Mục Lục

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc287556324)

[**1.1** **Giới thiệu** 1](#_Toc287556325)

[**1.2** **Mục tiêu và phạm vi đề tài** 3](#_Toc287556326)

[**1.3** **Kết quả dự kiến** 4](#_Toc287556327)

[**1.4** **Cấu trúc nội dung trình bày** 4](#_Toc287556328)

[**CHƯƠNG 2 : KHẢO SÁT CÁC NGHIÊN CỨU LIÊN QUAN** 5](#_Toc287556329)

[**2.1** **Cơ sở lý thuyết và các thuật ngữ liên quan** 5](#_Toc287556330)

[**2.1.1** **Giới thiệu về hỏi đáp** 5](#_Toc287556331)

[**2.1.2** **Nhận diện thực thể đặt tên** 5](#_Toc287556332)

[**2.1.3** **Gán nhãn từ loại** 6](#_Toc287556333)

[**2.1.4** **Bộ ba quan hệ và đồ thị khái niệm** 6](#_Toc287556334)

[**2.1.5** **Từ điển Wordnet** 8](#_Toc287556335)

[**2.2** **Các nghiên cứu liên quan** 9](#_Toc287556336)

[CHƯƠNG 3: PHƯƠNG PHÁP ĐỀ XUẤT XỬ LÝ CÂU HỎI 16](#_Toc287556337)

[**3.1** **Mở đầu** 16](#_Toc287556338)

[**3.2** **Tạo tập tin cấu hình ngữ nghĩa cho cơ sở dữ liệu** 17](#_Toc287556339)

[**3.3** **Các bước xử lý câu hỏi người dùng** 21](#_Toc287556340)

[**3.3.1** **Tiền xử lý** 21](#_Toc287556341)

[**3.3.2** **Liệt kê các bộ ba quan hệ về từ** 26](#_Toc287556342)

[**3.3.3** **Nhận diện thực thể** 29](#_Toc287556343)

[**3.3.4** **Sinh câu truy vấn SQL** 37](#_Toc287556344)

[**3.3.5** **Một số ví dụ** 44](#_Toc287556345)

[CHƯƠNG 4: HIỆN THỰC HỆ THỐNG 51](#_Toc287556346)

[**4.1** **Tổng quan về hệ thống** 51](#_Toc287556347)

[**4.1.1** **Mô hình hệ thống:** 51](#_Toc287556348)

[**4.1.2** **Các chức năng trong hệ thống** 53](#_Toc287556349)

[**4.2** **Thiết kế cài đặt hệ thống** 54](#_Toc287556350)

[**4.2.1** **Môi trường xây dựng hệ thống** 54](#_Toc287556351)

[**4.2.2** **Cấu trúc các lớp** 55](#_Toc287556352)

[**4.2.3** **Cơ sở dữ liệu DBLP** 55](#_Toc287556353)

[**4.2.4** **Giao diện chương trình** 58](#_Toc287556354)

[**4.2.5** **Đánh giá và nhận xét** 61](#_Toc287556355)

[**CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 63](#_Toc287556356)

[THAM KHẢO 64](#_Toc287556357)

[Tiếng Việt 64](#_Toc287556358)

[Tiếng Anh 64](#_Toc287556359)

[Web 65](#_Toc287556360)

[Phụ lục A : Tập 100 câu hỏi kiểm thử 67](#_Toc287556361)

[Phụ lục B : Danh sách các nhãn từ loại Penn Treebank 70](#_Toc287556362)

[Phụ lục C : Tập 18 luật nhãn từ loạt dùng để liệt kê bộ ba 71](#_Toc287556363)

[Phụ lục D : Tập tin cấu hình ngữ nghĩa cho cơ sở dữ liệu BDBLP 72](#_Toc287556364)

# CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

* 1. **Giới thiệu**

Hiện nay, cùng với sự phát triển mạnh mẽ của công nghệ thông tin nói chung và Internet nói riêng đã và đang phục vụ rất nhiều lợi ích cho con người. Song song với sự phát triển đó, là lượng thông tin ngày càng lớn dần từ đó dẫn đến nhu cầu tìm kiếm thông tin trên Internet ngày càng gia tăng. Để đáp ứng các nhu cầu ấy, hàng loạt các công cụ tìm kiếm đã được ra đời như Bing[[1]](#footnote-2), Yahoo![[2]](#footnote-3), Google[[3]](#footnote-4), … Các công cụ này hầu hết đều thực hiện việc tìm kiếm dựa trên các từ hay cụm từ khóa. Kết quả trả về của các công cụ thường là các trang liên kết (hoặc văn bản) có nội dung liên quan tới từ khóa tìm kiếm. Đây là cách tìm kiếm khá quen thuộc đối với đa số người dùng.

Vì hầu hết các công cụ tìm kiếm chủ yếu dựa trên từ khóa hay cụm từ nên khi thực hiện việc tìm kiếm sẽ cho ra rất nhiều trang web chứa từ khóa hay cụm từ đó. Để có được thông tin chính xác, người dùng cần tốn thời gian để duyệt rất nhiều các kết quả trả về từ công cụ tìm kiếm.

Do đó, một hệ thống hỏi đáp có khả năng sẽ phục vụ tốt hơn trong vấn đề tìm kiếm thông tin. Nếu hệ thống có thể hiểu đúng được ngữ nghĩa trong câu hỏi mà người dùng đưa vào, thì nó có thể tìm kiếm và đưa ra câu trả lời đầy đủ và chính xác nhất, thỏa mãn được nhu cầu người dùng. Hơn thế nữa, cách thức hỏi đáp sẽ làm máy tính trở nên “thông minh” hơn, tạo ra một hướng giao tiếp thân thiện giữa người và máy.

Trong lĩnh vực khoa học, nhu cầu tìm kiếm thông tin bài báo phục vụ cho các nghiên cứu và tham khảo là rất cần thiết. Một vài các thư viện số và công cụ như: CiteSeerX[[4]](#footnote-5), IEEE[[5]](#footnote-6), ACM[[6]](#footnote-7), ScienceDirect[[7]](#footnote-8), SSRN[[8]](#footnote-9), PaperCube[[9]](#footnote-10), Google Scholar[[10]](#footnote-11) ... đã và đang phục vụ rất tốt cho người dùng trong vấn đề này. Thông tin các bài báo trong những thư viện đều rất phong phú và đa dạng. Cho nên, các thư viện hay công cụ luôn cung cấp nhiều chức năng tìm kiếm như: tìm kiếm theo tiêu đề bài báo, theo tác giả, theo nhà xuất bản … nhằm đem lại hiệu quả và tiện dụng cho vấn đề tìm kiếm. Một số trang như PaperCube có khả năng hiển thị bài báo theo dạng mô hình đồ thị. Điều này giúp người dùng dễ dàng định hướng được trong việc tìm kiếm. Tuy nhiên, các thư viện này vẫn còn tìm kiếm chủ yếu dựa trên từ khóa do người dùng nhập vào. Việc duyệt tìm để có được đúng bài báo mong muốn trong rất nhiều kết quả trả về vẫn sẽ phải hao tốn một khoảng thời gian, tương tự như trường hợp các công cụ tìm kiếm thông tin ở trên. Nếu áp dụng hình thức hỏi đáp trong việc tìm kiếm các bài báo, thì có khả năng hệ thống sẽ hỗ trợ tốt và chính xác hơn khi tìm kiếm các bài báo trong thư viện.

Với nhận định như thế, đề tài hướng tới xây dựng một hệ thống hỏi đáp phục vụ cho nhu cầu tìm kiếm các bài báo khoa học. Nó sẽ cung cấp cho người dùng một giao diện hỏi đáp để tìm kiếm bài báo. Người dùng sẽ nhập vào một câu hỏi dưới dạng ngôn ngữ tự nhiên thay cho từ khóa. Kết quả trả về là các thông tin hoặc bài báo tương ứng với nội dung của câu hỏi, thõa mãn đúng theo câu hỏi người dùng đặt ra. Cuối cùng, các đánh giá về đề tài sẽ dựa trên thuật toán xử lý để hiểu được câu hỏi tự nhiên của người dùng.

* 1. **Mục tiêu và phạm vi đề tài**

Đề tài hướng tới xây dựng một trang web với mục đích phục vụ tìm kiếm các bài báo khoa học thuộc lĩnh vực công nghệ thông tin. Vậy đối tượng người dùng là các nhà nghiên cứu, các kỹ sư, sinh viên công nghệ thông tin, ... Cách thức tìm kiếm bao gồm :

* Tìm kiếm bằng từ khóa thông thường.
* Tìm kiếm bằng hình thức đặt ra một câu hỏi cho hệ thống thông qua giao diện hỏi đáp.

Câu hỏi và từ khóa người dùng nhập vào sẽ ở dạng ngôn ngữ là tiếng Anh.

Nguồn dữ liệu chứa thông tin về các bài báo sẽ được lấy từ nguồn DBLP[[11]](#footnote-12) (Digital Bibliography & Library Project). Đây là một kho thư viện điện tử lưu trữ thông tin của hơn 1,5 triệu bài báo [REF]. Dữ liệu DBLP luôn được cập nhật theo định kỳ (mới nhất tính đến 1/2011). Đây là một kho dữ liệu lớn và thích hợp để nhóm thực hiện đề tài. Ngoài ra, kho dữ liệu này còn được cung cấp miễn phí và tải về tại trang chủ của DBLP.

Dựa vào nguồn dữ liệu đã có trong DBLP, các câu hỏi của người dùng đặt ra chỉ giới hạn trong phạm vi các thông tin : tiêu đề, tên tác giả, năm công bố, nguồn gốc và nhà xuất bản. Các dạng câu hỏi dự kiến có thể xử lý được bao gồm:

* Yes/No Question: Câu hỏi nghi vấn.
  + Ví dụ: Are there any books written by Mike Holcombe in 2000 ?
* Wh-Question: Các câu hỏi bắt đầu với Who, What, Where, When ...
  + Ví dụ: What book did Philip K. Chan write in 1999 ?
* List question : Câu hỏi dạng liệt kê.
  + Ví dụ: Name all publications were publish in 2010

Các dạng trên dựa theo phân loại chuẩn trong tiếng Anh [REF] (Yes/no question và Wh-question) và phân loại List Question của tập câu hỏi chuẩn TREC[[12]](#footnote-13).

Cuối cùng, để đánh giá kết quả đề tài, một tập 102 câu hỏi thuộc 3 dạng câu hỏi trên được tạo ra để kiểm thử. Các câu hỏi trong tập này được xây dựng bằng tay dựa vào một số mẫu câu hỏi chuẩn trong TREC, và một số câu hỏi khác được cho là cần thiết trong tìm kiếm bài báo. (giới hạn bởi thông tin dữ liệu của DBLP)

* 1. **Kết quả dự kiến**

Xây dựng một hệ thống phục vụ tìm kiếm các thông tin bài báo với giao diện thân thiện và dễ dàng. Người dùng nhanh chóng nắm bắt được cách sử dụng và định hướng được thông tin tìm kiếm dựa trên cách tổ chức dữ liệu trên giao diện web.

Cung cấp một giao diện hỏi đáp cho người dùng để tìm các bài báo. Dựa trên thuật toán đề xuất, hệ thống dự kiến xử lý được 90% các câu hỏi trong tập kiểm thử.

* 1. **Cấu trúc nội dung trình bày**

Ở chương đầu, chúng tôi trình bày tổng quan về đề tài, nêu rõ mục đích và phạm vi thực hiện cũng như các kết quả dự kiến.

Chương tiếp theo - Chương 2 - sẽ trình bày nội dung khảo sát một số nghiên cứu liên quan đến đề tài. Trước đó, chương sẽ giải thích một số khái niệm để làm cơ sở để hiểu rõ các vấn đề trong phần khảo sát. Tiếp theo, sẽ giới thiệu những nghiên cứu của một số các tác giả liên quan đến vấn đề hỏi đáp và kèm theo các kết quả mà họ đạt được.

Chương 3 trình bày về phương pháp đề xuất để giải quyết vấn đề đặt ra trong đề tài là sử dụng câu hỏi để tìm kiếm các bài báo trong DBLP. Chương sẽ nêu ra từng bước mà câu hỏi được chuyển thành câu truy vấn SQL xuống cơ sở dữ liệu.

Chương 4 và 5 sẽ trình bày nội dung về hiện thực hệ thống, các đánh giá thực nghiệm. Cuối cùng sẽ là kết luận của đề tài và đưa ra hướng phát triển.

**CHƯƠNG 2 : KHẢO SÁT CÁC NGHIÊN CỨU LIÊN QUAN**

* 1. **Cơ sở lý thuyết và các thuật ngữ liên quan**
     1. **Giới thiệu về hỏi đáp**

Theo [REF] và [REF cô Hà, DTrường], thì hỏi đáp (QA) là một loại rút trích thông tin, cung cấp một tập các tài liệu (ví dụ như các trang Web hay tài liệu cục bộ). Hệ thống có thể đưa ra câu trả lời cho những câu hỏi đặt ra bằng ngôn ngữ tự nhiên. Bởi vì phải xử lý ngôn ngữ tự nhiên (NLP) nên QA phức tạp hơn các loại rút trích thông tin khác (như rút trích văn bản chẳng hạn) và đôi khi nó được xem như là bước kế tiếp của bộ máy tìm kiếm.

QA nghiên cứu giải quyết một phạm vi rộng lớn các loại câu hỏi khác nhau, bao gồm: sự việc, danh sách, sự xác định, như thế nào, tại sao, giả thuyết, ràng buộc ngữ nghĩa và các câu hỏi đa ngôn ngữ. QA tìm kiếm trên phạm vi rộng lớn các tập hợp khác nhau, từ tập hợp văn bản cục bộ đến nội dung trên các trang Web. Bao gồm hai phân hệ chính:

* **Closed-domain QA:** giải quyết những câu hỏi với những lĩnh vực xác định. Ví dụ như: y tế, giáo dục, khoa học, lịch sử ...
* **Open-domain QA:** giải quyết những câu hỏi về mọi lĩnh vực và có thể chỉ cung cấp kiến thức tổng quát chung chung. Những hệ thống như thế này thường có nhiều dữ liệu sẵn có để rút trích câu trả lời.
  + 1. **Nhận diện thực thể đặt tên**

Thực thể đặt tên là những thành phần của văn bảng thuộc vào những mục sau: tên người, tên tổ chức, địa điểm, thời gian, số lượng, phần trăm [19]… Trong câu hỏi người dùng hay trong văn bản, các thực thể đặt tên cũng thường hay xuất hiện.

Ví dụ trong câu *“What books were written by Philip K. Chan”* thì *Philip K. Chan* được xem là một thực thể đặt tên (tên tác giả).

Do thực thể đặt tên không theo một nguyên tắc và giới hạn nào. Nên trong vấn đề nhận biết thực thể đặt tên, ta thường áp dụng một số từ điển như Wordnet. Tuy nhiên, số lượng thực thể đặt tên trong Wordnet cũng có giới hạn.

GATE là một công cụ hỗ trợ mạnh mẽ trong việc xử lý văn bản. GATE hỗ trợ nhận diện các thực thể đặt tên như tên người, nơi chốn, năm .... Do GATE dựa trên các luật và một tập từ điển lưu trữ trong một danh sách gọi là Gazetteer.

* + 1. **Gán nhãn từ loại**

Trong phạm trù ngôn ngữ học, gán nhãn từ loại (part-of-speech tagging) còn được gọi là gán nhãn ngữ pháp hay định hướng từ loại. Đó là quá trình đánh dấu lên những từ trong một văn bản thuộc về một phần của bài ph át biểu (tài liệu). Quá trình gán nhãn dựa trên định nghĩa của từ, cũng như ngữ cảnh (Ví dụ: mối quan hệ của nó với các từ lân cận hoặc liên quan trong cùng một câu, cụm từ, hoặc một đoạn. Một hình thức đơn giản của việc này là việc xác định các từ như danh từ, động từ, tính từ, trạng từ ... (theo nguồn wikipedia)

Việc sử dụng gán nhãn từ loại sẽ giúp ta biết được vai trò của mỗi từ trong câu (danh từ, động từ hay tính từ ... ). Từ đó, sẽ hỗ trợ ta cho xử lý về sau như việc phân tích cú pháp, hoặc loại bỏ hay nhận biết các từ loại cần thiết trong câu, so sánh ngữ nghĩa giữa hai từ có cùng từ loại ...

Một số công cụ gán nhãn từ loại thường được biết đến: Wordnet, Gate pos tagger, OpenNLP ...

* + 1. **Bộ ba quan hệ và đồ thị khái niệm**

Theo [24] và [2] thì RDF là một mô hình lưu trữ theo dạng các bộ ba subject-predicate- object. Trong đó Subject là một chủ từ bắt đầu cho một bộ ba. Predicate là các vị từ ở giữa biểu diễn cho mối quan hệ giữa Subject và Object. Object là đối tượng mà subject có quan hệ tới.

Trong các web ngữ nghĩa hiện nay thường sử dụng mô hình lưu trữ RDF ([Resource Description Framework](http://www.w3.org/RDF/)) [24]. RDF là một mô hình lưu trữ các đối tượng (resource) và các mối quan hệ giữa chúng. Đây là một mô hình ngữ nghĩa đơn giản và được biểu diễn trong cú pháp XML.

