TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI: NGHIÊN CỨU MÔ HÌNH COKB VÀ XÂY DỰNG HỆ THỐNG GIẢI TOÁN RỜI RẠC

Giảng viên hướng dẫn: TS. NGUYỄN ĐÌNH HIỂN

Sinh viên thực hiện: NGUYỄN PHÚC HOÀI LINH

Lớp : CÔNG NGHỆ THÔNG TIN

Khoá : K58

Tp. Hồ Chí Minh, năm 2021

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI TP. HỒ CHÍ MINH

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI: NGHIÊN CỨU MÔ HÌNH COKB VÀ XÂY DỰNG HỆ THỐNG GIẢI TOÁN RỜI RẠC

Giảng viên hướng dẫn: TS. NGUYỄN ĐÌNH HIỂN

Sinh viên thực hiện: NGUYỄN PHÚC HOÀI LINH

Lớp : CÔNG NGHỆ THÔNG TIN

Khoá : K58

Tp. Hồ Chí Minh, năm 2021

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

-------***-------

Mã sinh viên:5851071042 .... **Họ tên SV:** Nguyễn Phúc Hoài Linh..............
Khóa: K58 .... **Lớp:** Công Nghệ Thông Tin...........................

1. **Tên đề tài:**

Nghiên cứu mô hình COKB và xây dựng hệ thống giải toán rời rạc

2. **Mục đích, yêu cầu:**

a) **Mục đích**

- Xây dựng được một hệ thống website có khả năng giải được các bài toán cấp bậc đại học. Website có hiệu quả cao
- Giúp các sinh viên công nghệ thông tin nói chung và các sinh viên đang học môn Toán rời rạc nói riêng có được một công cụ hỗ trợ việc học

b) **Yêu cầu**

Yêu cầu chức năng:

+ Hệ thống có khả năng được các vấn đề của Toán rời rạc

+ Thành phần lấy đầu vào của hệ thống gần gũi với người dùng

+ Cho phép người dùng xem được kết quả bài toán theo trình tự

Yêu cầu phi chức năng:

+ Tốc độ giải bài tập phải nhanh.

+ Nền tảng triển khai là website, công nghệ sử dụng là Nodejs

+ Website deploy lên Heroku.

3. **Nội dung và phạm vi đề tài:**

a) **Nội dung**

- Khảo sát kiến thức về môn Toán rời rạc
- Mô hình biểu diễn Toán rời rạc
- Thiết kế các thuật giải cho các vấn đề trên miền Tri thức toán rời rạc
- Hệ thống hỗ trợ giải toán thông minh môn Toán rời rạc cấp bậc Đại học

b) **Phạm vi**

- Nghiên cứu mô hình COKB và áp dụng để xây dựng mô hình miền tri thức Toán rời rạc
- Xây dựng website giải toán rời rạc trong ba chương Logic mệnh đề, đại số hàm Boolean và quan hệ hai ngôi

4. **Công nghệ, công cụ và ngôn ngữ lập trình:** Nodejs, Typescript và Javascript. Deploy trên Heroku.
5. **Các kết quả chính dự kiến sẽ đạt được và ứng dụng:** Xây dựng được một website giải được các bài toán trong các chương Logic mệnh đề, đại số Boolean và quan hệ hai ngôi.
6. **Giảng viên và cán bộ hướng dẫn**

   Họ tên: TS. Nguyễn Đình Hiển

   Đơn vị công tác: Trường Đại học Công nghệ thông tin, ĐHQG-HCM

   Điện thoại: 0918735299 Email: hiennd@uit.edu.vn

Ngày tháng 03 năm 2021	**Đã giao nhiệm vụ TKTN**
Trưởng BM Công nghệ Thông tin	**Giảng viên hướng dẫn**

ThS. Trần Phong Nhã

Đã nhận nhiệm vụ TKTN

Sinh viên: Nguyễn Phúc Hoài Linh Ký tên:

Điện thoại: 0335849481 Email: 5851071042@st.utc2.edu.vn

LỜI CẢM ƠN

Trong suốt khoảng thời gian hoàn thành đồ án tốt nghiệp này, em đã gặp không ít khó khăn và thử thách. Nhưng với sự giúp đỡ của các thầy cô, bạn bè và những người xung quanh đã giúp em hoàn thành được đồ án này.

Lời cảm ơn đầu tiên, em xin cảm ơn thầy TS. Nguyễn Đình Hiển, người đã bỏ thời gian và công sức đã chỉ dạy em hoàn thành khóa luận này. Em cũng chân thành cảm ơn, quý thầy cô giảng viên trường Đại học Giao Thông Vận Tải phân hiệu tại TPHCM nói chung và các thầy cô Bộ môn CNTT nói riêng, đã truyền đạt cho em những kiến thức để em có nền tảng như hiện nay.

Chân thành gửi lời cảm ơn đến các bạn trong lớp CNTT K58, những người luôn hỗ trợ hết mình cho các thành viên trong lớp mỗi khi ai đó gặp khó khăn. Luôn luôn chia sẻ các kiến thức để tập thể cùng phát triển.

Cuối cùng em xin cảm ơn những người thân đã luôn tạo điều kiện tốt nhất để em có thể hoàn thành khóa luận này, nhất là trong thời buổi dịch bệnh như ngày nay. Tất nhiên, bài báo cáo sẽ có nhiều thiếu sót, để có thể khắc phục những điều đó, em hi vọng rằng sẽ được sự giúp đỡ và góp ý chân thành từ phía các thầy, các cô.

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Tp. Hồ Chí Minh, ngày ….… tháng ….… năm ….…

Giảng viên hướng dẫn

Nguyễn Đình Hiển

MỤC LỤC

DANH MỤC CHỮ VIẾT TẮT

STT	Mô tả	Ý nghĩa	Ghi chú
1	COKB	Computational Objects Knowledge Base	
2	Expr	Expression	
3	GT	Giả thiết	
4	O	Kết luận	
5	C	Thành phần khái niệm	
6	H	Thành phần quan hệ kế thừa	

BẢNG BIỂU, SƠ ĐỒ, HÌNH VẼ

MỞ ĐẦU

Giới thiệu

Ngày nay, nhu cầu về việc đưa các tri thức của con người lên máy tính, nhằm hỗ trợ cho các ứng dụng trợ giảng hay hệ thống giáo dục thông minh, đang ngày càng được quan tâm. Các tri thức khi này sẽ được tổ chức thành các Hệ cơ sở tri thức trên máy tính.

Khi tiến hành xây dựng một Hệ cơ sở tri thức thì quá trình biểu diễn tri thức đóng một vai trò đặc biệt quan trọng. Một phương pháp biễu diễn tri thức phù hợp với miền tri thức cần biểu diễn sẽ giúp việc định nghĩa hệ cơ sở trở nên dễ dàng, đầy đủ. Đồng thời, điều đó cũng giúp việc xây dựng động cơ suy diễn trơ nên hiệu quả hơn.

Ngày nay, có nhiều phương pháp biểu diễn tri thức. Chẳng hạn biểu diễn bằng mạng ngữ nghĩa, fream-based,… Điểm chung của các phương pháp này là dễ triển khai, thích hợp với một hoặc một số miền tri thức. Tuy nhiên nhược điểm của chúng là ít linh động, không có tính bao quát cao. Mặc khác cũng có một số mô hình được xây dựng chặc chẽ, nhưng yêu cầu phải dựa trên một ngôn ngữ logic nào đó, khác với các ngôn ngữ lập trình phổ biến hiện nay. Do đó, gây khó khăn cho việc cài đặt và triển khai.

Mô hình tri thức các đối tượng tính toán (Computational Objects Knowledge Base - COKB) là một ontology dựa trên phương pháp hướng đối tượng. Trong mô hình này, các miền tri thức được đặc tả bởi một đối tượng từ một class được định nghĩa sẵn. Các đối tượng này có các thuộc tính và hành vi cụ thể. Tùy thuộc vào các miền tri thức phức tạp hay không mà chúng ta sẽ có các khía cạnh khác như quan hệ giữa các tri thức, toán tử, các hàm có trong miền tri thức. Cũng chính nhờ các tiếp cận này mà mô hình COKB có thể được cài đặt trên các ngôn ngữ lập trình có hỗ trợ hướng đối tượng. Từ đó, cũng giúp việc giao tiếp giữa người thiết kế hệ cơ sở tri thức và người triển khai trở nên dễ dàng. Việc này rất thích hợp cho các dự án thực tế. Tuy nhiên, mô hình COKB vẫn còn hạn chế. Đó là việc kết hợp giữa các miền tri thức với nhau còn khó khăn. Đòi hỏi phải có một thuật toán hỗ trợ khi triển khai chúng.

Hệ giải quyết vấn đề (Intelligent Problem Solver – IPS) được sử dụng trong lĩnh vực giáo dục trong việc hỗ trợ dạy học hoặc trong các hệ thống giáo dục trực tuyến. Hệ thống giải bài tập thông minh là một trong số đó. Các hệ thống này có thể đưa ra các lời giải theo từng bước dựa trên đầu vào input. Người dùng chỉ cần nhập đầu vào chính là đề bài cần giải và mục tiêu cần tìm. Hệ thống sẽ cho ra kết quả.

Hiện nay, các ứng dụng giải bài tập này đang được phát triển rất nhanh chóng. Có thể điểm mặt qua một vài hệ thống đang "hot" hiện nay như: Qanda, Photomath, Mathway,… Các ứng dụng trên đều đang có lượt tải và số lượng đánh giá tích cực khủng trên cửa hàng CHPlay, đây chính là dẫn chứng cho thấy các hệ giải bài tập thông minh đang ngày càng được quan tâm.

Để đánh giá các hệ giải bài tập này, có các tiêu chuẩn như sau:

- Hệ thống có thể giải tự động được các bài tập từ cơ bản đến nâng cao.
- Yêu cầu đầu vào phải giống với các bài tập trong các giáo trình, đầu ra phải trình bày từng bước, dễ hiểu, bám sát giáo trình và tương tự cách suy nghĩ của học sinh.
- Hệ thống phải giúp học sinh hiểu và cải thiện khả năng giải các bài toán tương tự. Hệ thống cũng phải bám sát các lý thuyết trong giáo trình của trường học.

Toán rời rạc là một môn học quan trọng đối với các sinh viên công nghệ thông tin. Nó bao gồm các chương logic mệnh đề, hàm Boolean, quan hệ hai ngôi, pương pháp đếm và đồ thị[10]. Các chương Logic mệnh đề, hàm Boolean và quan hệ hai ngôi giúp sinh viên cải thiện được khả năng tuy duy logic, suy luận cho sinh viên. Trong các chương này có các dạng bài tập chính là: rút gọn mệnh đề, chứng minh hai mệnh đề tương đương, suy diễn logic và xác định chân trị của biểu thức, đối với chương Logic mệnh đề. Trong chương đại số Boolean bài tập quan trọng nhất là rút gọn hàm Boolean bằng Karnaugh. Với chương quan hệ hai ngôi có các dạng bài tập xác định các tính chat của quan hệ, chứng minh quan hệ là quan hệ thứ tự/tương đương, xác định các thuộc tính của quan hệ thứ tự/tương đương.

Hiện nay, có rất nhiều các hệ thống hỗ trợ việc giải các bài toán liên quan đến Toán rời rạc, tuy nhiên phần lớn chúng không được giúp được việc hỗ trợ cho dạy học. Chúng thường đưa ra kết quả nhưng không đưa ra các lời giải cho bài tập đó.

