trong một cuốn sách thậm chí cuốn sách được viết bằng ngôn ngữ khác.

Một cuốn sách không có một chỉ mục gây khó khăn cho người sử dụng (NSD). Khó khăn tìm kiếm là do một chỉ mục không đầy đủ hoặc không có một chỉ mục nào cả.

Đối với thư viện số, chúng ta đang nói về dữ liệu lớn, hàng triệu trang văn bản ít có cấu trúc và không có một đầu mối theo ngữ cảnh nào như tựa đề. Duyệt ngẫu nhiên dữ liệu lớn bằng thủ công rất tốn kém và ngay cả tìm kiếm vét cạn bằng biện pháp cơ học là đắt. Nếu không có một chỉ mục có sẵn, sự trích lọc thông tin tất phải thất bại. Do đó, thành công của hệ truy tìm tài liệu IR quyết định thông tin lưu trữ được chỉ mục chính xác và đầy đủ.

Ở đây, chúng tôi khảo sát *chỉ mục tệp đảo* IFID, *chỉ mục tệp ký số* SFID và sau đó, đánh giá các phương pháp chỉ mục tài liệu sử dụng trong thư viện số.

Giả thiết một *cơ sở dữ liệu tài liệu* (CSDL) được coi là một tập các *tài liệu* riêng biệt, mỗi một được mô tả bởi một tập *thuật ngữ đại diện* và chỉ mục phải có khả năng nhận dạng tất cả tài liệu chứa tổ hợp của các thuật ngữ đã định rõ, hoặc theo một cách khác nào đó đánh giá là có liên quan tới tập thuật ngữ truy vấn. Như vậy, một tài liệu là đơn vị văn bản trả lại đáp ứng cho truy vấn.

Có những trường hợp nhạy cảm lựa chọn một tài liệu trong CSDL là một đoạn, hoặc thậm chí chỉ một câu của một tài liệu gốc.

Người thiết kế CSDL lựa chọn tính kết hạt của chỉ mục – giải pháp cho sự định vị thuật ngữ được ghi bên trong mỗi một tài liệu.

Trong giới hạn, nếu tính kết hạt của chỉ mục được lấy bằng một từ thì chỉ mục sẽ ghi định vị chính xác của mỗi một từ trong CSDL, như vậy, văn bản gốc có thể phục hồi từ chỉ mục. Ở trường hợp này, không có khả năng chỉ mục được lưu trữ trong không gian nhỏ hơn so với lượng tối thiểu có thể đối với văn bản chính dùng một giải thuật nén văn bản chuẩn. Nếu có thể thì phương pháp nén chỉ mục có thể được sử dụng bằng một giải thuật nén văn bản tốt hơn và dường như không thể có.

Khi tính kết hạt của chỉ mục thưa hơn – theo mức độ câu hoặc tài liệu – văn bản nhập có thể không còn được tái sinh từ chỉ mục và một biểu diễn kinh tế hơn trở nên có khả năng. Mỗi một đầu vào trong một chỉ mục mức tài liệu là một con trỏ hướng tới một tài liệu riêng biệt và đối với một chỉ mục có một triệu tài liệu, như một con trỏ lấy 20 bit không nén. Tuy nhiên, có thể làm giảm tới khoảng 6 bit đối với các CSDL tài liệu điển hình, quả thực là một sự tiết kiệm rất đáng giá.

Người thiết kế CSDL quyết định các thuật ngữ đại diện cho tài liệu văn bản nên được tạo lập như thế nào. Một khả năng đơn giản là lấy mỗi một trong số từ xuất hiện trong tài liệu và khai báo nó đúng nguyên văn là một thuật ngữ. Điều này có xu hướng cả mở rộng từ vựng của CSDL – số thuật ngữ riêng biệt xuất hiện – lẫn làm tăng số bộ nhận dạng tài liệu phải được lưu trữ trong chỉ mục. Có một từ vựng quá rộng không chỉ ảnh hưởng đến các yêu cầu không gian lưu trữ của hệ thống mà còn có thể làm cho nó khó dùng hơn vì có nhiều thuật ngữ truy vấn tiềm năng hơn phải xem xét khi trình bày chính xác các yêu cầu hệ

