**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**Bùi Đức Lực**

**Nguyễn Hải Đăng**

**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG WEBSITE HỎI ĐÁP THÔNG TIN BÓNG ĐÁ**

**CỬ NHÂN NGÀNH KHOA HỌC MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2016**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**BÙI ĐỨC LỰC \_ 12520636**

**NGUYỄN HẢI ĐĂNG \_ 12520556**

**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG WEBSITE HỎI ĐÁP THÔNG TIN BÓNG ĐÁ**

**CỬ NHÂNNGÀNH KHOA HỌC MÁY TÍNH**

**GIẢNG VIÊN HƯỚNG DẪN**

**THS. NGUYỄN TRỌNG CHỈNH**

**TP. HỒ CHÍ MINH, 2016**

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

* 1. …………………………………………. – Chủ tịch.
  2. …………………………………………. – Thư ký.
  3. …………………………………………. – Ủy viên.
  4. …………………………………………. – Ủy viên.

**Lời cám ơn!**

Được sự phân công của khoa Khoa Học Máy Tính trường Đại Học Công Nghệ Thông Tin, và sự đồng ý của Thầy giáo hướng dẫn Th.s Nguyễn Trọng Chỉnh nhóm đã thực hiện đề tài “Xây Dựng website hỏi đáp thông tin bóng đá”. Để hoàn thành tốt đề tài này, trước tiên, chúng em xin chân thành cảm ơn các thầy cô và ban giám hiệu trường ĐH Công Nghệ Thông Tin nói chung và các thầy cô trong khoa nói riêng đã tạo mọi điều kiện giúp đỡ chúng em trong quá trình học tập và nghiên cứu.

Kế đến, chúng em xin chân thành gửi lời cảm ơn đến thầy Nguyễn Trọng Chỉnh, là giáo viên hướng dẫn và cũng là người đã nhiệt tình chỉ bảo cho chúng em trong suốt quá trình hoàn thành đề tài.

Cuối cùng xin gửi lời cảm ơn đặc biệt đến gia đình luôn là một chỗ dựa vững chắc cũng như luôn tạo mọi điều kiện tốt nhất cho chúng em được học tập.

Tp Hồ Chí Minh, 10/06/2016

Nhóm sinh viên thực hiện

Bùi Đức Lực

Nguyễn Hải Đăng

MỤC LỤC

[Chương 1. TÊN CHƯƠNG 1 3](#_Toc367742496)

[1.1. Chủ đề cấp độ 2 3](#_Toc367742497)

[1.1.1. Chủ đề cấp độ 3 3](#_Toc367742498)

[1.1.2. Chủ đề cấp độ 3 3](#_Toc367742499)

[1.1.2.1. Chủ đề cấp độ 4 3](#_Toc367742500)

[Chương 2. TÊN CHƯƠNG 2 4](#_Toc367742501)

[2.1. Chủ đề cấp độ 2 4](#_Toc367742502)

[2.1.1. Chủ đề cấp độ 3 4](#_Toc367742503)

[2.1.1.1. Chủ đề cấp độ 4 4](#_Toc367742504)

[2.2. Chủ đề cấp độ 2 4](#_Toc367742505)

[2.2.1. Chủ đề cấp độ 3 4](#_Toc367742506)

[Chương 3. TÊN CHƯƠNG 3 5](#_Toc367742507)

[3.1. Chủ đề cấp độ 2 5](#_Toc367742508)

[3.1.1. Chủ đề cấp độ 3 5](#_Toc367742509)

[3.1.1.1. Chủ đề cấp độ 4 5](#_Toc367742510)

[3.2. Chủ đề cấp độ 2 5](#_Toc367742511)

DANH MỤC HÌNH VẼ

[Hình 1.1: Tên hình 1 3](#_Toc367742554)

DANH MỤC BẢNG

[Bảng 1.1: Tên bảng 1 3](#_Toc367742567)

[Bảng 2.1: Tên bảng 1 4](#_Toc367742568)

DANH MỤC TỪ VIẾT TẮT

TÓM TẮT KHÓA LUẬN

Khóa luận được thực hiện dưới dạng một ứng dụng web. Ứng dụng nhận input là câu hỏi dưới dạng ngôn ngữ tự nhiên và output là câu trả lời liên quan đến nội dung câu hỏi. Trong khóa luận này, nhóm thực hiện nghiên cứu ba vấn đề về lớn là: khai phá tri thức và tìm kiếm thông tin, rút trích thông tin, rút trích câu trả lời.

Vấn đề khai phá tri thức và tìm kiếm thông tin được thực hiện bằng công cụ tìm kiếm Bing và Google

Vấn đề rút trích thông tin được thực hiện bằng bộ thư viện lucene hổ trợ việc rút trích thông tin cần thiết từ nguồn dữ liệu lớn.

Vấn đề tìm kiếm câu trả lời được thực hiện bằng cách sử dụng biểu thức chính quy để rút trích câu trả lời.

MỞ ĐẦU

Với nhu cầu trao đổi thông tin của con người ngày càng cao, thông tin tràn ngập trên mọi phương tiện truyền thông, đặc biệt là sự phát triển rộng rãi của mạng toàn cầu Internet, hằng ngày con người phải xử lý một lượng thông tin khổng lồ. Những thắc mắc của người dùng dưới dạng truy vấn sẽ được tìm kiếm và trả về một cách ngắn gọn, súc tích, chính xác nhất những gì mà họ mong muốn. Đó chính là mục tiêu của hệ thống hỏi-đáp tự động. Rất nhiều hệ thống hỏi đáp thông tin qua mạng ra đời nhằm đáp ứng nhu cầu này.

Qua báo cáo này, Nhóm sẽ trình bày những gì đã tìm hiểu được về một Hệ Thống Hỏi Đáp Tự Động, các kỹ thuật liên quan đến việc xây dựng hệ thống. Từ đó xây dựng một Hệ Thống Hỏi Đáp Tự Động cụ thể.

Cấu trúc của bài báo cáo được trình bày như sau:

Chương 1: Giới thiệu vấn đề. Trong chương này giới thiệu về cụ thể về vấn đề, mục tiêu của hệ thống, giới hạn và giải pháp để giải quyết vấn đề.

Chương 2: Các kỹ thuật được sử dụng trong hệ thống. Chương này giới thiệu về hệ thống hỏi đáp (khái niệm, lịch sử phát triền, phân loại, mô hình chung của một hệ thống hỏi đáp, phương pháp đánh giá), giới thiệu sơ lược về khái niệm tìm kiếm thông và một số công cụ tìm kiếm thông tin (Bing API, Google API, Lucene), giới thiệu tổng quan về ứng dụng web (khái niệm, mô hình của ứng dụng web).

Chương 3: Phương pháp trả lời câu hỏi. Chương này trình bày về các quá trình cụ thể để trả lời câu hỏi về lĩnh vực bóng đá (Quá trình xác định dạng câu hỏi, phân tích câu hỏi, rút trích câu trả lời, xếp hạng câu trả lời, trả kết quả).

Chương 4: Phân tích thiết kế và cài đặt hệ thống. Chương này trình bày chức năng của hệ thống, mô hình của hệ thống với chức năng của từng phần trong hệ thống, các lớp được cài đặt để thể hiện chức năng của từng thành phần. Giới thiệu các vấn đề phần mềm đã giải quyết được.

Chương 5: Đánh giá, kết luận và hướng phát triển. Chương này đánh giá kết quả theo bộ số liệu, nêu kết luận và hướng phát triển.

# GIỚI THIỆU VẤN ĐỀ

## Giới thiệu bái toàn

Nguồn dữ liệu trên internet đang ngày một tăng lên không ngừng, cộng với nhu cầu trao đổi thông tin của con người ngày càng cao. Đặc biệt trong lĩnh vực bóng đá thì nguồn dữ liệu về thông tin bóng đá là rất lớn, và nhu cầu tìm kiếm thông tin về bóng đá của con người hàng ngày cũng nhiều không kém. Hiện nay các công cụ tìm kiếm như Bing hay Google sẽ giúp người dùng tìm kiếm được những thông tin cần thiết. Nhưng kết quả mà Bing hay Google trả về cho chúng ta sẽ không phải thông tin chính xác mà chúng ta cần tìm, mà sẽ là các đường link dẫn đến các trang chưa thông tin. Nguồn dữ liệu trả về này sẽ rất lớn có khi lên đến hàng nghìn trang web khác nhau. Do đó sẽ không tránh khỏi việc xuất hiện dữ liệu thừa không liên quan đến câu hỏi, đến thông tin người dùng mong muốn.Người sử dụng muốn có được thông tin mình cần phải tự duyệt và đọc lần lượt các trang web để xác định thông tin mình cần. Nếu một trang web lớn chứ nhiều dữ liệu thì việc tìm kiếm thông tin cần thiết sẽ là rất khó khăn và sẽ mất khá nhiều thời gian, chưa kể đến việc thông tin chúng ta cần sẽ không có trong link đó và chúng ta phải chuyển sang link khác.

Vậy vấn đề đặt ra ở đây là liệu có hệ thống nào, phần mềm nào sẽ giúp chúng ra việc đó, giúp chúng ta tìm kiếm thông tin cách chính xác mà lại không mất nhiều thời gian. Hệ thống giúp chúng ta làm điều này có tên là “Hệ thống hỏi đáp”. Trong đề tài này, nhóm sẽ xây dựng một ứng dụng dạng web, thực hiện việc trả lời những câu hỏi liên quan đến thông tin bóng đá do người dùng nhập vào.