Việc hình thành các bộ ba như trên sẽ có lợi cho việc lưu trữ ngữ nghĩa. Các bộ ba có thể kết hợp lại thành một đồ thị ngữ nghĩa (Hình 2). Đồ thị khái niệm (hay ngữ nghĩa ) như trong hình là sự kết hợp giữa các bộ ba quan hệ. Đây là một cách để biểu diễn tri thức dưới dạng đồ thị được John F. Sowa đưa ra năm 1976 .



Hình 2 – Thông tin được biểu diễn dưới dạng đồ thị liên kết bởi các bộ ba

(Hình lấy từ tài liệu tham khảo [24])

Với hình trên, các bộ ba được liên kết thành một đồ thị. Một độ thị như thế sẽ có thể biểu diễn cho một đoạn văn bản, một thông tin trên web, một câu nói ... Ta có thể dễ dàng thấy rằng “*the\_thirteenth\_floor*” là một “*movie*”. Do đó nó có mối quan hệ “*is\_a*”. Đồng thời nó cũng có mối quan hệ để thể hiện ngày phát hành là “*released\_in*” và có giá trị là năm “*1999*”. Vậy tóm lại, với cách hiểu như thế thì câu nói có thể là: *The Thirteenth Floor is a movie which are realesed in 1999*.

Dựa vào các bộ ba quan hệ này mà đã có một số nghiên cứu đã áp dụng để phục vụ cho vấn đề hỏi đáp như [2], [7], [8], [9] và [10]. Chủ yếu các nghiên cứu trình bày cách xây dựng các bộ ba từ câu hỏi và sử dụng chúng để truy vấn xuống một cơ sở dữ liệu đã lưu trữ sẵn các bộ ba như cơ sở dữ liệu MSQL/MYSQL hay Ontology. Ontology một dạng mô hình có thể lưu trữ dữ liệu theo dạng RDF.

* + 1. **Từ điển Wordnet**

WordNet: là một dạng từ điển ngữ nghĩa tiếng Anh, được tạo ra từ năm 1985 và đang được duy trì ở các phòng thí nghiệm khoa học nhận thức của Đại học Princeton dưới sự hướng dẫn của giáo sư tâm lý học A. George Miller. Các cơ sở dữ liệu và phần mềm công cụ đã được phát hành theo một giấy phép của BSD ([Berkeley Software Distribution](http://en.wikipedia.org/wiki/Berkeley_Software_Distribution)) và có thể được tải về và sử dụng một cách tự do. Các cơ sở dữ liệu cũng có thể được truy cập trực tuyến.

Trong WordNet, các từ được xếp vào các nhóm đồng nghĩa gọi là synsets (tập các từ có thể thay thế nhau trong một ngữ cảnh nào đó), cung cấp các định nghĩa ngắn, tổng quát và ghi lại những quan hệ ngữ nghĩa khác nhau giữa các nhóm từ đồng nghĩa. Các nhóm đồng nghĩa liên kết với nhau thông qua các quan hệ ngữ nghĩa được xây dựng thông qua các nghiên cứu về cách sử dụng, cách lưu trữ các tri thức về ngôn ngữ trong bộ não con người. Nhờ cách tổ chức như vậy, WordNet đã cung cấp nhiều tri thức hữu dụng cho việc xử lý ngôn ngữ tự nhiên. Nó được sử dụng với 2 mục đích: để tạo ra sự kết hợp giữa từ điển và từ điển đồng nghĩa có thể sử dụng hợp lý hơn, và để hỗ trợ phân tích văn bản tự động và ứng dụng trí tuệ nhân tạo.

Công cụ Similarity Wordnet[[13]](#footnote-14): là một công cụ phát triển dựa trên bộ từ điển Wordnet. Nó tính toán mức độ tương tự ngữ nghĩa giữa hai từ và cho ra kết quả là một giá trị trong khoảng [0,1]. Việc sử dụng công cụ này giúp ta vẫn hiểu được ngữ nghĩa của từ nếu người dùng nhập vào một từ đồng nghĩa.

Ví dụ: *author* và *writer* là hai từ có cùng ngữ nghĩa. Nếu sử dụng công cụ này, độ tương tự giữa hai từ sẽ là *1.0*.

* + 1. **Các nghiên cứu liên quan**

Với những hệ thống hỏi đáp nêu trên cho thấy rằng đã có những nghiên cứu về vấn đề xử lý câu hỏi tự nhiên phục vụ tìm kiếm thông tin. Qua khảo sát, nhiều bài báo đã trình bày nhữn nghiên cứu của họ về cách thức hỏi đáp bằng ngôn ngữ tự nhiên như tài liệu [1], [2], [7], [8] và[9].

Vấn đề chung của các nghiên cứu là hiểu được câu hỏi mà người dùng đưa vào. Câu hỏi bằng nhiều cách sẽ được chuyển thành dạng ngôn ngữ mà máy có thể hiểu được. Ví dụ như câu truy vấn SQL.

* Một nghiên cứu gần đây của Wael Salloum, 2009 [7]. Tác giả đưa ra hướng tiếp cận bằng cách mô hình các văn bản và câu hỏi người dùng thành một dạng đồ thị gọi là Conceptual Graph Formalism (CGF). Đồ thị khái niệm là một hình thức biểu diễn tri thức được John F. Sowa đưa ra năm 1976.

Nguồn dữ liệu của đề tài này là từ các văn bản (document) nói chung, không thuộc riêng về một lĩnh vực nào, và dữ liệu này phải có độ tin cậy cao. Một văn bản được tách thành nhiều câu. Và mỗi câu dựa trên việc phân tích cú pháp, mỗi nút trong cây cú pháp sẽ được rút khái niệm từ wordnet kết hợp với công cụ verbnet để xác định mối quan hệ giữa động từ với chủ từ, đối tượng. Câu sẽ được chuyển thành các Đồ thị khái niệm (gọi là Concept Graph (CG)). Cuối cùng các đồ thị này sẽ được lưu xuống cơ sở dữ liệu. Tương tự cho câu hỏi cũng chuyển thành một CG.

Cuối cùng, tác giả có đề cập tới một phép chiếu (projection operator) dùng để so sánh giữa CG của một câu hỏi với CG của một câu trong văn bản. Theo cách hiểu, thao tác này chỉ đơn giản là so sánh mức độ liên quan giữa đồ thị câu hỏi và đồ thị các câu trả lời. Đó là cách mà câu trả lời được rút ra, xếp hạng và trả về cho người dùng.

Ví dụ : với câu “Mark Twain wrote Tom Sawyer”, thì đồ thị khái niệm sẽ được tạo như sau:



Hình 1 – Ví dụ về một đồ thị khái niệm

( được lấy từ Figure 3. tài liệu tham khảo [2])

Trong đồ thị ở hình 1, theo cách hiểu của nhóm, *Author* là một lớp đối tượng và một thể hiện của nó là *Mark Twain*. Cũng như thế, *Novel* có thể hiện là Tom Sawyer. Giữa *Author* và *Novel,* ta có mối quan hệ *Write*. *Agent* và *Patient* được coi là các mối quan hệ trung gian: *Author* là tác nhân (agent) của hành động *write* và *Novel* là một thể bị động (patient) so với hành động *write*.

Tương tự câu hỏi cũng được chuyển thành đồ thị khái niệm. Vấn đề ở các câu hỏi là cần xác định đối tượng cần hỏi.

Ví dụ: *“ Who write Tom Sawyer? “* sẽ đuợc chuyển thành như sau:

*[write]-*

*(Agnt) -> [Person: \*] ?*

*(Ptnt) -> [Novel:Tom Sawyer]*

*Hay trình bày theo cách hiểu:*

*[Person: ?] -(Agnt)-[write]-(Ptnt)- [Novel:Tom Sawyer]*

Theo cách ghi ở đây, là một cách khác để thể hiện, *Person* là đối tượng cần biết trong câu hỏi , giữa *person* và *Tom Sawyer* có mối quan hệ *write*. *Agnt* và *Ptnt* là hai từ viết tắt của *Agent, Patient*. Đây là một cách thể hiện khác của đồ thị so với hình 1. Sau đó, công việc tiếp theo là so sánh ( projection operator) giữa đồ thị CG của câu hỏi với các đồ thị khái niệm CG mỗi câu để tìm câu trả lời.

Kết quả bài báo trên đạt được là biểu diễn văn bản và câu hỏi dưới dạng các CG. Bằng cách so sánh giữa các CG, tìm ra câu trả lời. Hướng đi tiếp theo của tác giả là tìm cách tổng hợp lại các câu trả lời với mục đích là đưa ra một câu trả lời duy nhất.

* Với một nghiên cứu trong nước, nhóm tìm hiểu được một đề tài “Xây dựng công cụ tìm kiếm tài liệu học tập bằng các truy vấn ngôn ngữ tự nhiên trên kho học liệu mở tiếng Việt” [2]. Với tên đề tài thì hệ thống hỏi đáp mà tác giả hướng tới là một hệ thống hỏi và trả lời trên ngôn ngữ tiếng việt.

Với cách tiếp cận của đề tài trên, câu hỏi tiếng việt được đưa vào hệ thống phân tích. Một tập 40 luật cú pháp tiếng việt được tác giả tạo ra sẵn. Nếu phân tích được, thành phần sẽ tạo ra một cây cú pháp tương ứng. Từ đó, cây cú pháp được chuyển thành cây sinh mã truy vấn bằng cách ánh xạ lên một số nút của cây cú pháp vào ontology. Cuối cùng, câu truy vấn SPARQL được tạo ra và lấy dữ liệu trả lời.

Ontology có thể xem là một kho dữ liệu được lưu trữ dưới dạng ngữ nghĩa. SPARQL là ngôn ngữ để truy vấn vào kho dữ liệu này [17,18].

Dữ liệu trong đề tài này được lấy từ kho học liệu mở vocw[[14]](#footnote-15). Các metadata được rút từ các trang của vocw và cập nhật dữ liệu vào ontology.

Kết quả của đề tài trên được đánh giá như sau: với 40 câu hỏi chuẩn thì hệ thống chính xác 100%, với 91 câu hỏi ngẫu nhiên thì độ chính xác là 84,62%. Hướng đi tiếp theo của đề tài là cải thiện bộ phân tích cú pháp với mục đích tạo ra bộ phân tích mạnh hơn và nhanh hơn. Đồng thời, việc mở rộng không gian tìm kiếm cũng được tác giả lưu ý tới. Vậy, đề tài bị hạn chế bởi 40 luật cú pháp, do việc xử lý ngôn ngữ Tiếng Việt còn gặp nhiều khó khăn.

* Một đề tài khác là “ Dịch câu truy vấn có cấu trúc sang đồ thị ý niệm: cách tiếp cận ít phụ thuộc vào cú pháp”, 2008 [3]. Theo ý kiến của tác giả, việc phân tích cú pháp để hình thành đồ thị ý niệm gặp khó khăn khi các câu truy vấn không đúng cú pháp. Vì thế tác giả đưa ra một phương pháp dịch câu truy vấn sang đồ thị ý niệm mà không dựa trên việc phân tích cú pháp câu.

Ở đề tài này, phương pháp tiếp cận dựa trên việc nhận diện các thực thể đặt tên, không đặt tên và từ quan hệ. Sau khi nhận diện được, thao tác tiếp theo là nhận diện lớp của thực thể mà không xác định được nó thuộc lớp nào ( như What, Who ...). Bước kế tiếp là gom các thực thể được cho là giống nhau, xác định những mối quan hệ ẩn mà không được nêu trong câu hỏi. Cuối cùng, xác định quan hệ giữa các thực thể, xóa bỏ những mối quan hệ không thích hợp và xây dựng đồ thị ý niệm.

Công cụ đựơc sử dụng trong đề tài này là GATE [20]. Do GATE có khả năng làm việc với Ontology, công cụ OCAT (Ontology-based Corpus Annotation tool) được sử dụng để chú thích các thực thể đặt tên. Ngoài ra, công cụ ANNIE được dùng để nhận diện các từ quan hệ và thực thể không đặt tên bằng các lập ra các danh sách trong Gazetteer. Ontology được sử dụng ở đây có tên là PROTON.

Tập dữ liệu câu hỏi kiểm nghiệm được lấy từ 440 câu hỏi của TREC 2002 và 445 câu hỏi của TREC 2007.

Việc đánh giá của đề tài dựa trên độ chính xác của việc tạo đồ thị ý niệm. Kết quả của đề tài trên nhóm xin trình bày theo bảng sau:

(Quá trình làm giàu Ontology trong nghiên cứu này chủ yếu thủ công bằng cách bổ sung các lớp và mối quan hệ)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tổng Số câu | Trước khi làm giàu  Ontology | Sau khi làm giàu  Ontology |
| TREC 2002 | 440 | 15.23% | 78.64% |
| TREC 2007 | 445 | 8.54% | 60.45% |

Bảng 4- Kết quả nghiên cứu tạo đồ thị ý niệm từ tài liệu [3]

* Với hệ thống eLSSNL (eLibrary Searching System by Natural Language) của tài liệu [1], câu truy vấn dưới dạng ngôn ngữ tự nhiên không được xây dựng thành đồ thị ý niệm như tài liệu [3] và [7].

Đối với hệ thống eLSSNL, tác giả dùng một phương pháp khác so với việc xây dựng đồ thị ý niệm. Câu truy vấn tự nhiên của người dùng được nhận dạng các giá trị thuộc tính tương ứng với các thuộc tính của bảng trong cơ sở dữ liệu. Do hệ thống này sử dụng dữ liệu miễn phí từ dự án Gutenberg[[15]](#footnote-16) chứa thông tin của khoảng 33,000 ebooks miễn phí.

Cách để hiểu câu truy vấn dưới dạng ngôn ngữ tự nhiên của người dùng được tài liệu [1] giải quyết bằng việc xây dựng một tập từ khóa dựa trên các chỉ định từ mà tác giả đã liệt kê. Sau đó, các giá trị hay từ khóa đó được nhận diện thuộc về thuộc tính nào trong bảng cơ sở dữ liệu.

Ví dụ:

*Nhận dạng giá trị cho thuộc tính Title/FriendlyTitle:*

+  DOM(Title) hoặc  DOM(FriendlyTitle) (DOM(fileld): là miền giá trị của một thuộc tính (field) trong table BOOKS )

+ Đi kèm theo các chỉ định từ: title/heading

*Nhận dạng giá trị cho thuộc tính NumDownload:*

+ Là số *(isNumber)*, ưu tiên tìm theo ID trước

+ Đi kèm theo các chỉ định từ: [more] than/>=/>; less/<=/<; top; best; most; numdownload.

Đối với từng thuộc tính của bảng sẽ có những điều kiện để nhận dạng như ví dụ trên. Cuối cùng bằng các phép kết hợp, câu truy vấn được sinh ra và tìm các dữ liệu thỏa mãn.

Ngoài ra, hệ thống eLSSNL còn hỗ trợ lưu lại các các câu truy vấn mà người dùng nhập vào và lưu trong một bảng dưới cơ sở dữ liệu. Người dùng còn được hỗ trợ gợi ý cho người dùng các câu truy vấn tương tự. Việc đó dựa vào việc so sánh các tập từ khóa được lưu lại trong bảng vừa đề cập và so sánh với tập từ khóa trong câu hỏi người dùng vừa nhập vào.

Theo đánh giá về hiệu quả tìm kiếm, hạn chế lớn nhất của eLSSNL là về mặt tốc độ. Nếu dùng người nhập vào một câu truy vấn tự nhiên mới hoàn toàn – tức là tập từ khóa không trùng với các tập từ khóa được lưu trữ- thì tốc độ tìm kiếm sẽ khá lâu nếu người dùng tìm trên nhiều giá trị thuộc tính.