thống. Do đó, thông thường mỗi một từ được chuyển đổi theo các cách sau đây trước khi được bao hàm trong chỉ mục. Thứ nhất là gộp dạng chữ – chuyển đổi tất cả chữ hoa thành chữ thường tương đương hoặc ngược lại. Thứ hai là quá trình truy gốc từ – nghĩa là, đối với tất cả hậu tố và các từ bổ nghĩa khác bị loại bỏ và được tiến hành sao cho các truy tìm tài liệu liên quan dù là dạng chính xác của từ là khác nhau. Cần chú ý nó có thể gây ra truy tìm tài liệu không liên quan. Thứ ba là loại trừ các từ bỏ qua – các từ phổ biến hoặc có ít nội dung thông tin sao cho sự sử dụng chúng trong một truy vấn không thể loại trừ bất kỳ tài liệu nào vì chúng có thể có mặt ở hầu hết tài liệu. Do đó, không thứ gì bị mất nếu chúng đơn giản bị loại trừ khỏi chỉ mục. Cuối cùng là phép thay thế từ đồng nghĩa, trong đó các từ đồng nghĩa được nhận dạng và chỉ mục dưới một thuật ngữ đại diện đơn.

Ở đây, chúng tôi thử nghiệm trên CSDL TREC của Viện tiêu chuẩn và công nghệ quốc gia Mỹ NIST

Bảng 1 – Cơ sở dữ liệu TREC.

Số tài liệu N	741856
Số thuật ngữ F	333338738
Số thuật ngữ riêng biệt n	535346
Số con trỏ chỉ mục *f*	134994414
Kích thước tổng (MB)	2070.29

CSDL TREC (Text REtrieval Conference) là một CSDL tài liệu rất lớn, có tổng cộng hơn 2 GB văn bản và gần 750000 tài liệu.

Định nghĩa từ được sử dụng để nhận dạng đối với chỉ mục:

Một từ là một dãy cực đại của các ký tự chữ và số, nhưng giới hạn tối đa 256 ký tự tổng cộng và tối đa bốn ký tự số.

Giới hạn là để tránh các dãy số trang trở thành dải dài của các từ riêng biệt. Không có giới hạn này, kích thước của từ vựng có thể bị phình ra không cần thiết.

2. CHỈ MỤC TỆP ĐẢO IFID

Chỉ mục là một cơ chế nhằm định vị thuật ngữ cho trước trong văn bản. Có nhiều cách để đạt được điều này. Ở các ứng dụng văn bản, cấu trúc phù hợp đơn giản nhất là *tệp đảo* (IF), đôi khi được coi là *tệp mục lục*.

Bây giờ, chúng tôi định nghĩa chính xác IFID. Đối với mỗi một thuật ngữ trong từ điển, một IF chứa một *danh sách đảo* **(**IL**)** lưu trữ một danh sách con trỏ tới tất cả xuất hiện của thuật ngữ đó trong văn bản chính, trong đó mỗi một con trỏ trong thực tế là số tài liệu mà thuật ngữ đó xuất hiện. IL đôi khi được coi là một *danh sách mục lục* và các con trỏ là *mục lục*. Đây là phương pháp chỉ mục tự nhiên nhất, gần tương ứng với chỉ mục của một cuốn sách và với cách dùng mục lục truyền thống.

Một IF yêu cầu một *từ vựng* - một danh sách tất cả thuật ngữ xuất hiện trong CSDL. Từ vựng trợ giúp một ánh xạ từ các thuật ngữ tới các IL tương ứng của chúng và ở dạng đơn giản nhất của nó là một danh sách các xâu và địa chỉ đĩa từ.

Bảng 2 - Văn bản mẫu; mỗi dòng là một tài liệu.

TÀI LIỆU	VĂN BẢN
1	Information retrieval is searching and indexing
2	Indexing is building an index
3	An inverted file is an index

4	Building an inverted file is indexing

Ví dụ về một IFID, xét văn bản mẫu ở bảng 2, với mỗi dòng được coi là một tài liệu để chỉ mục. IF sinh ra cho văn bản này được chỉ ra ở bảng 3, trong đó các thuật ngữ được gộp dạng nhưng không được truy gốc và không một từ nào bị bỏ qua. Nói chung, các IL cho một CSDL có độ dài rất khác nhau.