## Mục tiêu hệ thống.

Hệ thống được xây dựng với mục tiêu sau:

* Xây dựng giao diện trực quan, dễ dàng sử dụng.
* Phân tích được dạng câu hỏi do người dùng nhập vào dưới dạng ngôn ngữ tự nhiên.
* Rút trích được thông tin cần thiết từ nguồn dữ liệu khổng lồ trên internet dựa vào câu hỏi.
* Trả lời được các dạng câu hỏi một cách chính xác và nhanh chóng. Tiết kiệm thời gian tìm kiếm thông tin cho người dùng.

## Giải pháp cho hệ thống.

Để giải quyết được vấn đề của hệ thống, nhóm đã nghiên cứu và đưa ra giải pháp gồm các quá trình được thực hiện tuần tự sau:

* Thu thập dữ liệu liên quan: để thu thập dữ liệu liên quan đến mục đích câu hỏi sẽ sử dụng nguồn dữ liệu có sẵn trên internet, thông qua đó dùng công cụ tìm kiếm để lấy thông tin cần thiết tránh lấy thông tin thừa gây nhiễu dữ liệu.
* Thu thập bộ câu hỏi bao quát các dạng câu hỏi liên quan tới lĩnh vực, cụ thể là lĩnh vực bóng đá.
* Xác định dạng câu hỏi, phân tích câu hỏi do người dùng nhập vào dưới dạng ngôn ngữ tự nhiên.
* Thu hẹp không gian tìm kiếm. Từ nguồn dữ liệu thu thập được, thực hiện rút trích thông tin cần thiết liên quan đến mục đích của câu hỏi.
* Rút trích câu trả lời.

## Phạm vi đề tài.

Câu hỏi được giới hạn là kiểu câu đơn với nội dung câu hỏi được xác định:

* Thông tin về cầu thủ: năm sinh, tên, tuổi, chiều cao, số áo…
* Thông tin về trận đấu: thời gian diễn ra, thời gian kết thúc, kết quả của trận đấu…
* Thông tin về giải đấu: thời gian khai mạc, bế mạc, địa điểm,..

# CÁC VẤN ĐỀ LIÊN QUAN

## Giới thiệu về hệ thống hỏi đáp

### Tổng quan về hệ thống hỏi đáp tự động

Hệ thống hỏi-đáp tự động (Question Answering-QA) là một hệ thống được xây dựng để thực hiện việc tìm kiếm câu trả lời cho một câu hỏi của người dùng. Hệ thống hỏi-đáp tự động liên quan đến 3 lĩnh vực lớn là xử lý ngôn ngữ tự nhiên (Natural Language Processing), tìm kiếm thông tin (Information Retrieval) và rút trích thông tin (Information Extraction).

Hệ thống hỏi đáp tự động có thể coi như một lựa chọn thứ hai bên cạnh hệ thống trích chọn thông tin khi người dùng muốn tìm kiếm thông tin họ cần. Hệ thống trích chọn thông tin nhận đầu vào là các từ khóa và trả về tập các tài liệu liên quan (có chứa các từ khóa đó). Kết quả mà hệ thống trích chọn thông tin (máy tìm kiếm) trả lại cho người dùng là rất lớn, có thể lên đến hàng nghìn trang web mà phần nhiều không chứa thông tin người dùng mong muốn. Trong khi đó, hệ thống hỏi đáp nhận đầu vào là câu hỏi dưới dạng ngôn ngữ tự nhiên của người dùng, trả lại các đoạn văn bản ngắn (các snippet) chứa câu trả lời trực tiếp cho câu hỏi. Nghiên cứu về hệ thống hỏi đáp tự động hiện đang thu hút sự quan tâm của rất nhiều các nhà nghiên cứu từ các trường đại học, các viện nghiên cứu và cả các doanh nghiệp lớn trong ngành công nghệ thông tin, có ý nghĩa khoa học lẫn ý nghĩa thực tế. Rất nhiều các hội nghị thường niên về khai phá dữ liệu, trích chọn thông tin dành một chủ đề riêng cho các nghiên cứu về hệ thống hỏi đáp như TREC1 , CLEF2 … Bài toán xây dựng hệ thống hỏi đáp là một bài toán khó thuộc lĩnh vực xử lý ngôn ngữ tự nhiên. Chúng ta biết rằng ngôn ngữ tự nhiên vốn nhập nhằng, đa nghĩa, việc xác định được ngữ nghĩa của câu hỏi cũng như phát hiện ra câu trả lời là một thách thức không nhỏ. Không những vậy, giữa câu hỏi và câu trả lời còn tồn tại các quan hệ “ngầm” hay phụ thuộc vào ngữ cảnh.

### Lịch sử phát triển của hệ thống hỏi đáp

Nghiên cứu về QA đã bắt đầu từ những năm 60 của thế kỷ trước. Tuy nhiên, những khó khăn trong việc xử lý ngôn ngữ tự nhiên đã giới hạn việc nghiên cứu QA vào việc nghiên cứu các hệ chuyên gia trả lời câu hỏi trong các lĩnh vực nhỏ hẹp. Vào những năm gần đây, nghiên cứu về hệ QA tổng quát đã được đẩy mạnh cùng với sự xuất hiện của tiểu ban QA (QA Track) ở hội nghị TREC ( Text Retrieval Conference), sự thảo luận về QA ở diễn đàn CLEF (Cross Language Evaluation Forum).

Thiết kế một hệ thống hỏi-đáp không phải là khái niệm mới. Một số hệ thống đầu tiên đã được ra đời từ những năm 1960. Một ví dụ của một hệ thống như vậy là hệ thống BASEBALL được phát triển năm 1961 do nhóm tác giả Green, Chomsky, và Laughery . Hệ thống này được thiết kế để cung cấp các thông tin về các số liệu thống kê của liên đoàn bóng chày Mỹ. Một hệ thống khác tương tự như BASEBALL đã được phát triển bởi Woods (1973) và được đặt tên LUNAR. LUNAR có thể trả lời các câu hỏi liên quan đến các mẫu đá trở về từ tàu thăm dò mặt trăng Apollo. Hệ thống này dịch các câu hỏi thành một hay nhiều truy vấn cơ sở dữ liệu. Hệ thống TEAM được phát triển bởi Grosz (1983) đã có những đặc điểm cơ bản như một chuỗi các biểu diễn ngữ nghĩa và một lược đồ phiên dịch làm cho nó tốt hơn 2 hệ thống trước. Điểm tương đồng giữa cả ba hệ thống là tất cả đều sử dụng cơ sở dữ liệu để lưu trữ cơ sở tri thức. Việc thiết kế của cơ sở dữ liệu này và các dữ liệu có cấu trúc được xây dựng một cách thủ công bởi tất cả các chuyên gia trong các lĩnh vực tương ứng.

Xử lý ngôn ngữ tự nhiên trong quá trình thực hiện đã nhận được rất nhiều sự quan tâm trong những năm 1970. Một trong những hệ thống như thế là Margie (Schank, Goldman, Riesbeck, & Rieger, 1975). Hệ thống này đã có thể xử lý các tài liệu mà sau này có thể trả lời các câu hỏi cơ bản. Điều này được thực hiện bằng cách phân tích và tổ chức tài liệu của thông tin trong một mô hình tương tự như bộ nhớ của con người. Điều này đã được cải thiện hơn bởi Lenhert, Dyer, Johnson, Yang, và Harley (1983) trong hệ thống Boris. Boris và Margie là các hệ thống gần giống nhất với hệ thống hỏi-đáp lĩnh vực rộng (open-domain) hoạt động ngày nay.

Hệ thống hỏi-đáp nhận được rất nhiều mối quan tâm của cộng đồng các nhà nghiên cứu. Điều này bắt nguồn từ việc một lượng tài liệu lớn có gắn chỉ mục sẵn có trên Internet, cùng với các thành tựu trong lĩnh vực rút trích thông tin và sự gia tăng nhu cầu thương mại cho các sản phẩm QA trên toàn cầu. Một lợi thế nữa, dễ dàng cho việc phát triển các hệ thống QA phức tạp hơn, là các nguồn tài nguyên như là WordNet (Fellbaum, 1998) và OpenCyc (Curtis, Matthews, & Baxter, 2005) . WordNet cung cấp sự truy cập tới các thông tin ngữ nghĩa và các mối liên hệ ngữ nghĩa giữa các cụm từ. OpenCyC cung cấp sự truy cập tới thông tin ontology và tri thức thông dụng. Cả 2 hệ thống này đều được cung cấp miễn phí. Hội thảo TREC (Text Retrieval and Evaluation Conference) về lĩnh vực QA năm 1999 đã đóng góp một phần to lớn cho sự phát triển của lĩnh vực QA với 20 thành viên tham gia. Sự kiện này đã thúc đẩy các nhà nghiên cứu chia sẻ kinh nghiệm và so sánh kết quả của họ sử dụng một độ đo chung.