**Nhận xét:**

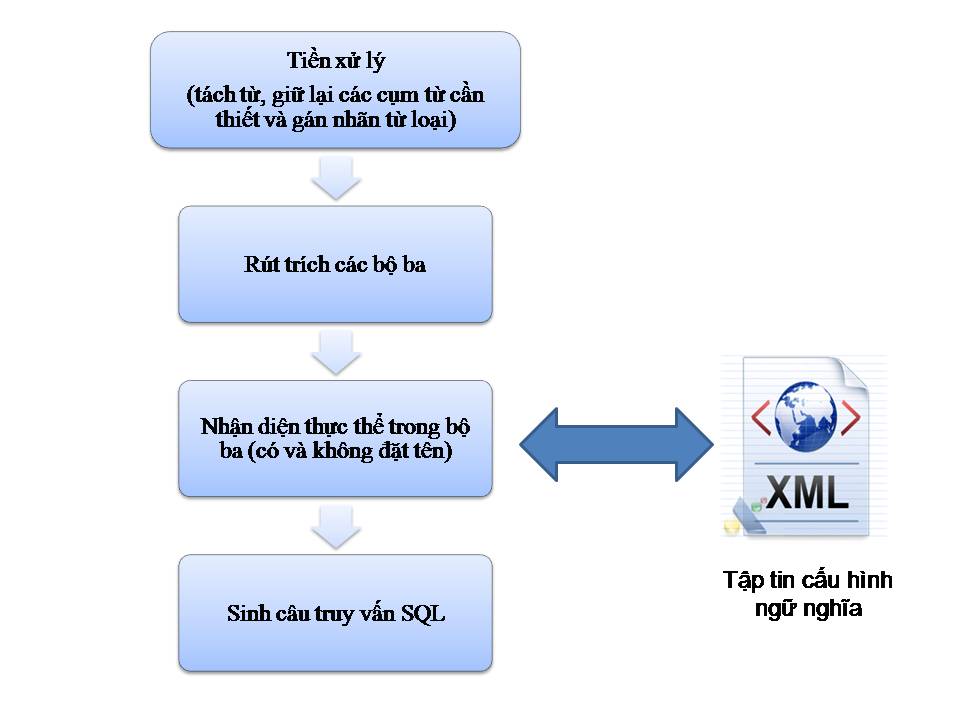
* Đối với các phương pháp tạo đồ thị ý niệm từ câu hỏi [3,7], thì việc tạo đồ thị dựa trên nhận diện các thực thể đặt tên, không đặt tên. Các công cụ để nhận diện có thể dùng là Wordnet hay một Ontology (PROTON). Với cách phân tích cú pháp để tạo đồ thị ý niệm của [7] thì có gặp khó khăn nếu người dùng nhập một câu có nhập nhằng về cú pháp (hoặc sai cú pháp). Vì thế nhóm thấy hướng tiếp cận không theo việc phân tích cú pháp của [3] sẽ có thể khắc phục được điều này.
* Tương tự với nghiên cứu của tài liệu [2], theo hướng hỏi đáp trên ngôn ngữ tiếng việt, việc phân tích cú pháp của dựa theo 40 luật của tác giả. Nếu câu không có cú pháp đúng theo 40 luật này thì quá trình phân tích cú pháp thất bại.
* Hệ thống eLSSNL có hướng đi gần giống với nhóm, do hệ thống phục vụ tìm kiếm các sách báo trong kho dữ liệu Gutenberg bằng câu truy vấn tự nhiên. Tuy nhiên, việc xác định các thuộc tính bằng các chỉ định từ chỉ cho ra các kết quả là các cuốn sách. Nếu người dùng muốn biết các thông tin thêm về tác giả, các cuốn sách mà tác giả đó đã viết thì hệ thống eLSSNL không hỗ trợ.
* Ngoài ra, để nhận diện được thực thể đặt tên, là các từ (cụm từ) chỉ về tên người, địa danh , tổ chức...[19] thì ta cần có một từ điển như Wordnet để nhận diện. Hoặc với một cách khác, ta có thể dùng một ontology như theo tài liệu [3]. Đối với đề tài của nhóm, thì các thực thể đặt tên trong dữ liệu DBLP là các title, publisher, source, ... có số lượng rất lớn. Do dữ liệu của DBLP chứa hơn 1,5 triệu bài báo. Vì thế, việc nhận diện thực thể đặt tên (như tên tiêu đề bài báo) theo các cách trên có thể sẽ ảnh hưởng nhiều tới thời gian xử lý và nhận diện.

# CHƯƠNG 3: PHƯƠNG PHÁP ĐỀ XUẤT XỬ LÝ CÂU HỎI

1. **Mở đầu**

Ở chương này đề tài trình bày các bước xử lý để chuyển câu hỏi tự nhiên thành câu truy vấn SQL. Câu truy vấn này sẽ được thực thi để tìm kiếm các bài báo trong thư viện DBLP, trả về cho người dùng.

Các bước xử lý được minh họa ở hình sau:

****Hình – Mô hình minh họa các bước xử lý câu hỏi người dùng

thành câu truy vấn SQL

Đầu tiên, câu hỏi sẽ qua thành phần tiền xử lý để thực hiện thao tác gán nhãn từ loại và một số đơn giản hóa động từ (ví dụ: đưa động từ trong tiếng Anh về dạng nguyên mẫu). Kết quả của thao tác tiền xử lý sẽ là một danh sách các từ, cụm từ được gán nhãn. Sau đó, danh sách sẽ được xử lý để rút ra các bộ ba dựa vào các luật về từ loại (ví dụ: danh từ (NN)- động từ(VB) - danh từ (NN)). Từ các bộ ba rút được, thành phần nhận diện các thực thể được thực thi và nhận biết các thực thể trong bộ ba. Ở bước này, đề tài xây dựng một tập tin XML cấu hình ngữ nghĩa cho cơ sở dữ liệu. Tập tin này miêu tả các mối quan hệ giữa các bảng và thuộc tính với nhau. Trong tập tin còn chứa các từ thể hiện cho các mối quan hệ (Ví dụ: từ *“write”*thể hiện cho mối quan hệ giữa bảng Publication và bảng Author). Việc nhận diện sẽ tiến hành bằng cách so khớp giữa từ quan hệ có trong bộ ba với các từ chỉ mối quan hệ lưu trữ trong tập tin XML. Cuối cùng, Các bộ ba sau khi được nhận diện các thực thể (có thể đóng vai trò là thuộc tính hay là đại diện của một bảng trong cơ sở dữ liệu) sẽ được chuyển thành câu truy vấn SQL. Việc hình thành câu truy vấn sẽ gồm 3 phần: tạo câu SELECT, FROM và WHERE.

Ở mục sau sẽ trình bày về cách để tạo ra tập tin cấu hình ngữ nghĩa phục vụ cho bước nhận diện thực thể. Và tiếp theo đó, chi tiết các bước để chuyển một câu hỏi thành câu truy vấn sẽ được nêu rõ.

1. **Tạo tập tin cấu hình ngữ nghĩa cho cơ sở dữ liệu**

Bởi vì thông tin trên cơ sở dữ liệu quan hệ chưa thật sự miêu tả rõ các mối quan hệ ngữ nghĩa giữa các bảng với nhau hoặc giữa bảng với các thuộc tính của bảng đó. Chẳng hạn, với một bảng có tên là Publication và nó có thuộc tính là publisher, ta có thể định nghĩa quan hệ giữa bảng Publication và thuộc tính publisher là “is published”.

Hơn nữa, tên của các bảng hay thuộc tính trong cơ sở dữ liệu đôi lúc được đặt bởi người dùng, điều đó gây ra sự khó khăn cho việc nhận dạng các thực thể của câu hỏi.

Ví dụ: với câu hỏi *Who write “*Active Database Systems*”*. Giả sử trong cơ sở dữ liệu, bảng chứa thông tin để trả lời câu hỏi vừa rồi có tên là dblp\_pub\_new. Đây là một tên được đặt bởi người tạo ra cơ sở dữ liệu, cho nên nó không mang một ngữ nghĩa nào hết. Vì thế, việc tận dụng các từ điển đồng nghĩa để xác định sự xuất hiện của tên bảng trong câu trở nên khó khăn. Nhưng nếu ta đặt tên bảng *dblp\_pub\_new* bằng một tên tượng trưng khác có nghĩa như *publication* thì việc nhận dạng sẽ dễ dàng hơn nhiều.

Đồng thời, cần có một thông tin để miêu tả các mối quan hệ giữa bảng và các thuộc tính trong bảng, ánh xạ giữa các bảng nếu các bảng đó có quan hệ với nhau. Điều này sẽ giúp ích nhiều trong việc phát sinh câu truy vấn SQL tự động. Từ một câu hỏi được đưa ra, dựa vào các đối tượng trong câu hỏi đó và những mối quan hệ ánh xạ của các bảng với nhau trong database, hệ thống sẽ sinh ra câu truy vấn.

Chính vì thế, nhóm đề xuất một lược đồ XML để mô tả thông tin của database và các mối quan hệ (hình 3).



Hình 3 – Lược đồ XML biểu diển mối quan hệ ngữ nghĩa cho cơ sở dữ liệu

Trong đó: (giá trị đầu là tên của nút cha, giá trị trong dấu “[]” là tên của nút con)

* Database[database-name]: tên của database.
* database [mapping-table]: là thông tin của table ở giữa dùng để ánh xạ mối quan hệ n-n giữa 2 table với nhau.
* mapping-table [table-key]: là khóa chính của table đầu tiên.
* mapping-table [related-table-key]: là khóa chính của table quan hệ.
* mapping-table [mapping-table-name]: là tên của table ánh xạ đã có sẵn trong database.
* mapping-table [name] tên của mapping-table do người dùng đặt.
* database[table]: thông tin của table trong database
* table [primary-key]: Khóa chính của table.
* table [table-name]: Tên của table.
* table[visible]: là thuộc tính dùng để thể hiện table đó có được hiện ra cho người dùng thấy hay không.
* table[table-alias]: tên gọi khác của table đó. Ví dụ: Một table có table-name là dblp\_pub\_new. Ta có thể đặt table-alias là Publication để người dùng dễ hiểu.
* table[field] thông tin thuộc tính của table.
* field[field-name] tên của thuộc tính trong database.
* field[field-alias] tên gọi khác của thuộc tính.
* field[type]: Kiểu dữ liệu của thuộc tính đó.
* field[visible] thể hiện thuộc tính đó có hiện cho người dùng thấy hay không.
* field[relation]: thể hiện mối quan hệ giữa table và thuộc tính.
* relation[relation-name] tên của mối quan hệ.
* relation[related-table] tên của table quan hệ. Thông tin này sẽ hiểu rõ hơn trong ví dụ.
* relation[reversed-relation] tên quan hệ đảo.
* relation[mapping-table] tên của table-mapping ở giữa 2 table

1. **Các bước xử lý câu hỏi người dùng**
   1. **Tiền xử lý**

Bước xử lí này chủ yếu sẽ thực hiện công việc gán nhãn từ loại cho các từ trong câu hỏi mà ngươì dùng nhập vào.

* Đầu vào: câu hỏi người dùng nhập vào.
* Đầu ra: Danh sách các từ đã được gán nhãn từ loại.

**Xây dựng danh sách nhãn từ loại**

Việc gán nhãn từ loại sẽ giúp ta dễ dàng xác định vai trò các từ trong câu. Từ đó, ta dễ dàng ánh xạ với các bộ từ điển ngữ nghĩa như Wordnet để so sánh đối chiếu, hay tìm đồng nghĩa, tìm khái niệm, tổng quá hóa...

Hiện nay, có nhiều công cụ gán nhãn từ loại được đưa ra như OpenNLP[[16]](#footnote-17), Annie Pos Tagger[[17]](#footnote-18) … Mặc dù các công cụ đó đã giúp người dùng có thể gán nhãn hầu như chính xác các từ loại trong câu tiếng anh, tuy nhiên với những câu có nội dung chứa những cụm từ là tên tác giả: Philip K. Chan, Mark Twain…hay tên bài báo: “Active database system”,… thì việc gán nhãn từ loại từng từ trong câu lại không chính xác về mặt ý nghĩa của cụm từ đó**.** Điển hình: Các công cụ đó chưa có thể hiểu được những từ viết hoa liên tiếp như Philip K. Chan, Mark Twain… hay cụm từ trong dấu ngoặc kép như “Active database system”,… mà thay vào đó là những từ riêng lẽ trong cụm từ đó.

Chẳng hạn với câu hỏi:

*Who is the author of* ***“A Distributed Object Manager for the Smalltalk-80 System”****?*

*Thì kết quả xử lý Pos Tagger với công cụ OpenNLP sẽ cho ra kết quả như sau:*



Kết quả sẽ là:

*Who(*WP) *is(*VBZ) *the(DT) author(NN) of(IN) “(NN)* ***A(DT) Distributed(NNP) Object(NNP) Manager(NNP) for(IN) the(DT) Smalltalk-80(JJ) System(NNP) ”(NNP) ?***

Trong đó:

WP: Đại từ câu hỏi.

VBZ: Động từ số ít.

JJ: Tính từ

NN: Danh từ số ít

NNP: Danh từ viết hoa số ít.

DT: Từ hạn định

IN: Giới từ.

(Tham khảo thêm các nhãn từ loại thông dụng ở phụ luc B)

Công cụ ANNIE POS TAGGER sẽ cho kết quả tương tự với khác biệt là dấu ngoặc kép thứ 2 được gán nhãn là NN (danh từ số ít) trong khi với công cụ OpenNLP là NNP(danh từ viết hoa số ít) và với từ ***Smalltalk-80*** thìxuất hiện nhãn NNP (danh từ viết hoa số ít) CD (bảng số) thay vì JJ( tính từ) như ở OpenNLP.



Kết quả:

*Who(*WP) *is(*VBZ) *the(DT) author(NN) of(IN) “(NN)* ***A(DT) Distributed(VBN) Object(NNP) Manager(NNP) for(IN) the(DT) Smalltalk-(NNP) 80(CD) System(NNP) ”(NN) ?***

Với kết quả chạy thử 2 công cụ trên, đề tài nhận thấy kết quả của 2 công cụ hầu như giống nhau. Cho nên đề tài chọn công cụ OpenNLP làm công cụ chính trong quá trình cài đặt. Đồng thời, với 2 kết quả đó cho thấy các cụm từ như ***“A Distributed Object Manager for the Smalltalk-80 System”*** đã bị mất đi ý nghĩa của nó.

Để không làm mất đi ý nghĩa của những cụm từ như tên tác giả, tên bài báo,…đề tài sẽ không đưa những cụm từ đó tham gia vào quá trình gán nhãn từ loại từ động mà gán nhãn bằng tay là danh từ viết hoa số ít(NNP). Những cụm từ được xác định theo điều kiện sau:

* Những từ viết hoa hoặc những cụm từ viết hoa liên tiếp.

Ví dụ: Mark Twain, Philip K. Chan...

* Những từ hoặc cụm từ trong dấu ngoặc kép.

Ví dụ: “Active Database Systems”.

* Những từ viết hoa tòan bộ

Ví dụ: ACM, IEEE …

Ví dụ: Với câu hỏi *Who is the author of* ***“A Distributed Object Manager for the Smalltalk-80 System.”****?*, ta gom những từ trong chuỗi “***A Distributed Object Manager for the Smalltalk-80 System.***” thành một cụm từ. Gán nhãn nó lại thành NNP(danh từ viết hoa số ít).

*Kết quả gán nhãn từ lọai(bằng OpenNLP): Who(WP) is(VBZ) the(DT) author(NN) of(IN)* ***“A Distributed Object Manager for the Smalltalk-80 System.”*** *(NNP)*

Các từ sau khi được gán nhãn sẽ được đưa vào danh sách. Ngoài ra, các mạo từ như: *the, a, an và một số ký hiệu đặc biệt như (?,:,;...)* được lọai bỏ khỏi danh sách do chúng không có ý nghĩa gì trong thao tác truy vấn.

**Đơn giản hóa động từ (cụm động từ)**

Mặc dù đã gán nhãn từ loại ở bước trên, nhưng các động từ trong câu vẫn cần được đơn giản hóa. Do đôi khi trong câu xuất hiện các cụm động từ hay động từ như has written, wrote,…có vai trò như nhau về mặt quan hệ. Như trong ví dụ sau:

* Philip K. Chan has written “Active database System”.
* Philip K. Chan wrote “Active database System”.

Ta nhận thấy hai động từ (cụm động từ) “wrote” và “has written” có mối quan hệ như nhau giữa Philip K. Chan và “Active database System”. Và 2 động từ(cụm động từ) này có thể được đơn giản hóa thành từ “write” để chỉ mối quan hệ trong 2 câu trên.

Do vậy, đề tài đưa ra một số quy tắc để đơn giản hóa động từ (cụm động từ) như sau:

* VBP/VBZ(1) + VBN(2) 🡪 VB (2).

Ví dụ: has written 🡪 write.

* VBP/VBZ (1) + VBN (2) + VBN (3) 🡪 VB (2) + VB(3)

Ví dụ : has been written 🡪 be write

* VBP/VBZ (1) + VBN (2) + VBG (3) 🡪 VB(3)

Ví dụ: has been writing 🡪 write.

* VBP/VBZ(1) + VBG(2) 🡪 VB (2).

Ví dụ: is writing 🡪 write.

* VBP/VBZ(1)-be + VBN(2) 🡪 VB(1) + VB(2) (VBP/VBZ là động từ TOBE)

Ví dụ: are written 🡪 be write

* VBP/VBZ(1) + VBG(2) + VBN(3) 🡪 VB(1) + VB(3)

Ví dụ: is being written 🡪 be write

* VBD(1) 🡪 VB(1).

Ví dụ: wrote🡪 write.

* VBD(1) + VBG(2) 🡪 VB(2).

Ví dụ: was writing 🡪 write.

* VBD(1) + VBN(2) 🡪 VB(1) + VB(2)

Ví dụ: was written 🡪 be write

* VBD(1) + VBG(2) + VBN(3) 🡪 VB(1) + VB(3)

Ví dụ: was being written 🡪 be write.

* MD(1) + VBN(2) 🡪 VBP(3)

Ví dụ: can write 🡪 write.