Rút kinh nghiệm trên, bằng việc kết hợp với mô hình COKB, em sẽ tiến hành biểu diễn miền tri thức Toán rời rạc, bao gồm ba chương Logic mệnh đề, Đại số hàm Boolean và Quan hệ hai ngôi. Song song đó, em sẽ tiến hành xây dựng các thuật giải nhằm giải quyết các bài toán cụ thể. Chương Logic mệnh đề bao gồm rút gọn biểu thức mệnh đề, chứng minh hai mệnh đề tương đương, suy diễn logic và xác định chân trị biểu thức. Chương đại số Boolean: rút gọn biểu thức mệnh đề bằng bìa Karnaugh. Quan hệ hai ngôi: xác định các tính chất quan hệ, chứng minh quan hệ là quan hệ thứ tự/tương đương, xác định các thuộc tính của quan hệ thứ tự/tương đương.

Hệ thống được xây dựng phải thõa mãn được các tiêu chuẩn đã nêu ra bao gồm. Có khả tự giải được các bài tập. Gần gũi, dễ hiểu và bám sát lý thuyết. Giúp học sinh có thể hiểu, củng cố và giải được các bài tập nâng cao.

Mục tiêu nghiên cứu

- Khảo sát lý thuyết về miền tri thức toán rời rạc.
- Tìm hiểu về mô hình COKB và áp dụng để xây dựng mô hình biểu diễn miền tri thức toán rời rạc
- Xây dựng thuật toán để giải các bài tập trong các chương Logic mệnh đề, đại số hàm Boolean và quan hệ hai ngôi.
- Tiến hành xây dựng hệ thống giải bài tập là một website.
- Khảo sát và đánh giá độ hiệu quả của hệ thống website.

Đối tượng nghiên cứu

Bài nghiên cứu tập trung nghiên cứu các đối tượng sao:

Mô hình Ops-Funcs, đây là một mô hình phần cốt lõi là tri thức về các toán tử và tích hợp với đó là các phương thức xử lý nó.

Các cơ sở lý thuyết của môn toán rời rạc, tập trung ở ba chương Logic mệnh đề, đại số hàm Boolean và quan hệ hai ngôi.

Các thuật toán nhằm giải quyết các bài toán trong mô toán rời rạc.

Các ngôn ngữ và công nghệ sau:

+ Javascript chuẩn es6.

+ TypeScript

+ Nodejs và freamwork Express.

+ Github và Heroku.

Phạm vi nghiên cứu

Cơ sở lý thuyết ở bài nghiên cứu này nằm trong ba chương đó là Logic mệnh đề, đại số Boolean và quan hệ hai ngôi.

Đối với chương quan hệ hai ngôi, do giới hạn về thời gian và trình độ của tác giả do đó, bài nghiên cứu chỉ nằm trong phạm vi số nguyên. Trong đó các toán tử được hỗ trợ là +,-,*/. Các quan hệ đại số cũng chỉ hỗ trợ một vài quan hệ như: chia hết, bội số, ước số, số chẵn, số lẻ, các phép so sánh =,>,<,>=,<=.

Hệ thống xây dựng trên thành một website có giao diện đơn giản, dễ hiểu.

Website được deploy trên heroku để thuận tiện việc truy cập.

Ý nghĩa và hạn chế

Mục đích

Xây dựng được một hệ thống website có khả năng giải được các bài toán cấp bậc đại học. Website có hiệu quả cao

Giúp các sinh viên công nghệ thông tin nói chung và các sinh viên đang học môn Toán rời rạc nói riêng có được một công cụ hỗ trợ việc học

Hạn chế

Do thời gian có hạn, mặc khác trình độ của tác giả nên hệ thống có nhiều sai xót là điều không thể tránh khỏi.

Hệ thống chỉ dừng lại ở mức ba chương trong môn học, chưa được mở rộng thêm.

Trên thực tế nhu cầu thực tế của sinh viên là cần một công cụ vừa tra cứu kiến thức vừa hướng dẫn giải bài tập. Hệ thống cần thêm chức năng ấy.

Cấu trúc báo cáo:

Chương 1: Giới thiệu về đề tài

Chương 2: Khảo sát kiến thức về môn Toán rời rạc

Chương 3: Mô hình biểu diễn Toán rời rạc

Chương 4: Thiết kế các thuật giải cho các vấn đề trên miền Tri thức toán rời rạc

Chương 5: Hệ thống hỗ trợ giải toán thông minh môn Toán rời rạc cấp bậc Đại học

Chương 6: Tổng kết

CHƯƠNG 1. KHẢO SÁT MIỀN TRI THỨC TOÁN RỜI RẠC

1.1. Lý thuyết Logic mệnh đề và đại số Boolean

A. Logic mệnh đề

Logic mệnh đề là một nhánh của ngành toán logic. Trong đó đối tượng nghiên cứu chính là các mệnh đề.

Mệnh đề là một câu phát biểu có chân trị đúng hoặc sai. Một mệnh đề có chân trị là 1 được xác định là một mệnh đề đúng, ngược lại mệnh đề có chân trị là 0 được gọi là mệnh đề sai. Không có mệnh đề nào có chân trị vừa đúng hoặc sai.

VD: "Hà Nội là thủ đô của Việt Nam"- là một mệnh đề với chân trị là đúng.

"Hôm nay trời lạnh quá!!" - không phải là một mệnh đề vì mang nhiều cảm tính, tùy thuộc vào mỗi người, do đó ta không thể xác định tính chất đúng hay sai cua câu nói này.

Ký hiệu: người ta thường dùng các chữ cái a,b,c,… làm ký hiệu cho mệnh đề. Chẳng hạn a= "Hà Nội là thủ đô của Việt Nam". Hoặc có thể ghi tắc là *a*.

Chú ý:

- Các mệnh đề có chân trị đúng sai tùy thuộc vào khoảng thời gian cụ thể. Có thể trong khoảng thời gian này, mệnh đề có chân trị là đúng nhưng trong khoảng thời gian khác nó mang chân trị là sai.
- VD: "Đội tuyển Việt Nam vừa giành chiến thắng trước đối thủ của mình". Mệnh đề này sẽ có chân trị đúng hoặc sai dựa vào kết quả từng trận đấu của đội tuyển Việt Nam.
- Có một số mệnh đề chưa xác định được chân trị tai hiện tại. Nhưng chắc chắn nó có giá trị đúng hoặc sai. VD: "Đội tuyển Việt Nam vượt qua vòng loại thứ 3 và tham dự WC 2022."

*** Các toán tử trong logic mệnh đề:** Cũng tương tự các lĩnh vực toán học khác, Logic mệnh đề cũng có toán tử tác động vào các mệnh đề. Các toán tử bao gồm: nhóm toán tử một ngôi và nhóm toán tử hai ngôi. Nhóm toán tử một ngôi chỉ gồm toán tử phủ định (¬). Nhóm toán tử hai ngôi bào gồm: ∧, ∨,→,↔. Kết quả của các phép toán được thể hiện đầy đủ tại bảng dưới đây.

p	q	$\neg p$	p∨q	p∧q	p→q	p↔q
1	1	0	1	1	1	1
1	0	0	1	0	0	0
0	1	1	1	0	1	0
0	0	1	0	0	1	1

Bảng 1.1 Bảng toán tử

***Phân loại:** Dựa vào cấu trúc của mệnh đề, cho một mệnh đề p cho trước ta có nhận xét sau:

- Nếu p là mệnh đề có chân trị luôn luôn là True (đúng) hoặc False (sai) thì p được gọi là hằng.
- Nếu p là một mệnh đề đơn. p được gọi là biến mệnh đề.
- p là kết quả của các phép toán logic với các biến mệnh đề khác thì p được gọi là mệnh đề phức tạp hay biểu thức mệnh đề.

*** Sự tương đương mệnh đề:**

Định nghĩa 2.1: Cho P và Q là hai mệnh đề. Ta nói rằng hai mệnh đề P, Q tương đương logic với nhau, nếu với mọi hệ chân lý gán cho các biến mệnh đề có mặt trong hai công thức đó thì chúng luôn nhận giá trị chân lý như nhau.

Ký hiệu: $P \equiv Q$.

Biểu thức $P \equiv Q$ được xác định thông quan một loạt các biểu thức tương đương khác. Các biểu thức này đã được chứng minh từ trước đó.

* **Các luật trong logic mệnh đề**: Trong logic mệnh đề tồn tai các luật bao gồm các luật tương đương logic và luật suy diễn logic.

Name	Rule	
Disjuntive	$p \vee q, \neg p \Rightarrow q$	
Conjunction	$p, q \Rightarrow p \wedge q$	$p \wedge q \Rightarrow p, q$
Modus Ponens	$(p \rightarrow q), p \Rightarrow q$	
Modus Tollens	$(p \rightarrow q), \neg q \Rightarrow \neg p$	
Hypothetical	$(p \rightarrow q), (q \rightarrow r) \Rightarrow (p \rightarrow r)$	
Resolution	$p \vee q, \neg p \vee r \Rightarrow q \vee r$	

Hình 1.1 Luật suy diễn

Mệnh đề tương đương		Tên gọi
$p \wedge True \equiv p$	$p \vee False \equiv p$	Luật đồng nhất
$p \vee True \equiv True$	$p \wedge False \equiv False$	Luật nuốt
$p \vee p \equiv p$	$p \wedge p \equiv p$	Luật lũy đẳng
$\neg(\neg p) \equiv p$		Luật phủ định kép
$p \vee q \equiv q \vee p$	$p \wedge q \equiv p \wedge q$	Luật giao hoán
$(p \vee q) \vee r \equiv p \vee (q \vee r)$	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Luật kết hợp
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$		Luật phân phối
$\neg(p \vee q) \equiv \neg p \wedge \neg q$	$\neg(p \wedge q) \equiv \neg p \vee \neg q$	Luật De Morgan
$p \wedge \neg p \equiv False$	$p \vee \neg p \equiv True$	Luật phần tử bù
$p \vee (p \wedge q) \equiv p$	$p \wedge (p \vee q) \equiv p$	Luật hấp thụ
$p \rightarrow q \equiv \neg p \vee q$		Luật phép kéo theo
$p \leftrightarrow q \equiv (p \rightarrow q) \vee (q \rightarrow p)$		Luật phép tương đương

Hình 1.2 Luật tương đương

B. Lý thuyết về đại số Boolean

Tương tự Logic mệnh đề, Hàm Boolean xoay quanh các biến mang hai giá trị 0 và 1. Tuy nhiên, điểm khác biệt giữa hai miền tri thức này đó là Logic mệnh đề dùng để biểu diễn các lý luận, suy diễn, trong khi Hàm Boolean thì không. Do đó, tại miền tri thức

Hàm Boolean chỉ chứa các phép toán “+”, “.” và “¬” mà không chứa các dấu “→” hay “↔”. Vì vậy các quy tắc rút gọn của Hàm Boolean không có hai luật về phép tương đương và phép kéo theo. Cũng như, không có các quy tắc suy diễn logic như Logic mệnh đề.