Năm 2000, Carbonell và các đồng nghiệp trong bài báo The Vision Statement to Guide Research in Question Answering and Text Summarization đã đưa ra các tư tưởng chung cho việc nghiên cứu Q&A. Theo đó một hệ thống hỏi đáp được người dùng đánh giá là hữu ích nếu đáp ứng được các tiêu chuẩn:

* Tính hợp lý về thời gian (Timeliness): Câu trả lời phải được đưa ra trong thời gian ngắn, ngay cả khi có hàng ngàn người dùng cùng truy nhập hệ thống một lúc. Các nguồn dữ liệu mới cần phải được tích hợp vào hệ thống ngay khi chúng sẵn sàng để có thế cung cấp cho người dùng câu trả lời cho những câu hỏi về các sự kiện có tính thời sự.
* Tính chính xác: Tính chính xác của hệ thống hỏi đáp tự động là cực kì quan trọng bởi việc đưa ra câu trả lời sai còn tai hại hơn nhiều là không đưa ra câu trả lời. Nghiên cứu về Q&A cần tập trung vào việc đánh giá tính đúng đắn của câu trả lời đưa ra, bao gồm cả phương thức để phát hiện các trường hợp mà dữ liệu hiện thời không chứa câu trả lời cho câu hỏi. Các thông tin mâu thuẫn trong dữ liệu cũng cần được tìm ra và các thông tin này cần được xử lý theo một cách phù hợp, nhất quán. Để đạt được sự chính xác, hệ thống Q&A cần được tích hợp các nguồn tri thức (world knowledge ) và cơ chế “bắt chước” việc suy luận thông thường (việc bắt chước có thể hiểu như là một quá trình học).
* Tính khả dụng: Hệ thống Q&A cần đáp ứng được các yêu cầu cụ thể của một người dùng. Các ontology trên từng miền cụ thể và ontology trên miền mở cần được tích hợp trong hệ thống. Hệ thống Q&A cần có khả năng khai phá câu trả lời 5 từ bất kì dạng dữ liệu gì (văn bản, web, cơ sở dữ liệu, …) và đưa ra câu trả lời dưới định dạng mà người dùng mong muốn, cho phép người dùng miêu tả ngữ cảnh của câu hỏi và cung cấp các thông tin giải thích, trích dẫn nguồn cho câu trả lời.
* Tính hoàn chỉnh: Câu trả lời hoàn chỉnh cho câu hỏi của người dùng là điều mà các hệ thống Q&A hướng tới. Trong nhiều trường hợp (câu hỏi về danh sách, nguyên nhân, cách thức…), các phần của câu trả lời nằm rải rác trong một văn bản, thậm chí trong nhiều văn bản. Vì vậy cần phải hợp nhất các phần này dựa trên các thông tin liên kết để tạo ra câu trả lời hoàn chỉnh.
* Tính thích hợp của câu trả lời: Trong ngôn ngữ tự nhiên, câu hỏi đưa ra luôn gắn với ngữ cảnh nào đó và câu trả lời cũng nằm trong một ngữ cảnh nhất định. Câu trả lời mà hệ thống Q&A đưa ra phải phù hợp ngữ cảnh với câu hỏi. Một hệ thống Q&A có khả năng giao tiếp (interactive Q&A) là cần thiết trong nhiều trường hợp bởi chuỗi các câu hỏi liên quan đến một vấn đề sẽ giúp làm sáng tỏ thông tin mà người dùng đang hỏi. Việc đánh giá một hệ thống Q&A cần hướng người dùng bởi ý kiến người dùng là đánh giá tốt nhất cho tính thích hợp của câu trả lời. Các tiêu chuẩn trên được đặt ra với mong muốn xây dựng được một hệ thống Q&A hoàn chỉnh. Tuy nhiên, không phải hệ thống nào cũng có khả năng thông minh và hoàn thiện như thế. Các nghiên cứu về Q&A hiện nay đang tập trung vào xây dựng hệ thống hỏi đáp có tính chính xác cao và có khả năng sử dụng nguồn dữ liệu web khổng lồ trên Internet.

Rất nhiều phương pháp tiếp cận khác nhau được sử dụng trong các hệ thống hỏi-đáp gần đây như phương pháp thống kê, phương pháp dựa trên luật, và các phương pháp tổng hợp. Hầu hết các hệ thống hỏi-đáp ngày càng trở nên phức tạp, thường sử dụng rất nhiều các mô-đun khác nhau, chẳng hạn như tìm kiếm thông tin, phân tích cú pháp câu, phân tích loại câu hỏi, phân tích ngữ nghĩa, và thậm chí dùng phương pháp suy luận (Moldovan et al., 2002) để đánh giá, và xếp hạng câu trả lời. Với sự gia tăng tính phức tạp của hệ thống hỏi-đáp, sẽ rất khó khăn để xác định hiệu quả của hệ thống đến từ hiệu quả của giai đoạn nào.

### Phân loại hệ thống hỏi đáp

Có nhiều cách phân loại hệ thống hỏi đáp dựa trên các tiêu chí khác nhau như: Phân loại theo miền ứng dụng, theo khả năng trả lời câu hỏi, theo cách tiếp cận giải quyết bài toán…

#### Phân loại theo miền ứng dụng

Hệ thống hỏi đáp miền mở (open domain Question answering): Hệ thống trả lời bất kỳ câu hỏi nào được đưa vào. Khó khăn cho hệ thống miền mở đó chính là việc xây dựng các tri thức nên cho việc trả lời cũng như phân tích câu hỏi, các phương pháp hiện nay thường sử dụng một số các ontology khái quát hay các mạng tri thức như: wikipedia, bách khoa từ điển... Tuy nhiên, dữ liệu cho việc trích rút câu trả lời là phong phú, dễ thu thập.

Hệ thống hỏi đáp miền đóng (close domain Question answering): Hệ thống tập trung vào trả lời các câu hỏi liên quan đến một miền cụ thể (giáo dục, y tế, thể thao...). Xây dựng hệ thống hỏi đáp miền đóng được coi là bài toán dễ hơn so với xây dựng hệ thống hỏi đáp miền mở vì có thể sử dụng các tri thức miền (thường là ontology của miền cụ thể).

Các nghiên cứu hiện nay về Q&A đang tập trung vào xây dựng hệ thống hỏi đáp trên miền mở, sử dụng nguồn dữ liệu phi cấu trúc (kho văn bản lớn hay dữ liệu web) để tìm câu trả lời. Các nghiên cứu mới và cải tiến những phương pháp cũ để có thể áp dụng cho nguồn dữ liệu web vốn đa dạng, nhiều “nhiễu” và trùng lặp đang rất được quan tâm.

#### Phân loại theo khả năng trả lời câu hỏi

Hệ thống có khả năng trả lời các câu hỏi liên quan đến sự vật, hiện tượng,... dựa trên việc trích ra câu trả lời có sẵn trong tập tài liệu. Câu trả lời là các chuỗi ký tự trong một tài liệu. Kỹ thuật chính được sử dụng là xử lý chuỗi và từ khóa.

Hệ thống có cơ chế lập luận đơn giản: Trích xuất các câu trả lời có sẵn trong tập tài liệu sau đó sử dụng các suy luận để tìm mối liên kết giữa câu trả lời và câu hỏi. Hệ thống sử dụng các nguồn tri thức như ontology về từng miền cụ thể và ontology chung.

Hệ thống trả lời các câu hỏi yêu cầu khả năng tổng hợp: Các phần của câu trả lời được trích rút từ nhiều tài liệu sau đó được tổng hợp lại thành câu trả lời hoàn chỉnh. Câu hỏi thường là về danh sách, về cách thức, nguyên nhân...

Hệ thống có khả năng giao tiếp với người dùng: Trả lời chuỗi các câu hỏi của người dùng về cùng một vấn đề. Ví dụ các câu hỏi của người dùng như: “Giáo sư A sinh năm nào? Ở đâu? Ông ấy đang công tác ở đâu?”.

Hệ thống có khả năng lập luận tương tự: Có thể trả lời các câu hỏi có tính chất suy đoán, câu trả lời ẩn trong tập tài liệu. Hệ thống cần trích ra các luận chứng và sử dụng lập luận tương tự để tìm ra câu trả lời.

#### Phân loại theo hướng tiếp cận

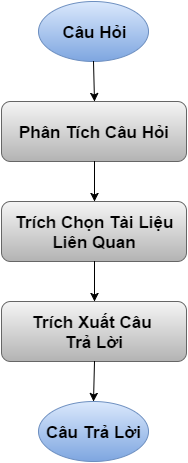
Hướng tiếp cận nông (shalow): Nhiều phương pháp sử dụng trong Q&A dùng các kĩ thuật dựa trên từ khóa để định vị các câu, đọan văn có khả năng chứa câu trả lời từ các văn bản được trích chọn về. Sau đó giữ lại các câu, đoạn văn có chứa chuỗi ký tự cùng loại với loại câu trả lời mong muốn (ví dụ các câu hỏi về tên người, địa danh, số lượng…).

Hướng tiếp cận sâu (deep): Trong những trường hợp khi mà hướng tiếp cận bề mặt không thể tìm ra câu trả lời, những quá trình xử lý về ngữ pháp, ngữ nghĩa và ngữ cảnh là cần thiết để trích xuất hoặc tạo ra câu trả lời. Các kĩ thuật thường dùng như nhận dạng thực thể (named-entity recognition), trích xuất mối quan hệ, loại bỏ nhập nhằng ngữ nghĩa,… Hệ thống thường sử dụng các nguồn tri thức như Wordnet, ontology để làm giàu thêm khả năng lập luận thông qua các định nghĩa và mối liên hệ ngữ nghĩa. Các hệ thống hỏi đáp dựa theo mô hình ngôn ngữ thống kê cũng đang ngày càng phổ biến.

