TRƯỜNG ĐẠI HỌC QUỐC TẾ SÀI GÒN
THE SAIGON INTERNATIONAL UNIVERSITY

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO NGHIÊN CỨU KHOA HỌC

KỸ THUẬT TÁCH TỪ TRONG CÂU TIẾNG VIỆT VÀ ỨNG DỤNG TÌM KIẾM THÔNG TIN TRÊN WEBSITE

Giảng viên hướng dẫn: ThS. Đặng Văn Thành Nhân

Sinh viên thực hiện:

- Trần Văn Đan Trường – 91011801418
- Võ Phước Sang – 81011801421

TP. Hồ Chí Minh, 2020

MỤC LỤC

TÓM TẮT ĐỀ TÀI

Đề tài nghiên cứu: Kỹ thuật tách từ trong câu tiếng Việt và Ứng dụng trong tìm kiếm thông tin trên website.

Tóm tắt: Tìm hiểu các kỹ thuật tách từ tiếng Việt và lập chỉ mục cho văn bản. Lựa chọn phương pháp tối ưu để tìm kiếm thông tin. Áp dụng với bài toán tách từ và tìm kiếm thông tin tiếng Việt trên website.

DANH MỤC CÁC CHỮ VIẾT TẮT

IR	Information Retrieval
HTML	HyperText Markup Language
LSI	Latent Semantic Indexing
sim	Similar
SVD	Singular Value Decomposition
Tdf	Term document frequency
Tf	Term frequency
URL	Uniform Resource Locator
VSM	Vector Space Model
WWW	Word Wide Web
XML	eXtensible Markup Language

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH VẼ

MỞ ĐẦU

- Ngày nay, lịch sử nhân loại đã bước sang một trang mới nhờ sự bùng nổ của công nghệ thông tin. Những thành tựu của ngành công nghệ thông tin là vô cùng to lớn, nó đã chi phối và làm thay đổi mọi mặt của đời sống xã hội, làm cho cuộc sống của con người văn minh, hiện đại hơn. Sự ra đời của Internet chính là bước tiến vĩ đại của nhân loại, là yếu tố quan trọng bậc nhất chi phối cuộc sống của chúng ta ngày nay. Nhờ có Internet thế giới trở nên 'phẳng' hơn, ở mọi nơi trên trái đất chúng ta đều có thể học tập và tìm kiếm thông tin.
- Theo guồng quay của cuộc sống, thế giới Internet ngày càng rộng lớn và phong phú hơn. Cứ mỗi phút trôi qua có thêm hàng triệu trang web được sinh ra để làm giàu cho vốn tài nguyên tri thức của nhân loại. Tuy nhiên, một trong những khó khăn của con người gặp phải trong việc khai thác thông tin là khả năng tìm chính xác thông tin họ cần trên web. Để trợ gúp công việc này, các hệ thống tìm kiếm trên web đã lần lượt được phát triển nhằm phục vụ cho nhu cầu tìm kiếm thông tin của người sử dụng. Phổ biến nhất là các hệ thống tìm kiếm theo từ khóa. Hiện nay có nhiều hệ thống hoạt động hiệu quả trên Internet như Google, Cốc Cốc, Baidu, Yandex, Bing, Yahoo…Tuy nhiên, phần lớn các công cụ tìm kiếm này là những sản phẩm thương mại và mã nguồn được giữ bí mật. Việc tìm kiếm thông tin tiếng Việt trên web vẫn chưa chính xác cao. Do đó, nhu cầu phải có một công cụ tìm kiếm "hiểu" và xử lý tốt văn bản tiếng Việt trên web đang là chủ đề được nhiều người quan tâm.
- Mục tiêu của đề tài này nhằm xây dựng một hệ thống tìm kiếm thông tin bằng tiếng Việt trên web có sử dụng các kết quả của xử lý ngôn ngữ tự nhiên tự động để xác định các chỉ mục và xếp hạng tìm kiếm là các từ của tiếng Việt.

CHƯƠNG 1. TỔNG QUAN VỀ TÁCH TỪ TIẾNG VIỆT

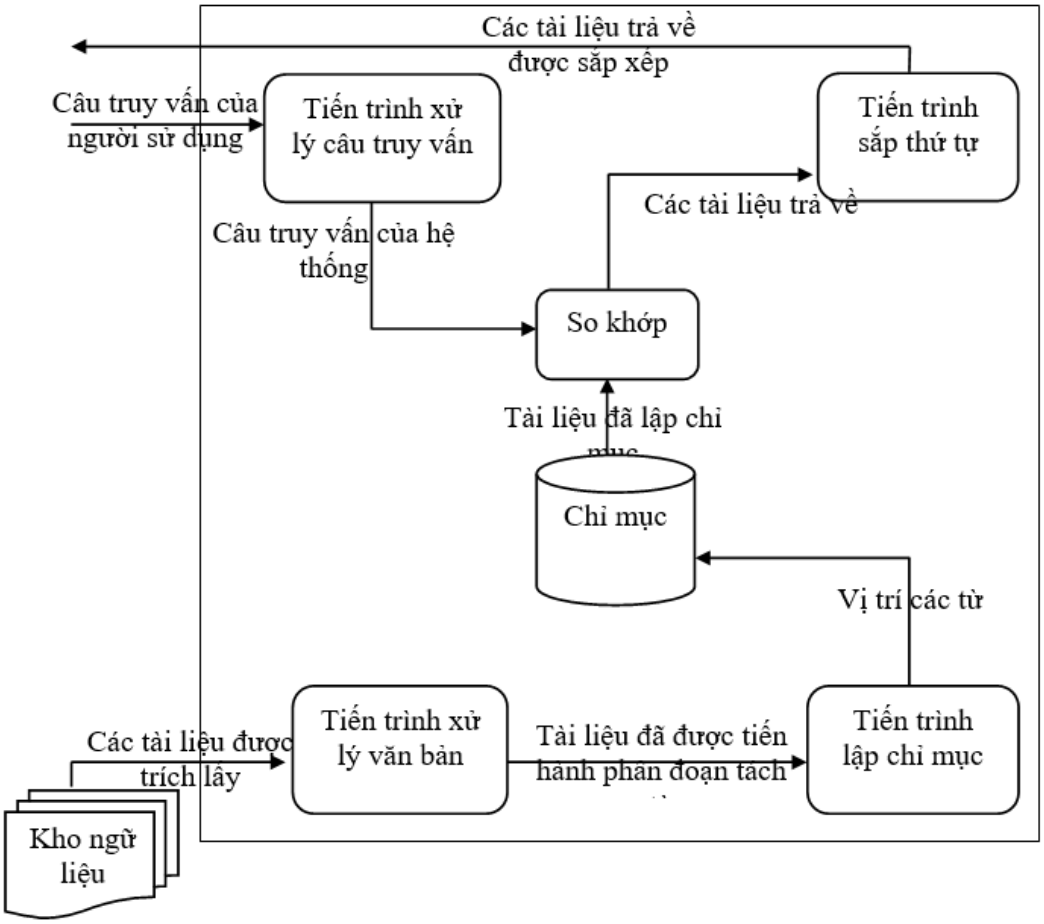
- Nội dung chương này nhằm giới thiệu tổng quan về tìm kiếm thông tin. Giới thiệu quy trình xây dựng một hệ thống tìm kiếm thông tin. Một số mô hình tìm kiếm thông tin trên Web phổ biến hiện nay. Đồng thời tóm lược một số khó khăn trong xây dựng hệ thống tìm kiếm thông tin tiếng Việt.

1.1. Giới thiệu về tìm kiếm thông tin

- Tìm kiếm thông tin Information Retrieval (IR) là tìm kiếm tài nguyên trên một tập lớn các dữ liệu phi cấu trúc được lưu trữ trên máy tính nhằm thỏa mãn nhu cầu về thông tin.
- Tìm kiếm thông tin là ngành khoa học liên quan đến việc phân tích, thiết kế và triển khai các hệ thống máy tính nhằm biểu diễn, tổ chức và truy cập khối lượng lớn thông tin được số hoá. Nền tảng của nó là khoa học thông tin (Information Science), nghiên cứu việc "tập hợp, tổ chức, lưu trữ, truy cập, phân loại thông tin".
- Mục đích của tìm kiếm thông tin là trả lại cho người dùng một tập các thông tin thỏa mãn nhu cầu của họ. Chúng ta định nghĩa rằng thông tin cần thiết là “câu truy vấn” (query) và các thông tin được chọn là “tài liệu” (documents). Mỗi cách tiếp cận trong tìm kiếm thông tin bao gồm hai phần chính: một là các kỹ thuật để biểu diễn thông tin (câu truy vấn, tài liệu) và hai là phương pháp so sánh các cách biểu diễn này. Mục đích là để thực hiện tự động qui trình kiểm tra các tài liệu bằng cách tính độ tương quan giữa các câu truy vấn và tài liệu. Quy trình này thành công khi nó trả về các kết quả được người dùng tạo ra khi so sánh câu truy vấn với các tài liệu.
- Các nghiên cứu trong lĩnh vực IR nhắm tới mục tiêu nâng cao chất lượng của các giai đoạn trong tìm kiếm, bao gồm 1) Tiếp nhận và phân tích yêu cầu từ người dùng; 2) Thực hiện việc tìm kiếm; và 3) Gửi trả kết quả cho người dùng. Các mô hình khác nhau được nghiên cứu, xây dựng nhằm tái biểu diễn câu truy vấn và tài liệu tìm kiếm, sau đó áp dụng các chiến lược tìm kiếm phù hợp.

1.1.1. Quy trình xây dựng hệ thống tìm kiếm thông tin

- Cách hoạt động cơ bản của một hệ thống tìm kiếm thông tin cổ điển.

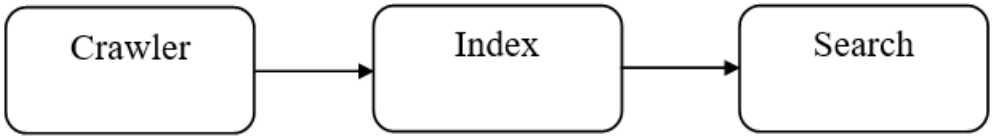


Hình 1.1: Mô hình hoạt động của hệ thống tìm kiếm thông tin

+ Ở giai đoạn đầu tiên, giai đoạn tiền xử lý, tài liệu thô của ngữ liệu được xử lý thành các tài liệu được tách từ, phân đoạn (tokenized documents) và sau đó được lập chỉ mục thành một danh sách các vị trí của từ (postings per terms).
+ Ở giai đoạn thứ hai, người sử dụng đưa ra một câu truy vấn (phi cấu trúc bằng ngôn ngữ tự nhiên) mô tả nhu cầu thông tin của họ. Hệ thống tìm kiếm thông tin sẽ biểu diễn câu truy vấn này thành những câu truy vấn có hoặc không có cấu trúc mà máy có thể hiểu được. Hệ thống tìm kiếm thông tin bắt đầu thực hiện đối chiếu để tìm ra các tài liệu, các yếu tố thông tin có thể trả lời và liên quan đến câu truy vấn.
+ Cuối cùng, các tài liệu, yếu tố thông tin tìm thấy được hiển thị thành một danh sách tài liệu và được sắp xếp theo thứ tự liên quan (ranked retrieved documents). Thông thường, những tài liệu, yếu tố thông tin có liên quan nhiều nhất được xếp trên những tài liệu ít liên quan hơn.

1.1.2. Các bộ phận cấu thành của hệ thống tìm kiếm thông tin

- Một hệ thống tìm kiếm thông tin hoạt động trên môi trường mạng thông tin (Internet) hay trên môi trường máy tính cá nhân (PC) đều gồm có thành phần chính sau:



Hình 1.2: Các bộ phận của máy tìm kiếm

+ Bộ thu thập thông tin – Crawler: đây là một chương trình tự động duyệt qua các cấu trúc siêu liên kết để thu thập tài liệu và một cách đệ quy nó nhận về tất cả các tài liệu có liên kết với tài liệu này. Đối với hệ thống tìm kiếm trên máy PC, người dùng thường chỉ định dữ liệu có sẵn trên máy tính. Robot được biết đến dưới nhiền tên gọi khác nhau: spider, web wandeere hoặc web worm…
+ Bộ lập chỉ mục – Index: thực hiện việc phân tích, trích chọn những thông tin cần thiết (thường là các từ đơn, từ ghép, cụm từ quan trọng) từ những dữ liệu mà robot thu thập được hoặc do người dùng chỉ định và tổ chức thành cơ sở dữ liệu riêng để có thể tìm kiếm trên đó một cách nhanh chóng, hiệu quả. Hệ thống chỉ mục là danh sách các từ khóa, chỉ rõ các từ khóa nào xuất hiện ở trang nào, địa chỉ nào…
+ Bộ tìm kiếm thông tin – Search Engine: bộ tìm kiếm tương tác với người sử dụng thông qua giao diện giao tiếp, có nhiệm vụ tiếp nhận và trả về những tài liệu thỏa yêu cầu người dùng. Bộ tìm kiếm tiếp nhận yêu cầu của người dùng, thực hiện một số biến đổi như sửa lỗi chính tả, chuẩn hóa từ khóa,…sau đó thực hiện so khớp với cơ sở dữ liệu do bộ lập chỉ mục tạo ra để lọc lấy danh sách các tài liệu thỏa mãn tốt nhất cho người dùng.

1.1.3. Các bước xây dựng hệ thống tìm kiếm thông tin

- Xây dựng một hệ thống tìm kiếm thông tin sẽ thực hiện qua 4 bước như sau: Tách từ tự động cho tập tài liệu, Lập chỉ mục cho tài liệu, Tìm kiếm, Sắp xếp các tài liệu trả về.

+ **Bước 1: Tách từ tự động cho tập các tài liệu**
  - Tách từ trong tiếng Anh chỉ dựa vào khoảng trắng. Tuy nhiên đối với tiếng Việt, giai đoạn tách từ này tương đối khó khăn. Cấu trúc ngữ pháp tiếng Việt rất phức

tạp, không chỉ đơn thuần dựa vào khoảng trắng để tách từ. Hiện nay có rất nhiều công cụ dùng để tách từ tiếng Việt, mỗi phương pháp có ưu điểm, nhược điểm riêng. Các phương pháp này sẽ được trình bày ở chương 3 mục 3.1.

+ **Bước 2: Lập chỉ mục cho tài liệu**

- Sau khi có được tập các từ đã được trích, ta sẽ chọn các từ để làm từ chỉ mục. Tuy nhiên, không phải từ nào cũng được chọn làm từ chỉ mục. Các từ có khả năng đại diện cho tài liệu sẽ được chọn, các từ này được gọi là keyword. Do đó, trước khi lập chỉ mục sẽ là giai đoạn tiền xử lý đối với các từ trích được để chọn ra các key word thích hợp. Ta sẽ loại bỏ danh sách các từ ít có khả năng đại diện cho nội dung văn bản dựa vào danh sách gọi là stop list. Đối với tiếng Anh hay tiếng Việt đều có danh sách stop list.

+ **Bước 3: Tìm kiếm và sắp xếp các tài liệu trả về**

- Đây là quá trình người dùng nhập câu hỏi và yêu cầu tìm kiếm, câu hỏi mà người dùng nhập vào cũng sẽ được xử lý, nghĩa là ta cũng sẽ thực hiện tách từ cho câu hỏi. Phương pháp tách từ cho câu hỏi cũng nên là phương pháp tách từ cho các tài liệu thu thập được để đảm bảo sự tương thích. Sau đó, hệ thống sẽ tìm kiếm trong tập tin chỉ mục để xác định các tài liệu liên quan đến câu hỏi của người dùng.
- Các tài liệu sau khi đã xác định là liên quan đến câu hỏi của người dùng sẽ được sắp xếp lại, bởi vì trong các tài liệu đó có những tài liệu liên quan đến câu hỏi nhiều hơn. Hệ thống sẽ dựa vào một số phương pháp để xác định tài liệu nào liên quan nhiều nhất, sắp xếp lại và trả về cho người dùng theo thứ tự ưu tiên.

1.2. Một số mô hình xây dựng hệ thống tìm kiếm thông tin

- Mục tiêu của các hệ thống tìm kiếm thông tin là trả về các tài liệu càng liên quan đến câu hỏi càng tốt. Vì thế đã có rất nhiều mô hình tìm kiếm thông tin nhằm tính toán một cách chính xác độ tương quan này. Sau đây là một số mô hình tìm kiếm thông tin cơ bản.