* MD(1) +VB(2) + VBN(2) 🡪VB(2) + VB(3)

Ví dụ: can be written 🡪 be write

Trong đó:

* VBZ, VBD, VBN, MD… là những tự loại trong phụ lục B.
* 1,2,3 …là số thứ tự của động từ

Để phục vụ cho bước đơn giản hóa động từ (cụm động từ) ở trên, đề tài sử dụng công cụ Wordnet để chuyển các động từ về dạng nguyên mẫu.

**Rút các bộ ba**

Bước này chủ yếu sẽ rút ra các bộ ba chứa trong câu hỏi, việc rút ra các bộ ba sẽ giúp ta xác định mối quan hệ giữa các đối tượng trong câu hỏi.

Ở bước rút các bộ ba, đầu vào sẽ là danh sách từ loại tạo ra từ bước [ref]. Đầu ra sẽ là tập các bộ ba quan hệ chứa trong câu hỏi, Tuy nhiên các bộ ba này chưa được nhận diện.

Theo cách tạo đồ thị ý niệm của Wael Salloum[REF], một số dạng mẫu của câu được xác định trong quá trình hình thành đồ thị. Những mẫu này dựa trên cây cú pháp của câu sau khi được phân tích. Ví dụ: “The human liver secretes the bile” thì có mẫu NP-VP-NP (Noun Phrase - Verb Pharse – Noun Pharse) tương ứng với câu :

(The(DT) human(ADJ) liver(NN)) (**NP**) - (secretes(VBZ))(**VP**) - (the(DT) bile (NN))(**NP**)

Ở đây, nhóm sử dụng nhãn từ loại thay cho cây cú pháp. Vì theo nhận định, nếu dùng cây cú pháp thông tin có thể bị sai lệch nếu người dùng nhập sai cú pháp. Ngoài ra, sử dụng gán nhãn từ loại sẽ dễ dàng xử lý hơn so với cây cú pháp (giữa duyệt cây và duyệt mảng). Từ đó, nhóm đề ra những luật từ loại như sau.

Gồm 19 luật từ loại:

1. (NN|NNP)-1 VB-2 (NN)-3 NNP-4-NA VB-5 --> NNP-4-NA,VB-4,(NN|NNP)-1 -->DEL
2. (NN)-1 WDT-2 (NN|NNP)-3-NA VB-4 -->(NN|NNP)-3-NA,VB-4,(NN)-1 -->DEL
3. (NN)-1 WDT-2 VB-3 VB-4 IN-5 (NN|NNP|CD)-6-NA --> NN-1,VB-3 VB-4 IN-5,(NN|NNP|CD)-6 -->DEL
4. ...

(Tham khảo 19 luật tại phụ lục C)

Trong luật, vế trái và vế phải phân cách bởi -->. Trong đó, vế trái là đầu vào, vế phải là đầu ra.

Vế trái là một dãy các từ loại mà danh sách các từ loại trong vế trái sẽ so khớp với các từ loại trong câu ở từng vị trí đã được đánh số.

Vế phải sẽ có 3 vế nhỏ phân cách với nhau bởi dấu phẩy, 3 vế nhỏ này lần lượt là chủ thể, từ quan hệ, đối tượng.

Trong mỗi vế sẽ có những từ loại, thông tin trong từ lọai sẽ cách nhau bởi dấu “–”. Trong đó:

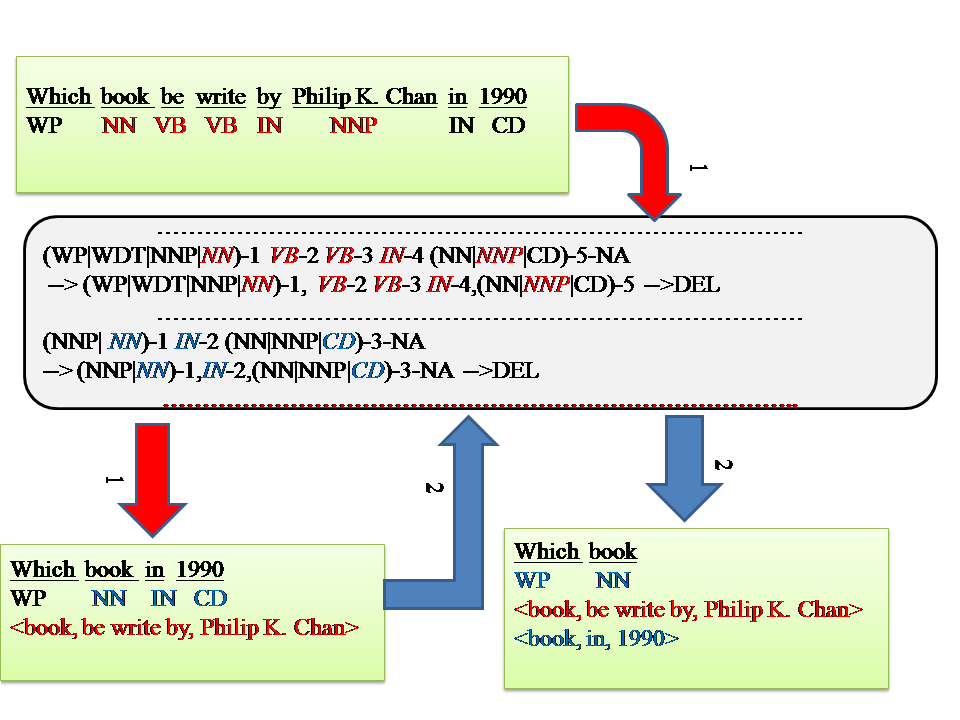
* Vế thứ 1 (NPP, NN, VBN...) là từ loại của từ.
* Vế thứ 2 (1, 2…) chỉ vị trí của từ loại trong luật.
* Vế thứ 3 (be, have, NA…) (có thể có hoặc không) chỉ giá trị của từ loại đó. Nếu là NA thì có nghĩa là từ loại tại vị trí tương ứng sẽ bao gồm một hay nhiều từ (cụm từ) có từ loại giống với từ loại ở cùng vị trí trong luật , và các từ loại được nối với nhau bởi các từ quan hệ như And, Or

Ví dụ: Who(WP) write(VB) DataBase(NNP) and(CC) Network Sytem(NNP).

Trong câu trên thì NNP sẽ có giá trị là *NA* vì nó bao gồm 2 danh từ liên tiếp nhau (là Database và Network System) và giữa chúng không có bất kỳ từ quan hệ là động từ hay giới từ nào.

* Từ DEL nói lên rằng các từ loại trong câu có được loại bỏ đi hay không sau khi rút được bộ ba. Việc xóa sẽ bắt đầu từ từ ví trí sau danh từ đầu tiên trong luật.

Ví dụ: Which book was written by Philip K. Chan in 1990?



Hình [num]: Ví dụ rút bộ ba.

Câu trên sẽ được chuyển thành:

Which(WP) book(NN) be(VB) write(VB) by(IN) Philip K. Chan(NNP) in(IN) 1990(CD)

Đầu tiên nó sẽ thõa mãn luật thứ 6 là:

(WP|WDT|NNP|***NN***)-1 ***VB***-2 ***VB***-3 ***IN***-4 (NN|***NNP***|CD)-5-NA --> (WP|WDT|NNP|***NN***)-1, ***VB***-2 ***VB***-3 ***IN***-4,(NN|***NNP***|CD)-5 -->DEL

và nó sẽ rút ra được bộ ba

<Book, be write by, Philip K.Chan>.

Tiếp theo luật này sẽ tiếp tục xóa các từ loại sau từ *book* (theo luật nó sẽ bắt đầu xóa các từ loại: be(VB), write(VB),by(IN), Philip K. Chan(NNP) ). Và trở thành:

Which(WP) book(NN) in(IN) 1990(CD)

Tiếp theo nó sẽ chạy các luật kế tiếp nó và so khớp với luật thứ 14 là :

(NNP|NN)-1 IN-2 (NN|NNP|CD)-3-NA --> (WP|WDT|NNP|NN)-1,IN-2,(NN|NNP|CD)-3-NA -->DEL

và nó rút ra được bộ ba

<book, in, 1990>.

Tiếp tục, luật này sẽ lại tiếp tục xóa các từ loại sau từ *book* (theo luật nó sẽ bắt đầu xóa các từ loại: in(IN), 1990(CD) ). Và trở thành:

Which(WP) book(NN)

Lúc này nó lại tiếp tục chạy tiếp các luật còn lại cho đến hết và không còn luật nào so khớp. Cuối cùng nó rút ra được 2 bộ ba:

<Book, be write by, Philip K.Chan>.

<book, in, 1990>.

* 1. **Nhận diện thực thể**

Ở bước này sẽ nhận diện loại thực thể của từng thực thể trong các bộ ba rút được từ bước Rút bộ ba. Với việc nhận diện thực thể, ta sẽ nhận biết mỗi thực thể là một trường hợp của bảng (table) hay thuộc tính (properties) trong cơ sở dữ liệu, từ đó sẽ phục vụ việc phát sinh câu truy vấn SQL cho bước sau.

Đầu vào của bước này chính là những bộ ba rút được từ bước trên.

Đầu ra chính là những bộ ba đã được nhận diện thực thể.

Để nhận diện loại thực thể của từng thực thể trong bộ ba, ta cần kết hợp các phương pháp: nhận diện thực thực thể đặt tên, nhận diện thực thể không đặt tên mà đề tài đưa ra ở bên dưới theo các bước sau:

Bước 1: Tìm tất cả bộ ba rút ra được từ tập tin cấu hình ngữ nghĩa (xem thêm ở phụ lục E)(gọi tập bộ ba rút được từ tập tin cấu hình ngữ nghĩa là tập Q) mà bộ ba đó có từ quan hệ giống hay đồng nghĩa (hoặc gần nghĩa) với từ quan hệ của bộ ba cần nhận diện.

Ví dụ: Với câu hỏi Which book was written by Philip K. Chan?

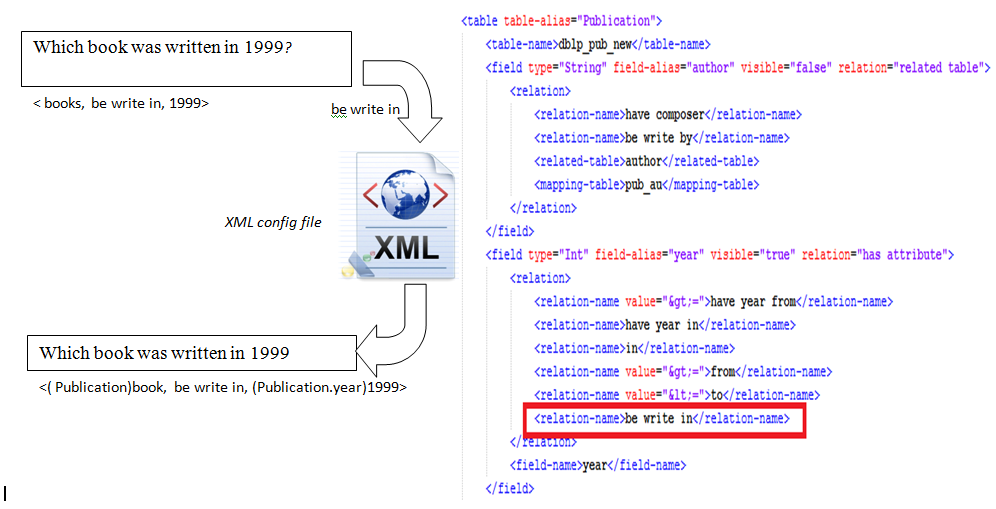
Thì sau khi qua bước rút bộ ba, ta có duy nhất 1 bộ ba cần nhận diện là < book,be write by, Philip K. Chan >.

Dựa vào từ điển Wordnet, ta dễ dàng xác định 2 động từ có đồng nghĩa hay gần nghĩa với nhau. Chẳng hạn từ write và từ compose sẽ được xác định là gần nghĩa khi tra vào từ điển Wordnet. Ta tạm gọi tập bộ ba tìm được này là tập R. Lúc này sẽ có 3 trường hợp:

* Số lượng bộ ba tìm thấy cho tập R bằng 1( size(R)=1). Ta có 2 trường hợp sau:

+ Bộ ba duy nhất không phải quan hệ ngược: Loại thực thể cho mỗi thực thể trong bộ ba cần nhận diện tương ứng với mỗi loại thực thể trong bộ ba duy nhất tìm thấy.

Ví dụ: Với câu hỏi: Which book is written in 1999?



Hình [num] Nhận diện bộ ba bằng từ quan hệ.

Bằng bước rút bộ ba, ta rút được bộ ba cần nhận diện:

<book, be write in, 1999>

Từ quan hệ “be write in” sẽ được so sánh với tất cả các từ quan hệ của các bộ ba ở phụ lục E. Ta sẽ tìm được 1 bộ ba duy nhất là:

(Publication, be write in, Publication.year)

Lúc này các thực thể trong bộ ba cần nhận diện <book, be write in, 1999> sẽ được nhận diện tương ứng với các loại thực thể trong bộ ba duy nhất (Publication, be write in, Publication.year). Ta sẽ có bộ ba được nhận diện sau:

<book: Publication, be write in, 1999: Publication.year>

+ Bộ ba duy nhất là quan hệ ngược: Ta hoán vị các thực thể trong bộ ba cần nhận diên. Sau đó, ta gán loại thực thể cho mỗi thực thể trong bộ ba cần nhận diện(đã được hoán vị) tương ứng với mỗi loại thực thể trong bộ ba duy nhất tìm.

Ví dụ: Với câu hỏi: Who publish “Active Database System” ?

Bằng bước rút bộ ba, ta rút được bộ ba sau:

<who, publish, “Active Database System”>

Từ quan hệ “publish” sẽ được so sánh với tất cả các từ quan hệ của các bộ ba ở phụ lục E. Ta sẽ tìm được 1 bộ ba duy nhất là:

(Publication, publish, Publication.publisher) (Quan hệ ngược)

Bộ ba tìm thấy ở dạng quan hệ ngược, nên ta hoán vị bộ ba cần nhận diện thành <who, publish, “Active Database System”> thành:

<“Active Database System”, publish, who>

Lúc này loại thực thể của từng thực thể trong bộ ba cần nhận diện <“Active Database System”, publish, who> (đã được hoán vị) sẽ tương ứng với các loại thực thể trong bộ ba:(Book, publish, publisher). Ta sẽ có bộ ba được nhận diện sau:

<“Active Database System”: Publication, publish, who: Publication.publisher>

* Số lượng bộ ba tìm thấy cho tập R nhiều hơn 1( size(R)>1):

Đầu tiên ta cần nhận diện các thực thể trong bộ ba rút được. Với cách thức nhận diện thực thể đặt tên( có nhãn từ loại là NNP,CD) ở mục [REF]và không đặt tên (có nhãn từ loại là NN) ở mục [REF] bên dưới, ta nhận thấy việc nhận diện thực thể chưa đặt tên chỉ là so sánh ngữ nghĩa giữa 2 danh từ, trong khi nhận diện thực thể đặt tên là đem giá trị của thực thể truy xuất vào database để tìm ra loại thực thể. Nếu database cần truy xuất có dữ liệu lớn thì việc nhận diện thực thể đặt tên sẽ tốn chi phí thời gian rất nhiều. Vì vậy, việc nhận diện sẽ ưu tiên cho thực thể chưa đặt tên trước hơn là thực thể đặt tên. Do đó, giả sử với một bộ ba rút được là <obj1, relation, obj2> thì thứ tự ưu tiên sẽ là:

* + - Nhận diện obj1(nếu obj1 là thực thể chưa đặt tên).
    - Nhận diện obj2(nếu obj2 là thực thể chưa đặt tên).
    - Nhận diện obj1(nếu obj1 là thực thể đặt tên).
    - Nhận diện obj2(nếu obj2 là thực thể đặt tên).

Sau mỗi lần nhận diện thực thể trong bộ ba (đặt tên hoặc chưa đặt tên), ta lại lấy loại thực thể vừa nhận diện được đem so sánh giống nhau với loại thực thể của các bộ ba trong tập R ở vị trí tương ứng. Sau đó loại bỏ trong tập R những bộ ba không thõa mãn điều kiện so sánh. (Chi tiết xem ở ví dụ bên dưới)

Ví dụ với câu hỏi: Which book is write by Philip K. Chan in 1999?

Câu trên sẽ được chuyển sang danh sách từ loại:

Which(WP) book(NN) be(VB) write(VB) by(IN) Philip K. Chan(NNP) in(IN) 1999(CD) ?

Ta rút được 2 bộ ba cần nhận diện:

<book, be write by, Philip K. Chan>

<book, in 1999>

Với bộ ba cần nhận diện <book, in, 1999> thì khi so sánh từ quan hệ vào tập Q, ta tìm được tập R có ba bộ ba sau:

(Publication, in, Publication.year)

(Publication, in, Publication.publisher)

(Publication, in, Publication.source)

Theo thứ tự ưu tiên vừa nêu, trong bộ ba <book, in, 1999>, ta nhận diện danh từ “book” (có nhãn từ loại là NN) trước tiên. Bằng phương pháp nhận diện thực thể chưa đặt tên được nêu ở mục [REF], ta nhận diện được “book” có loại thực thể là ***Publication***. Ta đem ***Publication*** đem so sánh với loại thực thể trong các bộ ba của tập R ở vị trí đầu tiên( vì book giữ vị trí đầu tiên trong bộ ba <book, in, 1999>).