Một truy vấn bao gồm một thuật ngữ đơn được trả lời bằng cách quét IL của nó và truy tìm mọi tài liệu mà nó trích dẫn. Đối với truy vấn Boole hội có dạng t_1 AND t_2 AND ... AND t_n , giao của các IL của thuật ngữ được tạo ra. Đối với truy vấn tuyển, trong đó toán tử là OR, phép hợp được thực hiện; đối với truy vấn phủ định dùng NOT, phép bù được thực hiện. IL thường được lưu trữ để làm tăng số tài liệu, sao cho các thao tác trộn khác nhau có thể được thực hiện thời gian tuyến tính theo kích thước của danh sách. Chẳng hạn, để định vị các tài liệu chứa *index* và *indexing* ở văn bản của bảng 2, các danh sách đối với hai từ – (2;5), (3;6) và (1;6), (2;1), (4;6) tương ứng.

Bảng 3 - IF đối với văn bản của bảng 2.

Số	Thuật ngữ	IL(tài liệu; vị trí)
1	an	(2;4), (3;1), (3;5), (4;2)
2	and	(1;5)
3	building	(2;3), (4;1)
4	file	(3;3), (4;4)
5	index	(2;5), (3;6)
6	indexing	(1;6), (2;1), (4;6)
7	information	(1;1)
8	inverted	(3;2), (4;3)
9	is	(1;3), (2;2), (3;4), (4;5)
10	retrieval	(1;2)
11	searching	(1;4)

Tính kết hạt của một chỉ mục là tính chính xác để nhận dạng vị trí của thuật ngữ. Một chỉ mục kết hạt thô chỉ có thể nhận dạng một khối văn bản, trong đó mỗi một khối lưu trữ vài tài liệu; một chỉ mục kết hạt trung bình lưu trữ các vị trí trong phạm vi số tài liệu; trong khi một chỉ mục kết hạt tinh đưa lại một câu hoặc số từ, có thể thậm chí một số byte. Chỉ mục thô yêu cầu lưu trữ ít hơn, nhưng trong khi truy tìm phải quét nhiều hơn văn bản thuần tuý để tìm các thuật ngữ. Với chỉ mục thô, các câu hỏi nhiều thuật ngữ thường làm tăng các *so khớp sai*, trong đó mỗi một thuật ngữ yêu cầu xuất hiện nơi nào đó trong khối, không chỉ trong giới hạn tài liệu giống nhau. Ở thái cực khác, chỉ mục mức từ cho phép các truy vấn bao hàm lân cận - chẳng hạn, *digital library* như một cụm từ đúng hơn hai từ riêng biệt *digital* và *library* - được trả lời nhanh bởi vì quan hệ mong muốn có thể được kiểm tra trước khi văn bản được truy tìm. Tuy nhiên, thêm thông tin định vị chính xác mở rộng chỉ mục tối thiểu bằng một nhân tử hai hoặc ba so với một chỉ mục mức tài liệu vì không chỉ có nhiều hơn con trỏ trong chỉ mục, mà còn mỗi một con trỏ đòi hỏi nhiều bit lưu trữ hơn bởi vì nó chỉ ra vị trí chính xác hơn. Trừ khi một phần đáng kể các truy vấn được chờ đợi dựa vào lân cận, tính kết hạt bình thường là các tài liệu riêng biệt. Sau đó, các truy vấn dựa vào lân cận và cụm từ có thể được điều khiển bởi phương pháp quét hậu truy tìm chậm hơn chút ít.

Bảng 4 - IF mức từ đối với văn bản của bảng 2.