### Các bước chung của hệ thống hỏi đáp

Một hệ thống hỏi đáp tự động thường gồm 3 bước chung sau:

* Bước 1: Phân tích câu hỏi: Bước phân tích câu hỏi tạo truy vấn cho bước trích chọn tài liệu liên quan và tìm ra những thông tin hữu ích cho bước trích xuất câu trả lời.
* Bước 2: Trích chọn tài liệu liên quan: Bước này sử dụng câu truy vấn được tạo ra ở bước phân tích câu hỏi để tìm các tài liệu liên quan đến câu hỏi.
* Bước3: Trích xuất câu trả lời: Bước này phân tích tập tài liệu trả về từ bước 2 và sử dụng các thông tin hữu ích do bước phân tích câu hỏi cung cấp để đưa ra câu trả lời chính xác nhất.



Các hệ thống hỏi đáp tự động hiện nay có kiến trúc rất đa dạng, tuy nhiên chúng đều bao gồm ba phần cơ bản như trên. Sự khác nhau chính giữa các hệ thống là ở quá trình xử lý trong từng bước, đặc biệt là ở cách tiếp cận trong việc xác định câu trả lời.

Cách tiếp cận theo trích chọn thông tin thuần túy (pure IR) là: chia nhỏ một tài liệu trong tập dữ liệu thành chuỗi các tài liệu con, trích chọn các tài liệu con có độ tương đồng lớn nhất với câu truy vấn (do bước phân tích câu hỏi tạo ra) và trả lại chúng cho người dùng.Thách thức lớn nhất ở đây là làm sao chia nhỏ được tài liệu thành các phần với kích cỡ tương ứng với kích cỡ của câu trả lời mà vẫn đủ lớn để có thể đánh chỉ mục được (nếu chia quá nhỏ thì số lượng tài liệu để đánh chỉ mục sẽ rất lớn, gây gánh nặng cho hệ thống trích chọn thông tin).

Cách tiếp cận theo xử lý ngôn ngữ tự nhiên (pure NLP) là: so khớp giữa biểu diễn ngữ pháp và (hoặc) biểu diễn ngữ nghĩa của câu hỏi với dạng biểu diễn ngữ pháp, ngữ nghĩa của các câu trong các tài liệu liên quan trả về. Khó khăn của cách tiếp cận này là hệ thống phải thực hiện việc phân tích ngữ pháp, ngữ nghĩa và so khớp đủ nhanh để đưa ra câu trả lời trong thời gian chấp nhận được, bởi số lượng các tài liệu cần xử lý là rất lớn trong khi các bước phân tích trên lại phức tạp và tốn nhiều thời gian.

Sự khác nhau trong cách trích xuất câu trả lời dẫn đến việc phân tích câu hỏi cũng trở nên đa dạng. Trong hướng tiếp cận theo trích xuất thông tin thuần túy, phân tích câu hỏi chỉ cần làm tốt việc tạo truy vấn, trong khi với hướng tiếp cận theo xử lý ngôn ngữ tự nhiên, câu hỏi cần được phân tích ngữ pháp, ngữ nghĩa một cách chính xác. Các hệ thống hiện nay thường là sự kết hợp giữa hai hướng tiếp cận, sử dụng hệ thống trích chọn thông tin để thu hẹp không gian tìm kiếm câu trả lời, đồng thời phân tích câu hỏi để tìm ra các thông tin về ngữ pháp, ngữ nghĩa nhằm tìm ra câu trả lời chính xác nhất. Kết quả của bước phân tích câu hỏi là đầu vào cho cả hai bước trích chọn tài liệu liên quan và trích xuất câu trả lời. Bước phân tích câu hỏi có ý nghĩa rất quan trọng, bởi nó ảnh hưởng đến hoạt động của các bước sau và do đó quyết định đến hiệu quả của toàn hệ thống. Chương 2 trình bày chi tiết các nội dung liên quan đến phân tích câu hỏi.

### Phương pháp đánh giá hệ thống hỏi đáp

Có hai phương pháp cơ bản dùng để đánh giá hệ thống hỏi đáp: trung bình thứ hạng đối xứng (MRR) và điểm trọng số tin cậy (CWS).

#### Trung bình thứ hạng đối xứng (MRR)

Chuẩn đánh giá này được đề xuất tại TREC-8. Chuẩn này được áp dụng cho các câu hỏi Factoid và non-Factoid như sau:

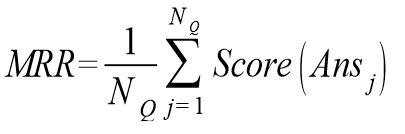
* Cho một tập câu hỏi mẫu Q có NQ câu hỏi.
* Tập tài liệu dùng để rút trích câu trả lời D.
* +Tập câu trả lời mẫu A={[ai, di]} là tập các cặp gồm câu trả lời ai và chỉ số di  của tài liệu trong D được dùng để rút trích câu trả lời đó.
* Hệ thống sẽ có 5 câu trả lời trong tập Ansq cho một câu hỏi q. Câu trả lời có thể là 50byte hoặc 250bytes kèm theo tài liệu dùng để trích câu trả lời.

Điểm số của kết quả trả lời Ansq là:

Score (Ansq) =

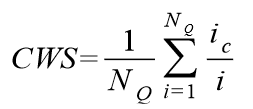
Với i<=5 là chỉ số của câu trả lời đún đầu tiên trong Ansq.

Nếu I > 5 thì Score(Ansq) = 0.

Trung bình thứ hạng đối xứng của hệ thống được tính là:

#### Điểm trọng số tin cậy (CWS)

Điểm trọng số tin cậy Confidence-Weighted Score (CWS) nhằm đánh giá khả năng hệ thống hiểu câu trả lời của nó như thế nào. Với số liệu được thiết lập như đánh giá trung bình thứ hạng đối xứng với rang buộc hệ thống chỉ được trả lời 1 câu trả lời. Điểm trọng số tin cậy được tính như sau:



## Giới thiệu về tìm kiếm thông tin

### Khái niệm về tìm kiếm thông tin.

Việc xây dựng một hệ thống hỏi đáp thì vấn đề bắt buộc phải có chính là nguồn thông tin. Để có được nguồn thông tin dữ liệu chúng ta phải thực hiện việc tìm kiếm thông tin, vậy tìm kiếm thông tin là gì?

Tìm kiếm thông tin theo thuật ngữ chuyên ngành có tên là “Information Retrieval” hay được viết tắt là IR, là quá trình tìm kiếm thông tin trên một nguồn dữ liệu có sẵn nào đó. Chúng ta đã biết kết quả của các hệ thống tìm kiếm như Bing hay Google sẽ trả về một lượng thông tin dữ liệu khá lớn có thể lên đến hàng nghìn trang web. Do đó việc tìm kiếm thông tin sẽ giúp ích chúng ta trong việc tìm kiếm lượng thông tin cần thiết và nội dung thông tin đề cập chính xác đến mục đích tìm kiếm của chúng ta.

### Công cụ tìm kiếm Bing

#### Giới thiệu về Bing

Bing (trước đây là Live Search, Windows Live Search và MSN Search) là [bộ máy tìm kiếm web](https://vi.wikipedia.org/wiki/M%C3%A1y_truy_t%C3%ACm_d%E1%BB%AF_li%E1%BB%87u) (được quảng cáo là một bộ máy "ra quyết định"), đại diện cho công nghệ tìm kiếm hiện nay của Microsoft. Được Giám đốc Điều hành của Microsoft [Steve Ballmer](https://vi.wikipedia.org/wiki/Steve_Ballmer) tiết lộ vào ngày [28 tháng 5](https://vi.wikipedia.org/wiki/28_th%C3%A1ng_5) năm [2009](https://vi.wikipedia.org/wiki/2009) tại hội nghị All Things D tại San Diego, Bing là một sự thay thế cho Live Search; bộ máy tìm kiếm này được đưa lên trực tuyến hoàn toàn vào ngày [3 tháng 6](https://vi.wikipedia.org/wiki/3_th%C3%A1ng_6) năm [2009](https://vi.wikipedia.org/wiki/2009).

#### Bing search API

Bing Search API là công cụ hỗ trợ các lập trình viên có thể lấy được các kết quả từ Bing trong các ứng dụng hoặc website, kết quả này dưới dạng JSON hoặc XML.

Hiện nay Bing Search API đã được chuyển qua Window Azure Marketplace.

Window Azure Marketplace cung cấp những chức năng:

* Cho phép truy cập các kết quả trang web, hình ảnh, tin tức và video cũng như những câu gợi ý tìm kiếm hay kiểm tra chính tả.
* Kết quả cải thiện và chính xác hơn so với API 2.0
* Hỗ trợ kiểu dữ liệu JSON, XML và Odata
* Có cơ hội kiếm tiền từ các ứng dụng của Window Azure Marketplace
* Có thể truy cập hàng trăm dữ liệu khác để phục vụ cho ứng dụng.