Ta rút ra được tập R có 3 bộ ba:

(Publication, in, Publication.year)

(Publication, in, Publication.publisher)

(Publication, in, Publication.source)

Tiếp tục, ta nhận diện thực thể thứ 2 là 1999 (có nhãn từ loại là CD). Bằng phương pháp nhận diện thực thể đặt tên được nêu ở mục [REF], ta nhận diện được “1999” có loại thực thể là ***Publication.year***. Ta đem ***Publication.year*** đem so sánh với loại thực thể trong các bộ ba của tập R ở vị trí thứ 2 (vì 1999 giữ vị trí thứ 2 trong bộ ba <book, in, 1999>). Ta rút gọn được tập R còn 1 bộ ba:

(Publication, in, Publication.year)

Nếu trong tập R có một bộ ba mà loại thực thể đầu tiên và loại thực thể thứ 2 của bộ ba đó lần lượt giống với loại thực thể đầu tiên và thứ 2 của các bộ ba còn lại. Ta xóa bỏ các bộ ba còn lại và giữ lại 1 bộ ba duy nhất. Sau đó xử lý giống trường hợp 1.

Ví dụ:

Nếu tập R là 3 bộ ba:

(Publication, have publisher, Publication.publisher)

(Publication, be publish by, Publication.publisher)

(Publication, be release by, Publication.publisher)

Ta chỉ giữ lại 1 bộ ba:

(Publication, have publisher, Publication.publisher)

* Nếu số lượng bộ ba tìm thấy cho tập R bằng 0( size(R)=0):

Gán R= Q. Rồi bắt đầu xử lý giống trường hợp 2.

Việc nhận diện thực thể sẽ được tối ưu bằng cách nhận diện các thực thể có cùng loại. Như vậy sẽ giảm được chi phí cho quá trình nhận diện. Việc nhận diện này theo một số quy tắc sau:

+ Trước thực thể đặt tên là một thực thể chưa đặt tên thì thực thể đặt tên được nhận diện là loại của thực thể chưa đặc tên.

Ví dụ: *Who write book “Active Database System”?* Với câu này thì book và “Active Database System” có cùng loại thực thể.

+ Các thực thể đặt tên được nối bởi các từ quan hệ And, OR.

Ví dụ: *Who* *write “Active Database System” and “My SQL Database”?*

Với câu hỏi này thì *“Active Database System”* và *“My SQL Database”* được nhận dạng cùng một loại thực thể.

**Nhận diện thực thể đặt tên**

Trong phạm vi đề tài, thực thể đặt tên là những từ (hay cụm từ) được xác định với nhãn từ loại là danh từ viết hoa số ít(NNP) hoặc bảng số(CD). Để nhận diện được thực thể đặt tên, ta chỉ cần ánh xạ giá trị của thực thể đặt tên đó vào cơ sở dữ liệu:

* Việc ánh xạ này sẽ được ưu tiên cho các thuộc tính đại diện của cho tên bảng (ví dụ: thuộc tính đại diện của Author sẽ là Name, của Publication là title) rồi mới xét đến các thuộc tính khác.

Ví dụ: Với câu hỏi What book did James Brown write in 1999 from ACM?

Ta sẽ có được danh sách từ loại như sau:

Which(WP) book() do(VB) James Brown(NNP) write(VB) in(IN) 1999(CD) from(IN) ACM(NNP)

Qua bước rút bộ ba, ta sẽ rút được 3 bộ ba:

< James Brown, write , book>

<book, in, 1999>

<book, from, ACM>

Xét 2 bộ ba <book, in, 1999> và <book, from, ACM>.

Để nhận diện loại thực thể của ACM là bảng(table) hay thuộc tính(property) nào trong cơ sở dữ liệu, ta đem giá trị “ACM” đem tra vào tất cả các thuộc tính của các bảng, ta tìm thấy ACM sẽ ở thuộc tính ***publisher*** của bảng ***Publication***. Do đó, ACM sẽ được nhận diện là ***Publication.publisher***. Tương tự thực thể với giá trị là 1999 sẽ được nhận diện là ***Publication.year***.

* Nếu giá trị của thực thể có kiểu dữ liệu là chuỗi(String) thì chỉ tìm ở các thuộc tính có kiểu dữ liệu là chuỗi(String), nếu kiểu dữ liệu là số(Integer) thì sẽ tìm ở những thuộc tính có kiểu dữ liệu là số(Integer)...

Ví dụ với 2 bộ ba <book, in, 1999>, <book, from, ACM>. Giá trị 1999 chỉ ánh xạ vào những thuộc tính có kiểu dữ liệu là Integer như year, còn giá trị ACM sẽ được ánh xạ vào các thuộc tính có kiểu dữ liệu là String như title,publisher, source…

**Nhận diện thực thể chưa đặt tên**

Ngược lại với thực thể đặt tên là thực thể không đặt tên, nó thường là các danh từ xuất hiện trong câu như writer, author, publisher, year…Trong phạm vi đề tài, các loại thực thể này được xác định với nhãn từ loại là danh từ số ít(NN). Với loại thực thể này, chúng không có giá trị mà đóng vai trò để nhận diện các bảng và thuộc tính phục vụ cho tạo câu lệnh SQL. Vì thế, thực thể không đặt tên được nhận biết để đánh dấu sự xuất hiện của những bảng hoặc thuộc tính trong cơ sở dữ liệu.

Tuy nhiên, trong câu hỏi tự nhiên, người dùng có thể tùy ý nhập vào một từ nào đó có cùng nghĩa với từ mình đã xác định (như tên của bảng, thuộc tính). Ví dụ:

Từ “writer” sẽ đồng nghĩa với từ “author” . Nếu như không nhận biết chúng là hai từ đồng nghĩa trả lời được những câu hỏi liên quan đến writer.

Nhóm đề xuất sử dụng công cụ Similarity Wordnet để hỗ trợ cho quá trình nhận dạng thực thể chưa đặt tên. Similarity Wordnet là một công cụ có thể dùng để đánh giá độ tương đồng về nghĩa giữa hai danh từ. Nếu giữa 2 danh từ hoàn tòan cùng nghĩa, thì mức đánh giá giữa hai danh từ đó là 1. Mức thấp nhất là 0, tức là hai danh từ đó không có liên quan gì về nghĩa với nhau.

Ví dụ: Với câu hỏi *Who is the writer of “*Active Database Systems*” ?*

Thực thể chưa đặt tên ở đây (ngoại trừ từ what) là writer. Giả sử trong cơ sở dữ liệu đã có chứa một bảng có tên là Author.

Bằng công cụ Similarity Wordnet, ta có thể nhận biết được từ *writer* là *author* (có độ tương đồng là 1) thay vì tên bảng nào khác trong cơ sở dữ liệu.

Như vậy, với cách này đầu tiên ta chỉ cần tìm các bảng mà tên của nó có độ tương đồng về nghĩa với thực thể không đặt tên trong một mức giới hạn. Theo quá trình thực nghiệm, thì nhóm chỉ xét độ tương đồng trên *6.5*. Theo một thứ tự ưu tiên, đầu tiên sẽ dùng công cụ so sánh với các tên bảng trong cơ sở dữ liệu. Nếu kết quả đều không thỏa, ta so sánh tiếp với các thuộc tính trong các bảng. Nếu kết quả vẫn tiếp tục không thỏa, ta chọn thuộc tính hay tên bảng có độ tương đồng cao nhất.

* 1. **Sinh câu truy vấn SQL**

Ở bước cuối cùng, từ các bộ ba, ta phát sinh ra câu truy vấn. Bước này thao tác chỉ đơn giản duyệt qua các bộ ba, rút các phần tử và đưa vào câu truy vấn SQL. Câu truy vấn được chia làm 3 phần SELECT, FROM và WHERE.

* **SELECT**: Chọn đối tượng để lấy dữ liệu. Đối tượng được xác định dựa trên câu hỏi của người dùng. Nhóm dùng một số quy tắc đơn giản tương tự như tài liệu [3] trình bày.
  + Với câu hỏi Who : đối tượng có thể là Person (Author) hoặc Organization (publisher). Tùy theo thực thể nhận diện được trong bộ ba mà quyết định đối tượng cho câu SELECT.
  + Với các hỏi What và Which: xác định dựa trên danh từ đứng sau nó (vd: Which books ... thì lấy từ books là đối tượng SELECT).
  + Các trường hợp khác nếu không xác định được thì mặc định là đối tượng sách trong cơ sở dữ liệu.
* **FROM**: Xác định các bảng xuất hiện trong các bộ ba và bỏ vào câu truy vấn FROM.
* **WHERE**: Bỏ các thuộc tính nhận diện được trong bộ ba vào câu truy vấn. Ngoài ra, còn xác định các kết nối giữa các bảng dựa trên mối quan hệ được đinh nghĩa sẵn trong tập tin cấu hình.

Từ các bộ ba, ta có thể phát sinh ra câu truy vấn như sau:

* **Đối tượng truy vấn (SELECT):**
* Nếu thực thể câu hỏi là thuộc tính:
* Thuộc tính đó có *related-table* (Xem tập tin cấu hình ngữ nghĩa ở phụ lục D):

Ta có câu truy vấn: SELECT `related-table[table-alias]`.\*

Ví dụ: Với câu hỏi: *Who write “Database”?* Ta sẽ rút và nhận diện được bộ ba <Who: *publication.author*,write, “Database”: *publication* >. Dựa vào thông tin cấu hình ở file XML (Xem thêm ở phụ lục D), publication.author là thuộc tính và thuộc tính đó đó có chứa related-table là Author. Nên ta có câu truy vấn:

SELECT Author.\*

* Thuộc tính không có *related-table*:

Ta có câu truy vấn: SELECT `table[table-alias]`.tên thuộc tính

Ví dụ: Với câu hỏi: *Who publish “Database”?* Ta sẽ rút và nhận diện được bộ ba <Who: *publication.publisher*, publish, “Database”: *publication* >. Dựa vào thông tin cấu hình ở file XML, *publication.publisher* là thuộc tính và thuộc tính đó đó không có chứa related-table nào. Nên ta có câu truy vấn:

SELECT `publication`.publisher

* Nếu thực thể câu hỏi là table:

Ta có câu truy vấn: SELECT ` table[table-alias]`.\*

Ví dụ: Với câu hỏi: *Which book is published by O’really?* Ta sẽ rút và nhận diện được bộ ba <book: *publication*, is publish by, “ O’really’ ”: *publisher* >. Dựa vào thông tin cấu hình ở file XML, *publication* là table. Nên ta có câu truy vấn:

SELECT `publication`.\*

**Nguồn truy vấn (FROM):**

Ta cần lập danh sách tất cả các tên table từ danh sách các bộ ba, nếu có thuộc tính có mapping-table hoặc related-table thì thêm mapping-table hoặc related-table đó vào nguồn truy xuất. Một table sẽ được đưa vào nguồn truy vấn FROM theo cấu trúc:

`table[table-name]` `table[table-alias]`

Ví dụ: Với câu hỏi: Who write “Active Database Systems” ?

Ta có bộ 3 quan hệ đã được nhận diện < Who:*Author* , write, “Active Database Systems” :*Publication* >

Ở đây, “Active Database Systems” là thuộc tính *publication* của *Author*, và nó có related-table là *publication*, mapping-table là *pub\_au*. Bởi vì mapping-table *pub\_au* có mapping-table [mapping-table-name] là `dblp\_author\_ref\_new` trùng với table[table-name] của Author nên ta sẽ không thêm vào.

Nên ta có được mẫu truy vấn sau:

SELECT DISTINCT `Author`.\*

FROM

`dblp\_author\_ref\_new` `Author`,`dblp\_pub\_new` `Publication`

Trường hợp nhiều bộ ba được nhận diện giống nhau, có cùng một thực thể giống nhau, và thực thể còn lại chỉ khác nhau ở giá trị (thực thể ở dạng giống nhau).

Ví dụ: Who is the author of "Working Models for Uncertain Data" and "Active Database Systems"?

Ta có các bộ ba đã được nhận diện là:

<author: *Author*, of, "Working Models for Uncertain Data": *Author .publication*>, <author:*Author*, of, "Active Database Systems": *Author .publication*>

Ở đây 2 bộ ba được nhận dạng giống nhau, chỉ khác nhau ở giá trị là "Working Models for Uncertain Data" và "Active Database Systems". Ta gọi Author là thực thể *không* ở dạng giống nhau và thực thể còn lại thì ngược lại.

Với trường hợp này thì nguồn *From* lại bao gồm những câu truy vấn con chứa *Select*, *From*, *Where*. Ta tiếp tục đặt tên cho những câu truy vấn con đó theo quy tắc:

table[table-alias] của\_thực\_thể\_không\_ở\_dạng\_giống\_nhau + Số thứ tự n€[1,+∞] (Ngoại trừ trường hợp đầu tiên không đánh số để tối ưu hóa câu truy vấn chính)

Trong đó :

+ Select: là table hay thuộc tính của thực thể đặt tên ở dạng giống nhau. Bước này giống như quá trình tạo câu select ở trên.

+ From: giống như bước tạo From ở trên: xác định tất cả các table trong bộ ba.

+Where: giống như bước tạo Where ở mục dưới.

Sau đó INNER JOIN tất cả các câu truy vấn con đó với nhau với điều kiện là

table[table-alias] + Số thứ tự n. table[primary-key] = table[table-alias] + Số thứ tự n+1. table[primary-key]

Ví dụ: Với ví dụ trên, ta có câu truy vấn cho from như sau:

FROM

(SELECT `Author`.\*

FROM `dblp\_author\_ref\_new` `Author`,`dblp\_pub\_new` `Publication`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Publication`.`title` = "Working Models for Uncertain Data" )) Author

INNER JOIN

(SELECT `Author`.\*

FROM `dblp\_author\_ref\_new` `Author`,`dblp\_pub\_new` `Publication`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Publication`.`title` = "Active Database Systems" )) Author1 ON Author.author = Author1.author

**Điều kiện truy vấn (WHERE):**

* Nếu thực thể là một table và có giá trị, ta dựa vào thuộc tính đại diện của bảng(table) đó trong tập tin cấu hình ngữ nghĩa. Ta tạm gọi thuộc tính đại diện đó là presentation\_field. Ta có cấu trúc sau:

`table[table-alias]`.`presentation\_field[field\_name]` = giá\_trị

Ví dụ: Với thực thể “Active Database System” được nhận diện là ***Publication***, ***Publication*** có:

table[table-alias]:Publication

presentation\_field: title

presentation\_field[field\_name]: title

Giá trị là “Active Database System”. Ta có đoạn query sau:

Publication.title = “Active Database System”

* Nếu thực thể là một thuộc tính:
* Nếu một thuộc tính có related\_table và có mapping table(quan hệ n-n), ta có thêm điều kiện sau(dựa vào thông tin tại XML):

table[table-alias].table[primary\_key] = mapping\_table[mapping-table-name]. mapping\_table[table-key]

AND mapping\_table[mapping-table-name].mapping\_table[related-table-key] = related-table[table-alias].related-table[primary-key]

Nếu mapping\_table[mapping-table-name] trùng với bất kì table[table-name] thì mapping\_table[mapping-table-name] trong câu truy vấn sẽ được thay bằng table[table-alias] của table đó.

Ví dụ: Với bộ ba <book:Publication,be write by, Philip K. Chan:Publication.author> thì:

Bảng Publication có (Xem thêm ở tập tin cấu hình ngữ nghĩa):

table[table-alias]: Publication

table[primary\_key]: id

mapping\_table: pub\_au

mapping\_table[table-key]: pub\_id

mapping\_table[related-table-key]: author

mapping\_table[mapping-table-name]:dblp\_ author\_ref\_new

Bảng Author có:

table[table-alias]: Author

table[primary\_key]: author

table[name]: dblp\_author\_ref\_new

Thuộc tính Publication.author có:

related-table: Author

related-table[table-alias]: Author

related-table[primary-key]: Author

Do đó, một đoạn câu truy vấn được sinh ra như sau:

`Publication`.`id` = `dblp\_author\_ref\_new`.`pub\_id` and `dblp\_author\_ref\_new`.`author` = `Author`.`author`

Vì bảng Author và mapping\_table có cùng chung một bảng trong cơ sở dữ liệu là ***dblp\_author\_ref\_new***, nên câu truy vấn trên được rút gọn thành:

`Publication`.`id` = `Author`.`pub\_id`

+ Nếu thuộc tính có related-table đồng thời thuộc tính đó có giá trị, ta làm tương tự với thực thể là table, ta tạm gọi thuộc tính đại diện cho related-table đó là related\_presentation\_field, ta có thêm điều kiện sau:

`related-table[table-alias]`.`related\_presentation\_field[field-name]` = giá\_trị\_thuộc\_tính

Ví dụ: Với bộ ba <book:Publication,be write by, Philip K. Chan:Publication.author>. Ta có thêm một số thông tin sau:

Với bảng Author:

presentation\_field:Author

presentation\_field[field-name]:author

Như vậy ta có đoạn câu truy vấn sau:

`Author`.`author` = ‘Philip K. Chan’

* Nếu một thuộc tính chỉ có mỗi related\_table (quan hệ n-1) , ta có điều kiện:

table[table-alias].field[field\_name] = related\_table[table\_alias].related-table[primary-key]

Trong đề tài không có ví dụ cụ thể cho trường hợp này, ở đây field[field\_name] được xem là khóa ngoại cho một bảng khác.