Số	Thuật ngữ	(Tài liệu; từ)
1	an	<4; (2;4), (3;1), (3;5), (4;2)>

2	and	<1; (1;5)>
3	building	<2; (2;3), (4;1)>
4	file	<2; (3;3), (4;4)>
5	index	<2; (2;5), (3;6)>
6	indexing	<3; (1;6), (2;1), (4;6)>
7	information	<1; (1;1)>
8	inverted	<2; (3;2), (4;3)>
9	is	<4; (1;3), (2;2), (3;4), (4;5)>
10	retrieval	<1; (1;2)>
11	searching	<1; (1;4)>

Bảng 4 chỉ ra văn bản của bảng 2 chỉ mục bởi số từ trong giới hạn số tài liệu, trong đó ký pháp ($x;y_1, y_2, ...$) chỉ thị từ cho trước xuất hiện trong tài liệu x bằng số từ $y_1, y_2, ...$ Để tìm các tài liệu chứa *index* và *indexing* cách nhau ít hơn hai từ, hai danh sách được trộn lại, nhưng lần này các cặp mục vào (một từ mỗi một danh sách) chỉ được chấp nhận khi số tài liệu giống nhau xuất hiện và các thành phần số từ khác ít hơn hai. Ở đây, không có một mục vào nào, như vậy không có một từ nào được đọc từ văn bản chính. IF thô hơn của bảng 3 cho hai so khớp sai, yêu cầu các dòng nhất định của văn bản được kiểm tra và loại bỏ.

Khi đó, chỉ mục lớn hơn. Có hai lý do: thứ nhất, có nhiều thông tin hơn được lưu trữ đối với mỗi một con trỏ - một số từ cũng như một số tài liệu - không ngạc nhiên thông tin vị trí chính xác hơn yêu cầu một mô tả dài hơn. Thứ hai, vài từ xuất hiện nhiều hơn một lần trong một dòng. Ở chỉ mục của bảng 3, sự xuất hiện trùng lặp được biểu diễn với một con trỏ đơn, nhưng ở chỉ mục mức từ của bảng 4, cả hai lần xuất hiện yêu cầu một mục vào. Một chỉ mục mức từ cần lưu trữ một trị cho mỗi một từ trong văn bản (F ở bảng 1), trong khi một chỉ mục mức tài liệu được lợi từ nhiều lần xuất hiện của chính một từ trong tài liệu và lưu trữ con trỏ ít hơn (*f* ở bảng 1).

Tổng quát hơn, một IF lưu trữ một tập địa chỉ có thứ tự - ở trường hợp cực đoan, 1 số từ ở 1 số câu trong 1 số đoạn trong 1 số chương trong 1 số tập bên trong 1 số tài liệu. Mỗi một vị trí thuật ngữ được coi là một vector ở dạng toạ độ. Tuy nhiên, trong mỗi một toạ độ danh sách địa chỉ thường được lưu trữ ở dạng minh hoạ ở bảng 4 và tất cả biểu diễn mô tả ở đây dễ dàng tổng quát hoá cho trường hợp nhiều chiều.

Do đó, từ nay về sau, giả sử chỉ mục là chỉ mục mức tài liệu đơn. Trên thực tế, căn cứ vào một tài liệu có thể được định nghĩa là một một đơn vị rất nhỏ, như một câu hoặc một đoạn, bằng cách nào đó chỉ mục mức từ chính là một trường hợp cực đoan, trong đó mỗi một từ được định nghĩa là một tài liệu.

IF không nén chiếm không gian đáng kể và có thể chiếm 50 đến 100% của chính không gian văn bản. Chẳng hạn, trong một bài văn tiếng Anh điển hình, từ trung bình chứa khoảng 5 ký tự và mỗi một từ thường kèm theo một hoặc hai dấu khoảng trống hoặc dấu chấm câu. Lưu trữ bằng số tài liệu 32 bit và giả sử không có bất kỳ sự trùng lặp từ bên trong tài liệu, như vậy, có thể có 4 byte của thông tin con trỏ IL đối với mỗi một 6 byte văn bản. Nếu một trường "số từ trong một tài liệu" 2 byte được cộng vào mỗi một con trỏ, chỉ mục tốn 6 byte cho khoảng từng 6 byte văn bản.