Như vậy, hàm Boolean tập trung vào việc biểu diễn chân trị của một hàm dựa vào các trường hợp các biến mệnh đề. Do đó nó được ứng dụng nhiều trong ngành điện-điện tử, biểu diễn các mạnh điện,…

1.2. Lý thuyết quan hệ hai ngôi.

1.2.1. Tập hợp

Định nghĩa 2.2: Trong toán học, tập hợp là sự tụ tập của một dãy số hữu hạn hay vô hạn các đối tượng nào đó. Những đối tượng này ta gọi là phần tử của tập hợp. Một tập hợp có thể có nhiều phần tử hoặc không có phần tử nào. Ký hiệu bằng một ký tự viết hoa.

***Biểu diễn một tập hợp:** để biểu diễn một tập hợp ta có nhiều cách để thực hiện nhưng ta sẽ quan tâm đến các cách sau:

- Phương pháp liệt kê. Nếu số lượng tập hợp là hữu hạn.

  VD: A = {1;2;3;4;5;6;7;8}.

- Phương pháp nêu đặc tả. Phương pháp này thường dùng để đánh dấu một tập hợp mà ta không thể xác định được số lượng phần tử hoặc số lượng phần tử quá lớn, như các tập số nguyên, số tự nhiên, số thực,… đồng thời các phần tử có chung một hoặc một số đặc điểm nào đó .
- Phương pháp được hiểu như sau, cho một tập hợp không gian K cho trước. Biểu diễn tập hợp P ∈ K, sao cho các phần tử Pi của P thoản mãn mọi điều kiện của tập Conditions. Ví dụ:

$$Số\ chẵn: P = SO\_CHAN = \{x \in N \mid x \vdots 2\}$$

- Tập hợp không có phần tử nào được gọi là tập rỗng. Ký hiệu: {} hay ∅.

1.2.2. Quan hệ

Định nghĩa 2.3: Một quan hệ hai ngôi từ tập A đến tập B là tập con của tích đề các R=A x B. Chúng ta sẽ viết a R b thay cho $(a, b) \in R$.

Từ định nghĩa 2.1.2 ta có các nhận xét sau về một quan hệ R:

- Một quan hệ R là một tập hợp. Do đó nó hoàn toàn có thể được biểu diễn như một tập hợp bình thường.
- Nếu quan hệ R từ chính tập A, thì quan hệ trên được gọi là quan hệ trên một tập A.
- Với các tập hợp A hữu hạn {a0,a1,…, aN}, ta có thể ánh xạ các phần tử (ai, aj) từ ai, aj ∈ A lên một ma trận M NxN. Khi đó M[ij] = 1 khi (ai, aj) ∈ R và ngược lại M[ij] = 0.

1.2.3. Các tính chất của quan hệ

Một quan hệ sẽ có các tính cơ bản như sau.

Tính phản xạ: Xét quan hệ R và tập hợp A. Quan hệ R có tính chất *phản xạ* nếu:

$$\forall\, a \in A\,,\ a\,R\,a$$

Tính đối xứng: Xét quan hệ R và tập hợp A. Quan hệ R có tính chất *đối xứng* nếu:

$$\forall\, a,b \in A,\ \ (a\,R\,b) \rightarrow (b\,R\,a)$$

Tính phản xứng: Xét quan hệ R và tập hợp A. Quan hệ R có tính chất *phản xứng*:

$$\forall\, a,b \in A,\ \ (a\,R\,b) \wedge (b\,R\,a) \rightarrow (a=b)$$

Tính bắc cầu: Xét quan hệ R và tập hợp A. Quan hệ R có tính chất *bắc cầu* nếu:

$$\forall\, a,b,c \in A,\ \ (a\,R\,b) \wedge (\,b\,R\,c) \rightarrow (a\,R\,c)$$

1.2.4. Phân loại quan hệ hai ngôi

Một quan hệ hai ngôi có thể được phân ra làm hai loại bao gồm: quan hệ tương đương và quan hệ thứ tự.

*** Quan hệ tương đương**

Định nghĩa 2.4: Quan hệ tương đương: Cho quan hệ R trên A, R được gọi là tương đương nếu R có các tính chất: phản xạ, đối xứng và bắc cầu.

VD: Quan hệ R trên tập số nguyên sao cho với hai số nguyên a,b bất kỳ ta luôn có a+b là số chẵn. Là một quan hệ tương đương vì chứa đủ ba tính chat phản xạ, đối xứng và bắc cầu.

Lớp tương đương: Cho R là quan hệ tương đương trên A và phần tử $a \in A$. Lớp tương ương chứa a được ký hiệu bởi $[a]_R$ hoặc [a] là tập:

$$[a]_R = \{b \in A|\ b\ R\ a\}$$

Dựa theo định nghĩa lớp tương đương ta có được định lý như sau, cho R là quan hệ tương đương trên tập A và $a,b \in A$, Khi đó

1. a R b nếu $[a]_R = [b]_R$.
2. $[a]_R \neq [b]_R$ nếu $[a]_R \cap [b]_R = \emptyset$.

Như vậy ta nhận thấy tập A có thể phân hoạch thành các lớp tương đương.

**** Quan hệ thứ tự:***

Định nghĩa 2.5 : Quan hệ R trên tập A là quan hệ thứ tự (thứ tự) nếu nó có tính chất phản xạ, phản xứng và bắc cầu. Ký hiệu "$\prec$".

Tính chất:

1. Với $x,y \in A$ và $x \prec y$ thì khi đó ta nói y là trội của x (x bị trội bởi y)
2. Trường hợp y là trội trực tiếp của x nếu y là trội của x và không tồn tại một trội z nào sao cho z bị trội bơi y và z là trội của z.
3. Xét phần tử $x \in A$:

- x gọi là phần tử tối đại, nếu x không bị trội bởi bât kỳ phần tử nào khác trong A.
- x được gọi là phần tử tối tiểu nếu x là trội của bất kỳ phàn tử trong A.
- x được gọi là giá trị lớn nhât nếu x là trội của mọi phần tử trong A
- x được gọi là giá trị nhỏ nhất nếu x bị trội bởi mọi phần tử trong A

CHƯƠNG 2. MÔ HÌNH BIỂU DIỄN TOÁN RỜI RẠC

2.1. Mô hình COKB

Như đã đề cập ở phần trước đó. Mô hình COKB được xây dựng dựa trên phương pháp hướng đối tượng. Do đó, các thành phần của mô hình sẽ được biểu diễn bởi các đối tượng. Các thành phần này bao gồm:

K = (C, H, R, Ops, Funcs, Rules)

Trong đó:

+ C là một tập hợp các khái niệm về các C-Object
+ H là một tập hợp các quan hệ phân cấp giữa các loại đối tượng.
+ R là tập hợp các khái niệm về các loại quan hệ trên các C-Object.
+ Ops là một tập hợp các toán tử.
+ Funcs là một tập hợp các hàm.
+ Rules là tập hợp các luật.

2.2. Xây dựng mô hình biểu diễn miền tri thức Toán rời rạc

Định nghĩa 3.1: Mô hình được em đề cập trong báo cáo này được gọi là *Ops-Funcs*, đây là mô hình được xây dựng dựa trên mô hình COKB. Nó được cấu tạo bởi:

K = (C,Ops,Rules)+ Funcs

A. Thành phần (C,Ops,Rules)

(C,Ops,Rules) là thành phần cốt lõi. Nhằm biểu diễn miền tri thức toán rời rạc. Nó bao gồm.

Thành phần C là tập các khái niệm. Với mỗi một đối tượng c ∈ C, có cấu trúc : *(Attr, m_Funcs)*. Trong đó Attr là tập thuộc tính của các khái niệm, m_Funcs là các hàm chức năng được dùng trong nội tại của đối đó. *Attr* của đối tượng *c* sẽ được đề cập tại các phần sau.

Các khái niệm trong miền tri thức Toán rời rạc trong ba chương Logic mệnh đề, đại số hàm Boolean và quan hệ 2 ngôi. Gồm các khái niệm sau:

Tên khái niệm	**Miền tri thức được khảo sát**
Biểu thức mệnh đề	Logic mệnh đề, đại số Boolean
Số nguyên	Quan hệ 2 ngôi
Biểu thức số nguyên	Quan hệ 2 ngôi
Tập hợp	Quan hệ 2 ngôi
Quan hệ 2 ngôi	Quan hệ 2 ngôi
Quan hệ tương đương	Quan hệ 2 ngôi
Quan hệ thứ tự	Quan hệ 2 ngôi

Bảng 2.1 Bảng các khái niệm trong miền tri thức

Thành phần Ops, là tập toán tử. Trong toán rời rạc, thì các chương Logic mệnh đề , đại số hàm Boolean có các toán tử logic $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$. Riêng chương quan hệ hai ngôi. Như đã đề cập nó chỉ diễn ra trong không gian số nguyên, do đó các toán tử trong miền tri thức này chính là các phép toán số học thông thường,+,-,*/. Nên ta không cần biểu diễn chúng.

Thành phần Rules, đây là thành phần biểu diễn các luật trong miền tri thức toán rời rạc. Chúng bao gồm các luật sau: luật mệnh đề tương đương, luật suy diễn logic và luật xác định tính chất quan hệ 2 ngôi.

$$\text{Rule} = R_{TuongDuong} \cup R_{SuyDien} \cup R_{QuanHe2Ngoi}$$

*** Luật mệnh đề tương đương $R_{TuongDuong}$**: có dạng: $p \equiv q$, trong đó p và q là hai biểu thức mệnh đề. Chẳng hạn:

- Tam đoạn phủ định kép: $\neg(\neg p) \equiv p$
    - Luật phần tử bù: $p \wedge \neg p \equiv False$ hoặc $p \vee \neg p \equiv True$

*** Luật suy diễn logic $R_{SuyDien}$ có dạng:** $A[a_1,a_2,\ldots,a_n] \rightarrow B[b_1,b_2,\ldots,b_n]$, với A,B là các tập biểu thức:

- Tam đoạn luận loại trừ: $[p \vee q, \neg q] \rightarrow [p]$

- Luật kết hợp: $[p, q] \rightarrow [p \wedge q]$

* **Luật về tính chất quan hệ 2 ngôi:** Luật này nhằm xác định một quan hệ 2 ngôi có thõa tính chất nào không. Do quan hệ có hai dạng biểu diễn nên ta có 2 cách thức xác định khác nhau.

Quan hệ có thể liệt kê được hữu hạn phần tử

Gọi M là ma trận được ánh xạ tử quan hệ R và R có tập mẫu là A. Khi đó các tính chất của R được xác định bằng.

- **Tính phản xạ**: $M_{ii} = 1$, vơi mọi i ∈ A.
- **Tính đối xứng**: Nếu $M_{ij} = 1$ thì $M_{ji} = 1$, vơi mọi i,j ∈ A.
- **Tính phản xứng**: Tính phản xạ phải hợp lệ và không tồn tại i,j ∈ A sao cho $M_{ij} = 1$ thì $M_{ji} = 1$
- **Tính bắc cầu**: Nếu $M_{xy} = 1$ và M_{yz} thì $M_{xz} = 1$, vơi mọi x,y,z ∈ A.

Quan hệ có cấu trúc đặc tả

Cho quan hệ 2 ngôi R và có *Quan hệ hai số nguyên a và b* là một vị từ P("lớn hơn", "bé hơn", "bằng", "chia hết", "số chẵn", "ước số",…). Gọi *SuyDien(A,B)* trong đó A,B là các vị từ *Quan hệ hai số nguyên,* để kiểm chứng quá trình suy diễn A→B là đúng hay sai.