Để sử dụng bing search api ta thực hiện tuần tự các bước sau:

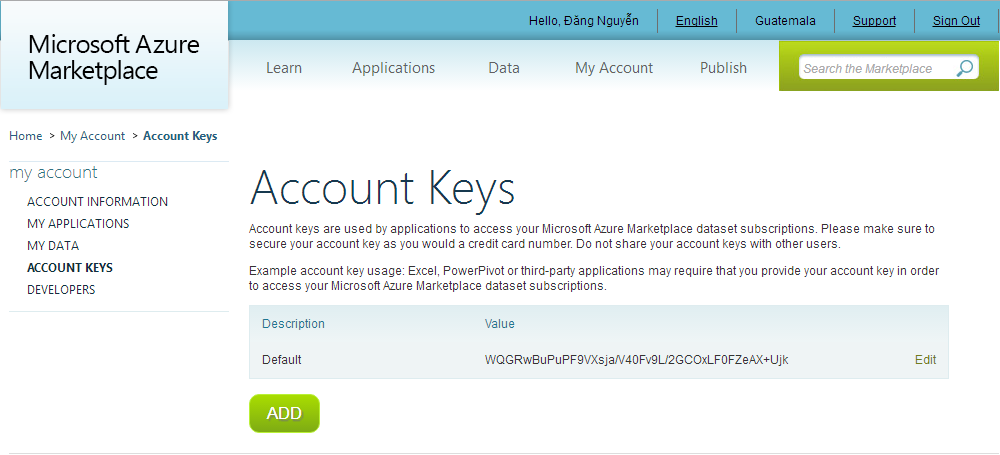
- Lấy Account key:

+ Truy cập vào <http://www.bing.com/developers/>

+ Click vào Get Started trong mục Search API

+ Bing đưa ra 1 vài lựa chọn tùy theo nhu cầu, ở đây chúng ta sẽ chọn 5000 query trong 1 tháng. Sau đó click vào Sign Up

+ Tạo 1 Window Azure Marketplace Account

+ Đăng kí Bing Search API

Trong ứng dụng java để sử dụng bing api ta cần phải thực hiện việc khai báo account key, urlpattern để có thể thực hiện kết nối tới bing và bắt đầu quá trình tìm kiếm:

* Khai báo một account key ta dùng cú pháp sau:
* Khai báo một UrlPattern:
* Khai báo một encode account key:

Sau khi đã khai báo xong, ta tiến hành kết nối URL:

Ở đây, kết quả trả về sẽ là một đối tượng Json, trong đó có chưa chuỗi các url, snippet, title.

Ví dụ xét câu hỏi: “Ai là vua phá lưới wc 2014?” ta được kết quả trả về như hình sau:



### Công cụ tìm kiếm Google

#### Giới thiệu về công cụ tìm kiếm google.

Google search là dịch vụ cung cấp chính và quan trọng nhất của công ty [Google](https://vi.wikipedia.org/wiki/Google). Dịch vụ này cho phép người truy cập tìm kiếm thông tin trên [Internet](https://vi.wikipedia.org/wiki/Internet), bao gồm các trang Web, hình ảnh & nhiều thông tin khác.

Bằng cách sử dụng các [Bot](https://vi.wikipedia.org/wiki/Bot) dò tìm và tạo chỉ mục ([index](https://vi.wikipedia.org/w/index.php?title=Index&action=edit&redlink=1)) các trang Web trên Internet,xây dựng cơ sở dữ liệu tìm kiếm cho các [máy chủ](https://vi.wikipedia.org/wiki/M%C3%A1y_ch%E1%BB%A7) của mình, khi có người truy cập và thực hiện tìm kiếm, kết quả tìm kiếm sẽ được lấy ra từ đây. Google search cũng cho phép người sử dụng khai báo trang web của họ với máy chủ của google, sau đó các máy chủ này sẽ sắp xếp thời gian để tạo chỉ mục cho các trang web được khai báo.

Để tìm kiếm, người sử dụng gõ vào ô tìm kiếm một hoặc một vài cụm từ gọi là từ khóa tìm kiếm (keywords). Dựa trên từ khóa tìm kiếm này, Google searchsẽ thực hiện tìm kiếm và hiển thị ra kết quả cho người sử dụng.

Việc sắp xếp thứ tự kết quả tìm kiếm phụ thuộc vào thứ hạng của nó theo phân cấp của Google search nhờ tổng hợp phức tạp từ [keyword](https://vi.wikipedia.org/w/index.php?title=Keyword&action=edit&redlink=1), [pagerank](https://vi.wikipedia.org/w/index.php?title=Pagerank&action=edit&redlink=1), [sitemap](https://vi.wikipedia.org/w/index.php?title=Sitemap&action=edit&redlink=1)... . Kết quả tìm kiếm cũng được phân loại theo đối tượng sử dụng khác nhau, nó phụ thuộc vào vị trí địa lý (lấy theo [IP](https://vi.wikipedia.org/wiki/IP) truy cập của người truy cập). Ngoài ra, Google search cũng sử dụng[cookies](https://vi.wikipedia.org/wiki/Cookie) và tính năng lưu trữ thói quen tìm kiếm của người dùng (cá nhân hóa kết quả tìm kiếm) để tạo ra kết quả tìm kiếm. Nói chung, việc sắp xếp và hiển thị kết quả củaGoogle search khá phức tạp và nó là bí mật công nghệ mà nhờ đó Google có thể chiếm lĩnh thị trường.

#### Cách sử dụng Jsoup để lấy kết quả trả về từ google.

Cũng giống với công cụ tìm kiếm bing, đối với công cụ tìm kiếm google cũng vậy. Chúng ta muốn lấy kết quả trả về, trước hết chúng ta phải khai báo các biến cần thiết, và sau đó sẽ kết nối tới google để lấy thông tin theo yêu cầu của câu hỏi.

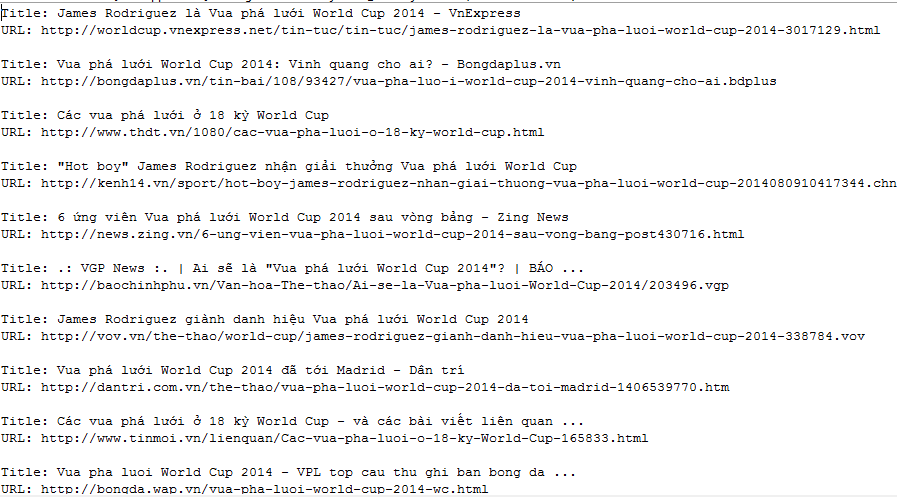
Việc thiết lập các biến được thực hiện như sau:



Sau khi đã khai báo các biến ta thực hiện việc kết nối url

Kết quả trả về của google khác với bing, google sẽ trả về một chuỗi các element, trong đó chưa các url và title của các link được trả về mà không có snippet như bing.

Ta lại xét ví dụ như trên: “Ai là vua phá lưới wc 2014?”, ta có kết quả trả về như hình:



## Giới thiệu thư viện VnTokenizer

VnTokenizer là công cụ tách từ tiếng Việt được nhóm tác giả Nguyễn Thị Minh Huyền, Vũ Xuân Lương và Lê Hồng Phương phát triển dựa trên phương pháp so khớp tối đa (Maximum Matching) với tập dữ liệu sử dụng là bảng âm tiết tiếng Việt và từ điển từ vựng tiếng Việt.

Tư tưởng của phương pháp này là duyệt một câu từ trái sang phải và chọn ra từ có nhiều tiếng nhất mà từ đó có mặt trong từ điển tiếng việt.

Công cụ sử dụng tập dữ liệu đi kèm là tập từ điển từ vựng tiếng Việt, danh sách các đơn vị từ mới bổ sung, được biểu diễn bằng ôtômat tối tiểu hữu hạn trạng thái, tệp chứa các biểu thức chính quy cho phép lọc các đơn vị từ đặc biệt (xâu dạng số, ngày tháng,…), và các tệp chứa các thống kê unigram và bigram trên kho văn bản tách từ mẫu.

Ví dụ xét câu sau: “Cristiano Ronaldo sinh năm nào?” sau khi thực hiện tách từ kết quả đạt được như sau

**Cristiano\_Ronaldo** có **biệt\_danh** là gì ?

## Giới thiệu thư viện Vntagger.

VnTagger là thư viện được phát triển bởi tác giả Lê Hồng Phương. VnTagger hổ trợ việc gán nhãn từ loại tự động có độ chính xác cao. Vntagger được phát triển dựa theo phương pháp “Entropy Maximum”.

Gán nhãn từ loại là quá trình xác định từ loại của mỗi từ trong phạm vi văn bản đó. Có nghĩa là việc gán nhãn từ loại sẽ phân loại các từ thành các lớp từ loại.