1.2.1. Mô hình tìm kiếm Boolean

- Đây là mô hình cơ bản và đơn giản dựa trên đại số Bool, sử dụng nguyên tắc so sánh chính xác khi tìm kiếm tài liệu. Hệ thống yêu cầu người sử dụng cung cấp câu truy vấn dưới hình thức là các từ khoá kèm theo các toán tử AND, OR, NOT.
- Mô hình vùng (Regions models) là một mở rộng của mô hình Boolean. Coi bộ sưu tập tài liệu như một chuỗi từ liên tục, mỗi chuỗi tuỳ ý các từ nối tiếp là một vùng. Các toán tử CONTAINING, CONTAINED_BY hay FOLLOWED_BY được bổ sung để so sánh tài liệu và yêu cầu.
- Điểm hạn chế lớn nhất của mô hình Boolean và mô hình vùng là chúng không hỗ trợ việc xếp hạng các tài liệu, không xử lý được vấn đề đồng nghĩa và đa nghĩa, có cú pháp phức tạp và dễ gây nhầm lẫn.

❖ Ưu điểm của mô hình Boolean:

- Đơn giản và dễ sử dụng.

❖ Nhược điểm của mô hình Boolean:

- Vì dựa trên phép toán logic nhị phân nên một văn bản được tìm kiếm chỉ xác định hai trạng thái: liên quan hoặc không với câu truy vấn. Số lượng văn bản trả về tuỳ thuộc vào số từ xuất hiện của câu truy vấn có liên quan hay không;
- Chuyển một câu truy vấn sang dạng boolean là không đơn giản;
- Văn bản trả về không được quan tâm đến thứ tự quan hệ với câu truy vấn.

1.2.2. Mô hình tính điểm và trọng số cho mục từ - Term weight

- Mô hình tìm kiếm Boolean chỉ trả về giá trị logic là có hoặc không có trong tài liệu tìm kiếm, kết quả trả về không có thứ hạng. Vai trò của các mục từ tìm kiếm là ngang nhau, chỉ xác định là có xuất hiện hay không trong tài liệu. Điều này đã dẫn đến kết quả tìm kiếm không được như mong muốn của người dùng. Để cải tiến mô hình này, người ta áp dụng cách tính điểm cho kết quả trả về, dựa trên trọng số của mục từ trên tài liệu.
- Mỗi mục từ trong ma trận từ chỉ mục được gán một trọng số, giá trị này phụ thuộc vào số lần xuất hiện của mục từ trên tài liệu chứa mục từ và tập tài liệu.
- Tính kết quả độ liên quan của câu truy vấn trên từng văn bản và sau đó sắp xếp kết quả trả về.

❖ Ưu điểm của mô hình tính điểm trọng số:

- Sử dụng trọng số cho từ chỉ mục khác trọng số nhị phân (non-binary). Trọng số từ chỉ mục không giới hạn bởi hai trị 0 hoặc 1, các trọng số này được sử dụng để tính toán độ đo tương tự của mỗi văn bản với câu truy vấn. Kết quả trả về có quan tâm đến thứ tự xuất hiện.

❖ **Nhược điểm của mô hình tính điểm và trọng số:**

- Kết quả tính trọng số chưa xét vai trò của các mục từ trong câu truy vấn. Có thể số lượng các mục từ như nhau nhưng vai trò khác nhau hoàn toàn.

1.2.3. Mô hình không gian vector – Vector Space Model (VSM)

- Mô hình không gian vector tính toán độ tương quan giữa câu hỏi và tài liệu bằng cách định nghĩa một vector biểu diễn cho mỗi tài liệu, và một vector biểu diễn cho câu hỏi [Salton, 1875].
- Trong đó, tài liệu và câu truy vấn được biểu diễn dưới dạng các vector. Một văn bản d được biểu diễn như một vector của các từ chỉ mục $d = (t_1,t_2,...,t_n)$ với t_i là từ chỉ mục thứ i ($1 \leq i \leq n$) (các giá trị có thể là số lần xuất hiện của term t_i trong văn bản d). Mỗi từ chỉ mục trong văn bản biểu diễn một chiều (dimension) trong không gian. Tương tự, câu truy vấn cũng được biểu diễn như một vector $q = (t_1,t_2,...,t_n)$.
- Sau khi đã biểu diễn tập văn bản và câu truy vấn thành các vector trong không gian vector, sử dụng độ đo cosin để tính độ đo tương tự giữa các vector văn bản và vector truy vấn, kết quả sau khi tính toán có thể được xếp hạng theo độ đo tương tự với vector truy vấn.

❖ **Ưu điểm của mô hình không gian vector:**

- Đơn giản, dễ hiểu;
- Đưa ra khái niệm phù hợp một phần; công thức xếp hạng cosin cho phép đồng thời xác định sự phù hợp và phục vụ sắp xếp danh sách kết quả..

❖ **Nhược điểm mô hình không gian vector**:

- Số chiều biểu diễn cho tập văn bản có thể rất lớn nên tốn nhiều không gian lưu trữ;
- Các văn bản trả về tuy cải thiện là có quan tâm đến việc xếp hạng các văn bản theo mức độ liên quan nhưng vẫn không có quan hệ về ngữ nghĩa với câu truy vấn. Các từ khoá được giả định độc lập và do đó mô hình không đánh giá được ngữ nghĩa của câu truy vấn tài liệu.

1.2.4. Mô hình xác suất – Probabilistic model

- Cho câu truy vấn của người dùng q và văn bản d trong tập văn bản. Mô hình xác suất tính xác suất mà văn bản d liên quan đến cấu truy vấn của người dùng. Mô hình giả thiết xác suất liên quan của một văn bản với câu truy vấn phụ thuộc cách biểu diễn chúng. Tập văn bản kết quả được xem là liên quan và có tổng xác suất liên quan với câu truy vấn lớn nhất.
- Trong mô hình này, câu hỏi được đặt ra là "Với xác suất nào, một tài liệu là phù hợp với một câu truy vấn". Từ năm 1960, Bill Maron and Larry Kuhns định nghĩa mô hình chỉ mục xác suất [[10]. Việc lập xác xuất P(T|D) của thuật ngữ T chứa trong tài liệu D ban đầu được lập bằng tay. Gọi P(D|T) là xác suất tài liệu D thoả mãn so với thuật ngữ T, luật Bayes được áp dụng như sau:

$$P(D|T) = \frac{P(T|D)P(D)}{P(T)} \qquad (1.1)$$

- Trong công thức, P(T) cố định ứng với một thuật ngữ T; P(D) được coi là xác suất ban đầu được xác định trước từ việc sử dụng tài liệu; giá trị P(T|D) được xác định trong quá trình xem xét tài liệu D và lập chỉ mục.

❖ **Ưu điểm của mô hình xác suất**:

- Văn bản được sắp xếp dựa vào xác suất liên quan đến câu truy vấn.

❖ **Nhược điểm mô hình xác suất:**

- Mô hình không quan tâm đến số lần xuất hiện của từ chỉ mục trong văn bản.
- Việc tính toán xác suất khá phức tạp và tốn nhiều chi phí.
- Mô hình không hỗ trợ cho việc biểu diễn tài liệu, việc này được thực hiện trong một quá trình độc lập.

1.2.5. Mô hình chỉ mục ngữ nghĩa ngầm – LSI

- Latent Semantic Indexing (LSI) là phương pháp tạo chỉ mục tự động dựa trên khái niệm để khắc phục hai hạn chế tồn tại trong mô hình không gian vector chuẩn về hai vấn đề đồng nghĩa (synoymy) và đa nghĩa (polysemy) [[11]. Với synoymy, nhiều từ có thể được sử dụng để biểu diễn một khái niệm, vì vậy hệ thống không thể trả về những văn bản liên quan đến câu truy vấn của người dùng khi họ sử dụng những từ trong câu truy vấn đồng nghĩa với những từ trong văn bản. Với polysemy, một từ có thể có nhiều nghĩa, vì vậy hệ thống có thể trả về những văn bản không liên quan. Điều

này thực tế rất thường xảy ra bởi vì các văn bản trong tập văn bản được viết bởi rất nhiều tác giả, với cách dùng từ rất khác nhau. Một cách tiếp cận tốt hơn cho phép người dùng truy vấn văn bản dựa trên khái niệm (concept) hay nghĩa (meaning) của văn bản.

- Mô hình LSI cố gắng khắc phục hai hạn chế trên trong mô hình không gian vector bằng cách chỉ mục khái niệm được tạo ra bởi phương pháp thống kê (phân tích SVD ma trận term – document A) thay cho việc sử dụng các từ chỉ mục đơn. Mô hình LSI dựa trên giả thiết là có các ngữ nghĩa ngầm (latent semantic) trong việc sử dụng từ: có nhiều từ biểu diễn cho một khái niệm và một khái niệm có thể được biểu diễn bởi nhiều từ.

❖ Ưu điểm mô hình chỉ mục ngữ nghĩa ngầm:

- Latent Semantic Indexing (LSI) là phương pháp tạo chỉ mục tự động dựa trên khái niệm để khắc phục hạn chế tồn tại trong mô hình không gian vector về hai vấn đề đồng nghĩa (synoymy) và đa nghĩa (polysemy) [[11].
- Việc giảm số chiều cải thiện đáng kể chi phí lưu trữ và thời gian thực thi.

❖ Nhược điểm mô hình chỉ mục ngữ nghĩa ngầm:

- Việc tìm kiếm cũng phải quyét qua tất cả các cột trong ma trận LSI nên cũng tốn chi phí và thời gian.

1.3. Một số hệ thống tìm kiếm thông tin hiện nay

1.3.1. Google Search

- Google đứng đầu danh sách Search Engine trên toàn cầu. Không chỉ ở Việt Nam mà trên toàn thế giới, Google luôn là lựa chọn hàng đầu khi một ai đó muốn tra cứu thông tin trên internet.
- Ngoài google.com là chung cho toàn thế giới bên cạnh đó thì mỗi quốc gia google điều có một tên miền riêng để tìm kiếm tập trung theo quốc gia đó. Với google ta có thể tìm kiếm web, hình ảnh, video, tin tức, map và nhiều chức năng khác nữa. Theo thống kê của ComCore trong năm 2015 thì Google chiếm tỷ lệ lớn khối lượng tìm kiếm trên toàn thế giới với hơn 63.9%.

1.3.2. Bing và Yahoo

- Bing là cái tên mới của MSN.com. Với tên miền mới và nhiều cải tiến vượt bậc trong những năm gần đây nhưng Bing.com vẫn chưa thể so sánh với Google về Search Engines.
- Yahoo là một trong những hệ thống tìm kiếm lâu đời. Tuy nhiên thời gian gần đây, Yahoo gần như không có nhiều cập nhật cho công cụ tìm kiếm.

1.3.3. Cốc Cốc

- Cốc Cốc là hệ thống mới và phổ biến ở Việt Nam hiện nay với sản phẩm nổi tiếng là trình duyệt web Coc Coc. Bên cạnh đó thì mảng quan trọng nữa là Search Engines thuần Việt để cạnh tranh với các đối thủ khác từ nước ngoài như Google. Cốc Cốc ngoài chức năng tìm kiếm web như các công cụ khác thì còn có chức năng tìm theo Toán học, Ngôi sao, Nhà nhà. Toán học thì chuyên giải các bài toán từ lớp 1 -12 đây là địa chỉ không thể thiếu cho học sinh tự học và tìm hiểu về toán. Ngôi sao là chuyên trang tìm kiếm các ngôi sao trong làng giải trí trong và ngoài nước với hình ảnh HD và thông tin tổng quát về ngôi sao. Nhà nhà tương tự như tìm kiếm Map nhưng thế mạnh ở đây là thuần Việt nên mọi sẽ được tìm kiếm chi tiết hơn Google. Đó cũng là những thế mạnh đã làm nên sự khác biệt để cạnh tranh với Google, bằng chứng cho thấy Cốc Cốc đang đứng vị trí số 1 tại Việt Nam theo thống kê của Alexa.

1.3.4. Một số hệ thống tìm kiếm thông tin khác

- Ask.com là Search Engines không giống như Google, Bing hay Yahoo mà nó chuyên về mảng tìm kiếm giải đáp câu hỏi của người sử dụng. Kết quả trả về cũng đa dạng có cả hình ảnh, video và các câu gợi ý trả lời.
- Yandex là máy tìm kiếm của Nga. Các chức năng chính như tìm kiếm Web, Image, Video, Map và có cả trình duyệt web Yandex Browser.
- Baidu là máy tìm kiếm lớn nhất của Trung Quốc và chiếm lĩnh thị phần tìm kiếm. Các chức năng nổi bật như tìm kiếm web, danh bạ web, map…

1.4. Khó khăn trong xây dựng một hệ thống tài liệu thông tin tiếng Việt

1.4.1. Khó khăn trong việc tách từ tiếng Việt

- Tách từ là giai đoạn khó khăn nhất khi xây dựng một hệ thống tìm kiếm thông tin tiếng Việt. Đối với tiếng Anh, việc xác định từ chỉ đơn giản dựa vào khoảng trắng để tách từ.
- Ví dụ. Câu: "I am a student" sẽ được tách thành 4 từ: I, am, a, student.
- Tuy nhiên, đối với tiếng Việt, tách từ dựa vào khoảng trắng chỉ thu được các tiếng. Từ có thể được ghép từ một hay nhiều tiếng. Từ phải có ý nghĩa hoàn chỉnh và có cấu tạo ổn định.
- Ví dụ. Câu: "Tôi là một sinh viên" được tách thành 4 từ: Tôi, là, một, sinh viên. Trong đó, từ "sinh viên" được hình thành từ 2 tiếng: "sinh" và "viên"
- Hiện nay, có rất nhiều phương pháp được sử dụng để tách từ tiếng Việt. Tuy nhiên, với sự phức tạp của ngữ pháp tiếng Việt nên chưa có phương pháp nào đạt được chính xác 100%. Và việc lựa chọn phương pháp nào là tốt nhất cũng đang là vấn đề tranh cãi.

1.4.2. Khó khăn về bảng mã tiếng Việt

- Không như tiếng Anh, tiếng Việt có rất nhiều bảng mã đòi hỏi phải xử lý. Một số công cụ tìm kiếm tiếng Việt hỗ trợ bảng mã rất tốt như Vinaseek, hỗ trợ mọi bảng mã (VNI, TCVN3, ViQR,…)

1.4.3. Một số khó khăn khác

- Tiếng Việt có các từ đồng nghĩa nhưng khác âm. Các công cụ hiện nay không hỗ trợ việc xác định các từ đồng nghĩa. Vì vậy kết quả trả về sẽ không đầy đủ.
- Ngược lại, có những từ đồng âm khác nghĩa. Các hệ thống sẽ trả về các tài liệu có chứa các từ đã được tách trong câu hỏi mà không cần xác định chúng có thực sự liên quan hay không. Vì vậy, kết quả trả về sẽ không chính xác.
- Một số từ xuất hiện rất nhiều nhưng không có ý nghĩa trong tài liệu. Các từ như: và, với, nhưng,…có tần số xuất hiện rất lớn trong bất cứ văn bản nào. Nếu tìm cách trả về các tài liệu có chứa những từ này sẽ thu được kết quả vô nghĩa, không cần thiết. Do đó, chúng ta cần tìm cách loại bỏ những từ này trước khi tìm kiếm.

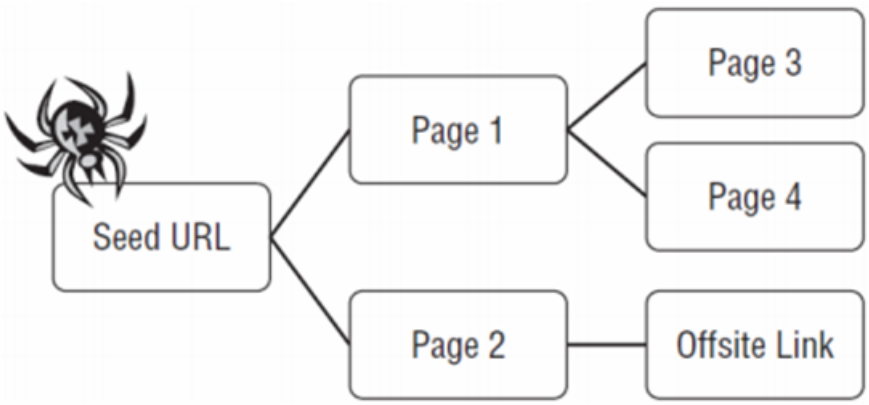
CHƯƠNG 2. QUY TRÌNH XÂY DỰNG HỆ THỐNG TÌM KIẾM THÔNG TIN TÁCH TỪ TIẾNG VIỆT

- Trong chương này, đề tài lần lượt trình bày các khái niệm cơ bản trong trình thu thập web. Phần lớn nội dung trong chương sẽ đi sâu vào tìm hiểu các thành phần cấu thành một trình thu thập thông tin, một số thuật toán Crawler hay áp dụng để thu thập dữ liệu.