* Nếu thuộc tính đó đã có giá trị, ta chỉ cần thêm điều kiện:

`table[table-alias]`.`field[field\_name]` toán\_tử giá\_trị\_thuộc\_tính

Toán\_tử: là những toán tử được định nghĩa sẵn cho mỗi từ quan hệ.

from(>=),in( =),to( <=), ….

Ví dụ: Với bộ ba đã được nhận diện: <book:Publication,from,1999:Publication.year>

Với thực thể 1999 có:

table[table-alias]: Publication

toán\_tử: >=

Ta có đoạn truy vấn sau:

`Publication`.`year` >= 1999

Ví dụ :Với câu hỏi : Who write “Active Database Systems”?

Với các bộ 3 đã được nhận diện < Who:*Author* , write, “Active Database Systems” :*Publication* >

, “Active Database Systems” là thuộc tính publication của author và có mapping-table-name là ref\_author\_book, có related-table là publication, đồng thời thuộc tính đại diện của publication là publication.title có giá trị là “Active Database Systems”, ta sinh được câu truy vấn sau:

SELECT DISTINCT `Author`.\*

FROM

`dblp\_author\_ref\_new` `Author`,`dblp\_pub\_new` `Publication`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND `Publication`.`title` = `Active Database Systems `

Ví dụ: Với câu hỏi “Who published “Active Database Systems”

Các bộ ba được tìm thấy <“Active Database Systems” : *Publication*,publish, Who : *Publication.publisher*. Ta có câu truy vấn:

SELECT Publication.publisher

FROM

`dblp\_pub\_new` `Publication`

WHERE

(`Publication`.`title` = “Active Database Systems” )

Sau khi kết hợp 3 phần, ta có được câu truy vấn hoàn chỉnh và lấy dữ liệu từ DBLP. Như mô hình hệ thống ở mục sau, thành phần xử lý câu trả lời được nêu ra chủ yếu để trình bày nội dung kết quả trên giao diện web. Ví dụ: thông tin trả về là các bài báo thì người dùng có thể nhấp chuột vào các bài báo để xem thông tin chi tiết (đường link).

* 1. **Một số ví dụ**

Các ví dụ được giải quyết với thông tin database trên dblp.

***Ví dụ 1:*** Câu hỏi “Who published “Active Database Systems”?

* Bước 1: Gán nhãn từ loại

Đầu vào : Câu hỏi.

Đầu ra: Danh sách các từ đã gán nhãn từ loại.

Kết quả: WP/Who VBD/published NNS/“Active Database Systems”.

* Bước 2: Đơn giản hóa danh từ, động từ, cụm động từ

Đầu vào: Danh sách nhãn từ loại ở bước 1

WP/Who VBD/published NNS/“Active Database Systems”.

Đầu ra: Danh sách nhãn từ loại đã được đơn giản hóa.

Kết quả: WP/Who VB/publish NNS/“Active Database Systems”.

* Bước 3: Rút bộ ba

Đầu vào: danh sách từ được gán nhãn ở bước 2.

WP/Who VB/publish NNS/“Active Database Systems”.

Đầu ra: Tập các bộ ba quan hệ.

Xử lý: Danh sách các từ được gán nhãn sẽ khớp với luật 13 trong tập luật (Xem thêm ở phụ lục C):

(WP|WDT|NNP|NN)-1 VB-2 (NN|NNP)-3-NA --> (WP|WDT|NNP|NN)-1,VB-2,(NN)-3

Kết quả: <Who, publish, “Active Database Systems”>

* Bước 4: Nhận diện thực thể

Đầu vào: Các bộ 3 quan hệ chưa được nhận diện ở bước 3.

<Who, publish, “Active Database Systems”>

Đầu ra: Bộ ba quan hệ đã được nhận diện.

Xử lý: Từ quan hệ “publish” sẽ được ánh xạ vào danh sách những bộ ba rút được từ file cấu hình ngữ nghĩa (những bộ ba này xem ở phụ lục E). Lúc này sẽ tìm thấy 1 bộ ba duy nhất:

(Publication, publish, Publication.publisher). (Quan hệ ngược trong tập tin XML cấu hình ngữ nghĩa <reverse-relation>)

Vì bộ ba duy nhất tìm thấy ở quan hệ ngược (reverse-relation) nên ta sẽ hoán vị các thực thể trong bộ ba <Who, publish, “Active Database Systems”> rồi gán loại thực thể tương ứng với bộ ba duy nhất tìm được:

<“Active Database Systems”:Publication, publish, Who: Publication.publisher>

Kết quả:

<“Active Database Systems”:Publication, publish, Publication.publisher>

* Bước 5: Sinh câu truy vấn  
  Đầu vào: Danh sách các bộ 3 quan hệ đã nhận diện ở bước 4.

<“Active Database Systems”:Publication, publish, Publication.publisher>

Đầu ra: Câu truy vấn SQL

*Đối tượng truy vấn :*Select `Publication`.`publisher`

*Nguồn truy vấn:*From `dblp\_pub\_new` `Publication`

*Điều kiện:* Where `Publication`.title = “Active Database Systems”

*Câu truy vấn:*

Select `Publication`.`publisher`

From `dblp\_pub\_new` `Publication`

Where `Publication`.title = “Active Database Systems”

***Ví dụ 2:*** Câu hỏi *Which books were written by Rafiul Ahad and Amelia Carlson in 2010 ?*

* Bước 1: Gán nhãn từ loại

Đầu vào : Câu hỏi.

Đầu ra: Danh sách các từ đã gán nhãn từ loại.

Kết quả:

WDT/Which books/NNS were/VBD written/VBN by/IN Rafiul Ahad/NNP and/CC Amelia Carlson/NNP in/IN 2011/CD

* Bước 2: Đơn giản hóa danh từ, động từ, cụm động từ

Đầu vào: Danh sách nhãn từ loại ở bước 1

WDT/Which books/NNS were/VBD written/VBN by/IN Rafiul Ahad/NNP and/CC Amelia Carlson/NNP in/IN 2011/CD

Đầu ra: Danh sách nhãn từ loại đã được đơn giản hóa.

Kết quả:

WP/Which book/NN be/VB write/VB by/IN Rafiul Ahad/NNP and/CC Amelia Carlson/NNP in/IN 2011/CD

* Bước 3: Rút bộ ba

Đầu vào: danh sách từ đã gán nhãn từ loại ở bước 2

WP/Which book/NN be/VB write/VB by/IN Rafiul Ahad/NNP and/CC Amelia Carlson/NNP in/IN 2011/CD

Đầu ra: Tập các bộ ba quan hệ.

Xử lý:

Danh sách các từ được gán nhãn đầu tiên sẽ khớp với luật 8 trong tập luật (Xem thêm ở phụ lục C):

(WP|WDT|NNP|NN)-1 VB-2 VB-3 IN-4 (NN|NNP|CD)-5-NA --> (WP|WDT|NNP|NN)-1,VB-2 VB-3 IN-4,(NN|NNP|CD)-5 -->DEL

Nên ta rút được 2 bộ ba:

<book, be write by, Rafiul Ahad>

<book, be write by, Amelia Carlson >

Đồng thời, trong luật có từ “DEL”, do đó sẽ xóa bỏ tạm thời các từ từ sau danh từ đầu tiên là book. Do đó, danh sách tạm các từ được gán nhãn từ loại là:

WP/Which book/NN in/IN 2011/CD

Tiếp tục nó lại so khớp với luật thứ 17 trong tập lục ở phụ lục C:

(NNP|NN)-1 (IN|TO)-2 (NN|NNP|CD)-3-NA --> (WP|WDT|NNP|NN)-1,(IN|TO)-2,(NN|NNP|CD)-3-NA -->DEL

Lúc này nó sẽ rút ra được 1 bộ ba nữa là:

<book,in, 2011>

Tiếp tục nó sẽ xóa các từ được gán nhãn từ loại từ sau danh từ đầu tiên và trở thành danh sách tạm thời sau:

WP/Which book/NN

Danh sách này không còn phụ hợp với luật nào khác, do đó kết quả trả về sẽ là ba bộ ba vừa rút được.

Kết quả:

<book, be write by, Rafiul Ahad>

<book, be write by, Amelia Carlson >

<book,in, 2011>

* Bước 4: Nhận diện thực thể

Đầu vào: Các bộ 3 quan hệ chưa được nhận diện ở bước 3.

<book, be write by, Rafiul Ahad>

<book, be write by, Amelia Carlson >

<book,in, 2011>

Đầu ra:Các bộ ba quan hệ đã được nhận diện.

Xử lý:

Với bộ ba: <book, be write by, Rafiul Ahad>

Từ quan hệ “be write by” sẽ được ánh xạ vào danh sách những bộ ba rút được từ file cấu hình ngữ nghĩa (những bộ ba này xem ở phụ lục E). Lúc này sẽ tìm thấy 1 bộ ba duy nhất:

(Publication, be write by, Publication.author).

Vì bộ ba tìm được là duy nhất nên ta gán loại thực thể cho bộ ba chưa nhận diện tương ứng với bộ ba duy nhất tìm được:

<book: Publication, be write by, Rafiul Ahad: Publication.author>

Tương tự, với bộ ba <book, be write by, Amelia Carlson > ta sẽ có bộ ba được nhận diện như sau:

<book: Publication, be write by, Amelia Carlson: Publication.author>

Với bộ ba chưa nhận diện <book, in, year>, từ quan hệ “in” sẽ được ánh xạ vào danh sách những bộ ba rút được từ file cấu hình ngữ nghĩa (những bộ ba này xem ở phụ lục E). Lúc này sẽ tìm thấy 3 bộ ba:

R={

<Publication,in, Publication.year>,

<Publication,in, Publication.source>,

<Publication,in, Publication.publisher>,

}

Lúc này, ta nhận diện thực thể “book”, vì thực thể “book” đã được nhận diện là “Publication” ở 2 bộ ba trên, nên ta không cần phải nhận diện thực thể “book” lần nữa. Ta đem so sánh loại thực thể “Publication” so sánh với các loại thực thể đầu tiên ở tập R. Lúc này, ta sẽ rút gọn tập R thành

R={

<Publication,in, Publication.year>,

<Publication,in, Publication.source>,

<Publication,in, Publication.publisher>,

}.

Tiếp theo, ta nhận diện thực thể 2001. Ta đem giá trị 2011 đem tra vào các thuộc tính Publication.year, Publication.source, Publication.publisher. Vì 2001 là kiểu số nên ta tra vào các thuộc tính có kiểu số (Integer) trong cơ sở dữ liệu (chỉ có Publication.year thỏa mãn), ta suy ra được 2001 có loại thực thể là Publication.year. Tiếp tục, ta đem loại thực thể Publication.year đem so sánh với các loại thực thể thứ 2 trong tập R. Ta rút được tập R chỉ có 1 bộ ba duy nhất:

R={

<Publication, in, Publication.year>

}.

Vì chỉ có 1 bộ ba duy nhất trong tập R, nên ta gán loại thực thể ở bộ ba duy nhất này cho bộ ba cần nhận diện <book, in, 2001>. Ta có bộ ba đã được nhận diện như sau:

<book:Publication, in, 2001:Publication.year>

Kết quả:

<book: Publication, be write by, Rafiul Ahad: Publication.author>

<book: Publication, be write by, Amelia Carlson: Publication.author>

<book:Publication, in, 2001:Publication.year>

* Bước 5: Sinh câu truy vấn  
  Đầu vào: Danh sách các bộ 3 quan hệ đã nhận diện ở bước 4.

<book: Publication, be write by, Rafiul Ahad: Publication.author>

<book: Publication, be write by, Amelia Carlson: Publication.author>

<book:Publication, in, 2001:Publication.year>

Đầu ra: Câu truy vấn MySQL

*Đối tượng truy vấn :*Select `Publication`.\*

*Nguồn truy vấn:*

Vì *Rafiul Ahad* và *Amelia Carlson* là 2 bộ ba có từ quan hệ và loại thực thể giống nhau, chỉ khác nhau ở giá trị. Nên nguồn truy vấn của nó là một câu truy vấn con:

FROM

(SELECT `Publication`.\*

FROM `dblp\_pub\_new` `Publication`,`dblp\_author\_ref\_new` `Author`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Author`.`author` = ‘*Rafiul Ahad’* )) Publication

INNER JOIN

(SELECT `Publication`.\*

FROM `dblp\_pub\_new` `Publication`,`dblp\_author\_ref\_new` `Author`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Author`.`author` = ‘*Amelia Carlson’* )) Publication1 ON Publication.id = Publication1.id

*Điều kiện:* (`Publication`.`year` = 1999 )

*Câu truy vấn:*

SELECT DISTINCT `Publication`.\*

FROM

(SELECT `Publication`.\*

FROM `dblp\_pub\_new` `Publication`,`dblp\_author\_ref\_new` `Author`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Author`.`author` = ‘*Rafiul Ahad’* )) Publication

INNER JOIN

(SELECT `Publication`.\*

FROM `dblp\_pub\_new` `Publication`,`dblp\_author\_ref\_new` `Author`

WHERE

`Publication`.`id` = `Author`.`pub\_id` AND (`Author`.`author` = ‘*Amelia Carlson’* )) Publication1 ON Publication.id = Publication1.id

WHERE

(`Publication`.`year` = 1999 )

# CHƯƠNG 4: HIỆN THỰC HỆ THỐNG

1. **Tổng quan về hệ thống**
   1. **Mô hình hệ thống:**

Hệ thống được xây dựng gồm hai luồng chính: người dùng nhập vào từ khóa để tìm bài báo và người dùng nhập câu hỏi đễ tìm kiếm.



Hình 6 – Mô hình hệ thống tìm kiếm bài báo

Chương trình hệ thống gồm hai luồng chính:

-Tìm kiếm theo từ khóa: Người dùng nhập vào các từ khóa để tìm kiếm. Từ khóa có thể là tên bài báo, tên nhà xuất bản và nguồn của bài báo. Kết quả trả về là các bài báo có thuộc tính chứa từ khóa mà người dùng đã nhập.

-Dùng câu hỏi để tìm kiếm: Thay vì nhập từ khóa. Hệ thống cho phép người dùng nhập vào một câu hỏi tự nhiên để tìm kiếm. Kết quả trả về sẽ là các bài báo phù hợp với các thông tin trong câu hỏi.

* 1. **Các chức năng trong hệ thống**

|  |  |  |
| --- | --- | --- |
| **Chức năng** | **Đầu vào** | **Đầu ra** |
| Tìm kiếm chung theo  từ khóa | Từ khóa là tên tiêu đề bài báo  (hoặc tên nguồn hoặc tên nhà xuất bản) | Danh sách các bài báo liên quan có tiêu đề (tên nguồn, tên nhà xuất bản) chứa từ khóa. |
| Tìm kiếm theo tên  tác giả | Tên tác giả | Danh sách các tác giả có tên tương tự, click chọn tác giả sẽ hiển thị các bài báo mà tác giả đó đã viết |
| Tìm kiếm nâng cao | Các thuộc tính mà người dùng có thể nhập: Năm (lớn hơn, nhỏ hơn, bằng), nhà xuất bản, nguồn, loại. | Danh sách các bài báo liên quan |
| Đa ngôn ngữ |  | Hiển thị đa ngôn ngữ |
| Tìm kiếm  hỏi đáp | Câu hỏi dạng ngôn ngữ tự nhiên | Các bài báo liên quan đến nội dung câu hỏi |

Bảng 5 – bảng các chức năng trong hệ thống

* Chức năng tìm kiếm: cung cấp chức năng tìm kiếm bài báo dựa vào từ khóa. Chức năng cho phép người dùng nhập vào từ khóa có liên quan đến tiêu đề, nguồn, nhà xuất bản.
* Chức năng tìm kiếm nâng cao: Cho phép người dùng nhập nhiều dữ liệu hơn để cho kết quả chính xác như mong muốn. Các thuộc tính mà người dùng có thể nhập: Năm (lớn hơn, nhỏ hơn hoặc bằng), nhà xuất bản, nguồn, loại.
* Chức năng tìm kiếm theo tên tác giả: Nhập vào tên tác giả, chương trình trả về các tác giả có tên tương tự. Khi chọn vào một tác giả sẽ hiển thị các bài báo mà tác giả đó đã viết. Các kết quả nhóm theo từng năm, giao diện hiển thị các bài báo mà tác giả đã viết trong mỗi năm một cách trực quan ( tương tự DBLP).
* Đa ngôn ngữ: Hỗ trợ giao diện hiển thị đa ngôn ngữ tiếng Anh-Tiếng Việt
* Chức năng hỏi đáp: Đây là một chức năng mới của chương trình. Người dùng nhập vào một câu hỏi tự nhiên duy nhất để tìm kiếm. Dựa trên phân tích câu hỏi chuyển thành câu truy vấn xuống cơ sở dữ liệu DBLP, chương trình trả về thông tin hoặc bài báo liên quan câu đến câu hỏi. Ngoài ra, người dùng được cung cấp một số câu hỏi mẫu có sẵn trên giao diện chức năng.