Đối với một văn bản có N tài liệu và một chỉ mục chứa *f* con trỏ, không gian tổng yêu cầu với biểu diễn thô là *f*.[logN] bit, giả sử con trỏ được lưu trữ bằng một số bit cực tiểu. Sử dụng con trỏ 20 bit để lưu trữ số tài liệu TREC cho IF 324 MB. Đây rõ ràng là một dạng nén so với số 32 bit tiện lợi hơn thường dùng khi lập trình, tuy thế, chỉ mục chiếm một phần

đáng kể của không gian lưu trữ văn bản. Đối với CSDL giống nhau, một IF mức từ sử dụng con trỏ 29 bit yêu cầu xấp xỉ 1200 MB.

Sự sử dụng một danh sách từ bỏ qua tiết kiệm đáng kể ở một IF không nén do các thuật ngữ thông thường chiếm một phần đáng kể của tổng số lần xuất hiện từ. Tuy nhiên, có nhiều cách tinh tế hơn để tiết kiệm được không gian như nhau và giữ lại tất cả thuật ngữ là từ chỉ mục. Cách tiếp cận thích hợp hơn của chúng ta là tất cả thuật ngữ được chỉ mục – cho dù, để xử lý truy vấn nhanh hơn, chúng chỉ bị bỏ qua khi có trong truy vấn.

3. CHỈ MỤC TỆP KÝ SỐ SFID

SFID là phương pháp chỉ mục khác. Sự tổ hợp nhất định của các trường hợp có thể xử lý truy vấn nhanh hơn IF, nhưng ở các tình huống giống nhau có khả năng đòi hỏi một lượng không gian lớn hơn. Tệp ký số (SF) đặc biệt thông dụng trong quá khứ bởi vì chúng hoàn toàn bị nén theo một nghĩa nào đó và như vậy, tốn ít không gian lưu trữ hơn IF không nén.

Ở đây, chúng tôi xét biểu diễn SF và so sánh với IF nén trong phạm vi của cả hai yêu cầu xử lý và giá lưu trữ.

Bảng 5 – Mã hoá chồng lên của tài liệu 2 đối với SF.

Thuật ngữ	Ký số thuật ngữ
indexing	0001 0000 1100 0100
is	0100 0100 0001 0000
building	0101 0011 0000 0000
an	0000 0100 0100 1100
index	1100 1000 0010 0000
Ký số bloc	1101 1111 1111 1110

SF là một phương pháp xác suất để chỉ mục văn bản. Mỗi một tài liệu có một *ký số* liên kết/ hoặc *bộ mô tả*, một xâu bit bắt nội dung tài liệu theo một nghĩa nào đó. Để tạo ra ký số cho một tài liệu, mỗi một thuật ngữ trong tài liệu được sử dụng để sinh ra một số giá trị băm và các bit của ký số tài liệu tương ứng với các giá trị băm được cài đặt cho nó.

Chẳng hạn, xét lại văn bản của bảng 2 và giả thiết các từ có ký số 16 bit trình bày ở bảng 5. Các ký số được xây dựng bằng cách lấy 3 hàm băm sinh ra các giá trị trong dải 1...16 và đặt các bit tương ứng vào ký số. Tức là, mỗi một từ được băm 3 lần dùng các hàm khác nhau và các bit chỉ thị như vậy là bit "1" ở ký số của từ đó. Những xung đột có thể dẫn đến ít hơn ba bit được cài đặt ở một số ký số thuật ngữ, nhưng không cần tính đến nó. Chẳng hạn, ở bảng 5 thuật ngữ *index* có 4 bit cài đặt. Mô tả các xâu bit như ký số là sự sử dụng rất thích hợp của từ - như với chữ ký con người.

Bây giờ, xét ký số tổng hợp khi các ký số của các từ ở mỗi một tài liệu được đặt chồng lên – tức là, hoặc đồng thời để làm ký số tài liệu. Chẳng hạn, tài liệu 2 của bảng 2 chứa 5 từ *indexing*, *is, building, an, index*, như vậy, ký số là 1101 1111 1111 1110.