2.1. Giới thiệu về Crawler

- Trình thu thập web (Web crawler) là một chương trình khai thác cấu trúc đồ thị của web di chuyển từ trang này qua trang khác. Thời kỳ đầu nó có những tên khá tượng hình như bọ web, rô-bốt, nhện và sâu, nhưng ngày nay tên gọi phổ biến nhất là vẫn là Crawler. Mặc dù vậy cụm từ 'thu thập' không lột tả được hết tốc độ của những chương trình này, vì chúng có tốc độ làm việc đáng kinh ngạc, có thể thu thập dữ liệu lên đến hàng chục ngàn trang trong vòng một vài phút.
- Nguyên lý hoạt động của một Crawler rất đơn giản, nó xuất phát từ những trang đầu tiên cho trước gọi là hạt giống (seed pages), và duyệt từ trang này đến trang khác thông qua những liên kết chứa trong những trang mà nó đi qua, quá trình này gọi là Crawling. Như vậy, Crawler sẽ duyệt vòng quanh và ngày một trải rộng phạm vi ra trên toàn bộ những Web Site trên Internet. Crawler tổng hợp nội dung (văn bản và những liên kết) từ những Web Site và lưu chúng vào trong cơ sở dữ liệu, lập chỉ mục và đánh giá PageRank cho những trang này dựa vào các thuật toán mà mỗi máy tìm kiếm sử dụng.
- Khi một Crawler được đưa lên trên môi trường Web, thông thường nó sẽ được khởi tạo bởi một vài trang Web. Việc đầu tiên mà Crawler sẽ làm trên những trang này là thu nhận tất cả các liên kết trên các trang Web, sau đó là đọc nội dung của các trang này và lần theo những liên kết mà Crawler đã tổng hợp trước đó để bắt đầu một quá trình thu thập mới. Những liên kết được tổng hợp từ những trang Web được đưa vào một khu vực chứa gọi là Crawl Frontier; các liên kết được đưa vào Crawl Frontier một cách có hệ thống theo một thuật toán mà mỗi Crawler sử dụng.
- Những liên kết trong Crawler Frontier đôi khi cũng hướng Crawler đến những trang Web mà Crawler đã duyệt qua trước đó, đây là một vấn đề phải giải quyết khi tiến hành phát triển một Crawler. Crawler sẽ duyệt qua những liên kết này cho đến khi nó

gặp một trang không chứa liên kết nào để tiếp tục (dead end) và rồi quay lùi về lại rồi tiếp tục duyệt tiếp cho đến khi tất cả các liên kết trong Web Site được duyệt qua hết. Hình 2.1 sẽ giúp hình dung đường đi của một Crawler.



Hình 2.1: Hành trình của Crawler

2.2. Cơ bản về hoạt động của Crawler

- Công việc của Crawler khá là phức tạp hơn rất nhiều so với những gì mà chúng ta gọi là "đọc" một Site. Crawler gửi một yêu cầu (Request) đến máy chủ chứa trang Web mà nó cần. Yêu cầu này sẽ được máy chủ đối xử tương tự như những yêu cầu của các trình duyệt (Browser) mà chúng ta vẫn thường dùng. Điểm khác biệt duy nhất giữa trình duyệt và Crawler chính là: Crawler chỉ lấy về nội dung các trang mà nó yêu cầu dưới dạng văn bản thuần (text-only). Crawler sẽ bỏ qua tất cả các nội dung thuộc định dạng đồ họa và những loại tập tin đa phương tiện khác (media file) như tập tin video, audio. Do đó thời gian để một Crawler thu về nội dung của một trang là nhanh hơn và cũng tốn ít băng thông mạng hơn rất nhiều so với một trình duyệt.
- Thông thường, không phải tất cả những tác vụ của một Crawler đều tiến hành một cách thuận lợi. Trên thực tế, khi người sử dụng mở một trình duyệt, nhập vào một URL và chờ đợi kết quả trả về. Có thể người dùng sẽ phải đợi một khoảng thời gian rất lâu để có thể xem được nội dung của trang Web đang duyệt, và cũng có thể họ sẽ không đủ kiên nhẫn để đợi cho đến khi trình duyệt hoàn thành việc nạp trang đó. Khi ấy người dùng sẽ chọn giải pháp là chuyển qua duyệt một trang khác. Với Crawler cũng vậy, nó cũng sẽ gặp trường hợp máy chủ mà nó gửi yêu cầu đến không trả lại yêu cầu trong một khoảng thời gian cho phép. Tại thời điểm đó, Crawler sẽ chuyển qua duyệt một liên kết khác và ghi nhớ lại liên kết mà nó đang gửi yêu cầu hiện tại.

Sau một thời gian, Crawler sẽ gửi yêu cầu trở lại đến trang Web đã ghi nhớ, nếu như tình trạng chờ đợi vẫn tiếp diễn thì sự việc chuyển qua một liên kết khác và quay trở lại sẽ lặp đi lặp lại theo một số lần nhất định. Nếu như vượt quá số lần này thì máy chủ đó sẽ được Crawler đưa ra một hình thức đối xử phù hợp mà mỗi nhà phát triển Crawler quy định, có thể là nó sẽ bị đưa vào blacklist, và sẽ không bao giờ xuất hiện trong hệ thống máy tìm kiếm của Crawler này.

- Do đó, nếu một Web Site mà tại thời điểm hiện tại có một số trang làm tốn rất nhiều thời gian để một Crawler tải về, cũng như một số lý do khác như là: Site hiện tại đang trong giai đoạn xây dựng, hoặc là nội dung không còn phù hợp với hiện tại… Chính vì vậy, Site đó không muốn Crawler viếng thăm những khu vực như vậy trong Site của mình. Có một quy ước được đề ra để Crawler hợp tác tốt hơn với những Site mà nó đi đến, chúng ta có thể gọi là những hướng dẫn cho một Crawler khi nó viếng thăm một Site. Những chỉ dẫn này được gọi với những cái tên như là Robot Exclusion Standard, Robot Exclusion Protocol, và được chỉ ra trong tập tin Robot.txt.

2.2.1. Tập tin Robot.txt

- Tập tin Robot.txt là một tệp định dạng văn bản thuần túy, nhằm chứa những khai báo về sự giới hạn và sự cho phép truy cập đối với một Crawler.
- Tất cả những chỉ dẫn đó được chứa trong tập tin với tên là robot.txt, và đây là nơi mà Crawler sẽ phải đọc đầu tiên khi tiến hành duyệt qua một Site nào đó. Nội dung của tập tin này tương tự như sau:

  *User-agent: **

  Disallow: /

- Trên là hai dòng thiết yếu của một tập tin robot.txt. Phần đầu tiên, *Useragent:* thông báo cho Crawler biết là loại Crawler nào sẽ áp dụng những điều luật bên dưới. Dấu (*) có nghĩa là sẽ áp dụng cho mọi Crawler. Dòng thứ hai Disallow: nêu ra phần nào trong một Site mà Crawler không được truy cập vào. Dấu (/) có nghĩa là mọi đường dẫn trong Site này đều không cho phép truy cập.
- Tập tin robot.txt phải luôn có dấu (:) đứng đằng sau những chỉ dẫn Useragent và Disallow. Nó chỉ ra rằng, đằng sau là những thông tin quan trọng mà Crawler sẽ phải quan tâm.

- Hiếm khi một Crawler lờ đi tất cả mọi đường dẫn trong một Web Site. Thay vào đó, sẽ có một số phần đặt biệt thay vì sử dụng dấu (/). Ví dụ như thư mục tạm thời trong Site, chỉ dẫn sẽ như sau:

  *User-agent: **

  Disallow: /tmp/
- Một ví dụ khác:

  *User-agent: **

  Disallow: /

  Disallow: /private/

  Disallow: /links/listing.html
- Nếu chúng ta muốn áp dụng chỉ dẫn này cho nhiều Crawler, việc cần làm là đưa tên của những Crawler lên phía trên của tập tin robot.txt. Ví dụ:

  User-agent: CrawlerName

  Disallow: /

  Disallow: /private/

  Disallow: /links/listing.html

  *User-agent: **

  Disallow: /tmp/
- Mỗi Crawler của một máy tìm kiếm được đặt bởi một tên khác nhau. Bên dưới là một số Crawler của các máy tìm kiếm nổi tiếng hiện nay:
  + Google: Googlebot
  + Bing: Bingbot
  + Yahoo! Web Search: Yahoo SLURP hoặc SLURP
  + Ask: Teoma
  + AltaVista: Scooter
  + LookSmart: MantraAgent
  + WebCrawler: WebCrawler
  + SearchHippo: Fluffy the Spider
- Để tìm hiểu thêm về Robot Exclusion Standard chúng ta có thể truy cập vào Web Robots Pages (www.robotstxt.org). Hiểu rõ về Robot Exclusion Standard sẽ giúp chúng ta điều khiển được các Crawler khi nó viếng thăm Web Site của mình.

- Trên thực tế, không phải bất kỳ một Site nào cũng cần phải có một tập tin robot.txt. Nhưng một điểm cần lưu ý là, không nên để một tập tin robot.txt không có nội dung bên trong Site. Một tập tin robot như vậy sẽ khiến cho Crawler ngầm định rằng, Site này không được truy cập bởi nó. Như vậy, việc sử dụng một tập tin robot trống rỗng cũng là một cách tốt nhất để cho một Web Site không xuất hiện trên bất kỳ một máy tìm kiếm nào cả.

2.2.2. Robots Meta Tag

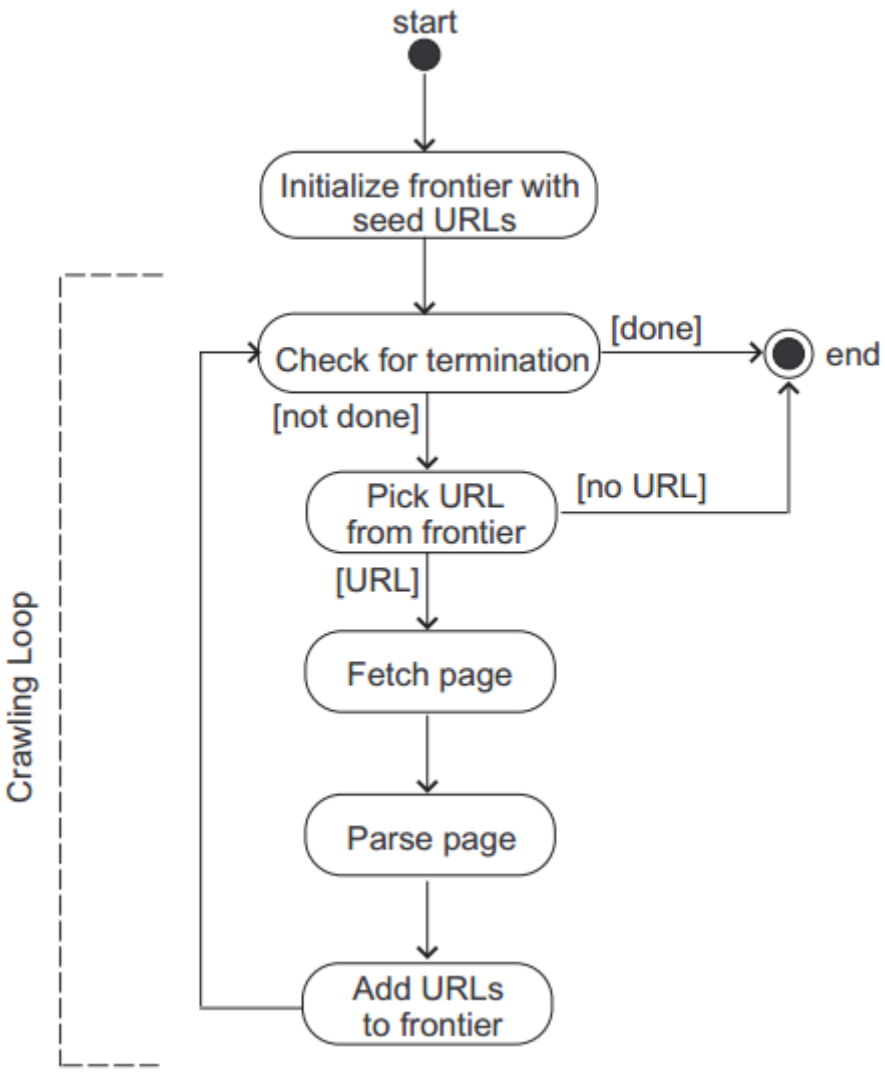
- Robots Meta Tags là một đoạn mã HTML nhỏ chèn vào giữa thẻ <HEAD>, và công dụng của nó cũng tương tự như việc sử dụng tập tin robot.txt. Đoạn mã bên dưới là một ví dụ:

```
<head>
<meta name="robots" content="noindex, nofollow">
<meta name="description" content="page description.">
<title>
        Web Site Title
</title>
</head>
```

- Sử dụng trong trường hợp, các website không muốn tạo ra file robot.txt vì một lý do nào đó.

2.3. Các kỹ thuật xây dựng Crawler

- Crawler là một chương trình hoạt động liên tục và lặp đi lặp lại, nó đi theo các bước và tuân theo các quy tắc nhất định. Hình 2.3 là mô hình quy trình làm việc cơ bản của một Crawler.



Hình 2.2: Quy trình hoạt động của Crawler

- Crawler duy trì một danh sách những URL chưa duyệt qua, danh sách này được gọi là Frontier. Frontier được khởi tạo bằng một số URL (seed URL) bởi người xây dựng hệ thống đưa ra. Mỗi vòng lặp của quá trình duyệt sẽ lấy trong Frontier một URL, và rồi tiến hành những chiến lược thu thập trang được chỉ dẫn bởi URL này thông qua giao thức HTTP. Sau đó phân tích trang đã lấy được để trích xuất ra những URL và những thông tin mà Crawler muốn thu thập, cuối cùng là kiểm tra xem những URL vừa lấy ra đã được Crawler duyệt qua hay chưa, rồi thêm những URL chưa được duyệt vào Frontier. Trước khi những URL này được thêm vào Frontier, chúng được Crawler cấp cho một điểm số, điểm số này nhằm đại diện cho giá trị của trang mà URL này hướng đến so với những trang khác của những URL khác. Với Google, có thể hiểu điểm số này như là PageRank. Quá trình thu thập sẽ dừng lại khi Crawler đã

thu thập được một số lượng trang nhất định hoặc cho đến khi Frontier không còn URL nào nữa.

2.3.1. Cấu trúc dữ liệu của URL Frontier

- Frontier là một danh sách công việc của một Crawler hay còn được gọi là To-do List. Frontier dùng để chứa những URL chưa được Crawler duyệt qua. Trong thuật ngữ của lĩnh vực tìm kiếm, Frontier là một danh sách mở (open list) của những lá (note) chưa được mở (unexpanded note), lá (note) ở đây chính là những trang Web trên Internet. Mặc dầu Frontier cần phải lưu trên Hard Disk để đáp ứng được một số lượng rất lớn URL được đổ vào trong khi Crawler duyệt xuyên qua những trang Web trên Internet. Nhưng Frontier vẫn có thể thiết kế để trở thành một kiểu dữ liệu có cấu trúc và lưu trữ trên bộ nhớ trong In-Memory (ví dụ như RAM). Căn cứ vào khả năng chứa của bộ nhớ trong, những nhà thiết kế có thể đưa ra được kích thước tối đa của một Frontier, điều này cũng đồng nghĩa với việc Frontier sẽ đưa ra giới hạn tối đa những URL mà nó có thể lưu trữ được. Với tình trạng bộ nhớ trong của các máy PC hiện nay, trung bình một Frontier sẽ có sức chứa cực đại là 100.000 URL. Đưa ra một giới hạn kích thước cho Frontier, chúng ta cần một cơ chế để đưa ra quyết định những URL sẽ bị loại bỏ khỏi Frontier hoặc sẽ bị lờ đi và không đưa vào Frontier khi giới hạn này bị chạm đến. Frontier có thể được điền vào một số lượng lớn các URL trong khoảng thời gian rất ngắn, ước tính cứ mỗi 10.000 trang Web mà Crawler đi qua thì nó sẽ thu về được 60.000 URL, tính trung bình là có 7 liên kết trong một trang.
- Trong trường hợp Crawler hoạt động theo cơ chế duyệt theo chiều rộng, Frontier có thể được cài đặt như là một hàng đợi FIFO. Và URL tiếp theo được đưa đến Crawler sẽ lấy từ phần tử đầu tiên của Frontier và URL mới khi thêm vào sẽ được thêm vào đằng sau cùng của Frontier. Công việc xác định xem URL đã từng nằm trong Frontier hay chưa cũng là một tác vụ tốn rất nhiều chi phí cho Crawler. Một giải pháp cho vấn đề này là sử dụng một phần bộ nhớ để cấp phát một HashTable dùng để chứa những URL mà Crawler đã duyệt qua (với URL chính là khóa) để hỗ trợ cho việc tra cứu một cách nhanh chóng, và Hash-Table này phải được giữ đồng bộ với Frontier.
- Cũng có một phương thức khác để cài đặt Frontier, gọi là hàng đợi ưu tiên (Priority Queue) được áp dụng cho những Crawler được biết đến như là những Best-First Crawler. Hàng đợi ưu tiên có thể là một mảng liên kết động, cái mà luôn được sắp

xếp theo điểm số ước lượng của những trang mà URL hướng đến. Tại mỗi bước, URL tốt nhất sẽ được lấy ra từ đầu danh sách của Frontier, và khi một trang được thu thập, những URL lấy ra từ nó được gán một điểm số dựa vào một Heuristic. Và cuối cùng, chúng được thêm vào Frontier với đảm bảo là trật tự về độ ưu tiên trong Frontier vẫn được bảo toàn. Một khi kích thước Frontier đạt đến giới hạn, khi đó chỉ những URL có độ ưu tiên cao nhất mới được giữ lại trong Frontier.