Ví dụ:

*-Câu hỏi:* How many publications were written by *Philip K. Chan*?

-Kết quả:

|  |
| --- |
| 1. [Experiments on Multi-Strategy Learning by Meta-Learning](http://localhost:8080/QAWeb/showPubDetail.do?id=107298)  2. [Learning Useful System Call Attributes for Anomaly Detection](http://localhost:8080/QAWeb/showPubDetail.do?id=234312)  3. [Incrementally Learning Rules for Anomaly Detection](http://localhost:8080/QAWeb/showPubDetail.do?id=235040)  …  …  36. [Systems for Knowledge Discovery in Databases](http://localhost:8080/QAWeb/showPubDetail.do?id=2138381) |

*-Câu hỏi:* Who wrote paper “[*Incrementally Learning Rules for Anomaly Detection*](http://localhost:8080/QAWeb/showPubDetail.do?id=235040)*.”*?

-Kết quả:

[*Denis Petrussenko*](http://localhost:8080/QAWeb/showPubsByAuthor.do?authorName=Denis%20Petrussenko)*,*

[*Philip*](http://localhost:8080/QAWeb/showPubsByAuthor.do?authorName=Denis%20Petrussenko)K. Chan,

1. **Thiết kế cài đặt hệ thống**
2. **Môi trường xây dựng hệ thống**

* Hệ điều hành: WindowXP/ Vista/ 7
* Chương trình chạy trên nền java 1.6
* Server: Glassfish 3.0
* Database: MySQL
* Framework MVC: Strut 1.3 theo mô hình MVC (Model – Controller- View)
* Các công cụ sử dụng: Wordnet, WNSimilarity, OpenNLP Postagger,

Hibernate 3.

1. **Cấu trúc các lớp**

* ***com.myapp.struts:*** gói chứa các lớp ngôn ngữ (gồm tiếng Anh và tiếng Việt).
* ***uit.qadbpss.core.search:*** nhân chứa các lớp xử lý tìm kiếm theo từ khóa, lớp thao tác với Wordnet.
* ***uit.qadbpss.core.wordnet:*** nhân chứa các lớp thao tác với bộ từ điển Wordnet và lớp thao tác các so sánh độ tương đồng ngữ nghĩa.
* ***uit.qadbpss.dbconfig:*** gói chứa các lớp cấu hình, lớp đại diện cho các thuộc tính trong bảng của cơ sở dữ liệu, lớp thao tác với file cấu hình XML.
* ***uit.qadbpss.model:*** lớp đối tượng đại diện cho một đối tượng dưới cơ sở dữ liệu ( dùng để truy xuất Hibernate).
* ***uit.qadbpss.preprocess:*** các lớp tiền xử lý câu hỏi của người dùng.
* ***uit.qadbpss.extracttriple:*** các lớp hỗ trợ rút các bộ ba trong câu hỏi.
* ***uit.qadbpss.entityrecog:*** Lớp nhận diện các đối tượng trong bộ ba.
* ***uit.qadbpss.generatequery:*** Lớp tạo câu truy vấn SQL từ các bộ ba rút ra được từ các lớp trên.
* ***uit.qadbpss.util:*** Lớp toàn cục cho cả hệ thống*.*Các lớp về kho ký tự CharPool, String Pool.

1. **Cơ sở dữ liệu DBLP**

Dữ liệu của đề tài lấy từ nguồn dữ liệu DBLP. Cấu trúc các bảng được trình bày như bên hình dưới:

****

* **Table Publication (dblp\_pub\_new)**

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Loại** | **Giải thích** |
| id | int(8) | Khóa toàn cục trong CSDL |
| dblp\_key | varchar(150) | Khóa trong file xml (một dạng khóa khác tổ chức trong dữ liệu xml) |
| title | longtext | Tiêu để bài báo |
| source | varchar(150) | Nguồn bài báo |
| source\_id | varchar(50) | Tham chiếu đến nguồn bài báo (đường dẫn trong dblp\_key) |
| series | varchar(100) | Series của bài báo ( chỉ dành cho kiểu book và proceedings |
| year | int(4) | Năm công bố |
| type | varchar(20) | Kiểu bài báo (vd: article, proceedings, book …) |
| volume | varchar(50) | Khối lượng của nguồn, nơi tác phẩm được xuất bản |
| number | varchar(20) | Số lượng nguồn nơi mà tác phẩm được công bố |
| month | varchar(30) | Tháng xuất bản |
| pages | varchar(100) | Số trang |
| ee | varchar(200) | Đường dẫn tới phiên bản điện tử |
| ee\_PDF | varchar(200) | Đường dẫn tới phiên bản điện tử dạng PDF |
| url | varchar(150) | Địa chỉ tới trang dblp ( bắt đầu với db/…) |
| publisher | varchar(250) | Tên nhà xuất bản |
| isbn | varchar(25) | Mã ISBN |
| crossref | varchar(50) | Khóa tham chiếu chéo tới một bài báo khác |
| titleSignature | varchar(255) | Tiêu đề bài báo (không có khoảng trắng) |
| doi | varchar(255) | Digital Object Identifier |
| mdate | date | Ngày chỉnh sửa |

* **Table Author (dblp\_author\_ref\_new)**

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Loại** | **Giải thích** |
| pub\_id | int(8) | Khóa tương ứng với id trong bảng dblp\_pub\_new |
| author | varchar(70) | Tên tác giả |
| editor | int(1) | Biến xác nhận tác giả có phải là người chỉnh sửa |
| author\_num | int(3) | Số thứ tự của tác giả |

* **Table Publication Reference (dblp\_ref\_new)**

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Loại** | **Giải thích** |
| id | int(8) | Khóa tương ứng với id trong bảng dblp\_pub\_new |
| dblp\_key | varchar(150) | Khóa dblp\_key của những bài báo được trích dẫn |

1. **Giao diện chương trình**

* **Giao diện chính khi khởi động chương trình**

****

* **Giao diện tìm kiếm theo từ khóa**

****

* **Giao diện chạy thực nghiệm**

**Chạy thực nghiệm với câu:** *Who is the author of "Working Models for Uncertain Data"*

**Kết quả:**

****

**Chạy thực nghiệm với câu:** *Which books were written by Rafiul Ahad and Amelia Carlson ?*

**Kết quả:**

****

**Chạy thực nghiệm với câu:** *Which papers cited to "Foundations of Databases"?*

**Kết quả:**

**

1. **Đánh giá và nhận xét**

Với phương pháp trên,việc xử lý khá dễ dàng do thao tác trên danh sách từ loại. Việc bổ sung các mẫu luật từ loại khá đơn giản và số lượng mẫu không quá lớn (19 mẫu).

Về cách nhận diện thực thể trong bộ ba, phương pháp dùng từ quan hệ sẽ không bị ràng buộc theo các phương pháp khác [1, 3, 5, 6] phải có một công cụ như GATE hay Wordnet để nhận diện các thực thể đặt tên. Vì đối với một kho dữ liệu xác định về các bài báo khoa học về công nghệ thông tin, thì GATE hay Wordnet không thể nhận diện hết được các thực thể đặt tên, do chúng chỉ nhận diện được các đối tượng thông dụng như (tên người, địa điểm, ngày tháng ...).

Tập kiếm thử gồm 100 câu dựa tên thông tin hiện có của cơ sở dữ liệu, nhu cầu tìm kiếm và một số dạng câu hỏi trong TREC. Độ chính xác khi rút bộ ba được trình bày ở bảng sau:

|  |  |
| --- | --- |
|  | Khả năng rút trích bộ ba |
| Độ chính xác | 88% |
| Số câu hỏi rút được bộ ba | 99 |
| Số câu hỏi rút đúng | 88 |
| Số câu hỏi rút sai | 8 |
| Số câu hỏi không rút được bộ ba | 1 |
| Tổng số câu | 100 |

Bảng 6 – Đánh giá khả năng rút trích bộ ba

Với kết quả này, các trường hợp không rút được bộ ba và rút thiếu chứng tỏ tập 19 mẫu luật từ loại vẫn chưa hoàn chỉnh và bao quát hết được tập mẫu. Điều này có thể giải quyết bằng cách bổ sung và chỉnh sửa lại luật. Với trường hợp bộ ba bị rút sai, nguyên nhân do một số nhập nhằng về ngữ nghĩa khiến bộ ba rút ra không đúng với ngữ nghĩa. Vấn đề này hiện chúng tôi vẫn chưa giải quyết được và đang tìm hướng khắc phục.

Chúng tôi đang hướng tới bổ sung một số câu hỏi dựa trên tham khảo từ một số nguồn khác và người sử dụng. Cuối cùng, chúng tôi mong muốn cung cấp cho người dùng một cách thức khác để tìm kiếm là sử dụng câu hỏi tự nhiên. Hướng đi tiếp theo của hệ thống sẽ phát triển thêm các chức năng gợi ý các câu hỏi tương tự cho người dùng và đề xuất ra một số hướng xử lý khi người dùng nhập không chính xác tên tác giả, tổ chức hay tiêu đề. Ngoài ra, chúng tôi có dự định sử dụng một cơ sở dữ liệu khác là Ontology – một cơ sở dữ liệu đã tổ chức theo ngữ nghĩa – để áp dụng phương pháp này.

**CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Với sự phát triển của công nghệ thông tin hiện nay, việc sử dụng một câu hỏi dưới dạng ngôn ngữ tự nhiên để tìm kiếm một thông tin hay tài liệu nào đó quả thật còn khá mới mẻ. Tuy nhiên, với sự có mặt của những hệ thống đã và đang hoạt động như : NSIR, START, Ask.com ... Điều đó đã chứng tỏ các nhà nghiên cứu đã có một mối quan tâm tới vấn đề hỏi đáp [1-6]. Vậy hướng đi của đề tài là một hướng đi đúng và thiết thực.

Hệ thống được nhóm xây dựng nên nhằm thỏa mãn một phần nào nhu cầu của người dùng trong vấn đề tìm kiếm các bài báo khoa học. Cách thức hỏi đáp không mới nhưng được đưa vào để khai thác nâng cao hiệu quả trong vấn đề tìm kiếm. Nhóm hy vọng, đây có thể là một cách thức mới trong tương lai nhằm giúp giao tiếp người và máy trở nên thân thiện hơn.

# TÀI LIỆU THAM KHẢO

## Tiếng Việt

1. *Lương Quý Tịnh Hà (2009****), Xây dựng công cụ tìm kiếm tài liệu học tập bằng các truy vấn ngôn ngữ tự nhiên trên kho học liệu mở tiếng Việt****,**luận văn Thạc Sĩ, Tp. HCM.*
2. *Cao Duy Trường (2008),* ***Dịch câu truy vấn có cấu trúc sang đồ thị ý niệm: cách tiếp cận ít phụ thuộc vào cú pháp****,**Tp.HCM****.***
3. *Đỗ Thị Thanh Tuyền (2008),* ***XÂY DỰNG HỆ THỐNG TRA CỨU THƯ VIỆN ĐIỆN TỬ BẰNG NGÔN NGỮ TỰ NHIÊN****, luận văn Thạc Sĩ Công Nghệ Thông Tin, Tp. HCM.*

## Tiếng Anh

1. *Lorand Dali, Delia Rusu, Blaz Fortuna, Dunja Mladenic and Marko Grobelnik (2009) ,* ***Question Answering Based on Semantic Grahps****, Department of Knowledge Technologies, Jožef Stefan Institute.*
2. *Meltem Serhatli and Ferda N. Alpaslan (2009),* ***An Ontology based Question Answering System on Software Test Document Domain****, World Academy of Science, Engineering and Technology 54 2009.*
3. *Wael Salloum (2009),* ***A Question Answering System based on Conceptual Graph Formalism****, Conference: The 2nd International Symposium on Knowledge Acquisition and Modeling (KAM 2009), IEEE Computer Society Press.*
4. *DongHyun Choi and Key-Sun Choi (2008)* ***Automatic Relation Triple extraction by dependency parse tree ,*** *Computer Science Department**Semantic Web Research Center, KAIST Daejeon, Korea.*
5. *Shiyan Ou, Constantin Orasan, Dalila Mekhaldi and Laura Hasler (2008),* ***Automatic Question Pattern Generation for Ontology-based Question****. In Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference (FLAIRS2008). Menlo Park, CA: AAAI Press., pp. 183 – 188.*
6. *Delia Rusu, Lorand Dali, Blaz Fortuna, Marko Grobelnik, Dunja Mladenic (2007) ,* ***Triplet Extraction from Sentences****, Department of Knowledge Technologies, Jožef Stefan Institute*
7. [*Stephan Bloehdorn*](http://www.springerlink.com/content/?Author=Stephan+Bloehdorn)*,* [*Philipp Cimiano*](http://www.springerlink.com/content/?Author=Philipp+Cimiano)*,* [*Alistair Duke*](http://www.springerlink.com/content/?Author=Alistair+Duke)*,* [*Peter Haase*](http://www.springerlink.com/content/?Author=Peter+Haase)*,* [*Jörg Heizmann*](http://www.springerlink.com/content/?Author=J%c3%b6rg+Heizmann)*,* [*Ian Thurlow*](http://www.springerlink.com/content/?Author=Ian+Thurlow) *and* [*Johanna Völker*](http://www.springerlink.com/content/?Author=Johanna+V%c3%b6lker) *(*2007) ***Ontology-Based Question Answering for Digital Libraries,*** [Lecture Notes in Computer Science](http://www.springerlink.com/content/0302-9743/) , Volume 4675/2007.
8. *Kenneth C. Litkowski (1999)* ***Question-Answering using Semantic Relation Triples*** *, In Proceedings of the 8th Text Retrieval Conference (TREC-8).*

## Trang web

1. Digital Bibliography & Library Project

<http://www.informatik.uni-trier.de/~ley/db/>

1. Wiki Question Types

[http://en.wikipedia.org/wiki/Question#Types](http://en.wikipedia.org/wiki/Question)

1. English Club

<http://www.englishclub.com/grammar/verbs-questions_types.htm>

1. Oxford

[http://oxforddictionaries.com/view/entry/m\_en\_gb0680290#m\_en\_gb0680290](http://oxforddictionaries.com/view/entry/m_en_gb0680290)

1. Learn English

<http://www.learnenglish.de/grammar/questiontext.htm>

1. Ontology

<http://www.w3schools.com/RDF/rdf_owl.asp>, <http://www.phpvn.org/index.php?topic=133.0;wap2>

1. SPARQL

<http://en.wikipedia.org/wiki/SPARQL>

1. Named Entity

<http://en.wikipedia.org/wiki/Named_entity_recognition>

1. GATE tool

<http://gate.ac.uk/>

1. NSIR

<http://tangra.si.umich.edu/clair/NSIR/html/nsir.cgi>

1. The START Natuaral Language Question Answering System

[http://start.csail.mit.edu](http://start.csail.mit.edu/)