Để thử nghiệm liệu một thuật ngữ truy vấn có xuất hiện ở một tài liệu đã cho, các giá trị của hàm băm cho thuật ngữ được tính toán. Nếu tất cả bit tương ứng trong bộ mô tả của tài liệu nào đó được cài đặt, thuật ngữ hầu như chắc chắn xuất hiện trong tài liệu đó. Hơn nữa, cho đến nay không thể nói thuật ngữ không xuất hiện trong tài liệu, dù tổ hợp các từ

khác nào đó có thể ngẫu nhiên cài đặt tất cả bit kiểm tra đối với thuật ngữ truy vấn. Để giải quyết sự không chắc chắn này, tài liệu phải được bắt và quét để kiểm tra rằng thuật ngữ thực sự xuất hiện. Xác suất của so khớp sai như thế có thể bị bắt ngẫu nhiên nhỏ bằng cách cài đặt một số bit cho mỗi một thuật ngữ và thực hiện ký số hiệu quả lớn, nhưng kiểm tra so khớp sai thường được đòi hỏi với các SFID và có thể thêm thực sự vào giá xử lý truy vấn vì mỗi một tài liệu kiểm tra phải được giải mã hoàn toàn, phân tích cú pháp thành các từ và các từ hoàn toàn được truy gốc.

SF trở nên hiệu quả hơn khi các truy vấn trở nên riêng biệt hơn bởi vì các truy vấn bao hàm sự kết hợp của một số thuật ngữ có thể kiểm tra nhiều bit hơn trong SF. Chỉ một bit cần bằng 0 để tạo ra sự so khớp sai và dẫn đến một xác suất nhỏ đối với so khớp sai khi số bit đã định rõ trong truy vấn lớn. Ngược lại, sử dụng một SF để xử lý các truy vấn thuật ngữ đơn đòi hỏi phải xử lý chậm, trong khi nhiều so khớp sai bị hạn chế, hoặc một SF lớn đến mức mỗi một thuật ngữ có thể cài đặt nhiều bit.

Chú ý rằng dù các truy vấn thuật ngữ đơn dường như hoàn toàn vô vọng trong một CSDL khổng lồ như Web, tuy nhiên chúng được sử dụng rất rộng rãi. Các truy vấn trung bình với các động cơ tìm kiếm Web chỉ chứa hai hoặc ba thuật ngữ.

Nếu truy vấn bao hàm các thuật ngữ phủ định, logic trở nên phức tạp hơn và có một sự đảo ngược qui luật đáng kể. Chẳng hạn, xét truy vấn NOT *index*. Ký số đối với *index* có bit 1, 2, 5 và 11 được cài đặt, như vậy, các tài liệu trong đó một trong số vị trí bit chỉ định không được cài đặt có thể được chấp nhận là câu trả lời không có kiểm tra sâu hơn bởi vì nếu thuật ngữ có mặt, bit nên được cài đặt.

4. ĐÁNH GIÁ VÀ KẾT LUẬN

SF có thể tạo ra các truy cập không cần thiết tới văn bản chính bởi vì các so khớp sai, nhưng chúng có thể được giảm không đáng kể nếu ký số là đủ lớn và một số bit thích hợp được cài đặt đối với từng thuật ngữ. Tương phản, nếu IL được truy cập theo độ dài tăng lên và từ vựng của CSDL được điều khiển trong bộ nhớ chính thì một IF yêu cầu truy cập đĩa tổng cộng không nhiều hơn so với một SF, ngoại trừ một ít trường hợp không hợp lý. Hơn nữa, các thao tác SF trở nên phức tạp nếu phép tuyển và phép phủ định được cho phép, SF không thể được sử dụng để trợ giúp các truy vấn xếp hạng. SF có thể đặc biệt tai hại khi độ dài bản ghi có thể thay đổi nhiều.

Đối với lựa chọn tham số điển hình, SFID lớn hơn 2 đến 3 lần IFID nén:

Hai hạn chế của hiệu năng SF - tốc độ chậm và kích thước chỉ mục yêu cầu lớn hơn - tạo ra một luận cứ đầy đủ ủng hộ IF nén đối với các ứng dụng bao gồm CSDL văn bản.