2.3.2. Bộ lọc địa chỉ

- Bộ lọc địa chỉ là một cấu trúc dữ liệu quan trọng thứ hai trong bất kỳ thể hiện nào của Crawler, nó nhằm lưu lại tất cả những URL mà Crawler đã đi qua và đã từng chứa trong Frontier. Mục đích của bộ lọc địa chỉ là nhằm hạn chế trường hợp Crawler thêm vào Frontier một cách lặp lại những URL đã thêm vào trước đó. Vì lý do này, người ta gọi nó là URL-Seen Test (UST) hay là Duplicate URL Eliminator (DUE). UST phải hỗ trợ những chức năng như: thêm, xóa, và kiểm tra sự tồn tại.
- Có nhiều cách thức để cài đặt cấu trúc này, một trong những phương pháp đơn giản nhất đã được nói đến và sử dụng rất nhiều, đó là sử dụng cấu trúc HashTable [[9].

2.3.3. Chiến lược thu thập và bộ phân tích trang Web (Fetching & parsing)

- Để có thể thu thập dữ liệu từ một trang Web, chúng ta cần một máy khách HTTP, máy này sẽ gửi một yêu cầu HTTP đến máy chủ và nhận lại kết quả trả về. Máy khách sẽ quy định một khoảng thời gian gọi là Time-Out để đảm bảo rằng không phải chi phí một lượng thời gian qúa lớn cho một máy chủ chậm chạp (Slow Server) hay là một trang Web quá lớn. Trên thực tế, người ta thường đưa ra hạn chế cho Crawler để chỉ tải về 10-20 KB đầu tiên của trang đang xử lý. Một khi nội dung của trang được lấy về, chúng ta cần phải phân tích nội dung của trang đó, và một phần nội dung của những trang này sẽ tác động trở lại hoạt động của Crawler sau này. Nội dung này có thể đơn giản chi là những liên kết (hyperlink/URL) hoặc có thể nó sẽ kéo theo những xử lý phức tạp hơn như là phân tích cấu trúc HTML của trang đó. Một trong những nhiệm vụ quan trọng trong quá trình phân tích của Crawler là chuyển đổi những URL được lấy về theo dạng chuẩn. Dưới đây là sự diễn giải chi tiết hơn về thành phần quan trọng này.

2.3.4. Trích xuất URL và sự chuẩn hóa

- Trích xuất URL là quá trình phân tích mã HTML của một trang Web và lấy ra những liên kết có trong trang đó. Chuẩn hóa URL là sự biến đổi những liên kết lấy về trở thành một dạng tiêu chuẩn và thống nhất về định dạng.
- Thông thường, để có thể lấy được những URL bên trong một tài liệu HTML, chúng ta có thể tìm đến những Anchor Tags (thẻ <A>) và lấy giá trị của thuộc tính HREF. Tuy nhiên, chúng ta cần phải chuyển đổi tất cả những URL đó trở về những URL có định dạng tuyệt đối. Đây là một bước rất quan trọng nhằm tránh trường hợp phân tích những URL có định dạng khác nhau nhưng lại hướng đến một trang Web duy nhất. Dưới đây là một số thủ tục tiêu biểu để chuẩn hóa một URL:
  + Chuyển đổi giao thức và tên của máy chủ về ký tự thường.
    - Ví dụ: chuyển từ

      HTTP://www.siu.edu.vn

      trở thành http://www.siu.edu.vn.
  + Loại bỏ những phần là Bookmark của URL.
    - Ví dụ như:

      http://www.siu.edu.vn/index.html#top

      trở thành http://www.mta.edu.vn/index.html
  + Chuyển đổi những ký tự đặt biệt (ví dụ: '~') theo một quy định thống nhất. Công việc này nhằm tránh trường hợp crawler nhầm lẫn giữa

    http://www.siu.edu.vn/~contact/

    và http://www.siu.edu.vn/%7Econtact/
  + Một vài URL, có thêm vào dấu chéo sau cùng (http://www.siu.edu.vn/) và cũng có một số trường hợp trong cùng Site đó lại không như vậy (http://www.siu.edu.vn). Nhưng vậy ta phải định nghĩa một quy tắc, hoặc là đưa vào hoặc là loại bỏ.
  + Loại bỏ chỉ dẫn '..' và thư mục cha của nó trong URL. Ví dụ:

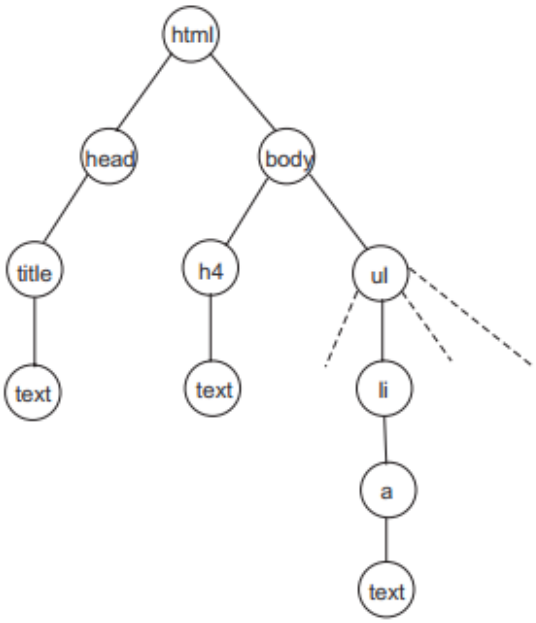
    /%7Epant/BizIntel/Seeds/../ODPSeeds.dat

    trở thành /%7Epant/BizIntel/ODPSeeds.dat
- Việc thống nhất và áp dụng những quy tắc chuẩn hóa là rất quan trọng với Crawler, những quy tắc này sẽ tăng hiệu suất phân tích nếu như nó được định nghĩa một tập luật tối ưu.

2.3.5. *Mô hình thẻ HTML dạng cây*

- Các trình thu thập có thể lấy ra giá trị của các URL hoặc một nội dung bất kỳ trong một trang web bằng cách kiểm tra phạm vi thẻ tag HTML chứa chúng. Để làm được điều này, trình thu thập có thể sử dụng mô hình thẻ HTML dạng cây và phân tích cấu trúc DOM (Document Oject Model) của mô hình này. Phân tích cấu trúc DOM giúp trình thu thập có thể duyệt các node trên cây này và chỉ lấy ra phần nội dung mà nó cần. Hình 2.3 cho ta thấy một mô hình cây tương ứng với một mã nguồn URL.

```
<html>
<head>
<title>Projects</title>
</head>
<body>
<h4>Projects</h4>
<ul>
  <li> <a href="blink.html">LAMP</a> Linkage analysis with multiple processors.</li>
  <li> <a href="nice.html">NICE</a> The network infrastructure for combinatorial exploration.</li>
  <li> <a href="amass.html">AMASS</a> A DNA sequence assembly algorithm.</li>
  <li> <a href="dali.html">DALI</a> A distributed, adaptive, first-order logic theorem prover.</li>
</ul>
</body>
</html>
```



Hình 2.3: Mô hình cây tương ứng với một mã nguồn URL

- Có thể thấy thẻ <html> là gốc của cây, các thẻ bên trong nó là các node mở rộng, và dữ liệu text là lá của cây.
- Trên thực tế, không phải văn bản HTML nào cũng được viết đúng quy chuẩn như ví dụ trên. HTML là ngôn ngữ không phân biệt chữ hoa hay chữ thường (hai thẻ <tr> và <TR> đều là một). Các phần tử HTML cần có một thẻ mở và một thẻ đóng, tuy

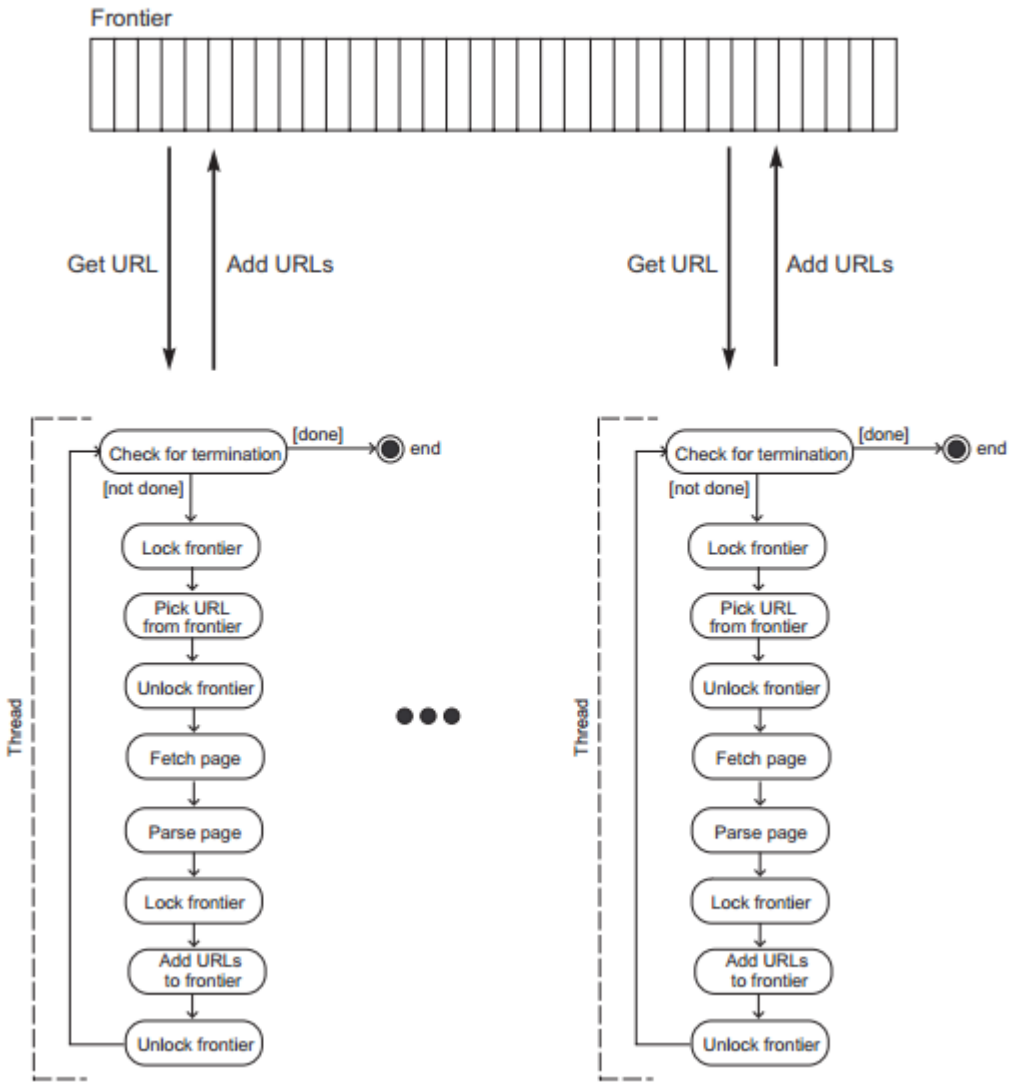
nhiên điều này không luôn luôn đúng, có nhiều phần tử không cần thẻ đóng, ví dụ các thẻ
, <hr> và <li>. Ngoài ra khi lồng nhau, các phần tử HTML cũng không cần thiết phải lồng nhau theo đúng thứ tự (tức là thẻ nào mở trước thì phải đóng sau). Ví dụ sau là hợp lệ trong HTML:

 Cộng hòa xã hội chủ nghĩa Việt Nam
Độc lập tự do hạnh phúc

- Vì vậy trước khi lập mô hình cây cho một mã nguồn HTML chúng ta cần một quá trình chuyển đổi các tài liệu HTML tồi thành các tài liệu HTML tiêu chuẩn, quá trình này gọi là chuẩn hóa các trang HTML. Quá trình này bao gồm việc chuyển đổi các thẻ sang dạng chữ thường, chèn thêm các thẻ bị và sắp xếp lại thứ tự các thẻ trong tài liệu HTML. Chuẩn hóa trang HTML là rất cần thiết để việc lập mô hình cây được chính xác. Nếu như trình thu thập chỉ cần lấy các liên kết hoặc văn bản hoặc một phầnvăn bản thì có thể ta không cần sử dụng tới mô hình cây mà chỉ cần sử dụng kỹ thuật bóc tách HTML đơn giản. Trình bóc tách như vậy cũng được hỗ trợ trong nhiều ngôn ngữ lập trình.

2.3.6. Crawler đa tiến trình

- Crawler đa tiến trình là một chương trình Crawler với nhiều tiến trình tiến hành thu thập dữ liệu tại cùng một thời điểm.
- Một vòng lặp tuần tự cho quá trình thu thập bỏ phí khá nhiều thời gian rỗi cho những tài nguyên trong hệ thống. Ví dụ như thời gian rỗi của CPU (trong quá trình chờ đợi việc truy cập Network, hệ thống lưu trữ) hoặc thời gian rỗi của Network (trong quá tình xử lý của CPU). Để tận dụng được những khoảng thời gian này, xử lý đa tiến trình (Multi-threading) là một biện pháp. Mỗi tiến trình sẽ nắm giữ một vòng lặp phục vụ cho việc thu thập. Hình 2.4 là mô hình cơ bản của hệ thống Crawler đa tiến trình:



Hình 2.4: Mô hình đa tiến trình của Crawler

- Mỗi tiến trình bắt đầu công việc của mình bằng cách khóa hàng đợi Frontier để lấy ra URL tiếp theo cho quá trình thu thập, sau khi lấy được một URL trong Frontier, tiến trình này sẽ mở khóa cho Frontier để cho phép những tiến trình khác truy cập vào. Và rồi, Frontier tiếp tục được khóa trở lại khi tác vụ thêm những URL mới được duyệt vào. Những bước khóa và mở khóa là các phương thức cơ bản để đồng bộ hóa dữ liệu bên trong Frontier khi đã được chia sẻ cho rất nhiều tiến trình dùng chung đến Frontier trong hệ thống.
- Theo lẽ tự nhiên, Crawler đa tiến trình cũng có lúc gặp phải tình trạng Frontier trống rỗng như Crawler đơn tiến trình. Nhưng cách ứng xử của Crawler đa tiến trình với tình trạng này sẽ khác với Crawler đơn tiến trình. Nếu một tiến trình gặp phải tình trạng không còn URL trong Frontier để lấy ra, tiến trình này sẽ không lập tức đưa

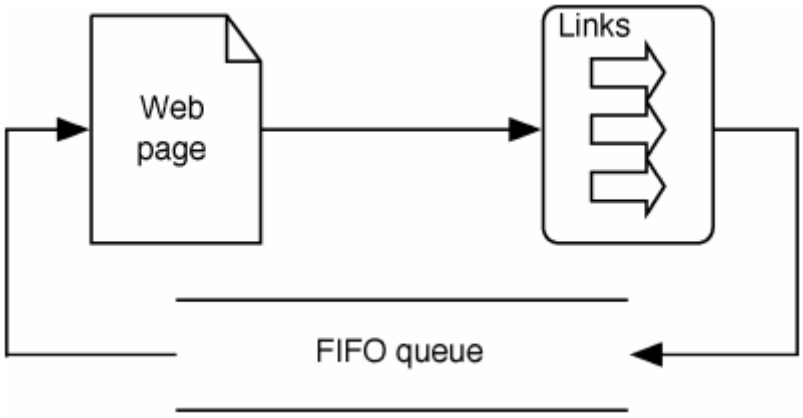
vòng lặp của nó đến trình trạng Dead-End (kết thúc). Có thể, đang có một tiến trình khác đang tiến hành phân tích một trang Web nào đó và sẽ có những URL mới sẽ được thêm vào Frontier. Một giải pháp cho vấn đề này là đặt tiến trình hiện hành vào trình trạng chờ (Sleep) khi bắt gặp trạng thái trống rỗng của Frontier. Khi tiến trình này hoạt động trở lại (Wake-up), nó sẽ kiểm tra lại Frontier. Chúng ta phải có một cơ chế quản lý được cả trạng thái của những tiến trình đang chờ, nếu như tất cả các tiến trình trong hệ thống đều rơi vào trạng thái chờ. Quá trình thu thập của Crawler đa tiến trình sẽ kết thúc.