- **Tính phản xạ**: Nếu P(a,a) đúng thì R có tính phản xạ.
- **Tính đối xứng:** Cho P1(a,b) và P2(b,a), giả sử P1 đúng thì nếu SuyDien([P1],P2) đúng thì R có tính đối xứng.
- **Tính phản xạ:** Nếu P(a,a) đúng và **Tính đối xứng** là sai thì R có quan hệ phản xạ.
- **Tính bắc cầu**: Cho P1(a,b), P2(b,c) và P3(a,c). Giả sử P1, P2 đúng thì SuyDien([P1,P2],P3) đúng thì R có tính chất bắc cầu.

B. Thành phần Funcs

Thành phần này bao gồm các hàm được sử dụng trong hệ thống:

Checker(P,R). Đây là hàm dùng để kiểm tra biểu thức *P* có áp dụng cho luật *R* hay không. Trả về một biểu thức, nếu không áp dụng được trả về NULL. Cấu trúc cụ thể được đề cập mục sau.

Replace(p,q,r): hàm tiến hành thay thế biểu thức p bằng q trong biểu thức gốc r.

Replace($A[a_1, a_{1,\dots,} a_n],B[b_1, b_{2,\dots,} b_n],p$): hàm thay thế một tập các biểu thức con *A* bằng một tập các biểu thức con *B* trong biểu thức gốc *p*.

Copy(p) : tạo ra một bản sao của biểu thức p.

Length(p): hàm lấy độ phức tap của một biểu thức. Nhằm so sánh trong quá trình đơn giản hóa.

Contain(p,q): kiểm tra biểu thức p có chứa biểu thức q hay không.

Not(p): phủ định một biểu thức.

Simplify(p): hàm rút gọn *p*, trả về *g* và các bước chuyển đổi *cd*

Equivalent (p,q): chứng minh hai mệnh đề tương đương, nếu có trả về tập ChuyenDoi [], nếu không trả về [].

Karnaugh (f): Ánh xạ một hàm Boolean lên biểu đồ Karnaugh.

BigCell(Kar): Trả về danh sách các tế bào lớn trong biểu đồ Karnaugh.

SimplestBoolean(bigCell): Trả về các đa thức tối giản từ một biểu đồ Karnaugh.

PredicateNumber(a,b): định nghĩa về quan hệ 2 số nguyên.

Hasse(): Tạo biểu đồ Hasse

2.3. Mô hình biểu diễn Logic mệnh đề và đại số Boolean

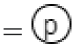
2.3.1. Logic mệnh đề

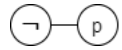
Định nghĩa về một biểu thức mệnh đề:

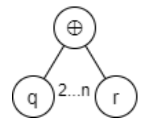
1. Các giá trị 1 (True) hoặc 0 (False) là một biểu thức mệnh đề, và chúng được gọi là một biểu thức hằng
2. Một biến p, chỉ có hai giá trị là 0 hoặc 1 là một là một biểu thức mệnh đề. Nó được gọi là một biến mệnh đề.
3. Nếu p là một biểu thức mệnh đề thì khi đó $\neg p$ cũng là một biểu thức
4. Với p và q là các biểu thức mệnh đề thì p $\oplus$ q là một biểu thức mệnh đề (với $\oplus$ là các phép toán $\wedge$, $\vee$,$\rightarrow$,$\leftrightarrow$).

Biểu diễn cấu trúc biểu thức mệnh đề:

Cho một biểu thức p. Khi đó cây T(p) biểu diễn p như sau:

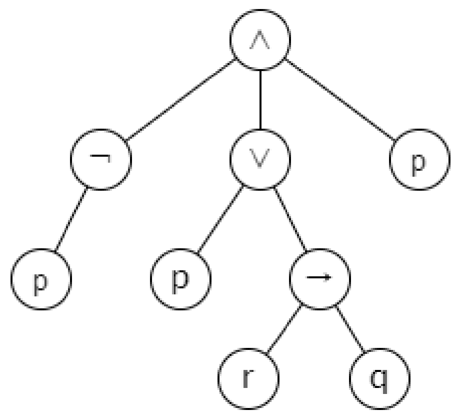
1. Nếu p là một hằng hoặc p là một biến mệnh đề thì T(p) = 

2. Nếu p = ¬q thì T(p) = 

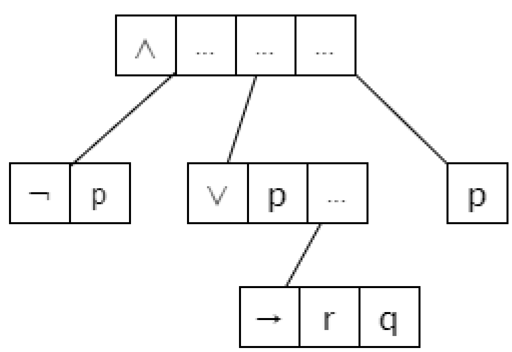
3. Nếu p = q ⊕ … ⊕ r thì T(p) = 

4. Nếu p là biểu thức con của q thì T(p) là nhánh con của T(q)

Ví dụ biểu thức $A = \neg p \wedge (p \vee (r \rightarrow q)) \wedge p$ được biểu diễn như sau:



Hoặc ta có thể viết gọn thành:



Lúc này khi duyệt cây trên, ta hoàn toàn có thể biểu thức trên lại thành dạng hậu tố:

$$A = (\wedge,(\neg,p),(\vee,p,(\rightarrow,r,q)),p) = (\wedge,B,C,p).$$

Trong đó: B=(¬,p), D=(→,r,q), C=(∨,p,D)

Tuy nhiên cách thức này sẽ phát sinh ra một nhước điểm nhỏ đó là việc biểu diễn sẽ gom các mệnh đề có cùng toán tử lại với nhau. Do đó, khi biểu diễn các mệnh đề có nhiều toán tử ta cần phải nhóm các mệnh đề lại bằng cặp dấu "( )", nhằm tạo ra các biểu thức con. Chẳng hạn, ta không nên đưa một mệnh đề như thế này: $P = a \vee b \wedge c$,mà hãy nhóm nó lại thành $P = a \vee (b \wedge c)$ hoặc $P = (a \vee b) \wedge c$. Điều này cũng hoàn toàn hợp với ngôn ngữ giao tiếp thường ngày của chúng ta.

Thành phần C được xây dựng dựa trên cấu trúc trên. Cụ thể, thành phần C biễu diễn một biểu thức mệnh đề có dạng như sau:

C = (Attr, M_Funcs)

Trong đó, *M_Funcs* là các hàm hành vi của một biểu thức mệnh đề. Bao gồm các phương thức quan trọng đó là ***getTruthValue()*** tính toán chân trị của một mệnh đề. Hàm ***createID()*** xây dựng chuỗi hậu tố của của biểu thức đó. Thành phần Attr của sẽ được cấu tạo bao gồm

- Truth_Value: một biến Boolean, thể hiện chân trị một biến mệnh đề hay hằng
- Id: chuỗi hậu tố của biểu thức
- O: là toán tử logic của một biểu thức
- Childs: là tập các biểu thức con cảu biểu thức đang xét.
- Parent : là biểu thức cha của biểu thức đang xét
- Primes: là tập các biến mệnh đề cấu tạo nên biểu thức.

2.3.2. Mô hình cho miền tri thức đại số Boolean

Định nghĩa đại số Boolean:

1. Các hằng 0 và 1 là một Boolean
2. Các biến a,b,c,…. Cũng là một Boolean
3. Nếu p và q là các biểu thức thì p.q , p+q và ¬p cũng là Boolean.

Như vậy ta thấy rằng, đại số Boolean khá tương đồng với Logic mệnh đề.

Định nghĩa Hàm Boolean:

Hàm Boolean là một ánh xạ sao cho:

$$f{:}B^n \rightarrow B, \text{ trong đó } B = \{0,1\}$$

Vậy hàm Boolean là một hàm số có dạng: $f(x_1,x_2,...,x_n)$, với $x_1,x_2,...,x_n$ là các biến chỉ nhận hai giá trị 0 và 1, f cũng nhận giá trị 0 và 1.

Ta ký hiệu F_n để đại diện cho một hàm Boolean có n biến.

Lưu ý: Cũng vì Hàm Boolean là một hàm n biến, nên ta có thể dễ dàng biểu diễn chúng qua một đa thức. Có nhiều cách để biểu diễn một hàm Boolean, tuy nhiên để theo chuẩn của các giáo trình giảng dạy trong các trường đại học, bài nghiên cứu này chỉ xét việc biểu diễn hàm Boolean thông qua *dạng nối rời chính tắc* của nó.

Định nghĩa về dạng nối rời chính tắc:

Xét một hàm Boolean F_n có n biến:

1. Đối các biến x_i và phủ định của nó là ¬x được gọi là một *từ đơn*
2. *Đơn thức* là tích khác không của một số hữu hạn các *từ đơn*.
3. *Từ tối tiểu* là một đơn thức nhưng có đầy đủ n *từ đơn*
4. *Dạng nối rời chính tắc* là công thức biểu diễn F_n thành tổng của các *từ tối thiểu*.

Sau khi trình bày về cấu trúc của một hàm Boolean ta sẽ tiến hành xây dựng mô hình cho nó.

Trong đại số hàm Boolean. Việc thao tác với biểu đồ Karnaugh là hết sức quan trọng. Bao gồm một số thao tác quan trọng như sau. *Ánh xạ hàm Boolean lên một biểu đồ Karnaugh* và *xác định các tê bào lớn của Karnaugh*. Các công đoạn trên được định nghĩa trong các hàm ***Karnaugh(f)*** và ***BigCell(Kar).***

*** Ánh xạ hàm Boolean lên biểu đồ Karnaugh:**

Để có thể ánh xạ một đa thức chuẩn tắc lên một bìa Karnaugh cần các quy tắc cụ thể. Bìa Karnaugh là tập hợp một hoặc nhiều ma trận với một số thứ tự cụ thể.

-Với hàm Boolean 3 biến ta có bìa Karnaugh như sau:

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$	0	2	6	4
C	1	3	7	5

Bảng 2.2 Bảng Karnaugh 3 biến

-Với hàm Boolean 4 biến ta có bìa Karnaugh như sau:

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{CD}$	0	4	12	8
$\overline{C}$D	1	5	13	9
CD	3	7	15	11
$C\overline{D}$	2	6	14	10

Bảng 2.3 Karnaugh 4 biến

-Với hàm Boolean 5 biến ta có bìa Karnaugh như sau:

	$\overline{BC}$	$\overline{B}C$	BC	$B\overline{C}$		$\overline{BC}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{DE}$	0	4	12	8		16	20	28	24
$\overline{D}E$	1	5	13	9		17	21	29	25
DE	3	7	15	11		19	23	31	27
$D\overline{E}$	2	6	14	10		18	22	30	26
$\overline{A}$						A			

Bảng 2.4 Bìa Karnaugh 5 biến

Để có thể ánh xạ các đa thức chính tắc, ta cần phải xem số lượng các từ đơn trong hàm Boolean. Sau đó xét từng đơn thức để ánh xạ chúng lên bìa.

VD: Giả sử cho một hàm Boolean 4 biến f, có đơn thức abc$\overline{d}$. Ta có thể ánh xạ nó lên bìa Kar(f):

$$\text{abc}\overline{d} \xrightarrow[nhị\ phân]{} 1110 \xrightarrow[thập\ phân]{} = 14.$$

Vậy abc$\overline{d}$ nằm ở ô có đánh số 14 ở Kar(f).