1. Ask.com

<http://www.ask.com>

1. RDF ([Resource Description Framework](http://www.w3.org/RDF/))

<http://www.xml.com/pub/a/2001/01/24/rdf.html?page=2>

1. Conceptual graph

<http://en.wikipedia.org/wiki/Conceptual_graph>

1. Wordnet

<http://wordnet.princeton.edu/>

1. Pos tagging

<http://en.wikipedia.org/wiki/Part-of-speech_tagging>

1. Question\_answering

<http://www.wordiq.com/definition/Question_answering>

1. <http://en.wikipedia.org/wiki/Question_answering>

# Phụ lục A : Tập 100 câu hỏi kiểm thử

|  |
| --- |
| 1. Who is the author of the book "Question Classification using Head Words and their Hypernyms"? 2. Who is the author of "Working Models for Uncertain Data" 3. Who is the author of the paper "Question Classification using Head Words and their Hypernyms"? 4. Who is the composer of the book "Question Classification using Head Words and their Hypernyms"? 5. Who is the creator of the book "Question Classification using Head Words and their Hypernyms"? 6. Who write books in 1999 ? 7. Who composed books in 2010? 8. Who write books from 1999 to 2010 ? 9. Who published books from 1999 to 2000 ? 10. Who published books in 1999 ? 11. Who released books in 2006 ? 12. Who wrote books for Springer in 2010 ? 13. Who wrote papers for ACM from 2009 to 2010 ? 14. Who wrote "Question Classification using Head Words and their Hypernyms"? 15. Who published "Question Classification using Head Words and their Hypernyms"? 16. Who is the publisher of book "Working Models for Uncertain Data"? 17. Who is the distributor of publication "Working Models for Uncertain Data"? 18. Which books were written by Rafiul Ahad and Amelia Carlson in 2010 ? 19. Which books were written by Rafiul Ahad from 1999 to 2010 ? 20. Which papers were written by Amelia Carlson from 1999? 21. Which books were published by O'Reilly or ACM in 1999 ? 22. Which publications were published by O'Reilly or ACM in 1999 ? 23. Which books were published by O'Reilly in 1999 ? 24. Which books were published by Springer from 1999 to 2000? 25. Which publications were published by O'Reilly or ACM from 2009 to 2010? 26. Which books did Richard L. Muller or James Brown write in ACM ? 27. Which books did Richard L. Muller write in ACM ? 28. Which authors write books for ACM? 29. Which authors wrote books for Springer from 1999? 30. Which authors composed books for O'Reilly ? 31. Which authors composed books in 1999 ? 32. Which authors write books in 2010 ? 33. Which books does Rafiul Ahad wrote with Amelia Carlson? 34. Which publication have source from TREC in 2010? 35. Which papers refer to "Foundations of Databases"? 36. Which publications refer to "Foundations of Databases"? 37. Which documents cited to "Foundations of Databases"? 38. Which books are published in 1999 and written by "Ken Arnold"? 39. Which books are written by "Ken Arnold" and published by IEEE ? 40. Which books are written by "Ken Arnold" and have source from DBPL ? 41. Which papers are composed by "Ken Arnold" and have source from DBPL ? 42. Which publishers released books in 2010 ? 43. Which publishers released book "Data Structures and Algorithms"? 44. Which publishers released book from 2009 to 2010? 45. How many papers were written by Rafiul Ahad ? 46. How many papers were written by Rafiul Ahad in 2010 ? 47. How many publisher did "Philip K. Chan" cooperate with? 48. How many publisher did "Philip K. Chan" work with? 49. How many papers were written by Raul Rato and Rui Gomes in 2010? 50. How many books have source from TREC ? 51. How many papers derived source from TREC ? 52. How many authors wrote book in 2009 ? 53. How many publisher released book in 2010? 54. How many publications refer to "Foundations of Databases"? 55. How many books cited to "Foundations of Databases"? 56. What are titles of books written by Marcus Thint ? 57. What are headings of books written by Marcus Thint ? 58. What are names of books written by William Kent ? 59. What papers did Jennifer Widom write ? 60. What books did Jennifer Widom write ? 61. What book did Philip K. Chan write in 1999 ? 62. What book did Philip K. Chan write from 1999 to 2000? 63. What book did James Brown write in 1999 from ACM? 64. What are the titles of the books published by O’reilly in 1999 ? 65. What composer wrote "Java 2D Graphics" 66. What books has isbn 1-56592-484-3 67. What books has doi 10.1145/360271.360274 68. What composer wrote books from 1999 in ACM? 69. What books were written by "Philip K. Chan" from ACM? 70. What books were published by ACM or Springer in 2010? 71. What publications have resulted from TREC in 1999? 72. What books refer to "Foundations of Databases" 73. What publications have resulted from TREC? 74. What year is "Foundations of Databases" written in? 75. What year is "Java 2D Graphics" written in? 76. What references of the book "Foundations of Databases" ? 77. What citations of the book "Foundations of Databases" ? 78. Does Rafiul Ahad write books in 2010 ? 79. Does Rafiul Ahad write books from 2005 to 2010 ? 80. Did you find me some books publish by O'Reilly ? 81. Do you know any books written in 2010 and publish by ACM ? 82. Did IEEE publish books in 2010 ? 83. Did you have any books were publish by ACM ? 84. Are there any books published by IEEE in 2000 ? 85. Are there any books written by Mike Holcombe in 2000 ? 86. Name some books Richard L. Muller write for Springer 87. Name all publications were publish in 2010 88. Name papers which were published from 1999 to 2001 in ACM ? 89. Name publications which result from DBPL in 2010 ? 90. List some books which are written by Jennifer Widom from 1999 91. List all books were published by Springer in 2010 92. List some books were published by ACM 93. List authors Who wrote books for Springer in 2010 94. Find books which are composed by Rafiul Ahad and Amelia Carlson 95. Find books are composed by Rafiul Ahad and Amelia Carlson 96. Find authors who wrote books for ACM from 2008 to 2009 97. Find publications have resulted from ACM Annual Conference 98. Show me some books were written by Lorand Dali 99. Show me any books which published in ACM and written by Lorand Dali. 100. Alfred V. Aho is author of "Data Structures and Algorithms" , isn't it? |

# 

# Phụ lục B : Danh sách các nhãn từ loại Penn Treebank

(<http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html> )

|  |  |  |
| --- | --- | --- |
| Number | Tag | Description |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.  28.  29.  30.  31.  32.  33.  34.  35.  36. | CC  CD  DT  EX  FW  IN  JJ  JJR  JJS  LS  MD  NN  NNS  NNP  NNPS  PDT  POS  PRP  PRP$  RB  RBR  RBS  RP  SYM  TO  UH  VB  VBD  VBG  VBN  VBP  VBZ  WDT  WP  WP$  WRB | Coordinating conjunction  Cardinal number  Determiner  Existential *there*  Foreign word  Preposition or subordinating conjunction  Adjective  Adjective, comparative  Adjective, superlative  List item marker  Modal  Noun, singular or mass  Noun, plural  Proper noun, singular  Proper noun, plural  Predeterminer  Possessive ending  Personal pronoun  Possessive pronoun  Adverb  Adverb, comparative  Adverb, superlative  Particle  Symbol  *to*  Interjection  Verb, base form  Verb, past tense  Verb, gerund or present participle  Verb, past participle  Verb, non-3rd person singular present  Verb, 3rd person singular present  Wh-determiner  Wh-pronoun  Possessive wh-pronoun  Wh-adverb |

# 

# Phụ lục C : Tập 18 luật nhãn từ loạt dùng để liệt kê bộ ba

|  |
| --- |
| 1. (NN|NNP)-1 VB-2-have NN-3 IN-4 (NNP|CD)-5-NA --> (NN|NNP)-1,VB-2 NN-3 IN-4,(NNP|CD)-5 -->DEL 2. (NN|NNP)-1 VB-2-have NN-3 (NNP|CD)-4-NA --> (NN|NNP)-1,VB-2 NN-3,(NNP|CD)-4 -->DEL 3. (NN|NNP)-1 VB-2 NN-3 NNP-4-NA VB-5 --> NNP-4-NA,VB-5,(NN|NNP)-1 -->DEL 4. NN-1 WDT-2 (NN|NNP)-3-NA VB-4 -->(NN|NNP)-3-NA,VB-4,NN-1 -->DEL 5. NN-1 WDT-2 VB-3 VB-4 IN-5 (NN|NNP|CD)-6-NA --> NN-1,VB-3 VB-4 IN-5,(NN|NNP|CD)-6 -->DEL 6. (NN|NNP)-1 VB-2 (NN|NNP)-3-NA VB-4 --> (NN|NNP)-3-NA,VB-4,(NN|NNP)-1 -->DEL 7. (WP|WDT|NNP|NN)-1 VB-2 VB-3 IN-4 (NN)-5 (NNP|CD)-6-NA --> (WP|WDT|NNP|NN)-1,VB-2 VB-3 IN-4,(NNP|CD)-6 -->DEL 8. (WP|WDT|NNP|NN)-1 VB-2 VB-3 IN-4 (NN|NNP|CD)-5-NA --> (WP|WDT|NNP|NN)-1,VB-2 VB-3 IN-4,(NN|NNP|CD)-5 -->DEL 9. (WP|WDT|NNP|NN)-1 VBN-2 IN-3 (NN|NNP|CD)-4-NA --> (WP|WDT|NNP|NN)-1,VB-2 IN-3,(NN|NNP|CD)-4 -->DEL 10. NN-1 NNP-2-NA VB-3 --> NNP-2-NA,VB-3,NN-1 -->DEL 11. (WP|WDT|NNP|NN)-1 VB-2-result IN-3 NNP-4-NA --> (WP|WDT|NNP|NN)-1,VB-2 IN-3,NNP-4 -->DEL 12. (WP|WDT|NNP|NN)-1 VB-2 (NN)-3 NNP-4-NA --> (WP|WDT|NNP|NN)-1,VB-2,NNP-4 13. (WP|WDT|NNP|NN)-1 VB-2 (NN|NNP)-3-NA --> (WP|WDT|NNP|NN)-1,VB-2,(NN)-3 14. (WP|WDT|NNP|NN)-1 VB-2 IN-3 NN-4 NNP-5-NA --> (WP|WDT|NNP|NN)-1,VB-2,NNP-5 15. (WP|WDT|NNP|NN)-1 VB-2 (IN|TO)-3 (NN|NNP)-4-NA --> (WP|WDT|NNP|NN)-1,VB-2,(NN|NNP)-4 16. (NNP|NN)-1 IN-2 (NN)-3 (NNP|CD)-4-NA --> (WP|WDT|NNP|NN)-1,IN-2,(NNP|CD)-4-NA 17. (NNP|NN)-1 (IN|TO)-2 (NN|NNP|CD)-3-NA --> (WP|WDT|NNP|NN)-1,(IN|TO)-2,(NN|NNP|CD)-3-NA -->DEL 18. (WP)-1 NN-2 VB-3 NNP-4 VBN-5 IN-6 --> NNP-4,VB-5 IN-6,NN-2-->DEL |

# 

# Phụ lục D : Tập tin cấu hình ngữ nghĩa cho cơ sở dữ liệu BDBLP

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <database xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="config.xsd">  <table table-alias="Publication">  <table-name>dblp\_pub\_new</table-name>  <field type="String" field-alias="author" visible="false" relation="related table">  <relation>  <relation-name>have author</relation-name>  <relation-name>have writer</relation-name>  <relation-name>have composer</relation-name>  <relation-name>be write by</relation-name>  <relation-name>be compose by</relation-name>  <related-table>author</related-table>  <mapping-table>pub\_au</mapping-table>  </relation>  </field>    <field type="String" field-alias="title" visible="true" relation="has attribute" presentation="true">  <relation>  <relation-name>have title</relation-name>  <relation-name>have heading</relation-name>  </relation>  <field-name>title</field-name>  </field>  <field type="String" field-alias="publisher" visible="true" relation="has attribute">  <relation>  <relation-name>have publisher</relation-name>  <relation-name>be publish by</relation-name>  <relation-name>be release by</relation-name>  <relation-name>for</relation-name>  <relation-name>from</relation-name>  <relation-name>in</relation-name>  <reversed-relation>publish</reversed-relation>  <reversed-relation>have publish</reversed-relation>  <reversed-relation>release</reversed-relation>  <reversed-relation>have release</reversed-relation>  </relation>  <field-name>publisher</field-name>  </field>    <field type="String" field-alias="source" visible="true" relation="has attribute">  <relation>  <relation-name>have source</relation-name>  <relation-name>have source from</relation-name>  <relation-name>have source in</relation-name>  <relation-name>from</relation-name>  <relation-name>in</relation-name>  <relation-name>result from</relation-name>  </relation>  <field-name>source</field-name>  </field>    <field type="Int" field-alias="year" visible="true" relation="has attribute">  <relation>  <relation-name value="&gt;=">have year from</relation-name>  <relation-name>have year in</relation-name>  <relation-name>in</relation-name>  <relation-name value="&gt;=">from</relation-name>  <relation-name value="&lt;=">to</relation-name>  <relation-name>be write in</relation-name>  </relation>  <field-name>year</field-name>  </field>  <field type="String" field-alias="isbn" visible="true" relation="has attribute">  <relation>  <relation-name>have isbn</relation-name>  </relation>  <field-name>isbn</field-name>    </field>  <field type="String" field-alias="doi" visible="true" relation="has attribute">  <relation>  <relation-name>have doi</relation-name>  </relation>  <field-name>doi</field-name>    </field>    <field type="String" field-alias="reference" visible="false" relation="mapping table">  <relation>    <relation-name>cite</relation-name>  <relation-name>cite to</relation-name>  <relation-name>refer to</relation-name>  <relation-name>refer</relation-name>  <mapping-table>pub\_ref</mapping-table>  <related-table>reference</related-table>  </relation>  </field>  <primary-key>id</primary-key>    </table>  <table table-alias="Author">  <table-name>dblp\_author\_ref\_new</table-name>  <field type="String" field-alias="publication" visible="false" relation="related table">  <relation>  <relation-name>write</relation-name>  <related-table>publication</related-table>  <mapping-table>au\_pub</mapping-table>  </relation>  </field>  <field type="String" field-alias="publisher" visible="false" relation="related table">  <relation>  <relation-name>write for</relation-name>  <relation-name>cooperate</relation-name>  <related-table>publication</related-table>  <mapping-table>au\_pub</mapping-table>  </relation>  <field-name>publisher</field-name>  </field>  <field type="String" field-alias="name" visible="true" relation="has attribute" presentation="true">  <relation>  <relation-name>have name</relation-name>  </relation>  <field-name>author</field-name>  </field>  <primary-key>author</primary-key>  </table>  <table table-alias="Reference">  <table-name>dblp\_pub\_new</table-name>  <field type="String" field-alias="title" visible="true" relation="has attribute" presentation="true">  <relation>  <relation-name>have title</relation-name>  <relation-name>have heading</relation-name>  </relation>  <field-name>title</field-name>  </field>  <primary-key>dblp\_key</primary-key>  </table>  <database-name>dblp</database-name>  <mapping-table name="ref\_pub">  <table-key>dblp\_key</table-key>  <related-table-key>id</related-table-key>  <mapping-table-name>dblp\_ref\_new</mapping-table-name>  </mapping-table>  <mapping-table name="pub\_ref">  <table-key>id</table-key>  <related-table-key>dblp\_key</related-table-key>  <mapping-table-name>dblp\_ref\_new</mapping-table-name>  </mapping-table>  <mapping-table name="pub\_au">  <table-key>pub\_id</table-key>  <related-table-key>author</related-table-key>  <mapping-table-name>dblp\_author\_ref\_new</mapping-table-name>  </mapping-table>  <mapping-table name="au\_pub">  <table-key>author</table-key>  <related-table-key>pub\_id</related-table-key>  <mapping-table-name>dblp\_author\_ref\_new</mapping-table-name>  </mapping-table>  </database> |

PHỤ LỤC E: NHỮNG BỘ BA RÚT ĐƯỢC TỪ TẬP TIN CẤU HÌNH NGỮ NGHĨA

(Publication, have author, Publication.Author)

(Publication, have writer, Publication.Author)

(Publication, have composer, Publication.Author)

(Publication, be write by, Publication.Author)

(Publication, be compose by, Publication.Author)

(Publication, have title, Publication.title)

(Publication, have heading, Publication.title)

(Publication, have publisher, Publication.publisher)

(Publication, be publish by, Publication.publisher)

(Publication, be release by, Publication.publisher)

(Publication, for, Publication.publisher)

(Publication, from, Publication.publisher)

(Publication, in, Publication.publisher)

(Publication, publish, Publication.publisher) (Quan hệ ngược)

(Publication, have publish, Publication.publisher) (Quan hệ ngược)

(Publication, release, Publication.publisher) (Quan hệ ngược)

(Publication, have release, Publication.publisher) (Quan hệ ngược)

(Publication, have source, Publication.source)

(Publication, have source from, Publication.source)

(Publication, have source in, Publication.source)

(Publication, from, Publication.source)

(Publication, in, Publication.source)

(Publication, result from, Publication.source)

(Publication, have year from, Publication.year)

(Publication, have year in, Publication.year)

(Publication, in, Publication.year)

(Publication, from, Publication.year)

(Publication, to, Publication.year)

(Publication, be write in, Publication.year)

(Publication, have isbn, Publication.isbn)

(Publication, have doi, Publication.doi)

(Publication, cite, Publication.Reference)

(Publication, cite to, Publication.Reference)

(Publication, refer to, Publication.Reference)

(Publication, refer, Publication.Reference)

(Author, write, Author.Publication)

(Author, write for, Author.publisher)

(Author, cooperate, Author.publisher)

(Author, have name, Author.name)

(Reference, have title, Reference.title)

(Reference, have heading, Reference.title)

1. <http://www.bing.com/> [↑](#footnote-ref-2)
2. <http://www.yahoo.com/> [↑](#footnote-ref-3)
3. <http://www.google.com/> [↑](#footnote-ref-4)
4. <http://citeseer.ist.psu.edu/> [↑](#footnote-ref-5)
5. <http://www.ieee.org/> [↑](#footnote-ref-6)
6. <http://portal.acm.org/> [↑](#footnote-ref-7)
7. <http://www.sciencedirect.com/> [↑](#footnote-ref-8)
8. <http://www.ssrn.com/> [↑](#footnote-ref-9)
9. <http://papercube.peterbergstrom.com/> [↑](#footnote-ref-10)
10. <http://scholar.google.com/> [↑](#footnote-ref-11)
11. <http://dblp.uni-trier.de/xml/> [↑](#footnote-ref-12)
12. <http://trec.nist.gov/> [↑](#footnote-ref-13)
13. http://sourceforge.net/projects/wn-similarity/ [↑](#footnote-ref-14)
14. <http://www.vocw.edu.vn/> [↑](#footnote-ref-15)
15. <http://www.gutenberg.org/wiki/Main_Page> [↑](#footnote-ref-16)
16. <http://opennlp.sourceforge.net/> [↑](#footnote-ref-17)
17. <http://gate.ac.uk/gate/doc/plugins.html> [↑](#footnote-ref-18)