2.4. Một số giải thuật Crawler

- Khi một trang được phân tích bởi Crawler, những liên kết trong trang đó sẽ được đưa vào danh sách của những trang chưa được phân tích, danh sách này chính là Frontier. Một trong những bước quan trọng nhất trong tiến trình hoạt động của một Crawler là xác định liên kết tiếp theo phù hợp nhất với tiêu chí của Crawler để tiến hành bước phân tích tiếp theo.
- Việc thiết kế thuật toán để lựa chọn liên kết tiếp theo trong Frontier nhằm phục vụ cho vòng lặp kế tiếp trong quá trình di chuyển của Crawler là bước mấu chốt để quyết định nét đặt trưng của Crawler. Nói một cách khác, một Crawler duyệt trên Internet nhằm mục đích liệt kê ra tất cả thông tin của tất cả các trang Web trên Internet sẽ được thiết kế khác với một Crawler chỉ liệt kê thông tin của những trường Đại Học hoặc chỉ là liệt kê những trang Web có nội dung giới thiệu phim ảnh. Đối với Crawler đầu tiên, thứ tự và độ ưu tiên của việc lựa chọn URL tiếp theo trong Frontier là không mấy quan trọng. Nhưng đối với Crawler thứ hai, Crawler mà chỉ tìm duyệt những trang Web của các trường đại học, thì công việc lấy URL tiếp theo trong Frontier cần phải có sự cân nhắc đến cấu trúc của URL, như là chỉ lấy ra hoặc thêm vào những URL với domain là .edu. Trong khi đó, Crawler thứ ba ở ví dụ trên, lại phải căn cứ vào nội dung và những từ khóa ở trong Site, nó chỉ duyệt qua những Site có chủ đề là movie. Vì vậy, khi thiết kế một Crawler, các nhà thiết kế cần phải cân nhắc thuật toán nào cần phải cài đặt để quyết định tính hiệu quả và thu thập đúng dữ liệu của những nhà cung cấp dịch vụ tìm kiếm đề ra.

2.4.1. Thuật toán tìm kiếm theo chiều rộng (Breadth-First)

- Thuật toán này được biết đến trong 1994 bởi WebCrawler. Giải thuật này sử dụng một hàng đợi FIFO, nó luôn lấy ra URL đầu tiên phía đầu hàng đợi và thêm vào những URL mới vào cuối hàng đợi. Có một điểm lưu ý là một khi Frontier đầy, Crawler chỉ có thể đưa thêm một URL vào sau mỗi lần xử lý một trang. Breadth-First Algorithm bởi có cơ chế như Hình 2.5:



Hình 2.5: Mô hình Crawler dò tìm theo chiều rộng

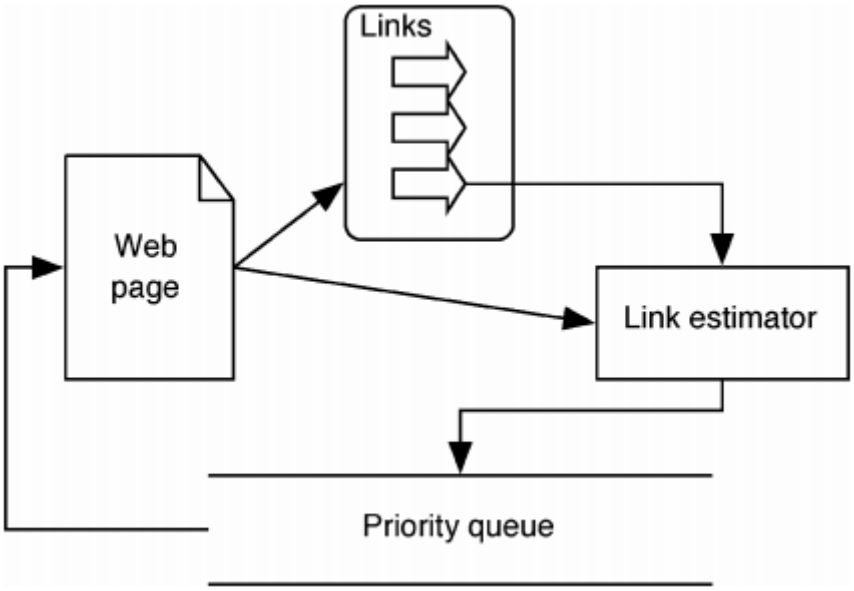
```
Breadth-First (starting_urls) {
   foreach link (starting_urls) {
      enqueue(frontier, link);
   }
   while (visited < MAX_PAGES) {
      link := dequeue_link(frontier);
      doc := fetch(link);
      enqueue(frontier, extract_links(doc));
      if (#frontier > MAX_BUFFER) {
         dequeue_last_links(frontier);
      }
   }
}
```

Hình 2.6: Mô hình hoạt động của thuật toán Breadth-First

- Breadth-First Crawler được xem như là Crawler cơ bản, nó hoàn toàn không đặt nặng công việc tìm kiếm của mình vào một lĩnh vực nào đó. Và nó phù hợp để cài đặt cho những máy tìm kiếm Primary Search Engine.

2.4.2. Thuật toán tìm kiếm tối ưu (Best-First)

- Giải thuật Best-First được trình bày bởi Cho et al và Hersovice et al (1998). Ý tưởng cơ bản là căn cứ vào một vài tiêu chí được ước lượng trước để lựa chọn ranhững liên kết được nhận định là tốt nhất trong quá trình phân tích để đưa vào Frontier. Việc lựa chọn những liên kết được quyết định bởi những phép tính so sánh sự tương đương về mặt ngữ nghĩa của nội dung bên trong trang của liên kết và những từ khóa trong chủ đề của máy tìm kiếm. Cũng như thế, sự tương đương giữa trang P và các từ khóa trong chủ đề được sử dụng để ước tính mức độ phù hợp của những trang được liên kết đến bởi P. URL với sự ước tính tốt nhất sẽ được đưa vào Frontier đầu tiên. Hình 2.7 sẽ giúp hiểu rõ hơn về quy trình hoạt động của Best-First Crawler:



Hình 2.7: Mô hình Crawler dò tìm theo (Best-First)

```
BFS (topic, starting_urls) {
   foreach link (starting_urls) {
      enqueue(frontier, link, 1);
   }
   while (visited < MAX_PAGES) {
      link := dequeue_top_link(frontier);
      doc := fetch(link);
      score :=  sim(topic, doc);
      enqueue(frontier, extract_links(doc), score);
      if (#frontier > MAX_BUFFER) {
         dequeue_bottom_links(frontier);
      }
   }
}
```

Hình 2.8: Thuật toán tìm kiếm tối ưu (Best-First)

- Hàm sim () trả về cosine độ tương đồng gữa topic và page:

$$sim(q,p) = \frac{\sum_{k \in q \cap p} f_{kq} f_{kp}}{\sqrt{\left(\sum_{k \in p} f_{kp}^2\right)\left(\sum_{k \in q} f_{kq}^2\right)}} \quad (2.1)$$

trong đó, q là topic, p là số page đã tải xuống và f_{kd} là tần số của k trong d

CHƯƠNG 3. CÁC KỸ THUẬT LƯU TRỮ

- Chương này nhằm cung cấp các kỹ thuật trong việc lưu trữ cho hệ thống tìm kiếm thông tin. Trình bày một số kỹ thuật tách từ cho tiếng Việt. Đồng thời, với các mô hình tìm kiếm thông tin đã trình bày ở chương 1, trong chương này sẽ tập trung vào phân tích kỹ phương pháp lập chỉ mục cho tài liệu tiếng Việt theo mô hình chỉ mục ngữ nghĩa ngầm LSI.

3.1. Một số kỹ thuật tách từ trong tiếng Việt

- Tách từ là giai đoạn đầu tiên của một hệ thống tài liệu thông tin. Tách từ cho tài liệu là công việc quan trọng. Đối với tiếng Anh chỉ đơn giản dựa vào khoảng trắng để tách từ. Nhưng đối với tiếng Việt không thể dựa vào khoảng trắng được vì tiếng Việt là ngôn ngữ đơn lập.

3.1.1. fnTBL (Fast Transformation-based learning)

- Phương pháp học dựa trên sự biến đổi (TBL) là cách tiếp cận dựa trên ngữ liệu đã đánh dấu. Theo cách tiếp cận này, để huấn luyện cho máy tính biết cách nhận diện ranh giới từ tiếng Việt ta có thể cho máy học trên ngữ liệu hàng vạn câu hỏi tiếng Việt đã được đánh dấu ranh giới từ đúng.
- Ý tưởng chính của phương pháp là để giải quyết một vấn đề nào đó ta sẽ áp dụng phép biến đổi, tại mỗi bước, phép biến đổi nào cho kết quả tốt nhất sẽ được chọn và được áp dụng lại với vấn đề đã đưa ra. Thuật toán kết thúc khi không còn phép biến đổi nào được chọn.

❖ **Ưu điểm:**

- Đặc điểm của phương pháp này là khả năng tự rút ra quy luật của ngôn ngữ.
- Nó có những ưu điểm của cách tiếp cận dựa trên luật (vì cuối cùng nó cũng dựa trên luật được rút ra) nhưng nó khắc phục được khuyết điểm của việc xây dựng các luật một cách thủ công bởi các chuyên gia.
- Các luật thử nghiệm tại chỗ để đánh giá độ chính xác và hiệu quả của luật (dựa trên ngữ liệu huấn luyện).
- Có khả năng khử được một số nhập nhằng như “The singer sang a lot of a??as ”, thì hệ có thể xác định được “a??as” là “arias” (dân ca) thay vì “areas” (khu vực) của các mô hình ngôn ngữ theo kiểu thống kê.

❖ **Nhược điểm:**

- Phương pháp này "dùng ngữ liệu có gán nhãn ngôn ngữ để học tự động các qui luật đó" [Đinh Điền, 2004]. Đây là việc rất khó, tốn kém nhiều về thời gian và công sức.
- Hệ phải trải qua một thời gian huấn luyến khá lâu để có thể rút ra các luật tương đối đầy đủ.
- Cài đặt phức tạp.

3.1.2. Longest Matching

- Phương pháp Longest Matching tách từ dựa vào từ điển có sẵn. Theo phương pháp này, để tách từ tiếng Việt ta đi từ trái qua phải và chọn từ có nhiều âm tiết nhất mà có mặt trong từ điển, rồi cứ tiếp tục cho từ kế tiếp cho đến hết câu. Thuật toán được trình bày trong [Chih –Hao Tsai, 2000].
- Dạng đơn giản nhất được dùng giải quyết nhập nhằng từ đơn. Giả sử có một chuỗi ký tự (tương đương với chuỗi tiếng trong tiếng Việt) $C_1, C_2, \ldots, C_n$. Ta bắt đầu từ đầu chuỗi. Đầu tiên kiểm tra xem C_1 có phải là từ hay không, sau đó kiểm tra xem $C_1\ C_2$ có phải là từ hay không. Tiếp tục tìm cho đến khi tìm được từ dài nhất. Từ có vẻ hợp lý nhất sẽ là từ dài nhất. Chọn từ đó, sau đó tìm tiếp như trên cho những từ còn lại cho đến khi xác định được toàn bộ chuỗi từ.
- Dạng phức tạp: Qui tắc của dạng này là phân đoạn có vẻ hợp lý nhất là đoạn ba từ với chiều dài tối đa. Thuật toán bắt đầu từ dạng đơn giản. Nếu phát hiện ra những cách tách từ gây nhập nhằng (Ví dụ như C_1 là từ, $C_1\ C_2$ cũng là từ), ta xem các chữ kế tiếp để tìm tất cả các đoạn ba từ có thể bắt đầu với C_1 và $C_1\ C_2$. Ví dụ ta có được những đoạn sau:

  $C_1 \quad C_2 \quad C_3C_4$

  $C_1C_2 \quad C_3\ C_4 \quad C_5$

  $C_1\ C_2 \quad C_3\ C_4 \quad C_5\ C_6$
- Chuỗi dài nhất sẽ là chuỗi thứ ba. Vậy từ đầu tiên của chuỗi thứ ba (C_1C_2) sẽ được chọn. Thực hiện lại các bước cho đến khi được chuỗi từ hoàn chỉnh.

❖ **Ưu điểm:**

- Với phương pháp này, ta dễ dàng tách được chính xác các ngữ/câu như: "hợp tác xã|| mua bán", "thành lập|| nước|| Việt Nam|| dân chủ|| cộng hòa",…Các tách từ

đơn giản, nhanh, chỉ cần dựa vào từ điển. Trong tiếng Hoa, cách này đạt độ chính xác 98,41% trong [Chih –Hao Tsai, 2000].

❖ **Nhược điểm:**

- Độ chính xác của phương pháp phụ thuộc hoàn toàn vào tính đủ và tính chính xác của từ điển. Phương pháp này sẽ tách từ sai trong các trường hợp: "học sinh|| học sinh|| học", "trước|| bàn là|| một|| ly|| nước".

3.1.3. Mô hình tách từ bằng WFST và mạng Neural

- Mô hình mạng chuyển dịch trạng thái hữu hạn có trọng số WFST (Weighted Finit State Transducer) đã được [Richard et al, 1996] áp dụng để tách từ tiếng Trung Quốc. Ý tưởng cơ bản là áp dụng WFST kết hợp với trọng số là xác suất xuất hiện của mỗi từ có trong ngữ liệu. Dùng WFST duyệt qua câu cần quét.
- Cách duyệt có trọng số lớn nhất sẽ là cách tách từ được chọn. Giải pháp này cũng đã được áp dụng trong [Đinh Điền et al, 2001] kèm với mạng Neural để khử nhập nhằng.
- Hệ thống tách từ tiếng Việt của [Đinh Điền et al, 2001] gồm hai tầng: tầng WFST ngoài làm việc tách từ còn xử lý thêm các vấn đề liên quan đến đặc thù của tiếng Việt như từ láy, tên riêng…và tầng mạng Neural dùng để khử nhập nhằng nếu có.

3.1.4. Phương pháp dựa trên thống kê từ Internet và thuật toán di truyền

- Phương pháp dựa trên thống kê từ Internet và thuật toán di truyền IGATEC – Internet and Genetics Algorithm based Text Categorization for Documents in Vietnames [H. nguyen et al, 2005] là một hướng tiếp cận mới cho việc tách từ với mục đích phân loại văn bản mà không cần dùng đến một từ điển hay tập huấn luyện nào. Trong hướng tiếp cận này, tác giả kết hợp thuật toán di truyền (Genetics Algorithm - GA) với dữ liệu thống kê được trích xuất từ Internet tiến hóa một quần thể gồm các cá thể là các khả năng tách từ trong câu.

❖ **Ưu điểm:**

- Phương pháp không cần sử dụng bất cứ tập huấn luyện hoặc từ điển nào. Tương đối đơn giản và tốn ít thời gian huấn luyện.

❖ **Hạn chế:**

- So với các phương pháp trước, IGATEC có độ chính xác thấp hơn Maximum Matching và WFST nhưng vẫn chấp nhận được đối với mục đích tách từ dành cho phân loại văn bản.

3.2. Phương pháp lập chỉ mục.

- Với mỗi mô hình tìm kiếm thông tin thì tương ứng có các cách tính toán chọn dữ liệu chỉ mục khác nhau. Trong phần này, luận văn tập trng vào trình bày kỹ thuật lập chỉ mục theo mô hình LSI. Đây cũng chính là cơ sở để xây dựng ứng dụng minh họa tại chương 4.

3.2.1. Xác định các từ chỉ mục

- Để xác định các từ chỉ mục cho tài liệu, các bước:
  + **Bước 1:** Tách từ
  + **Bước 2:** Bỏ các từ không có giá trị, các từ này được gọi là (stop word).
  + **Bước 3:** Loại bỏ hậu tố - chuẩn hoá từ (stemming): Giữ lại các từ có nội dung (content word) sử dụng như các từ chỉ mục, ngoài ra trước khi sử dụng nó như các từ chỉ mục cho tập văn bản ta sử dụng thuật toán stemming để giữ lại từ gốc của từ đó. Ví dụ: trong văn bản có các từ computer, computing, computed sử dụng thuật toán stemming để lấy từ gốc comput. Sử dụng thuật toán stemming làm giảm số từ chỉ mục hay giảm số chiều cho vector biểu diễn văn bản nhưng vẫn giữ được nội dung của văn bản. Đối với tiếng Việt bước này không cần phải thực hiện.
  + **Bước 4:** Tập các từ còn lại được sử dụng như các từ chỉ mục cho toàn bộ văn bản, số từ chỉ mục này chính là số chiều cho mỗi vector biểu diễn văn bản.