Ví dụ: trong tiếng Việt ta có câu hỏi sau:

“**Cristiano Ronaldo có biệt danh là gì?**”

Sau khi gán nhãn từ loại cho câu trên sẽ trở thành:

**Cristiano\_Ronaldo/N có/V biệt\_danh/N là/V gì/P?**

## Thư viện tìm kiếm lucene

Lucene là thư viện hổ trợ việc tìm kiếm được phát triển bởi Dough Cutting. Lucene cung cấp các hàm cơ bản để hổ trợ việc đánh chỉ mục và tìm kiếm. Muốn sử dụng lucene, chúng ta phải có sẵn nguồn dữ liệu. Dữ liệu này có thể là tập hợp các tập tin có định dạng PDF, text, hay các trang web HTML.

Lucene là thư viện tìm kiếm thông tin có khả năng xử lí và mở rộng cao, cho phép chúng ta tích hợp vào trong ứng dụng của mình.

Lucene gồm 2 thành phần chính là: thành phần tạo chỉ mục (index) và thành phần tìm kiếm (search). Đây là hai thành phần quan trọng co việc tìm kiếm của thư viện này.

## Biểu thức chính quy.

### Khái niệm

Biểu thức chính quy (Theo tiếng Anh: Regular Expression) là một chuỗi miêu tả một bộ các chuỗi khác theo những quy tắc cú pháp nhất định. Biểu thức chính quy thường được dùng trong các trình biên tập văn bản, các công cụ tìm kiếm thông tin dựa trên các mẫu quy định.

Biểu thức chính quy hoạt động dựa trên nguyên tắc là so khớp dựa vào khuôn mẫu, các khuôn mẫu này được xây dựng dựa trên những quy tắc cơ bản của biểu thức chính quy.

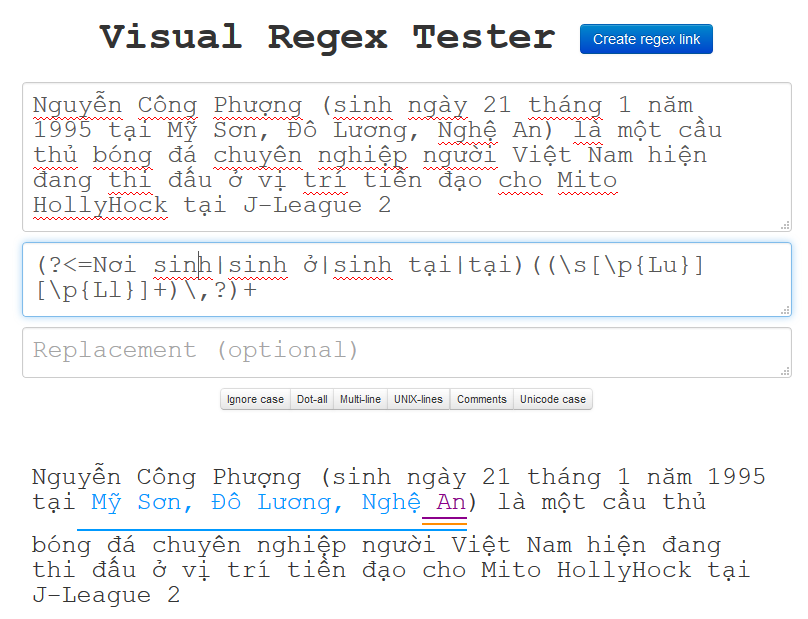
Để hiểu rõ hơn về biểu thức chính quy ta xét ví dụ sau: Ta có một đoạn dữ liệu có chưa thông tin về năm sinh của cầu thủ Nguyễn Công Phượng.

“**Nguyễn Công Phượng (sinh ngày 21 tháng 1 năm 1995 tại** [**Mỹ Sơn**](https://vi.wikipedia.org/wiki/M%E1%BB%B9_S%C6%A1n,_%C4%90%C3%B4_L%C6%B0%C6%A1ng)**,** [**Đô Lương**](https://vi.wikipedia.org/wiki/%C4%90%C3%B4_L%C6%B0%C6%A1ng_%28huy%E1%BB%87n%29)**,** [**Nghệ An**](https://vi.wikipedia.org/wiki/Ngh%E1%BB%87_An)**) là một cầu thủ** [**bóng đá**](https://vi.wikipedia.org/wiki/B%C3%B3ng_%C4%91%C3%A1) **chuyên nghiệp người** [**Việt Nam**](https://vi.wikipedia.org/wiki/Vi%E1%BB%87t_Nam) **hiện đang thi đấu ở vị trí** [**tiền đạo**](https://vi.wikipedia.org/wiki/Ti%E1%BB%81n_%C4%91%E1%BA%A1o_%28b%C3%B3ng_%C4%91%C3%A1%29) **cho** [**Mito HollyHock**](https://vi.wikipedia.org/wiki/C%C3%A2u_l%E1%BA%A1c_b%E1%BB%99_b%C3%B3ng_%C4%91%C3%A1_Mito_HollyHock) **tại** [**J-League 2**](https://vi.wikipedia.org/wiki/Gi%E1%BA%A3i_b%C3%B3ng_%C4%91%C3%A1_chuy%C3%AAn_nghi%E1%BB%87p_h%E1%BA%A1ng_nh%C3%AC_Nh%E1%BA%ADt_B%E1%BA%A3n)**.**”

Để có thể lấy thông tin về năm sinh của cầu thủ này ta thực hiện xây dựng biểu thức chính quy. Biểu thức này là một mẫu sẽ so khớp với đoạn dữ liệu và trả về kết quả là năm sinh.

**(?<=Nơi sinh|sinh ở|sinh tại|tại)((\s[\p{Lu}][\p{Ll}]+)\,?)+**

Kết quả này nhóm dùng công cụ test online là “visual regex tester”



### Các quy tắc cú pháp cơ bản

Các quy tắc cú pháp cơ bản của biểu thức chính quy được trình bày ở bảng dưới đây:

|  |  |
| --- | --- |
| **Ký tự** | **Trường hợp sử dụng** |
| . | Bất kỳ ký tự nào. |
| [abcd] | Bất kỳ ký tự nào nằm trong “[ ]” Ví dụ [abcd] thì “ac” phù hợp còn “ef” không phù hợp. |
| [^abcd] | Ngược với ở trên không có ký tự nào trong “[ ]”. |
| [a-z] | Ký tự bất kỳ nằm trong khoảng a đến z. |
| \w | bất kỳ ký tự nào nằm trong khoảng [a-zA-Z\_0-9]. |
| \W | Ngược với \w các ký tự như $,@,?,!, v.v. |
| \d | Số. |
| \D | không phải số. |
| \s | khoảng trắng. |
| \S | không khoảng trắng. |

Các quy tắc cú pháp unicode trong biểu thức chính quy:

|  |  |
| --- | --- |
| Ký tự | Trường hợp sử dụng |
| \p{L} or \p{Letter} | Bất kì loại kí tự từ bất kì ngôn ngữ nào. |
| \p{Ll} or \p{Lowercase\_Letter} | Chữ viết thường có một biến thể là chữ hoa. |
| \p{Lu} or \p{Uppercase\_Letter} | Một kí tự hoa mà có một biến thể chữ thường. |
| \p{Lt} or \p{Titlecase\_Letter} | Một kí tự mà xuất hiện ở đầu của một từ khi chỉ có chữ cái đầu tiên |
| \p{L&} or \p{Cased\_Letter} | Kí tự tồn tại trong chữ thường và chữ hoa biến thể ( sự kết hợp của Ll , Lu và Lt ) . |

Các quy tắc cú pháp định lượng:

|  |  |
| --- | --- |
| **Ký Tự.** | **Trường hợp sử dụng.** |
| \* | không hoặc nhiều. |
| + | một hoặc nhiều. |
| ? | 0 hoặc có 1 ký tự. |
| {n} | bắt buộc có n ký tự. |
| {n,} | bắt buộc tối tiểu n ký tự hoặc nhiều. |
| {n,m} | bắt buộc tối tiểu n ký tự hoặc tối đa m ký tự. |

## Ứng dụng web

### Khái niệm ứng dụng web

# PHƯƠNG PHÁP TRẢ LỜI CÂU HỎI

## Xác định dạng câu hỏi

Việc hệ thống tìm đúng được tài liệu liên quan đến câu hỏi, hay xác định được nội dung câu hỏi sẽ hỏi về vấn đề gì và sau đó là xác định regex để tìm câu trả lời phụ thuộc khá nhiều vào quá trình xác định dạng câu hỏi. Do đó quá trình xác định dạng câu hỏi quyết định một phần không nhỏ đến hiệu năng của hệ thống.

Quá trình xác định dạng câu hỏi nhận input là câu hỏi dưới dạng ngôn ngữ tự nhiên do người dùng nhập vào. Dựa vào tập câu hỏi khảo sát sẽ xây dựng biểu thức chính quy cho mỗi dạng câu hỏi.

Để xác định được dạng câu hỏi, ta sử dụng các keyword trong câu hỏi. Việc xác định keyword trong câu hỏi được thực hiện thông qua việc thu thập bộ câu hỏi liên quan, từ đó đưa ra tập các keyword chung để xác định dạng câu hỏi phù hợp.