Ưu điểm lớn của SF là không cần duy trì một từ vựng trong bộ nhớ khi xử lý truy vấn. Ở một CSDL với một vốn từ rất giàu, một IFID phải yêu cầu một từ vựng, phần đáng kể của tổng kích thước chỉ mục và nếu truy cập nhanh được yêu cầu, từ vựng này cần được điều khiển trong bộ nhớ. Giá sử dụng một IF nhưng với từ vựng trên đĩa là một truy cập đĩa thêm cho từng thuật ngữ truy vấn; nếu truy vấn điển hình có trên mười hoặc trăm thuật ngữ, IF có thể yêu cầu truy cập đĩa nhiều hơn so với SF vì nó không bao giờ cần truy tìm nhiều hơn một bitslice cho từng thuật ngữ trên truy vấn SF nhiều thuật ngữ để làm giảm tỷ suất so khớp sai tới mức có thể chịu được. Như vậy, khi từ vựng quá lớn đối với bộ nhớ chính, SF có thể yêu cầu chỉ một nửa truy cập đĩa của IF.

Ở một số trường hợp, SF yêu cầu thời gian bộ xử lý nhỏ hơn trong khi giải quyết các truy vấn hội so với IF nén, đặc biệt là nếu phần cứng chuyên dụng có sẵn để thực hiện bitslice và các phép toán. Nhưng các trường hợp này không tăng thêm trong thực tế. Chẳng

hạn, đối với một truy vấn Boole xây dựng từ 33 thuật ngữ thông thường nhất trong TREC, một cơ chế IF nén phải giải mã và trộn gần 15 triệu số tài liệu, tốn 10 giây về thời gian xử lý. Ở truy vấn như nhau, SFID đơn giản truy tìm và cộng đồng thời hầu hết 33 bitslice, mỗi một trong khoảng 91 KB.

SF có thể thích hợp ở các ứng dụng trong đó độ dài bản ghi bị ràng buộc và vốn từ rất lớn, hoặc trong đó nhiều truy vấn bao gồm các thuật ngữ thông thường. Tuy nhiên, so với IF nén, SF không làm giảm không gian chỉ mục, SF cũng không đưa ra cách giải quyết nhanh hơn đối với các truy vấn hội điển hình. Hơn nữa, SF xử lý kém với các truy vấn không hội, không đưa ra ưu điểm đặc biệt nào đối với CSDL động và đòi hỏi các tài nguyên tương tự hoặc hơn trong khi tạo lập CSDL.

Các tổ hợp lai của IF và SF được đưa ra, trong đó phương pháp chỉ mục đối với mỗi một thuật ngữ dựa trên tần suất thuật ngữ. Đề xuất khác là lưu trữ một IF đầy đủ và từ đó, lấp đầy bất kỳ không gian đĩa còn lại bằng bitvector trên một đĩa chỉ cho phép đọc, mặt khác không gian bị lãng phí. Dù cho các sơ đồ lai như thế hoá ra đề nghị phương án tốt nhất của cả hai phương pháp, truy tìm trở nên cài đặt phức tạp hơn. Hơn nữa, một sơ đồ nén phù hợp tránh tất cả suy xét này vì các thuật ngữ thông thường được lưu trữ rất cô đặc. IF nén chắc chắn là phương pháp chỉ mục hữu ích nhất một CSDL lớn các tài liệu văn bản có độ dài có thể thay đổi.

Tóm lại, chúng ta rút ra quy luật chỉ mục tài liệu văn bản trong thư viện số là:

Ở hầu hết các ứng dụng, IF thực hiện tốt hơn SF trong phạm vi của cả hai kích thước chỉ mục và tốc độ truy vấn.

TÀI LIỆU THAM KHẢO

[1] Arms W.Y. *Digital Libraries*, MIT Press, Cambridge, 2003.

[2] Barth A., Breu M., Endres A., de Kemp A. *Digital Libraries in Computer Science*, Springer, Berlin, 1998.

[3] Fox E.A., *Advanced Digital Libraries*, Virginia Polytechnic Institue and State University, 2000.

[4] Journal of Network and Computer Applications, *Special Issue of JNCA on Digital Libraries* 20 (1-2), 1997.

[5] Lesk M. *Practical Digital Libraries*, Morgan Kaufmann, San Francisco, 1997.

[6] National Institute of Standards and Technology, *NIST Special Publication Text Retrieval Conference* (TREC).