3.2.2. Xây dựng ma trận từ chỉ mục (Term – Document) A

- Sau khi đã xác định được các từ chỉ mục (giả sử có m từ chỉ mục), cho tập văn bản có n tài liệu. Mỗi văn bản được vector hoá thành một vector m chiều. Các thành phần của mỗi vector được đánh trọng số sử dụng hàm:

$$w_{ij} = l_{ij} \times g_i \times n_j \qquad (3.1)$$

- Trong đó:
  - l_{ij} là trọng số cục bộ của từ chỉ mục i trong văn bản j - là hàm đếm số lần xuất hiện của mỗi từ chỉ mục trong một văn bản.
  - g_i là trọng số toàn cục của từ chỉ mục i - là hàm đếm số lần xuất hiện của mỗi từ chỉ mục trong toàn bộ tập văn bản.
  - n_j là hệ số được chuẩn hoá của văn bản j - là hệ số cân bằng chiều dài của các văn bản trong tập văn bản.

Các công thức tính trọng số cục bộ của từ chỉ mục l_{ij}

- l_{ij} bằng số lần xuất hiện của từ chỉ mục trong văn bản (term FREQ):

$$l_{ij} = f_{ij}(FREQ) \tag{3.2}$$

- Trong đó f_{ij} là số lần xuất hiện của từ chỉ mục i trong văn bản j (tf: term frequency).
- Phương pháp này về cơ bản đã tính được điểm số hay thứ hạng của kết quả trả về. Tuy nhiên, có vấn đề xảy ra là nếu các mục từ xuất hiện nhiều thì trọng số của nó càng lớn, nhưng những mục từ xuất hiện nhiều lại là các mục từ thường dùng, ít có ý nghĩa (trong tiếng Anh gọi là các stop-word). Hàm logarithms được sử dụng để điều chỉnh lại số lần xuất hiện của một từ chỉ mục trong một văn bản, bởi vì một từ chỉ mục xuất hiện 10 lần trong một văn bản không hẳn có độ đo quan trọng gấp 10 lần so với một từ chỉ xuất hiện 1 lần. Công thức tính bằng hàm logarithms
- l_{ij} được tính bằng logarithms

$$l_{ij} = \begin{cases} 1 + \log f_{ij} & if \quad f_{ij} > 0 \\ 0 & if \quad f_{ij} = 0 \end{cases} (LOGA) \tag{3.3}$$

  - Trong đó: f_{ij} là số lần xuất hiện của từ chỉ mục i trong văn bản j

    l_{ij} được tính theo Augumented normalized term frequency
- Trong những văn bản dài có xu hướng lặp lại các từ nhiều lần, một công thức tính trọng số cục bộ khác là sự kết hợp giữa BNRY và FREQ để tạo thành hàm ATF1:

$$l_{ij} = \begin{cases} 0.5 + 0.5(f_{ij}/x_j) & if \quad f_{ij} > 0 \\ 0 & if \quad f_{ij} = 0 \end{cases} (ATF1) \tag{3.4}$$

  - Trong đó x_j là số lần xuất hiện lớn nhất của các từ chỉ mục trong văn bản j;

    f_{ij} là số lần xuất hiện của từ chỉ mục i trong văn bản j.
- Với công thức trên, l_{ij} thay đổi từ 0.5 đến 1.0 cho các từ chỉ mục xuất hiện trong văn bản. Bảng 3.1 dưới đây tổng hợp các hàm tính trọng số cục bộ l_{ij}.

Bảng 3.1: Bảng tính các hàm trọng số cục bộ l_{ij}

$l_{ij} =$	**Tên hàm**	**Viết tắt**
$1 \; if \; f_{ij} > 0$	Binary	BNRY

$0\ if\ f_{ij} = 0$		
f_{ij}	Within_document frequency	FREQ
$1 + \log f_{ij}\ if \quad f_{ij} > 0$ $0 \quad if \quad f_{ij} = 0$	Log	LOGA
$(1 + \log f_{ij})/(1 + \log a_j)$ if $f_{ij} > 0$ 0 if $f_{ij} = 0$	Normalized log	LOGN
$0.5 + 0.5(f_{ij}/x_j)\ if \quad f_{ij} > 0$ $0 \quad if \quad f_{ij} = 0$	Augumented normalized term frequency	ATF1

Các công thức tính trọng số toàn cục của từ chỉ mục g_i

- Trọng số toàn cục (global weight) chỉ giá trị "phân biệt" (discrimination value) của mỗi từ chỉ mục trong toàn bộ tập văn bản. Các hàm tính trọng số toàn cục dựa trên ý nghĩa: số lần xuất hiện ít của một từ chỉ mục trong toàn bộ văn bản có giá trị phân biệt cao hơn. Một hàm tính trọng số toàn cục thông dụng là IDF (inverted document frequency [**Error! Reference source not found.**].
- g_i được tính bằng phương pháp tính tần suất nghịch đảo (idf - inverted document frequency).

$$idf_t = \log \frac{N}{df_t} \tag{3.5}$$

- g_i được tính bằng phương pháp tính xác suất nghịch đảo (IDFB - Probabilistics inverse).

$$g_i = \log\left(\frac{N - df_t}{df_t}\right) \tag{3.6}$$

- g_i được tính bằng phương pháp tuần suất xuất hiện toàn cục của mục từ t (IGFF - Global frequency IDF).

$$g_i = \frac{F_i}{df_t} \tag{3.7}$$

  - Trong đó, ý nghĩa của các tham số trong hàm:
    - N là tổng số tài liệu trong bộ dữ liệu
    - df_t là tổng số tài liệu có chứa mục từ t

- F_i là số lần xuất hiện của từ chỉ mục i trong toàn bộ văn bản

Bảng 3.2: Bảng các hàm trọng số toàn cục g_i

g_i	Tên hàm	Viết tắt
$\log\frac{N}{df_t}$	Inverse document frequency	*IDFB*
$\log\left(\frac{N-df_t}{df_t}\right)$	Probabilistics inverse	*IDFP*
$\frac{F_i}{df_t}$	Global frequency IDF	*IGFF*

Công thức tính hệ số chuẩn hoá n_j

- Hệ số chuẩn hoá - là hệ số cân bằng chiều dài của các văn bản trong tập văn bản, công thức thức tính là công thức chuẩn hoá cosines (COSN):

$$N_j = \frac{1}{\sqrt{\sum_{i=0}^{m}\left(G_i L_{ij}\right)^2}} \tag{3.8}$$

3.2.3. Phân tích giá trị đơn (Singular Value Decomposition - SVD)

- Đầu tiên, ma trận thuật ngữ - tài liệu được biểu diễn dưới dạng:

$$t_i^T \rightarrow \begin{matrix} \\ d_j \\ \downarrow \end{matrix} \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix}$$

  - Trong đó mỗi dòng tương ứng với một thuật ngữ, xác định quan hệ (số lần xuất hiện, hay trọng số) của thuật ngữ đối với các tài liệu.

$$t_i^T = [x_{i,1} \quad \cdots \quad x_{i,n}]$$

  - Tương tự, mỗi tài liệu được biểu diễn dưới dạng

$$d_j = \begin{bmatrix} x_{1,j} \\ \vdots \\ x_{m,j} \end{bmatrix}$$

- Phân tích SVD của ma trận thuật ngữ - tài liệu A thành dạng $A = USV^T$

  + Trong đó:

- U là ma trận trực giao, các vector dòng của U là các vector thuật ngữ.
- S là ma trận đường chéo, có các giá trị $\sigma_1,...,\sigma_t$ suy biến.
- V^T là ma trận trực giao, các vector cột của V là các vector văn bản.

– Khi rút gọn ma trận S, giữ lại một số k phần tử đầu tiên và rút gọn tương ứng các ma trận U và V^T, sẽ tạo ra một xấp xỉ gần đúng cho ma trận từ chỉ mục A. Điều quan trọng hơn là qua đó, có thể ánh xạ các thuật ngữ và tài liệu vào một không gian ngữ nghĩa (concept space).

❖ **Ví dụ 3.1:** Phân tích SVD ma trận $A = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$ thành dạng $A = USV^T$

+ **Bước 1:** Tính ma trận chuyển vị A^T từ ma trận A và thực hiện phép nhân $A.A^T$

$$A = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix} \rightarrow A^T = \begin{bmatrix} 4 & 3 \\ 0 & -5 \end{bmatrix}$$

$$A.A^T = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}\begin{bmatrix} 4 & 3 \\ 0 & -5 \end{bmatrix} = \begin{bmatrix} 25 & -15 \\ -15 & 25 \end{bmatrix}$$

+ **Bước 2:** Xác định giá trị riêng và vector riêng của ma trận A (singular value).
  - Để tính được giá trị đơn của ma trận A, cần xác định giá trị riêng (eigenvalues) của A và sắp xếp theo thứ tự giá trị tuyệt đối giảm dần. Tính căn bậc hai của giá trị riêng này sẽ thu được giá trị đơn của A (singular value).
  - Tính giá trị riêng: bằng cách tính nghiệm của phương trình đặc trưng $det(A^T A - c * I) = 0$. Trong đó: A^T là ma trận chuyển vị của A; I là ma trận đơn vị;
  - Giải phương trình

$$det(A^T A - c * I) = |A^T A - c * I| = 0$$

$$det(A^T A - c * I) = (25 - c)(25 - c) - (-15)(-15) = 0$$

$$\Leftrightarrow c^2 - 50c + 400 = 0 \rightarrow c_1 = 40, c_2 = 10$$

  - Tính giá trị đơn s của ma trận A bằng căn bậc hai của giá trị riêng. Từ các giá trị c_1, c_2 tính được các giá trị riêng s tương ứng: $s_i = \sqrt{c_i}$

$$\rightarrow s_1 = \sqrt{40} = 6.3245; s_2 = \sqrt{10} = 3.162$$

+ **Bước 3:** Tạo ma trận đường chéo S với các giá trị suy biến $s_1 > s_2 > s_3 > \cdots$ bằng cách đưa các giá trị đơn (đã tính được ở bước 2) sắp xếp giảm dần trên đường chéo.

$$S = \begin{bmatrix} 6.3245 & 0 \\ 0 & 3.1622 \end{bmatrix}$$

+ **Bước 4:** Tính ma trận V^T

- V^T là ma trận trực giao, các vector cột của V là các vector văn bản.
- Tương ứng với giá trị riêng, ta tính vector riêng tương ứng với giá trị riêng bằng cách giải phương trình tuyến tính: $(A.A^T - c_i * I)X_i = 0$, với X_i là vector có số dòng bằng số dòng của ma trận A.
- Với giá trị c_1 = 40

$$A.A^T - c * I = \begin{bmatrix} 25-c & -15 \\ -15 & 25-c \end{bmatrix} = \begin{bmatrix} 25-40 & -15 \\ -15 & 25-40 \end{bmatrix} = \begin{bmatrix} -15 & -15 \\ -15 & -15 \end{bmatrix}$$

– Giải phương trình: $(A.A^T - c_1 * I)X_1 = 0$

$$\Leftrightarrow \begin{bmatrix} -15 & -15 \\ -15 & -15 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

– Giải hệ phương trình: $\begin{cases} -15x_1 - 15x_2 = 0 \\ -15x_1 - 15x_2 = 0 \end{cases} \rightarrow x_2 = -x_1$

$$X_1 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ -x_1 \end{bmatrix}; L = \sqrt{x_1^2 + x_2^2} = x_1\sqrt{2}$$

$$X_1 = \begin{bmatrix} x_1/L \\ -x_1/L \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0.7071 \\ -0.7071 \end{bmatrix}$$

– Tương tự với c_2=10

$$X_1 = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}$$

– Ta có ma trận:

$$V = [X_1 \quad X_2] = \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \rightarrow V^T = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

+ **Bước 5:** Tính ma trận U

– U là ma trận trực giao của ma trận A, các vector dòng của U là các vector thuật ngữ: $U = AVS^{-1}$

$$S^{-1} = \begin{bmatrix} \frac{1}{s_1} & 0 \\ 0 & \frac{1}{s_2} \end{bmatrix}; \frac{1}{s_1} = \frac{1}{6.3245} = 0.1581; \frac{1}{s_2} = \frac{1}{3.1622} = 0.3162$$

$$S^{-1} = \begin{bmatrix} 0.1581 & 0 \\ 0 & 0.3162 \end{bmatrix}$$

$$U = AVS^{-1} = \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix} \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} \begin{bmatrix} 0.1581 & 0 \\ 0 & 0.3162 \end{bmatrix}$$

$$= \begin{bmatrix} 0.4472 & 0.8944 \\ 0.8944 & -0.4472 \end{bmatrix}$$

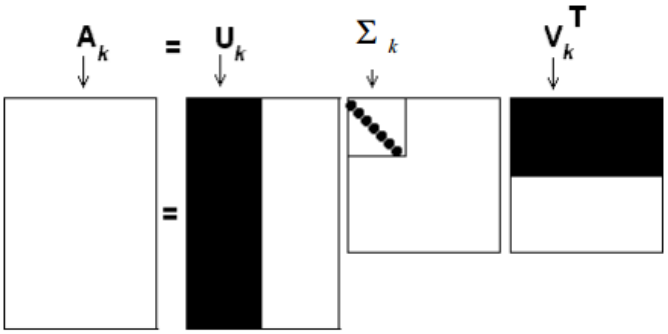
+ **Bước 6:** Tính $A = USV^T$

$$A = USV^T = \begin{bmatrix} 0.4472 & 0.8944 \\ 0.8944 & -0.4472 \end{bmatrix} \begin{bmatrix} 6.3245 & 0 \\ 0 & 3.1622 \end{bmatrix} \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

$$= \begin{bmatrix} 3.9998 & 0 \\ 2.9999 & -4.9997 \end{bmatrix} \approx \begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$$

3.2.4. Xây dựng ma trận xấp xỉ A_k

- Ma trận xấp xỉ $A_k = U_k S_k V_k^T$ có hạng là k. Trong đó, các cột của U_k là k cột đầu tiên của U, các cột của V_k^T là k cột đầu tiên của của V^T và S_k là ma trận đường chéo cấp k x k với các phần tử nằm trên đường chéo là k giá trị suy biến lớn nhất của A. Hình 2.2 dưới đây biểu diễn ma trận xấp xỉ A_k có hạng là k (trong đó $\sum_k$ chính là S_k).



Hình 3.1: Biểu diễn ma trận xấp xỉ A_k có hạng là k

- Trong mô hình LSI, ma trận A_k là xấp xỉ của ma trận từ chỉ mục (term – document A) được tạo ra có ý nghĩa rất quan trọng: phát hiện sự kết hợp ngữ nghĩa giữa các từ chỉ mục được sử dụng trong toàn bộ tập văn bản, loại bỏ những thay đổi trong cách sử dụng từ gây ảnh hưởng xấu đến phương pháp truy tìm theo từ chỉ mục. Vì sử dụng không gian LSI k chiều, nhỏ hơn rất nhiều so với số từ chỉ mục (m từ chỉ mục) nên sự khác nhau không quan trọng trong các từ "đồng nghĩa" được loại bỏ. Những từ chỉ mục thường xuyên xuất hiện cùng nhau trong các văn bản sẽ nằm gần nhau khi biểu diễn trong không gian LSI k chiều, ngay cả các từ chỉ mục không đồng thời xuất hiện

trong cùng một văn bản. Vì vậy, các văn bản không chứa các từ chỉ mục xuất hiện trong câu truy vấn cũng có thể có độ đo tương tự cao với câu truy vấn.

3.2.5. Chọn hệ số k trong mô hình LSI

- Trong mô hình LSI, việc chọn hệ số k để xây dựng ma trận xấp xỉ là một việc hết sức quan trọng đến hiệu quả của thuật toán. Việc chọn hệ số k như thế nào là tối ưu vẫn còn là một bài toán mở, chọn hệ số k quá nhỏ hay quá lớn cũng ảnh hưởng đến hiệu quả truy tìm của thuật toán. Theo các tài liệu nghiên cứu về LSI. Qua thực nghiệm trên các tập dữ liệu văn bản cụ thể, các tác giả chọn k từ 50 đến 100 cho các tập dữ liệu nhỏ và từ 100 đến 300 cho các tập dữ liệu lớn.
- Tuy nhiên các nghiên cứu trên chỉ đưa ra con số k cụ thể dựa vào thực nghiệm trên các tập dữ liệu mẫu cụ thể. Về tổng quát không thể sử dụng các con số trên cho các ứng dụng thực tế khi mà tập dữ liệu có thể chưa xác định trước (có thể tập dữ liệu rất nhỏ hoặc rất lớn). Một phương pháp đề nghị chọn hệ số k gần đây nhất (2003) được đưa ra bởi Miles Efron, tác giả sử dụng phương pháp phân tích giá trị riêng (Eigenvalue) của ma trận từ chỉ mục (Term – Document A) và sử dụng kiểm định thống kê để chọn hệ số k tốt nhất trên dãy các hệ số k được chọn thử nghiệm.
- Ta có thể tính độ sai số của phép xấp xỉ tạo ma trận A_k từ ma trận từ chỉ mục A bằng công thức sau:

$$\min_{rank(B)=k} \|A - B\|_F^2 = \|A - A_k\|_F^2 = \sigma_{k+1}^2 + \cdots + \sigma_{rank\,(A)}^2 \qquad (3.9)$$

- Từ công thức trên ta có thể tính được tỉ lệ thay đổi của ma trận A_k có hạng k so với ma trận A ban đầu là:

$$s = \frac{\|A - A_k\|_F^2}{\|A\|_F^2} = \frac{\sum_{i=k+1}^{rank\,(A)} \sigma_i^2}{\sum_{j=1}^{rank\,(A)} \sigma_j^2} \qquad (3.10)$$

- Vậy thay vì chọn hệ số k thủ công, hệ thống có thể tự động chọn hệ số k dựa vào tỉ số thay đổi của A_k so với A theo chuẩn F. Bài toán bây giờ cần giải quyết là chọn sai số s nào là tốt.
- Gọi error là độ đo sai số cho phép giảm hạng ma trận term – document A thành ma trận A_k có hạng k thì:

$$error = \begin{cases} \|A - A_k\|_F = \left(\sum_{i=k+1}^{rank\,(A)} \sigma_i^2\right)^{\frac{1}{2}} & sai\ số\ tuyệt\ đối \\ \dfrac{\|A - A_k\|_F}{\|A\|_F} & sai\ số\ tương\ đối \end{cases} \quad (3.11)$$

3.3. Tập tin nghịch đảo tài liệu

- Trong các mô hình tìm kiếm thông tin, khái niệm tập tin nghịch đảo tài liệu thường được nhắc đến như một phần không thể thiếu. Mục tiêu giúp quá trình đối sánh câu truy vấn và tài nguyên đã được lập chỉ mục được nhanh.