***Xác định tế bào lớn của bìa Karnaugh.**

Định nghĩa 3.3: Cho một hàm Boolean f với n biến, Kar(f) là bìa Karnaugh ánh xạ từ hàm f, ta có các định nghĩa sau:

- Tế bào T là một hình chữ nhật gồm 2^n ô và T ∈ Kar(f).
- Tế bào T là một tế bào lớn nếu T thõa hai điều kiện:

  + T là một tế bào và T ∈ Kar(f).

  + Không tồn tại T’ nào sao cho T’ ≠ T và T ∈ T’ ∈ Kar(f).

- Từ các định nghĩa trên ta có nhận xét:
- Một từ tối thiểu thuộc hàm f, thì khi ánh xạ sẽ là một tế bào T của bìa Kar(f).
- Để nhóm các tế bào thành một tế bào lớn hơn, các tế bào cần có các từ đơn giống nhau.

2.4. Mô hình biểu diễn quan hệ hai ngôi

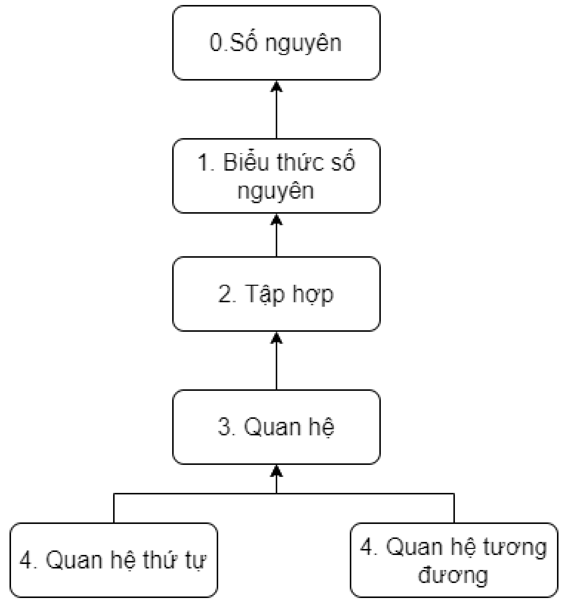
Trong bài nghiên cứu này, các quan hệ hai ngôi mà chúng ta xem xét được nằm gói gọn trong tập không gian số nguyên Z. Tức là cấu trúc của các tập hợp hay quan hệ đều là các số nguyên.

Do đó, ta sẽ có được các tập khái niệm cụ thể như sau:

Tên khái niệm	**Cấp độ**
Số nguyên	0
Biểu thức số nguyên	1
Tập hợp	2
Quan hệ 2 ngôi	3
Quan hệ tương đương	4
Quan hệ thứ tự	5

Bảng 2.5 Các khái niệm trong quan hệ 2 ngôi

Các khái niệm được miêu tả như sau:



Hình 2.1 Quan hệ kế thừa giữa các khái niệm

A. Số nguyên

Khái mức cơ sở tương đương với các số nguyên thông thường. Ta không cần định nghĩa ra khái niệm này.

B. Biểu thức số nguyên

Định nghĩa 3.5: Biểu thức con của một biểu thức.

Gọi p là một biểu thức, khi đó q và r là các biểu thức con của p nếu p = q $\otimes$ r ($\otimes$ là các phép +,-,*,/). Lưu ý các thứ tự và cấp bậc biểu thức dựa vào chuỗi hậu tố của biểu thức mà ta đang xét.

Được xây dung trên các số nguyên và toán tử +,-,*,/. Cấu trúc của khái niệm được dịnh nghĩa như sau:

$$C= (Attr,m\_Funcs).$$

Trong đó, *m_Funcs* chứa các hàm như Simplify nhằm rút gọn biểu thức. *Postfix* tạo chuỗi hậu tố. Thành phần *Attr* bao gồm:

+ sign: thể hiện dấu của biểu thức. âm hay dương.

+ operator: thể hiện phép toán, +,-,*,/ .

+ id: chuỗi hậu tố.

+ value: giá trị biểu thức.

+Childs: biểu thức con

+Parent: biểu thức cha.

C. Tập hợp, quan hệ 2 ngôi, quan hệ tương đương và quan hệ thứ tự

Cấp	Tên khái niệm	Attr	M_Rules
0	Tập hợp	-*elements:* tập các phần tử. -*condition:* đặc tả bằng "quan hệ đại số" -*id:* chuỗi biểu diễn tập hợp: + Nếu tập hợp có hữu hạn các element thì chuỗi này bao gồm các phần tử element + Nếu tập hợp là dạng đặc tả, thì id là một cấu trúc đặc tả.	Không có
1	Quan hệ	-*Không gian mẫu:* là một tập hợp, nó đại diện cho tập hợp mà quan hệ đang xét tới. -*Tính chất:* là các tính chất của một quan hệ hai ngôi. Gồm 4 tính chất.	Không có
2	Quan hệ tương đương	-*Lớp tương đương:* Tập hợp các lớp tương đương của quan hệ tương đương	Quy tắc về phân hoạch tập hợp.
2	Quan hệ thứ tự	-*Hasse* - *Giá trị lớn nhất, nhỏ nhất* - *Tối tiểu , tối đại.*	Quy tắc về các phần tử là tối tiểu và tối đại

Bảng 2.6 Phân loại thành phần C quan hệ 2 ngôi

Trong quan hệ hai ngôi, chúng ta có các quan hệ hai số nguyên đây là các vị từ được định nghĩa trong hàm **PredicateNumber**, và có cấu trúc như sau

<expr1> R <expr2> hoặc R<expr1>

Trong đó: *expr1, expr2* là các biểu thức số nguyên.

“R” là sự ràng buộc về mặt toán học giữa các biểu thức expr1, expr2.

Các quan hệ này được định nghĩa như sau:

Ràng buộc **R** = “=”, “>=”, “<=”

Ký tự	“=”, “<=” , “>=”
Biểu diễn	expr1 **R** expr2
Ý nghĩa	epxr1 bằng (lớn hơn hoặc bằng, bé hơn hoặc bằng) expr2
Điều kiện	- expr1.Value **R** expr2.Value - expr1.Id = expr2.Id
Tính chât	- Với a , thì a **R** a. - Nếu: a **R** b, thì b **R** a - Nếu: a **R** b và b **R** c thì a **R** c

Bảng 2.7 Quan hệ =,>=,<=

- Ràng buộc **R** = “>”, “<”

Ký tự	>,<
Biểu diễn	expr1 **R** expr2
Ý nghĩa	epxr1 lớn hơn(nhỏ hơn) expr2
Điều kiện	- expr1.Value **R** expr2.Value
Tính chât	- Nếu: a **R** b và b **R** c thì a **R** c

Bảng 2.8 Quan hệ <,>

- Ràng buộc **R** = “Chia hết”

Ký tự	“:”
Biểu diễn	expr1 **R** expr2
Ý nghĩa	epxr1 chia hết expr2
Điều kiện	- expr1.Value mod expr2.Value = 0 - expr1.Value = 0 - expr1 = k * expr2, với k là một biểu thức tùy ý
Tính chât	- Với a , thì a **R** a và -a **R** a. - Nếu a **R** b và b **R** c thì a **R** c. - Nếu a **R** c và b **R** c thì a+b **R** c, a-b **R** c , a*b **R** c .

Bảng 2.9 Quan hệ chia hết

- Ràng buộc **R** = “Là số chẵn”

Viết tắt	is Odd
Biểu diễn	**R** expr1
Ý nghĩa	epxr1 là một số chẵn
Điều kiện	- expr1.Value mod 2 = 0 - expr1 = 2*k, với k là một biểu thức tùy ý
Tính chât	- Nếu **R** a thì **R** (-a). - Nếu **R** a và **R** b thì **R** (a+b), **R** (a-b), **R** (a*b).

Bảng 2.10 Quan hệ só chẵn

- Ràng buộc **R** = “là ước số của”

Viết tắt	Is the divisor of
Biểu diễn	expr1 **R** expr2
Ý nghĩa	epxr1 là ước của expr2

Điều kiện	- expr2.Value mod expr1.Value = 0 - expr2.Value = 0 - expr2 = k * expr1, với k là một biểu thức tùy ý
Tính chât	- Với a , thì a **R** a và -a **R** a. - Nếu a **R** b và b **R** c thì a **R** c. - Nếu b **R** a và c **R** b thì c **R** a - Nếu a **R** c và b **R** c thì a+b **R** c, a-b **R** c , a*b **R** c .

Bảng 2.11 Quan hệ ước số

- Ràng buộc **R** = "là bội số của"

Viết tắt	Is the multiple of
Biểu diễn	expr1 **R** expr2
Ý nghĩa	epxr1 là bội của expr2
Điều kiện	- expr1.Value mod expr2.Value = 0 - expr1.Value = 0 - expr1 = k * expr2, với k là một biểu thức tùy ý
Tính chât	- Với a , thì a **R** a và -a **R** a. - Nếu a **R** b và b **R** c thì a **R** c. - Nếu b **R** a và b **R** c thì a **R** c. - Nếu a **R** c và b **R** c thì a+b **R** c, a-b **R** c , a*b **R** c .

Bảng 2.12 Quan hẹ bội số

CHƯƠNG 3. THIẾT KẾT THUẬT GIẢI CHO CÁC VẤN ĐỀ TOÁN RỜI RẠC BẬC ĐẠI HỌC

3.1. Bài toán trên mô hình COKB

Trong các chương trình giảng dạy môn toán rời rạc, ở các chương logic mệnh đề, đại số Boolean và quan hệ hai ngôi, ta sẽ có các dạng bài tập chính sau:

- Rút gọn biểu thức logic
- Suy diễn logic
- Chứng minh hai mệnh đề tương đương
- Xác định chân trị biểu thức logic
- Tối giản hóa hàm Boolean
- Xác định các tính chất của một quan hệ hai ngôi.
- Xác định một quan hệ hai ngôi có là quan hệ tương đương hay không và đưa ra các lớp tương đương
- Xác định một quan hệ có là quan hệ thứ tự hay không và đưa ra các thuộc tính của nó.

Các dạng bài tập của miền tri thức Toán rời rạc được phân ra làm hai loại như sau:

A. Bài tập trên chính đối tượng

Dạng bài tập này là cho một đối tượng cụ thể. Các đối tượng này có tập Attr. Mục tiêu của bài toán là tìm ra một thuộc tính trong Attr thông qua các hàm m_Funcs.

Các dạng bài tập này bao gồm: Xác định tính chất quan hệ 2 ngôi, chứng minh quan hệ thứ tự/tương đương, tìm các thuộc tính quan hệ thứ tự/tuong đương, xác định chân trị mệnh đề.

B. Dạng bài tập trên mô hình COKB

Bài tập dạng này sẽ cho một đối tượng, và để giải toán này cần phải sử dụng các thành phần khác trong mô hinh COKB ở đây là mô hình Ops-funcs.