Ví dụ: Muốn xác định đâu là dạng câu hỏi về tên người ta xác định như sau:

* Tập các keyword thu thập có liên quan đến dạng câu hỏi về tên người: Ai, người nào, cầu thủ nào, huấn luyện viên nào, là ai, là người nào, là cầu thủ nào, là huấn luyện viên nào,…
* Xác định regex tương ứng cho câu hỏi: **“(Ai|ai|Người nào|người nào|Cầu thủ|cầu thủ)(.\*)") || question.matches("(.\*) ai (.\*)”**
* Câu hỏi nhập vào dưới dạng String, do đó sẽ dùng hàm **“matches”** của String để xác định: “**question.matches("(Ai|ai|Người nào|người nào|Cầu thủ|cầu thủ)(.\*)") || question.matches("(.\*) ai (.\*)")**”
* Kết quả trả về sẽ là một số, con số này sẽ đại diện cho loại câu hỏi, mỗi loại câu hỏi sẽ có mốt số riêng biệt. Con số này sẽ được dùng để xác định biểu thức chính quy thích hợp cho loại câu hỏi đó.

### Quy trình xác định dạng câu hỏi.

Quá trình xác định dạng câu hỏi bao gồm các bước sau:

* Thu thập tập câu hỏi mẫu. Khoảng 100 câu hỏi bao gồm các dạng câu hỏi được giới hạn trong đề tài và được trình bày trong phần phụ lục.
* Xác định đặc điểm riêng của từng dạng câu hỏi.
* Xây dựng mẫu so khớp cho từng dạng câu hỏi.
* Đưa ra dạng câu hỏi tương ứng.

### Cách xác định cho những dạng câu hỏi của hệ thống.

Hệ thống được xây dựng để trả lời được 16 dạng câu hỏi phổ biến liên quan đến thông tin bóng đá.

#### Dạng câu hỏi về tên người.

Ví dụ: xét câu hỏi “**Ai giành quả bóng vàng FiFa 2014**”

* Tập keyword xác định được: ai, người nào, cầu thủ nào, là ai, là cầu thủ nào, là người nào
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(Ai|ai|Người nào|người nào|Cầu thủ|cầu thủ|HLV|Huấn luyện viên|uấn luyện viên)(.\*)") || question.matches("(.\*) ai (.\*)")) {

System.***out***.println("Question--> people");

**return** QuestionResult.***PEOPLE***;

}

#### Dạng câu hỏi năm sinh.

Ví dụ: xét câu hỏi “**Cristiano Ronaldo sinh năm nào?**”

* Tập keyword được xác định: sinh, bao nhiêu, năm nào, năm bao nhiêu, thời gian nào…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(Đội bóng|Đội bóng nào|đội bóng|đội bóng nào|CLB nào|Câu lạc bộ nào|CLB)(.\*)")

|| question.matches("(.\*)(đội bóng|đội bóng nào|CLB nào|Câu lạc bộ nào)(.\*)")) {

System.***out***.println("Question--> Team");

**return** QuestionResult.***TEAM***;

}

}

#### Dạng câu hỏi về số áo cầu thủ.

Ví dụ: xét câu hỏi “**Messi mang áo số bao nhiêu?**”

* Tập keyword được xác định: mang áo, mặc áo, số áo, mang số áo,

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(mang áo|mặc áo|mang số áo|số áo)(.\*)") || question.matches("(Số áo|số áo)(.\*)")) {

System.***out***.println("Question--> info");

**return** QuestionResult.***NUMBER***

}

#### Dạng câu hỏi về vị trí thi đấu trên sân.

Ví dụ: xét câu hỏi “**Robben thi đấu ở vị trí nào?”**

* Tập keyword được xác định: vị trí

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)vị trí(.\*)")) {

System.***out***.println("Question--> info");

**return** QuestionResult.***POSITION***;

}

#### Dạng câu hỏi về nơi sinh của cầu thủ?

Ví dụ: xét câu hỏi “**Steven Gerrard sinh ra tại đâu?”**

* Tập keyword được xác định: sinh ra…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(sinh ra|Sinh ra|lớn lên)(.\*)")) {

System.***out***.println("Question->where\_people");

**return** QuestionResult.***WHERE\_BORN***;

}

#### Dạng câu hỏi về chiều cao của cầu thủ.

Ví dụ: xét câu hỏi “**Steven Gerrard có chiều cao là bao nhiêu?”**

* Tập keyword được xác định: chiều cao, cao

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(Chiều cao|chiều cao)(.\*)") || question.matches("(.\*)(chiều cao)(.\*)")) {

System.***out***.println("Question--> info");

**return** QuestionResult.***HEIGHT***;

}

#### Dạng câu hỏi về biệt danh cầu thủ.

Ví dụ: xét câu hỏi “**Ronaldo có biệt danh là gì?”**

* Tập keyword được xác định: biệt danh

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)biệt danh là gì(.\*)") || question.matches("Biệt danh của(.\*)")) {

System.***out***.println("Question--> nickname");

**return** QuestionResult.***NICKNAME***;

}

#### Dạng câu hỏi địa điểm diễn ra trận đấu.

Ví dụ: xét câu hỏi “**Trận chung kết Champion League 2015-2016 tổ chức ở đâu?”**

* Tập keyword được xác định: diễn ra, tổ chức…

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(tổ chức|diễn ra|xảy ra)(.\*)")) {

System.***out***.println("Question-->where");

**return** QuestionResult.***WHERE***;

}

#### Dạng câu hỏi thời gian bắt đầu diễn ra sự kiện

Ví dụ: xét câu hỏi “**Euro 2016 bắt đầu khi nào?”**

* Tập keyword được xác định: khai mạc, bắt đầu, diễn ra, khi nào…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(khai mạc|bắt đầu|diễn ra)(.\*)")& question.matches("(.\*)(khi nào|lúc nào)(.\*)")) {

System.***out***.println("Question>Time(Start)");

**return** QuestionResult.***TIME\_START***;

}

#### Dạng câu hỏi thời gian kết thúc diễn ra trận đấu.

Ví dụ: xét câu hỏi “**Euro 2016 kết thúc khi nào?”**

* Tập keyword được xác định: bế mạc, kết thúc, khi nào…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(bế mạc|kết thúc)(.\*)")& question.matches("(.\*)(khi nào|lúc nào)(.\*)")) {

System.***out***.println("Question>Time(Start)");

**return** QuestionResult.***TIME\_START***;

}

#### Dạng câu hỏi kết quả trận đấu.

Ví dụ: xét câu hỏi “**Tỉ số trận chung kết Champion League 2014-2015 là bao nhiêu?”**

* Tập keyword được xác định: tỉ số, kết quả

…

* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(Tỉ số|tỉ số|Kết quả|kết quả )(.\*)")) {

System.***out***.println("Question-->Result(Match)");

**return** QuestionResult.***MATCH\_RESULT***;

}

#### Dạng câu hỏi về tổng số bàn thắng trong sự nghiệp.

Ví dụ: xét câu hỏi “**Beckham ghi được tổng cộng bao nhiêu bàn thắng trong sự nghiệp? ”**

* Tập keyword được xác định: tổng cộng, bàn thắng…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(bàn thắng)(.\*)") ) {

System.***out***.println("Question--> info");

**return** QuestionResult.***SUM\_GOAL***;

}

#### Dạng câu hỏi về tổng số trận thi đấu trong sự nghiệp.

Ví dụ: xét câu hỏi “**Beckham thi đấu tổng cộng bao nhiêu trận?”**

* Tập keyword được xác định: tổng cộng, thi đấu, chơi…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(thi đấu|chơi)(.\*)")& question.matches("(.\*)(tổng cộng)(.\*)") ) {

System.***out***.println("Question--> info");

**return** QuestionResult.***SUM\_MATCH***;

}

#### Dạng câu hỏi về đội bóng nào.

Ví dụ: xét câu hỏi “**Đội bóng nào vô địch Champion League 2014-2015”**

* Tập keyword được xác định:Đội bóng, CLB, Câu lạc bộ…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(Đội bóng|Đội bóng nào|đội bóng|đội bóng nào|CLB nào|Câu lạc bộ nào|CLB)(.\*)")|| question.matches("(.\*)(đội bóng|đội bóng nào|CLB nào|Câu lạc bộ nào)(.\*)")) {

System.***out***.println("Question--> Team");

**return** QuestionResult.***TEAM***;

}

}

#### Dạng câu hỏi về tên sân vận động.

Ví dụ: xét câu hỏi “**SVĐ của Real Madrid tên là gì?”**

* Tập keyword được xác định:SVĐ, Sân vận động…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("Sân vận động của(.\*)(là gì|có tên là gì|tên gọi là gì)(.\*)")) {

System.***out***.println("Question--> SVD");

**return** QuestionResult.***STADIUM***;

}

#### Dạng câu hỏi về sơ đồ thi đấu

Ví dụ: xét câu hỏi “**Barcelona thi đấu với sơ dồ nào?”**

* Tập keyword được xác định: sơ đồ…
* Quá trình xây dựng mẫu so khớp và đưa ra dạng câu hỏi như sau:

**if** (question.matches("(.\*)(đội hình|sơ đồ|Đội hình|Sơ đồ)(.\*)")) {

System.***out***.println("Question--> so do");

**return** QuestionResult.***SQUAD***;

}

## Phân tích câu hỏi

Khi một câu hỏi được người dùng nhập vào, câu hỏi sẽ được phân tích trước khi đưa vào hàm tìm kiếm.

Sẽ có hai dạng để phân tích câu hỏi. Một dạng dùng để tìm kiếm tài liệu từ Bing và Google và một dạng dùng để lucene tìm kiếm đoạn tài liệu thích hợp.