3.3.1. Phân biệt giữa tập tin nghịch đảo và tập tin trực tiếp

- Tập tin trực tiếp (direct file) là tập tin mà chính các mục thông tin đã cung cấp thứ tự chính của tập tin.
- Ngược lại, tập tin nghịch đảo (inverted file) được sắp xếp theo chủ đề, mỗi chủ đề lại bao gồm một tập các mục thông tin.
- Giả sử có một tập các tài liệu, mỗi tài liệu chứa danh sách các từ. Nếu một từ xuất hiện trong một tài liệu, ghi số 1. Ngược lại, ghi 0. Khi đó tập tin trực tiếp và tập tin nghịch đảo sẽ lưu trữ như sau:

Bảng 3.3: Cách tập tin nghịch đảo lưu trữ

	Tài liệu 1	Tài liệu 2	Tài liệu 3
Từ 1	1	0	1
Từ 2	1	1	0
Từ 3	0	1	1
Từ 4	1	1	1

Bảng 3.4: Cách tập tin trực tiếp lưu trữ

	Tài liệu 1	Tài liệu 2	Tài liệu 3	Tài liệu 4
Từ 1	1	1	0	1
Từ 2	0	1	1	1
Từ 3	1	0	1	1

3.3.2. Sử dụng tập tin nghịch đảo để lập chỉ mục

- Trong hệ thống tìm kiếm thông tin, tập tin nghịch đảo có ý nghĩa rất lớn, giúp việc truy cập đến các mục thông tin được nhanh chóng. Giả sử khi người dùng nhập một câu truy vấn, hệ thống sẽ tách thành 2 từ là "từ 1" và "từ 2". Dựa vào tập tin nghịch đảo, ta dễ dàng xác định được các tài liệu có liên quan đến 2 từ này để trả về cho người tìm kiếm. Tuy nhiên, khó khăn chính của tập tin nghịch đảo là khi thêm một tài liệu mới, tất cả các từ có liên quan đến tài liệu này đều phải được cập nhật lại. Ví dụ khi thêm tài liệu 4 có chứa 2 từ "từ 3" và "từ 4" vào tập tin nghịch đảo.

Bảng 3.5: Thêm một tài liệu mới vào tập tin nghịch đảo

	Tài liệu 1	Tài liệu 2	Tài liệu 3	Tài liệu 4
Từ 1	1	0	1	0
Từ 2	1	1	0	0
Từ 3	0	1	1	1
Từ 4	1	1	1	1

- Rõ ràng việc này tốn một chi phí lớn nếu tập tin nghịch đảo rất lớn. Trong thực tế, tập tin nghịch đảo tài liệu có thể chứa hàng trăm ngàn từ. Tuy nhiên, trong các hệ thống tìm kiếm thông tin, người ta chỉ cập nhật lại tập tin tại một khoảng thời gian định kỳ. Vì vậy, tập tin nghịch đảo vẫn được sử dụng để lập chỉ mục.

3.4. Truy vấn và xếp hạng thông tin

- Đây là quá trình người dùng nhập câu hỏi và yêu cầu tìm kiếm, câu hỏi mà người dùng nhập vào cũng sẽ được xử lý, nghĩa là ta cũng sẽ thực hiện tách từ cho câu hỏi. Sau đó, hệ thống sẽ tìm kiếm trong tập tin chỉ mục để xác định các tài liệu liên quan đến câu hỏi của người dùng.
- Để truy vấn trong mô hình LSI, vector truy vấn q được so sánh với các vector cột trong ma trận xấp xỉ A_k của ma trận term – document A.
- Gọi e_j là vector đơn vị thứ j có số chiều n (cột thứ j của ma trận đơn vị n x n), vector cột thứ j của ma trận A_k là $A_k e_j$. Độ đo cosines của các góc giữa vector truy vấn q và các vector văn bản trong ma trận A_k được tính:

$$\cos\theta_j = \frac{\left(A_k e_j\right)^T q}{\left\|A_k e_j\right\|_2 \|q\|_2} = \frac{\left(U_k S_k V_k^T e_j\right)^T q}{\left\|U_k S_k V_k^T e_j\right\|_2 \|q\|_2} = \frac{e_j^T V_k S_k (U_k^T q)}{\left\|S_k V_k^T e_j\right\|_2 \|q\|_2} \quad (3.12)$$

- Có một cách tiếp cận khác cho thủ tục truy vấn trong mô hình LSI, các văn bản có thể được so sánh với nhau bằng cách tính độ đo cosines các vector văn bản trong

"không gian văn bản" (document space) – chính là so sánh các vector cột trong ma trận V_k^T. Một câu truy vấn q được xem như là một văn bản và giống như một vector cột được thêm vào ma trận V_k^T. Để thêm q như một cột mới vào V_k^T ta phải chiếu q vào không gian văn bản k chiều.

- Từ công thức ma trận $A_k = U_k S_k V_k^T$ ta suy ra $S_k^{-1} U_k^T A_k = V_k^T$ (vì $U_k U_k^T = I_k$ vậy ta có $V_k = A_k^T U_k S_k^{-1}$.
- Áp dụng tương tự cho vector truy vấn q: $q_k = q^T U_k S_k^{-1}$
- Tính độ liên quan giữa vector truy vấn q và vector tài liệu d_i trong ma trận V_k^T bằng công thức sau:

$$sim(q,d) = sim(q^T U_k S_k^{-1}, d^T U_k S_k^{-1}) = \frac{q.d}{|q|.|d|} \qquad (3.13)$$

- Sắp kết quả trả về theo giảm dần độ liên quan.

❖ **Ví dụ 3.2:** Mô tả quá trình lập chỉ mục từ tập văn bản cho trước, tính toán độ liên quan của câu truy vấn và sắp kết quả trả về. Cho tập văn bản sau:

d1: Shipment of gold damaged in a fire

d2: Delivery of silver arrived in a silver truck

d3: Shipment of gold arrived in a truck

Câu truy vấn q: *gold silver truck*

+ **Bước 1:** Xây dựng ma trận team-document A như sau:

Term	*d*1	*d*2	*d*3	*q*
↓	↓	↓	↓	↓
a	1	1	1	0
arrived	0	1	1	0
damaged	1	0	0	0
delivery	0	1	0	0
fire	A= 1	0	0	q=0
gold	1	0	1	1
in	1	1	1	0
of	1	1	1	0
shipment	1	0	1	0
silver	0	2	0	1
truck	0	1	1	1

+ **Bước 2:** Phân tích SVD ma trận A: $A = USV^T$

$$V=\begin{bmatrix} -0.4945 & 0.6492 & -0.5780 \\ -0.6458 & -0.7194 & -0.2556 \\ -0.5817 & 0.2469 & 0.7750 \end{bmatrix} \rightarrow V^T=\begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \\ -0.5780 & -0.2556 & 0.7750 \end{bmatrix}$$

$$U=\begin{bmatrix} -0.4201 & 0.0748 & -0.0460 \\ -0.2995 & -0.2001 & 0.4078 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.1576 & -0.3046 & -0.2006 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.2626 & 0.3749 & 0.1547 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.3151 & -0.6093 & -0.4013 \\ -0.2995 & -0.2001 & 0.4078 \end{bmatrix}; S=\begin{bmatrix} 4.0989 & 0.0000 & 0.0000 \\ 0.0000 & 2.3616 & 0.0000 \\ 0.0000 & 0.0000 & 1.2737 \end{bmatrix}$$

+ **Bước 3:** Chọn hệ số k để tính ma trận xấp xỉ A_k. Giả sử k =2

$$U \approx U_k=\begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3749 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix}; S \approx S_k=\begin{bmatrix} 4.0989 & 0.0000 \\ 0.0000 & 2.3616 \\ 0.0000 & 0.0000 \end{bmatrix}$$

$$V \approx V_k=\begin{bmatrix} -0.4945 & 0.6492 \\ -0.6458 & -0.7194 \\ -0.5817 & 0.2469 \end{bmatrix} \rightarrow V^T \approx V_k{}^T=\begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \end{bmatrix}$$

+ **Bước 4:** Thực hiện truy vấn và sắp kết quả

Các vector tài liệu d trong ma trận V_k^T, mỗi cột thể hiện 01 vector tương ứng:

$d_1(-0.4945, 0.6492)$

$d_2(-0.6458, -0.7194)$

$d_3(-0.5817, 0.2469)$

- Áp dụng công thức q: $q = q^T U_k S_k^{-1}$

$$= [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1] \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3749 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \begin{bmatrix} \frac{1}{4.0989} & 0.0000 \\ 0.0000 & \frac{1}{2.3616} \end{bmatrix}$$

$$q = q^T U_k S_k^{-1} = [-0.2140 \quad -0.1821]$$

- Áp dụng công thức (3.13) Tính độ liên quan giữa vector q với từng vector tài liệu:

$sim(q, d_1) = \frac{(-0.2140).(-0.4945)+(0.1821).(0.6492)}{\sqrt{(-0.2140)^2+(-0.1821)^2}\sqrt{(-0.4935)^2+(0.6492)^2}} = -0.0541$

$sim(q, d_2) = 0.9910$

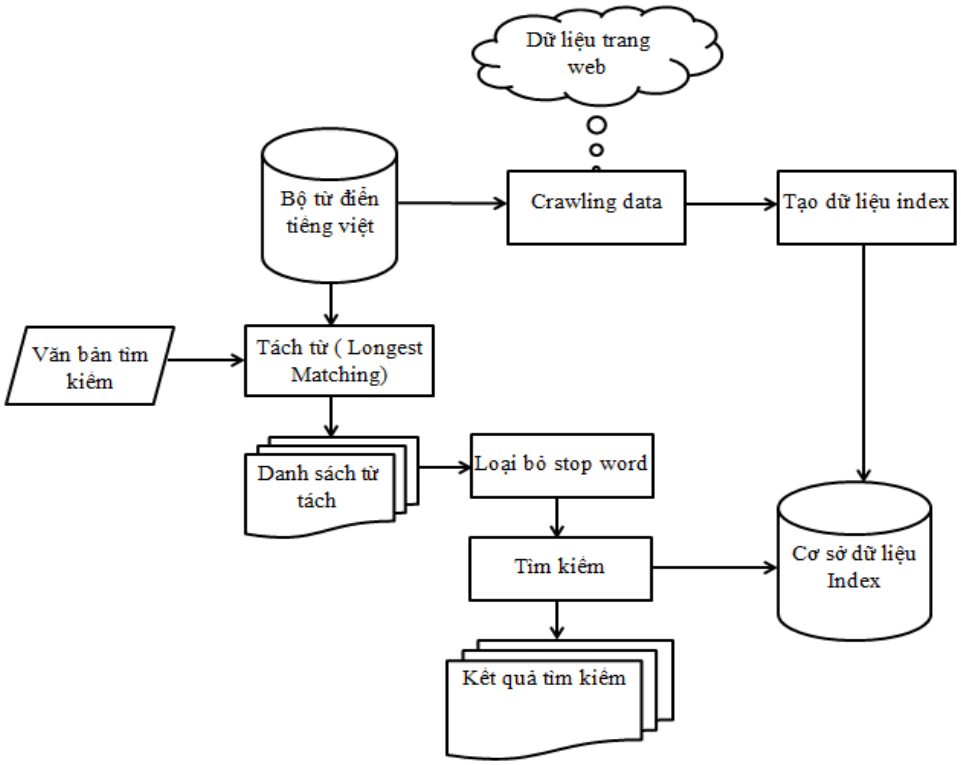
$sim(q, d_3) = 0.4487$

- Sắp độ liên quan giảm dần: $d_2 > d_3 > d_1$

CHƯƠNG 4. ÁP DỤNG VÀO TÌM KIẾM THÔNG TIN TRÊN WEB

4.1. Giới thiệu bài toán

- Chương trình được xây dựng hệ thống tìm kiếm thông tin có sử dụng tách từ tiếng việt nhằm giải quyết bài toán tách từ và tìm kiếm thông tin tiếng việt trên Web.



Hình 4.1: Sơ đồ hệ thống tìm kiếm có sử dụng tách từ tiếng Việt

  + **Đầu vào**: Gồm bộ từ điển được lưu trữ trong máy tính dưới dạng không nén, dữ liệu lấy từ Web crawler về.
  + **Đầu ra**: Danh sách các tập văn bản chứa từ hay cụm từ trong câu truy vấn, tách từ tiếng Việt.
- Nhập 1 câu tách từ trong câu đó ra tìm kiếm từ đơn hoặc từ ghét trong câu đó nếu có sẽ hiện lên đoạn chứa từ cần tìm được tô đậm trên Web.
- Với đầu vào và đầu ra của bài toán như trên thì chương trình phải đáp ứng các yêu cầu sau:
  + Chương trình cho phép thu thập và tạo chỉ mục tài liệu;
  + Cho phép cập nhật lại chỉ mục mỗi khi có tài liệu mới được đưa vào hệ thống.

+ Cho phép người dùng nhập vào câu truy vấn, sau đó dùng phương pháp tách từ để tách câu vừa nhập vào.
+ Cho phép người dùng nhập vào câu truy vấn, sau đó thực hiện tìm kiếm các tài liệu liên quan đến câu truy vấn.
+ Sắp xếp các tài liệu theo thứ tự giảm dần về độ tương quan của tài liệu và câu truy vấn, sau đó hiển thị kết quả cho người dùng.
+ Chương trình sử dụng ngôn ngữ lập trình C#, Server Explorer
+ Công cụ lập trình Visual Studio 2019.
+ Lưu trữ dữ liệu: Web Crawler
+ Hệ thống tìm kiếm sẽ được xây dựng theo mô hình Boolean, không gian Vector VSM và tính trọng số.
+ Các tài liệu tiếng Việt và câu truy vấn sẽ được tách từ theo phương pháp kỹ thuật Longest Matching.

4.2. Chức năng của chương trình

- Chương trình được xây dựng với các chức năng chính sau:
  + Lập chỉ mục cho các từ tạo nên tài liệu.
  + Chọn lọc các từ có giá trị phân biệt cao làm chỉ mục.
  + Tách từ từ các tài liệu.
  + Cập nhật lại chỉ mục khi thêm tài liệu mới.
  + Hiển thị kết quả tìm kiếm cho người dùng.

4.3. Lập chỉ mục

4.3.1. Lớp lập chỉ mục

- Đầu vào của lớp lập chỉ mục là nội dung trang Web, kết quả là danh sách chỉ mục được lưu vào cơ sở dữ liệu.
- Các hàm chính: Lấy danh sách các trang Web từ địa chỉ Websites.
- GetListUrlFromWeb (địa chỉ Website)

  {

  // Đọc qua nội dung trang Web và lưu lại tất cả các địa chỉ trong đó.

  }

  Tạo chỉ mục cho từng trang Web.

CreateIndex (địa chỉ trang Web)

{

// Đọc nội dung trang Web và tách từ sau đó lưu lại thành chỉ mục.