Đối với miền tri thức toán rời rạc, mô hình chung cho bài tập dạng này có dạng: O→G. Vói O và G tương đương giả thiết và kết luận. Các dạng bài tập bao gồm Rút gọn

Rút gọn biểu thức logic, Suy diễn logic, Tối giản hóa hàm Boolean và Chứng minh hai mệnh đề tương đương

3.2. Thiết kế thuật toán cho các bài toán logic mệnh đề

Tính chất:

1. Nếu biểu thức p ≡ q thì q ≡ p.
2. Cho một luật r có các biến variables (được nêu ở chương trước đó). Nếu ta thay thế các biến trong r bằng các biểu thức khác. Thì ta nhận được một biểu thức mới, tương đương r.
3. Cho một biểu thức *f* ,có chứa một biểu thức con *g*. Nếu ta thay thế *g* bằng một biểu thức, thì cũng tạo ra một biểu thức mới tương đương với *f*
4. Cho biểu thức *f* và *g* .Nếu qua k lần biến đổi *f* theo các luật tương đương ta nhận một biểu thức mới chính là *g*, thì ta nói *f* và *g* tương đương nhau

Định nghĩa 4.1: một biểu thức *p* được gọi là *một trạng thái* của *q* nếu *p* tương đương *q*. Như vậy một biểu thức *p* tương đương với *q*, có thể được hiểu như sau:

$$p \equiv \underbrace{p_1 \equiv p_2 \equiv \cdots \equiv p_n}_{n} \equiv q$$

$$\text{hoặc: } p \overset{P_T}{\rightarrow} q$$

Trong đó: p_1, $p_{1,\dots,}$ p_n là các trạng thái trung gian, sinh ra bằng cách chuyển đổi *p* qua các luật tương đương. P_T là các trạng thái của *p* và *q*.

Định nghĩa 4.2: Cho hai biểu thức mệnh đề p và q, hai biểu thức p và q được gọi là *đồng dạng*, ký hiệu : "≈". Nếu tồn tại các biểu thức p_1, p_2,…, p_n ,sao cho khi thay thế các biểu thức này cho các *biến mệnh đề* q_1, q_2,…, q_n của biểu thức q ta được q=p.

Nhân xét 4.1: Như vậy, để kiểm tra hai biểu thức mệnh đề p và q có đồng dạng nhau không có hai cách:

1. Nếu thay thế các *biến mệnh đề* p_1, p_2,…, p_n của p bằng các *biến mệnh đề* q_1, q_2,…, q_m của q mà p = q thì p≈q.

2. Nếu thay thế các *biến mệnh đề* $p_1, p_2, \ldots, p_n$ của p bằng các *biểu thức con* q_1, $q_2, \ldots, q_m$ của q mà p = q thì p≈q.

VD: A = p^q^q , B = a^b^a, suy ra A≈B vì nếu thay các biến mệnh đề p,q của A bằng các biến mệnh đề a,b của B thì A=B.

A = p^p, B = (a^b)^(a^b), suy ra A≈B vì nếu thay thế biến mệnh đề p của A bằng biểu thức con của B là (a^b), thì A=B.

Nhận xét 4.2:

1. Cho một luật tương đương r và một biểu thức mệnh đề p. Luật r có thể áp dụng cho p nếu:
   + r.Left ≈ p hoặc r.Right ≈ p.
   + r.Left ≈ p' mà p' ∈ p. Hoặc r.Right ≈ p' mà p' ∈ p.
2. Cho một luật suy diễn r, r có tập biểu thức tiền đề h. Luật r có thể có thể áp dụng vào tập giả thiết sự kiện GT của đề bài, nếu các biểu thức h_i (thuộc h) ≈ h_i' ∈ GT.

Định nghĩa 4.3: Suy diễn lôgic là lập luận mà trong đó kết luận được rút ra từ các sự kiện được biết trước theo kiểu: nếu các tiền đề là đúng thì kết luận phải đúng. Nghĩa là các sự kiện cho trước đòi hỏi rằng kết luận là đúng.

3.2.1. Rút gọn mệnh đề

3.2.1.1. Khái niệm

Định nghĩa 4.4: cho hai biểu thúc *f* và *g*. Ta nói rằng, *f* "đơn giản hơn" (ký hiệu "<<") *g* nếu:

$$Length(f) < Length(g)$$

Trong đó hàm, *Length(p)* đo độ phức tạp của một biểu thức. Độ phức tạp này phụ thuộc vào số lượng và loại toán tử của biểu thức p. Cụ thể như sau:

STT	TOÁN TỬ	ĐỘ PHỨC TẠP
1	"( )"	0.1
2	¬	0.5
3	∨,∧	1

4	→	2
5	↔	6
6	NONE	0

Bảng 3.1 Độ phức tạp toán tử

Như vậy, với một biểu thức *p,* mảng các biểu thức con c và bảng độ phức tạp *SC_DIFF* như trên thì:

$$\text{Length(p)} = \sum_{0<i<length(c)} SC\_DIFF(c.O)$$

Ví dụ p = ¬x∨y, Length(p) = 1.5.

Định nghĩa 4.6: trạng thái "simplest" của biểu thức *f* là được ký hiệu *simplest(f)*

Xác định yêu cầu: Bài toán rút gọn có thể được hiểu như sau. Cho một biểu thức *f*, việc rút gọn *f* chính là việc tìm biểu thức *g* tương đương với *f*, có độ phức tạp là nhỏ nhất.

g ≡ *f and* ∀*h, g* << *h*. Do đó *g* là simplest của *f.*

3.2.1.2. Cải tiến luật tương đương cho bài toán rút gọn

Nhắc lại *nhận xét 4.2*, ta có thể áp dụng luật r cho biểu thức p trong hai trường hợp. Một là một vế của r đồng dạng với p. Hai là tồn tại biểu thức đồng dạng với một trong hai vế của r, mà biểu thức đó là biểu thức con của p. Khi này p có thể áp dụng được luật r để biến đổi thành một *trạng thái* nào đó của p.

Tuy nhiên do yêu cầu của bài toán rút gọn mệnh đề là làm giảm đi độ phức tạp của mệnh đề đó. Cho nên ta chỉ cần quan tâm đến chiều hướng mà luật r chuyển đổi biểu thức đơn giản hơn thôi (VD: vế trái r là điều kiện xét và vế phải là kết quả trả về).

Do đó ta chỉ cần xây dựng các luật để kiểm tra và trả về theo đúng yêu cầu đó:

Tên luật	**Điều kiện**

- Luật đồng nhất	-Nếu p có toán tử là phép hội và p có biểu thức con là TRUE. -Hoặc p có toán tử là phép tuyển và p có biểu thức con là False.
Luật nuốt	-Nếu p có toán tử là phép tuyển và p có biểu thức con là TRUE. -Hoặc p có toán tử là phép hội và p có biểu thức con là False.
Luật lũy đẳnt	-Nếu p có hai biểu thức con giống nhau
Luật phần tử bù	-Nếu p có hai biểu thức con đối nhau
Luật phủ định kép	-Nếu p có toán tử là phủ định và biểu thức con của p cũng có toán tử là phép phủ định
Luật kéo theo	-p có toán tử là phép kéo theo
Luật phép tương đương	-p có toán tử là phép tương đương

Bảng 3.2 Các điều kiện áp dụng một luật tương đương

Tuy nhiên, với các luật như *De Morgan*, *kết hợp* và *phân phối*. Cần xem xét cả hai chiều hướng, do các luật này có tạo ra các trạng thái trung gian cho bước chuyển đổi tiếp theo.

3.2.1.3. Xây dựng thuật toán

Trước khi đề cập đến các ý tưởng xây dựng thuật toán để giải quyết vấn đề. Chúng ta cần xem lại cách tính độ phức tạp của một biểu thức. Ta nhận thấy rằng, độ phức tạp của một biểu thức p sẽ phụ thuộc vào ba yếu tố đó là toán tử của p, số lượng biểu thức con của p và độ phức tạp của các biểu thức con của p.

Do đó, để tiến hành rút gọn một biểu thức mệnh đề p thì chúng ta sẽ:

- Tối giản toán tử của p.
- Giảm số lượng biểu thức con của p.
- Rút gọn các biểu thức con của p.

Ý tưởng: Cho biểu thức *f*. Tại một thời điểm bất kỳ, có thể đơn giản hóa *f* thông qua hai cách. Một là tìm kiếm các luật r có thể áp dụng *f* khiến *f* . Hai là tiến hành đơn giản hóa các biểu thức con của nó. Cả hai cách trên tạo ra các trạng thái khác nhau cho *f*. Ta sẽ chọn trạng thái nào đơn hơn, gọi là *f'* .Nếu *Length(simplest(f)) = Length(f')* thì đặt *simplest(f) = f'*. Nếu *f'* là một trạng thái không tốt (tức là phức tạp hơn trạng thái cũ). Và nếu sau k lần chuyển đổi tiếp theo mà f' độ phức tạp của f' vẫn lớn hơn simplest(f) thì ta kết thúc quá trình này

Thuật giải:

Input: p- một biểu thức đầu vào, tập luật cải tiến R

Output: g là *simplest(p), trans* –danh sách các bước chuyển đổi trung gian .

Thực hiện:

B1:

- Đặt simplest(p)=p.

- Tìm kiếm các luật có thể áp dụng cho p, ưu tiên các luật làm giảm độ phức tạp p.

- Tiến hành rut gọn các con của p qua hàm *visitChilds(deep)*- deep thể hiện chiều sâu để duyệt trong cây biểu thức *Tree(p).* Tính từ chiều sâu biểu thức hiện tại.

- So sánh và chọn trạng thái tốt nhất gáng cho p.

- So sánh simplest(p) và p.

+ Nếu simplest(p)>>p thì simplest(p)=p;

+ Ngược lại tính trước k lần chuyển đổi nếu p>>simplest(p) thì lặp lại B1, ngược lại trả về kết quả simplest(p).

B2: Tiến hành rút gọn các biểu thức con.

B3: Lặp lại B1 một lần nữa.

3.2.2. Chứng minh hai biểu thức tương đương

Dựa vào công thức 1.1 . Ta có được nhận xét sau:

Nhận xét:

1. Cho hai biểu thức *f* và *g* . Nếu *g* là *simplest* (*f)* thì *g* tương đương *f.*

2. Cho hai biểu thức *f* và *g*. Nếu *simplest(f) = simplest(g)* thì *f* tương đương với *g*. Ngược lại, Nếu *simplest(f) ≠ simplest(g)* thì *f* không tương đương với *g* .

*__Xác định vấn đề:__ bài toán chúng minh hai mệnh đề tương đương được hiểu như sau. Cho biểu thức T có dạng $p \equiv q$. Tìm các trạng thái biến đổi P_T của p sao cho $p \xrightarrow{P_T} q$. Nếu:

1. simplest (p) = q thì P_T = Simplify(p).
2. simplest (q) = p thì P_T = Reverse(Simplify(q))
3. Simplify(p).Lastest = Simplify(q).Lastest thì P_T = (Simplify(p) ∪ Simplify(q)) / (Simplify(p) ∩ Simplify(q))

Thuật giải:

Input: biểu thức T có dạng $p \equiv q$, tập luật tương đương được cải tiến R

Output: Result -tập các bước chuyển đổi.

Thực hiện:

Đặt T_p = Simplify(p) và T_q = Simplify(q).

, $simplest_p$ = Lastest(T_p) và $simplest_q$= Lastest(T_q)

if: $simplest_p$ = q then đặt Result = T_p

else if $simplest_q$ = p then đặt Result = Reverse (T_q)

else if $simplest_q$ = $simplest_p$ then Simplify(p) ∪ Simplify(q)) / (Simplify(p) ∩ Simplify(q))

3.2.3. Suy diễn logic

Xác định vấn đề: bài toán suy diễn logic được biểu diễn như sau. Cho tập các biểu thức giả thiết GT và biểu thức mục tiêu G.Tìm các suy diễn từ các luật R. để chứng minh G ∈ GT.