### Phân tích câu hỏi dùng để tìm kiếm bằng bing và google

Đối với dạng phân tích câu hỏi để hổ trợ việc tìm kiếm của Bing và Google thì khá đơn giản. Đầu tiên input nhận vào sẽ là câu hỏi dưới dạng String, sau đó ta thực hiện thay thế khoảng trắng trong câu hỏi thành dấu “+”.

**public** String parseQuestion(String str) {

String result = **null**;

result = str.replace(' ', '+');

System.***out***.println(result);

**return** result;

}

Ví dụ: khi nhập vào câu hỏi “**Ai là vua phá lưới world cup 2014**”. Qua quá trình phân tích câu hỏi trên sẽ trở thành:

“**Ai+là+vua+phá+lưới+world+cup+2014**”

.

### Phân tích câu hỏi dùng để tìm kiếm bằng lucene.

Đối với dạng phân tích cho việc tìm kiếm trong lucene nhóm sẽ bao gồm các bước: phân đoạn từ, gán nhãn từ loại, loại bỏ từ không cần thiết trong câu hỏi. Để thực hiện việc phân tích này, nhóm dùng hai bộ thư viện hổ trợ là VnTokenizer và VnTagger của tác giả Lê Hồng Phương và sử dụng biểu thức chính quy để loại bỏ stopword.

Ở quá trình phân đoạn từ và gán nhãn từ loại, input sẽ là câu hỏi được người dùng nhập vào, output là kết quả trả về sau khi đã phân đoạn và gán nhãn. Để thực hiện phân đoạn từ và gán nhãn ta thực hiện như sau:

Ví dụ với câu hỏi tiếng việt: “Nguyễn Công Phượng sinh ra ở đâu?”, sau quá trình phân đoạn từ và gán nhãn ta được như sau:

String segmented = tokenizer.segment(input);

System.***out***.println(segmented);

String tagged = tagger.tagging(segmented);

System.***out***.println(tagged);

**Nguyễn\_Công\_Phượng/Np sinh\_ra/X ở/E đâu/P ?/?**

Sau khi đã phân đoạn từ và gán nhã từ loại, ta thực hiện việc loại bỏ stopword trong câu hỏi. Stopword là những từ để hỏi như: ai, đâu, nào, bao nhiêu, gì. Sử dụng biểu thức chính quy để giữ lại những từ cần thiết. Qua quá trình khảo sát trên tập câu hỏi nhất định, nhóm nhận thấy những từ được giữ lại bao gồm những từ loại như: Danh từ, động từ, tính từ, trạng từ… đại diện cho những từ này là: Np, N, V, M, E, A, L, X. Do đó ta xây dựng được biểu thức chính quy sau:

**String regex\_ = "\\S+/(N|Np|V|M|E|A|L|X)";**

Đối với câu hỏi trên, sau khi sử dụng biểu thức chính quy để rút trích ra nhữn từ cần thiết giữ lại, ta sẽ được một tập các từ riêng biệt là: Nguyễn\_Công\_Phượng, sinh\_ra, ở. Vì vậy ta cần phải nối các từ này lại với nhau để thành một chuỗi: **Nguyễn\_Công\_Phượng sinh\_ra ở**. Cuối cùng sẽ dùng hàm “replace” của String dể thay thế các kí tự “\_” thành các khoảng trắng. Như vậy câu hỏi ban đầu sẽ trở thành: **Nguyễn Công Phượng sinh ra ở**.

Ta biết đối tượng QueryParser khi được gọi trong quá trình tìm kiếm sẽ tách câu query thành 2 loại là **Phrases** và **Single Term**. Nghĩa là từ “**Nguyễn Công Phượng**” sẽ được phân tích thành “**Nguyễn**” or “**Công**” or “**Phượng**”. Điều này sẽ khiến cho kết quả tìm kiếm bị sai lệch không như mục đích nếu như tài liệu nào đó không liên quan đến thông tin cần thiết nhưng chưa nhiều từ Nguyễn, Công, Phương. Vì vậy nhóm sẽ sử dụng việc đóng ngoặc kép cho danh từ riêng về dạng Phrases để kết quả tìm kiếm được tốt hơn. Quá trình này được xử lí như sau:

**public** String PatternRepalace(String text) {

String segmented = tokenizer.segment("ABC" + text);

String tagged = tagger.tagging(segmented);

String noun\_list = "";

String regex\_replace = "\\S+/Np|/M";

Pattern pattern = Pattern.*compile*(regex\_replace);

Matcher matcher = pattern.matcher(tagged);

**while** (matcher.find()) {

noun\_list += matcher.group() + "|";

}

// -----System.out.println("noun\_list:" + noun\_list);

noun\_list = noun\_list.replace("\_", " ").replaceAll("/Np|/M", "").replace("|", "").replaceAll("\\|$", "");

**if** (noun\_list == "") {

noun\_list = " ";

}

**if** (!(noun\_list.toLowerCase().contains(" "))) {

noun\_list = " ";

}

String[] noun\_list\_array = noun\_list.split("\\|");

String result = text;

**for** (String noun : noun\_list\_array) {

**if** (noun != "") {

result = result.replace(noun, "\"" + noun + "\"");

}

}

**return** result;

}

Các bước thực hiện trên chỉ là quá trình tiền xử lí trước khi thực hiện việc phân tích câu hỏi để lucene tìm kiếm. Câu query được thực hiện dùng để tìm kiếm phải được phân tích dưới dạng chuẩn của lucene. Việc này được thực hiện thông qua đối tượng queryparser của lucene.

**Query q = new QueryParser("contents", analyzer).parse(QueryParseTitle(querystr, "title") + " OR " + querystr);**

Với câu hỏi trên thì kết quả câu query như sau:

**title:nguyễn title:công title:phượng title:sinh title:ra title:ở contents:"nguyễn công phượng" contents:sinh contents:ra contents:ở**

## Rút trích thông tin câu trả lời

Đối với việc rút trích thông tin câu trả lời, nhóm sử dụng biểu thức chính quy để thực hiện việc tìm kiếm câu trả lời. Ý tưởng của việc rút trích câu trả lời rất đơn giản. Dựa vào dạng câu hỏi đã được xác định ở quá trình xác định dạng câu hỏi, từ đó xây dựng một biểu thức chính quy cụ thể cho dạng câu hỏi đó.

Để có thể sử dụng biểu thức chính quy, nhóm sử dụng 2 thư viện hổ trợ có sẵn trong trình biên dịch eclipse là “Pattern” và “Matcher”.

Quá trình rút trích được thực hiện như sau:

Ta thấy hai thành phần chính được sử dụng là pattern và matcher.

Pattern pattern = Pattern.*compile*(regex);

Matcher matcher = pattern.matcher(sentence);

**while** (matcher.find()) {

tempResult = matcher.group();

// -----System.out.println(tempResult);

tempResult = tempResult.replaceAll("[^\\u00BF-\\u1FFF\\u2C00-\\uD7FF\\w\\s]", "").replaceAll("\\s+$", "");

result.add(tempResult);

}

* Pattern: là một đối tượng mẫu, một phiên bản phiên dịch của biểu thức chính quy. Pattern không có cấu tủ public, để sử dụng chúng ta chỉ cần gọi phương thức tĩnh là compile với tham số truyền vào là một biểu thức chính quy.
* Matcher: Là một phương tiện để sơ khớp dữ liệu với đối tượng pattern đã được tạo ra trước đó. Matcher cũng không có cấu tử public, chúng ta sẽ lấy đối tượng này thông qua phương thức matcher() của đối tượng patter. Tham số truyền vào trong phương thức matcher là đoạn văn bản cần kiểm tra để rút trích thông tin theo mẫu biểu thức chính quy.

Xét ví dụ cụ thể sau:

String regex = "\\d+";

String text = "22/12/1994";

Pattern pattern = Pattern.*compile*(regex);

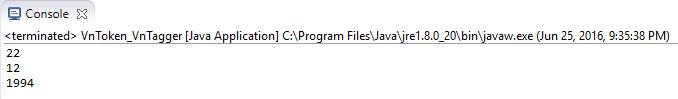
Matcher matcher = pattern.matcher(text);

**while** (matcher.find()) {

System.***out***.println(matcher.group());

}

Trong ví dụ đơn giản này, chúng ta sẽ thực hiện rút trích ra những con số từ đoạn text. Kết quả như sau:



## Xếp hạng câu trả lời.

Kết quả sau khi sử dụng biểu thức chính quy để rút trích sẽ không đảm bảo rằng sẽ chính xác. Vì vậy cần có quá trình xếp hạng câu trả lời để có được câu trả lời tốt nhất.

Để xếp hạng câu trả lời, nhóm thực hiện việc đánh trọng số xuất hiện cho n câu trả lời đầu tiên. Với n câu trả lời đầu tiên sẽ là những câu trả lời tốt nhất được biểu thức chính quy trả về dựa vào dữ liệu đã được xếp hạng bởi lucene. Việc đánh trọng số và xếp hạng được thực hiện như sau:

* Lấy ra n câu trả lời đầu tiên từ tập câu trả lời có được.
* Lần lượt tính số lần xuất hiện của n câu trả lời đầu tiên trong tổng số câu trả lời.
* Sắp xếp n câu trả lời theo thứ tự giảm dần số lần xuất hiện.

# PHÂN TÍCH THIẾT KẾ VÀ CÀI ĐẶT HỆ THỐNG

Mô hình chung hệ thống:



TÀI LIỆU THAM KHẢO

Theo chuẩn IEEE