}

Lưu chỉ mục xuống cơ sở dữ liệu

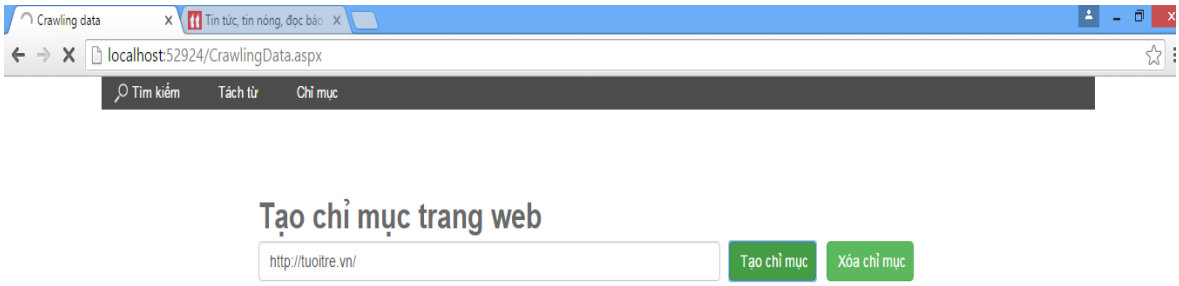
SaveIndex (danh sách chỉ mục)

{

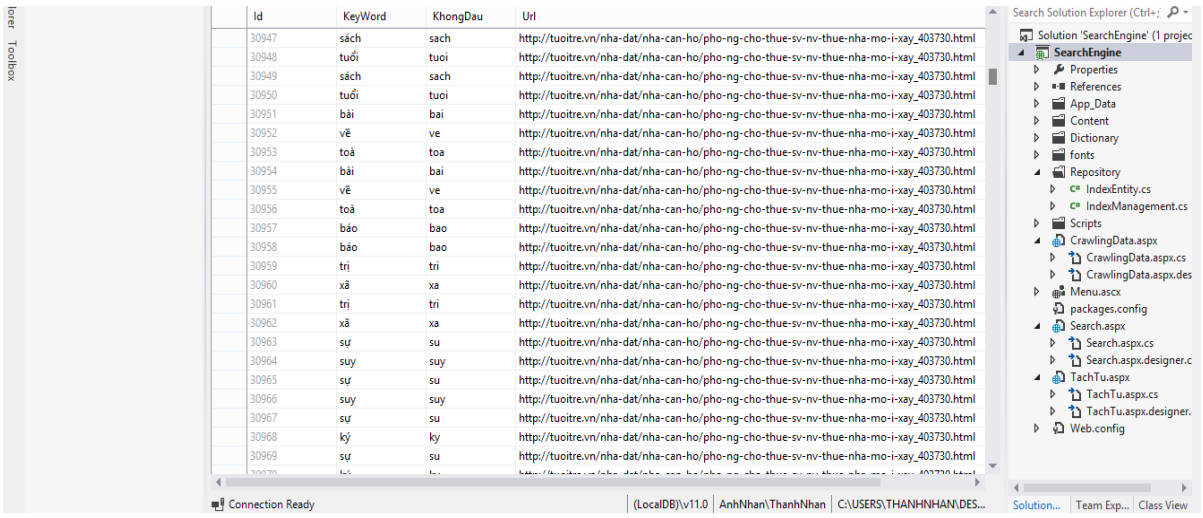
// Lưu danh sách chỉ mục xuống cơ sở dữ liệu .

}

4.3.2. Giao diện lập chỉ mục



Hình 4.2: Màn hình tạo chỉ mục



Hình 4.3: Màn hình lấy dữ liệu index

4.4. Tách từ

- Tách từ là rất quan trọng trong chương trình có thể thực hiện đúng và chính xác việc phân loại hay không là nhờ kết quả của việc tách từ đúng hay sai.

4.4.1. Lớp tách từ

- Lớp tách từ sẽ có nhiệm vụ tách các từ được nhập vào bởi người dùng
- Lớp tạo chỉ mục: Đọc dữ liệu từ trang Web được nhập vào bởi người dùng, sau đó phân tích và tạo chỉ mục cho từng từ và lưu lại vào cơ sở dữ liệu.

4.4.2. Các hàm chính

- Hàm tách câu tìm kiếm thành từ:
- Thuật toán:

void TachTu (câu văn bản)

{

// Loại bỏ dấu chấm, dấu phẩy trong văn bản.

// Tạo danh sách từ ngữ bằng cách chia văn bản dựa trên khoảng trắng của các từ.

// Loại bỏ từ rỗng.

While (duyệt qua danh sách các từ)

{

// Gán từ đầu tiên trong danh sách là từ tìm được.

// Kiểm tra từ vừa tìm được và từ kế tiếp xem có tồn tại trong từ điển hay không.

// Nếu tồn tại sẽ tiếp tục vòng lặp.

// Nếu không tồn tại sẽ thêm từ tìm được trước đó vào danh sách kết quả tách từ.

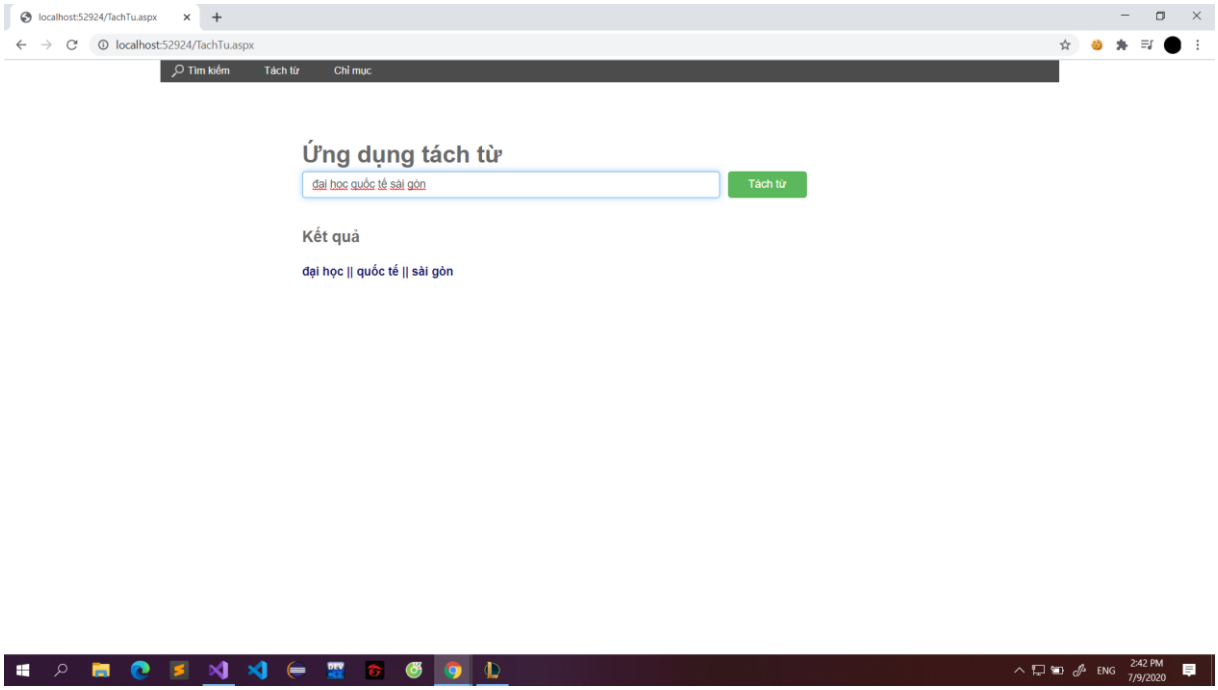
// Gọi hàm XacDinhTu.

} }

- Ví dụ: Chuỗi đầu vào: “Sinh viên VN: động lực cho những sáng tạo mới, tầm nhìn mới”.
- Chuỗi đầu vào =“Sinh viên VN: động lực cho những sáng tạo mới, tầm nhìn mới” trả về là mảng chuỗi chứa các tiếng ={“động”, “lực”, “cho”, “những”, “sáng”, “tạo”, “mới”}
- Hàm XacDinhTu(): gộp các tiếng lại thành từ, so sánh trong từ điển tiếng việt và ta sẽ lưu lại các từ này vào mảng từ.
- Thuật toán:

```
void XacDinhTu (mảng các tiếng)
{
        B1: gán từ = tiếng đầu tiên.
        B2: So sánh từ có trong từ điển hay không.
        B3: Nếu từ có trong từ điển và có 2 tiếng trở lên thì ta sẽ lưu lại.
        B4: Nếu trong mảng tiếng vẫn còn thì:
                từ: = từ + tiếng tiếp theo.
        Ngược lại kết thúc hàm.
        B5: Quay lại B2
}
```

4.4.3. Giao diện tách từ



Hình 4.4: Màn hình chi tiết tách từ

4.5. Tìm kiếm

- Lớp tìm kiếm sẽ có nhiệm vụ tách từ câu hỏi, loại bỏ các từ trong danh sách Stopword, sau đó tìm các từ khóa của câu hỏi do người dùng nhập vào, cuối cùng tìm kiếm trong cơ sở dữ liệu các chỉ mục tương ứng và hiển thị cho người dùng xem kết quả.

4.5.1. Các hàm chính:

- Đọc danh sách từ điển

ReadDictionaryData()

{

Đọc danh sách từ file xml

}

Hàm tách từ

WordDivision()

{

}

Hàm tìm kiếm

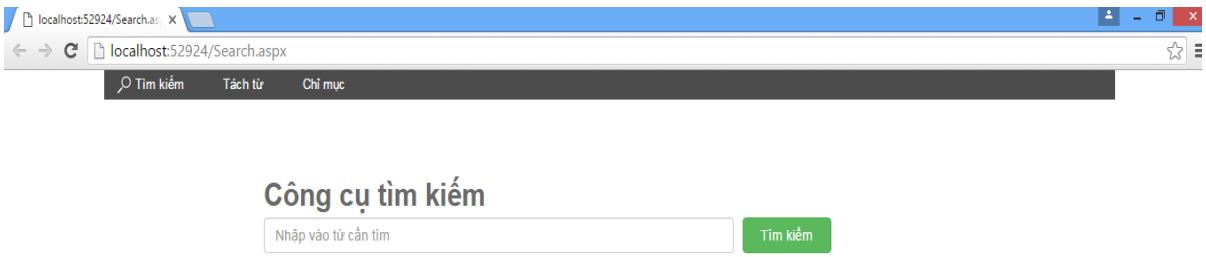
SearchFromDatabase()

{

// Đọc dữ liệu từ cơ sở dữ liệu và kiểm tra với danh sách tách từ để lấy ra kết quả tìm kiếm và hiển thị cho người dùng

}

4.5.2. Giao diện tìm kiếm



Hình 4.5: Màn hình tìm kiếm

4.6. Kết quả thực nghiệm

Tình huống thử nghiệm	Thời gian (Phút)	Số lượng từ thực tế	Đếm bằng chương trình	Phần trăm chính xác
1	0.5	200	200	100%
2	1.2	1000	1000	100%
3	3	Tất cả	Tất cả	95%

Thời gian lấy dữ liệu từ Web về lưu vào cơ sở dữ liệu

Tình huống thử nghiệm	Thời gian (Giây)	Số lượng từ thực tế	Phần trăm chính xác
1	20	2	100%
2	23	4	100%
3	25	8	100%

Thời gian tách từ và đọc dữ liệu đưa lên màn hình

Tình huống thử nghiệm	Thời gian (Giây)	Số lượng từ thực tế	Đếm bằng chương trình	Phần trăm chính xác
1	15	2	2	100%
2	16	7	7	100%
3	16	8	8	100%

Thời gian tra cứu các từ cần tìm

KẾT LUẬN VÀ KIẾN NGHỊ

1. Kết luận

- Kỹ thuật tách từ tiếng việt đã và đang là một vấn đề mang tính thời sự của Công nghệ thông tin. Đề tài này đã hoàn thành được những yêu cầu đề ra.
- Về lý thuyết. Tách từ và tìm kiếm thông tin tiếng việt là một bài toán khó và thú vị. Khó bởi vì vấn đề văn bản cần phải xử lý ngôn ngữ tiếng việt, mà như chúng ta đều biết, ngôn ngữ tự nhiên là muôn hình, phong phú cả về từ vựng, cú pháp và phức tạp về ngữ nghĩa. Nhưng cũng là bài toán thú vị vì với mỗi ngôn ngữ khác nhau thì phải thực hiện những cách xử lý khác nhau đối với ngôn ngữ. Đề tài này đã đề cập được một số vấn đề mang tính chất cơ sở về một số kỹ thuật tách từ và tìm kiếm thông tin theo nội dung trong tài liệu mô hình Boolean, không gian vector có tính trọng số. Những vấn đề liên quan đến đề tài như các phương pháp tách từ và phương pháp lập chỉ mục đã được nghiên cứu khá công phu theo cả chiều rộng và chiều sâu.Về lý thuyết: Tìm kiếm thông tin tiếng Việt trên web là một bài toán khó và thú vị. Khó bởi vì vấn đề văn bản cần phải xử lý ngôn ngữ, mà như chúng ta đều biết, ngôn ngữ tự nhiên là muôn hình, phong phú cả về từ vựng, cú pháp và phức tạp về ngữ nghĩa. Nhưng cũng là bài toán thú vị vì với mỗi ngôn ngữ khác nhau thì phải thực hiện những cách xử lý khác nhau đối với ngôn ngữ. Bản luận văn này đã đề cập được một số vấn đề mang tính chất cơ sở: crawler dò tìm thông tin, phương pháp tách từ, phương pháp lập chỉ mục và sắp kết quả trả về, một số mô hình tìm kiếm thông tin hiện nay.
- Về thực nghiệm: Đề tài đã xây dựng chương trình thực nghiệm tìm kiếm thông tin tiếng Việt trên web với đầy đủ các tính năng: Crawler dò tìm thông tin trên web, lập chỉ mục và kỹ thuật tìm kiếm thông tin bằng mô hình lập chỉ. Chương trình hỗ trợ giao diện Web cho người sử dụng tìm kiếm, được phát triển trên môi trường Visual Studio 2019.
- Nhìn chung: đề tài đã hoàn thành được những yêu cầu đề ra và có một số ưu điểm như sau:
  + Nghiên cứu được cách thức hoạt động của một hệ thống tìm kiếm thông tin có sử dụng tách từ tiếng Việt.
  + Hệ thống tách từ tiếng Việt khá chính xác.
  + Các yêu cầu liên quan đến lập chỉ mục và tra cứu.

+ Tìm kiếm khá nhanh: các tài liệu trả về được sắp xếp khá chính xác.
+ Tóm tắt được nội dung trả về.
+ Giao diện thân thiện, dễ dùng.

2. Khuyến nghị

- Đây là một đề tài có tính thực tế cao. Với nhiệm vụ là nghiên cứu, đề tài đã đáp ứng được một số yêu cầu cơ bản đặt ra. Tuy nhiên, việc ứng dụng chỉ mục ngữ nghĩa ngầm LSI sao cho hiệu quả vẫn đang là bài toán mở. Do đó hướng phát triển của đề tài như sau:
  + Xây dựng tính năng truy tìm thông tin trên web theo định kỳ.
  + Xây dựng bộ từ điển có độ chính xác cao.
  + Thêm chức năng cập nhật Singular Value Decomposition (SVD)
  + Nghiên cứu tính năng tự động hóa chọn hệ số k cho tính toán A_k

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1]. Đinh Điền (2004), "*Xử lý ngôn ngữ tự nhiên" ,* Đại học khoa học Tự nhiên Tp. Hồ Chí Minh.

Tiếng Anh:

[2]. Cherukuri Aswani Kumar, Suripeddi Srinivas (2006), *"Latent Semantic Indexing Using Eigenvalue Analysis for Efficient Information Retrieval"*, Comput. Sci, Int. J. Appl. Math., Vol. 16, No. 4, pp. 551–558.

[3]. Canfora, G. and L. Cerulo (2004), *"A Taxonomy of Information Retrieval Models and Tools"*, Journal of Computing and Information Technology, 12 (3): p. 175-194.

[4]. Filippo Menczer, Gautam Pain, Padmini Srinivasan (November - 2004), *Topical Web Crawlers: Evaluating Adaptive Algorithms,* Indiana University.

[5]. Christopher D. Manning, Prabhakar Raghavan and Schütze (2008) "*Introduction Information Retrieval"*, Cambridge University Press.

[6]. Jerri Ledford (2009), *Search Engine Optimization Bible,* Second Edition, published by Wiley Publishing, Inc.

[7]. Kontostathis (2007), *"Essential Dimensions of latent sematic indexing (LSI)"*, Proceedings of the 40th Hawaii International Conference on System Sciences.

[8]. Gautam Pant, Padmini Srinivasan, and Filippo Menczer (2004), *Crawling the Web*, The University of Iowa, Iowa City IA 52242, USA.

[9]. Marc Najork, Allan Heydon (2001), *High-Performance Web Crawling,* 130 Lytton Avenue Palo Alto, California 94301.

[10]. Maron, M.E., Kuhns, J.L (1960), *"On Relevance, Probabilistic Indexing and Information Retrieval"*, J. ACM 7, 216-244.

[11]. Michael W. Berry, Zlatko Drmac, Elizabeth R. Jessup (1999), *"Matrix, Vector Space, and Information Retrieval"*, Siam Review, Vol 41, No. 2, pp. 335 – 352.