Đinh nghĩa: một lời giải suy diễn được định nghĩa bởi cấu trúc sau:

Reasoning = (Id, Expr,Parent_Id, Rule)

Trong đó: Id là thứ tự của lời giải, nếu id = 0 thì đó là các giả thiết của đề bài

Expr là kết quả thu được sau bước suy diễn này, Parent_Id: tập hợp các cơ sở để có được bước suy diễn này

Rule: Luật được áp dụng

Ý tưởng: Ta sẽ dùng suy diễn tiến để thực hiện quá trình suy luận. Lần lượt lấy các thành phần của tập giả thiết, rồi áp dụng các luật đã được định nghĩa cho mệnh đề đang xét đó để tạo ra các sự kiện hay ràng buộc khác. Quá trình này được dừng lại cho đến khi, hoặc không thể tiến hành suy diễn được nữa hoặc đã tìm thấy mục tiêu O từ tập GT.

Phương pháp này có ưu điểm là đơn giản, dễ cài đặt. Tuy nhiên, cách thức này đôi khi sẽ nảy sinh ra các bước dư thừa, khiến không gian tìm kiếm trở lên lớn, gây tốn kém tài nguyên. Đôi khi các giả thiết dư thừa cũng làm cho quá trình suy diễn trở nên phức tạp.

Vì thế, em xin được phép đề xuất một giải pháp để hạn chế vấn đề này. Đó là sau mỗi một phép biến đổi hay suy diễn, chúng ta sẽ sắp xếp lại tập giả thiết sao cho nó càng gần mục tiêu O càng tốt. Cụ thể, các giả thiết có các biểu thức sơ cấp và toán tử càng giống với mục tiêu thì sẽ được ưu tiên duyệt trước, các giả thiết còn lại sẽ được xếp phía sau. Những giả thiết là sự kiện sẽ có độ ưu tiên thấp nhất đề hạn chế duyệt chúng. Sau quá trình sắp xếp trên ta lại thực hiện tiếp quá trình biến đổi và suy diễn cho đến khi gặp điều kiện dừng.

Để miêu tả quá trình sắp xếp này, chúng ta xem xét ví dụ. Cho mô hình suy diễn sau:

$$\begin{array}{l} p \vee q \\ \overline{p} \\ \overline{q} \vee r \\ s \rightarrow \overline{r} \\ \hline \therefore \overline{s} \end{array}$$

Ta tiến hành sắp xếp lại tập giả thiết:

Giả thiết	**Loại**	**Số lượng biểu thức sơ cấp giống nhau**	**Điểm**
p ∨ q	Ràng buộc	0	0

¬p	Sự kiện	0	-1
¬q ∨ r	Ràng buộc	0	0
s → ¬r	Ràng buộc	1	1

Bảng 3.3 Giả thiết sau khi được sắp xếp

Tập giả thiết sau khi đươc sắp xếp lại

Giả thiết	Điểm
s → ¬r	1
p ∨ q	0
¬q ∨ r	0
¬p	-1

Thuật giải cho bài toán suy diễn:

Input: Tập giả thiết GT và mục tiêu G

Ouput: Tập Result

Thực hiện:

B1: *Known* := *A*;*Sol* := [ ];

B2: *flag* := true;

while *flag* **do**

flag := false;

for *r* in **Rules do**

if (*r* có thể áp dụng cho *Known*) **then**

flag := true;

Add *r* into *Sol*.

Update *Known* bằng *r*(*Known*)

if ($B \subseteq$ *Known*) **then** **Goto** B3;

fi;

od; # for

od; # while

B3 3:

if ($B \subseteq Known$) **then**

Suy diễn $A \Rightarrow B$ đúng và *Sol* là kết quả.

else

Suy diễn $A \Rightarrow B$ sai

3.2.4. Xác định chân trị biểu thức

Xác định yêu cầu: cho một biểu thức mệnh đề p, kết quả trả về là một ma trận 2^MxN .

Với M là số lượng biến mệnh đề cấu thành nó

N là số lượng biểu thức trong p kể cả p.

Thuật giải:

Input: biểu thức p

Output: ma trận MT MxN

Thực hiện:

B1: Duyệt và ghi tất cả các biến mệnh đề có trong mệnh đề P vào trong tập A.

B2: Đếm tất cả các biêu thức có trong P (kể cả P). Và tạo ra ma trận MT kích thước MxN.

B3: Từ tập A, sinh ra tất cả trạng thái ứng với các trường hợp giá trị có thể xảy ra cho các biến mệnh đề, và lưu chúng vào mảng T.

B4: Ứng với từng trạng thái T_i có trong T. Ta tiến hành cập nhật chân trị của các mệnh đề dựa vào giá trị của các biến mệnh đề và các toán tử, theo chiều hướng từ mệnh đề cấp thấp nhất đến mệnh đề cấp cao nhất. Ghi nhận chân trị đó vào MT.

B5: lặp lại bước 3 đến khi T không còn nữa. Kết quả cuối cùng là ma trận MT

3.3. Thiết kế thuật giải giải rút gọn bìa Karnaugh

Trong miền tri thức đại số Boolean, bài tập chính đó là đơn giản hóa hàm Boolean. Để giải được dạng bài này, chúng ta cần thực hiện bốn quá trình sau:

- Ánh xạ đa thức biểu diễn hàm Boolean lên bìa Karnaugh
- Xác định các tế bào lớn
- Xác định các nhóm tế bào
- Xác định các đa thức tối giản nhất

A. Ánh xạ đa thức biểu diễn hàm Boolean lên bìa Karnaugh

Định nghĩa 4.3: tế bào trong bìa Karnaugh được biểu diễn bằng một chuỗi nhị phân, nó được tao ra bằng cách liên kết các biến trong đa thức Boolean lại với nhau. Do đó, một hàm Boolean có n biến sẽ có 2^n tê bào trong bìa Karnaugh tương ứng.

Ma trận biểu diễn bìa Karnaugh là một mảng ma trận, do các bài tập trong chương trình dạy ở các trường từ 3-5 biến. Do đó bìa Karnaugh được biểu diễn thông qua một mảng không gian 3 chiều.

Thuật toán:

Input: chuỗi đa thức.

Output: mảng ba chiều được ánh xạ từ đa thức.

Các bước:

B1: Xác định số lượng biến, khởi tạo mảng ba chiều M theo *Quy tắc thứ tự Bìa Karnaugh*

B2: Với mỗi một đơn thức T_i của đa thức F_n , sẽ được chuyển thành một số C_i = ChuyenDoi(T_i). Các số đó được thêm vào mảng A.

B3: Tiến hành duyệt mảng M nếu bất kỳ phần tử nào của M không thuộc A. Ta gáng phần tử đó là -1.

B. Xác định tế bào lớn

Biểu diễn tế bào lớn: Do các tế bào được biểu diễn thông qua các số tự nhiên (khác -1). Nên tế bào lớn được biểu diễn lả một mảng các số tự nhiên.

Ý tưởng: Do số lượng các biến của đề bài thường được cố định từ 3-5 biến. Vì thế, ta hoàn toàn xác định được tế bào lớn nhất có thể xuất hiện . Một tế bào lớn có dạng 2^n ô, nên trường hợp khả thi nhất là tế bào 16 ô, tế bào 8 ô,…, tế bào 1 ô. Để có thể lọc dược các tế bào này, ta cần tạo một khung cố định rồi lần lượt duyệt qua bìa Karnaugh. Bộ khung được xây dựng dựa vào số lượng các biến trong bìa Karnaugh, cụ thể:

**Bìa Karnaugh 3 biến:*

- Nhóm tế bào 16 biến: không có.
- Nhóm tế bào 8 biến : không khả thi.
- Nhóm tế bào 4 biến:

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$				
C				

Ô vuông 4

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$				
C				

Đối xứng qua cột

Hình 3.1 Nhóm tế bào 4 của bìa Karnaugh 3 biến

- Nhóm tế bào 2:

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$				
C				

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$				
C				

	$\overline{AB}$	$\overline{A}$B	AB	A$\overline{B}$
$\overline{C}$				
C				

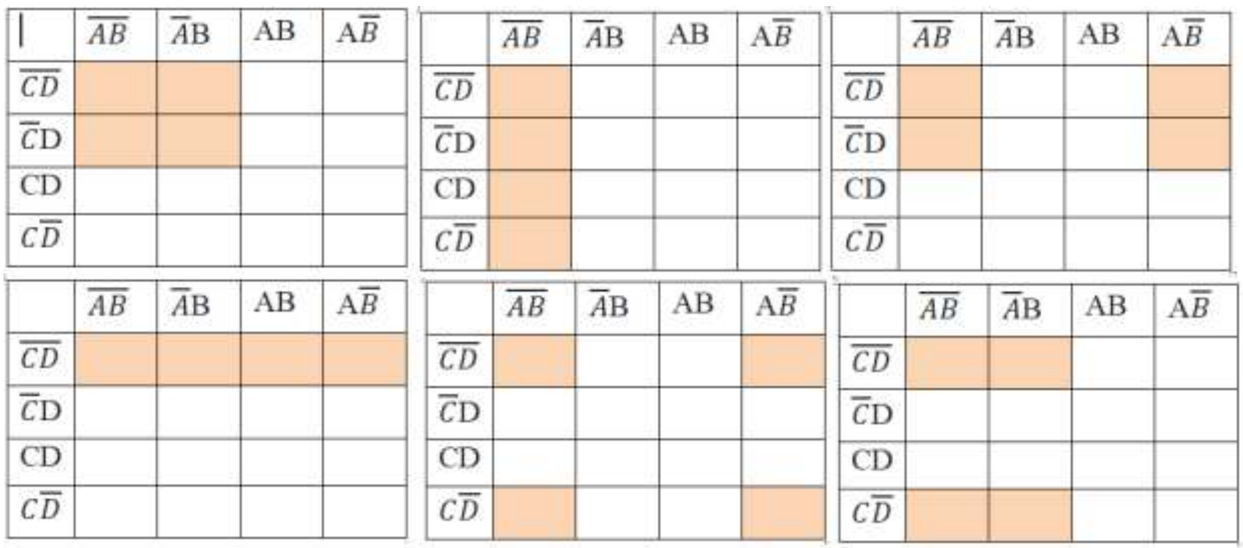
Hình 3.2 Nhóm tế bào 2 của bìa Karnaugh 3 biến

**Bìa Karnaugh 4 biến:*

- Nhóm tế bào 16 ô: không thả khi.
- Nhóm tế bào 8 ô

Hình 3.3 Nhóm tế bào 8 của bìa Karnaugh 4 biến

- Nhóm tế bào 4:



Hình 3.4 Nhóm tế bào 4 bìa Karnaugh 4 biến

- Nhóm tế bào 2

	$\overline{AB}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{CD}$				
$\overline{C}D$				
CD				
$C\overline{D}$				

	$\overline{AB}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{CD}$				
$\overline{C}D$				
CD				
$C\overline{D}$				

	$\overline{AB}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{CD}$				
$\overline{C}D$				
CD				
$C\overline{D}$				

	$\overline{AB}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{CD}$				
$\overline{C}D$				
CD				
$C\overline{D}$				

Hình 3.5 Nhóm tế bào 3 của bìa Karnaugh 4 biến

C. Xác định các nhóm tế bào lớn

Sau khi đã ánh xạ hàm Boolean lên một bìa Karnaugh, bước tiếp theo ta cần tìm các nhóm tế bào. Sao cho toàn bộ các phần tử của mỗi nhóm tế bào vừa bao phủ hết kar(f).

Thuât giải như sau:

- Đầu tiên, duyệt qua ma trận biểu diễn bìa Karnnaugh, thống kê ở từng ô có bao nhiêu tế bào đã bao phủ nó.

- Sau khi thống kê hoàn thành. Với các ô chỉ có một tế bào được bao phủ, tế bào đó chính là một tế bào thiết yếu và ta cần lưu giữ nó. Đồng thời với tế bào này, ta phải duyệt và đánh dấu cho các ô mà tế bào này chứa đựng.

- Sau khi hoàn thành bước trên, nếu vẫn còn xót một ô nào trong Kar(f), thì ô đó được trên hai tế bào lớn bao phủ. Trường hợp này ta tiến hành lấy từng trường hợp của các tế bào này. Từ đó tìm được tất cả các đa thức của f.

D. *Xác định đa thức tối giản.*

Định nghĩa 4.4: hai biểu thức đơn giản hơn.

Cho hai công thức đa thức của một hàm Bool :F = m1+m2 +…. +m_k và G =M1 + M2 +… + M_l.Ta nói rằng công thức F đơn giản hơn công thức G nếu tồn tại đơn ánh h: {1,2,..,k} → { 1,2,…, l} sao cho với mọi I ∈ {1,2,..,k} thì số từ đơn của m_i không nhiều hơn số từ đơn của $M_{h(i).}$

Sau khi tìm được tất cả đa thức của f. Dựa vào *định nghĩa hai biểu thức đơn giản hơn*, đa thức f được xem là tối giản nhất nếu f có số đơn thức là ít nhất.

3.4. Thiết kế thuật giải các bài toán quan hệ hai ngôi.

Các bài toán của quan hệ hai ngôi bao gồm: xác định các tính chất của quan hệ, chứng minh quan hệ tương đương và chứng minh quan hệ thứ tự.

A. Xác định các tính chất của quan hệ

Đối với bài toán trên quan hệ R có cấu trúc danh sách phần tử, được hiểu như sau. Cho R, R có một tập các phần tử *elements*. Với mỗi một tập luật r_i ∈ Rules, thì R.tinhChat$_i$=r_i (R).

Các luật xác định tính chất được quy định trong **mục 3.3**

B. *Xác minh quan hệ tương đương.*

Đối với bài toán này ta cần chia ra các trường hợp cụ thể:

Tập không gian cụ thể - Quan hệ cụ thể

Xét một quan hệ R có tập không gian A. Nếu R và A được biểu diễn bằng danh sách phần tử. Xác định một nhóm các lớp tương đương phan hoạch A như sau:

Bước 1: Ánh xạ R thành một ma trận M.

Bước 2: Tạo một mảng label chỉ định nhãn của từng thành phần trong A. Đặt index = 0.

Bước 3: Nêu label[index] là rỗng thì đặt label[index] = index.

Bước 4: Tiến hành duyệt trên ma trận M. Nếu có bất kỳ $M_{index\ j}$ nào khác 0, label[j] là rỗng và index ≠ j. Thì ta gáng label[j] = index, và đặt index = j rồi lặp lại bước 2.

Bước 5: tiến hành duyệt lại label để xem phần tử nào chưa xet nhãn để đặt index là thành phần đó rồi lại tiếp tục lặp bước 2.

Bước 6: các phần tử trong A đã được dánh nhãn chính là các lớp cần tìm.

Tập không gian cụ thể - Quan hệ đặc tả

Với một R là dạng đặc tả nhưng có tập không gian mẫu A xác định hữu hạn các phần tử. Ta tiến hành chuyển R thành dạng cấu trúc danh sách phần tử. Bằng cách duyệt từng phần tử trong A, rồi xét với các phần tử khác có trong A, xem thõa mãn điều kiện không. Sau đó, chuyển thành trường hợp *Tập không gian cụ thể* - *Quan hệ cụ thể*.

Tập không gian đặc tả - Quan hệ đặc tả

Với trường hợp này hệ thống chỉ hỗ trợ dạng *quan hệ đại số* chia hết và số chia là một số nguyên. Giả sử số nguyên đó là m, để tìm các lớp tương ứng của R, ta chỉ cần tìm các số a sau cho, a mod m = k. (k từ 0 đến m-1). Do đó nó m lớp tương đương.

C. Xác minh quan hệ tương đương.

Bài toán chỉ xét trong trường hợp A là một tập hợp có hữu hạn các phần tử. Việc giải quyết bài toán xác minh quan hệ tương đương sẽ xoay quanh đối tượng biểu đồ Hasse.

Định nghĩa 4.5: Biểu diễn cấu trúc biểu đồ Hasse.

1. Một đỉnh của biểu đồ Hasse gọi là một node, mỗi node có giá trị bằng một số và một danh sách các node cha(là các node trội của nó) và node con (là các node bị trội của nó).
2. Hasse được biểu diễn bằng một ma trận, với các node không có cha gọi là các seed. Các node không có con gọi là các leaf.

Xây dựng Hasse được thực hiện theo quy tắc sau:

B1: Ánh xạ R thành một ma trận M.

B2: Nếu Hase đang rỗng ta khởi tạo một node tương đương phần tử trên A. Rồi gáng nó vào *seed* trong Hasse.

B3: Duyệt Hasse từ trên xuống nếu:

+ Giá trị i hiện tại, mà có bất kỳ một j nào trong *Hasse* làm M_{ij} = 1 thì ta khởi tạo một node i nằm tầng trên node j và gáng cha của I là j, con của j là i.

+ Nếu không có bất kỳ phần tử nào Hasse có quan hệ với i thì gáng i vào *seed* của *Hasse*

- Đối với các thuộc tính khác của quan hệ thứ tự thì:

- Tối tiểu: là tập seed của Hasse.

- Tối đại là tập các leaf của Hasse.

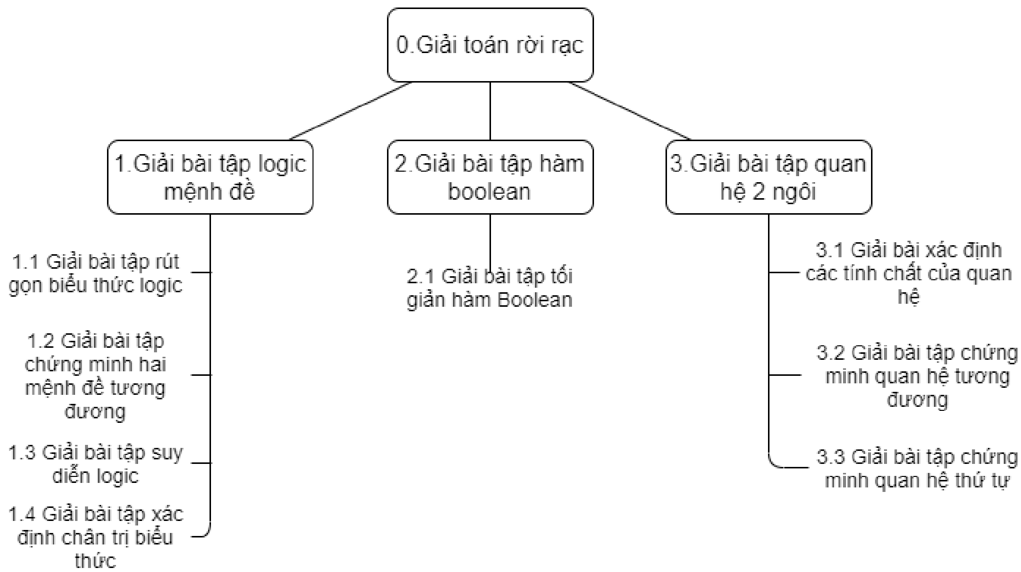
- Giá trị nhỏ nhất: khi seed có một giá trị duy nhất thì nó chính là giá trị nhỏ nhất.

- Giá trị lớn nhất: khi tập leaf có duy nhất một phần tử thì nó là giá trị lớn nhất.

CHƯƠNG 4. HỆ THỐNG GIẢI TOÁN RỜI RẠC

Hệ thống giải toán rời rạc được xây dựng thành một website đơn giản. Chương trình hỗ trợ giải các bài tập theo từng bước, phù hợp cho những ai đang có mong muốn tham khảm.

4.1. Phân tích nghiệp vụ hệ thống



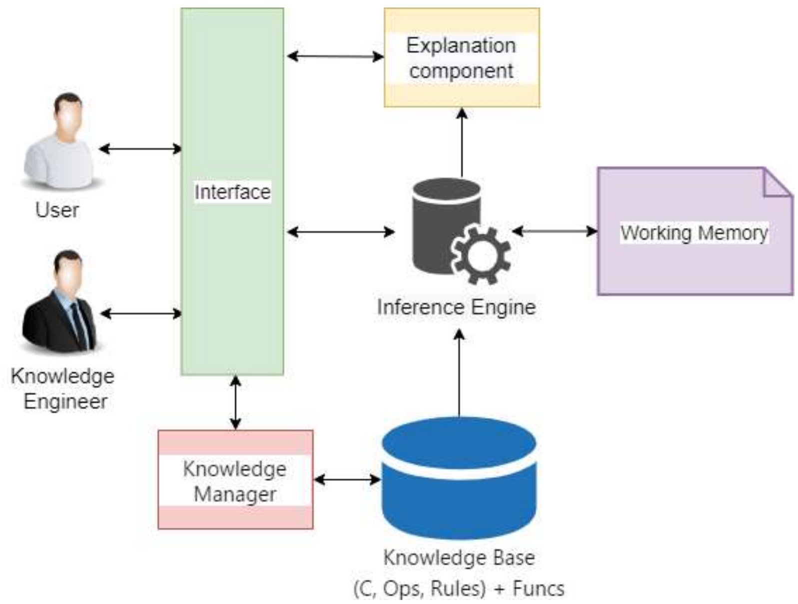
Sơ đồ 4.1 Sơ đồ nghiệp vụ của hệ thống

Hệ thống được xây dựng có các chức năng chính như sau:

- Giải bài tập liên quan logic mệnh đề: người dùng tiến hành nhập chuỗi biểu thức. Hệ thống tiến hành đưa ra lời giải theo đúng yêu cầu của người dùng như: rút gọn biểu thức, chứng minh hai mệnh đề tương đương nhau, suy diễn logic và xuất ra chân trị của một biểu thức.
- Giải bài tập hàm Boolean: chỉ có duy nhất một bài tập được hỗ trợ, đó là tối giản hàm Boolean. Đầu vào là chuỗi đa thức hoặc danh sách các ô của Kar(f). Lời giải là tất cả các đa thức tối giản của f.
- Giải tập chương quan hệ hai ngôi: xác định các tính chất của một quan hệ, chứng minh và tìm các thuộc tính của quan hệ hế tương đương. Chứng minh và tìm các tính chất của quan hệ thứ tự. Lưu ý: các quan hệ này chỉ nằm trong tập số nguyên Z.

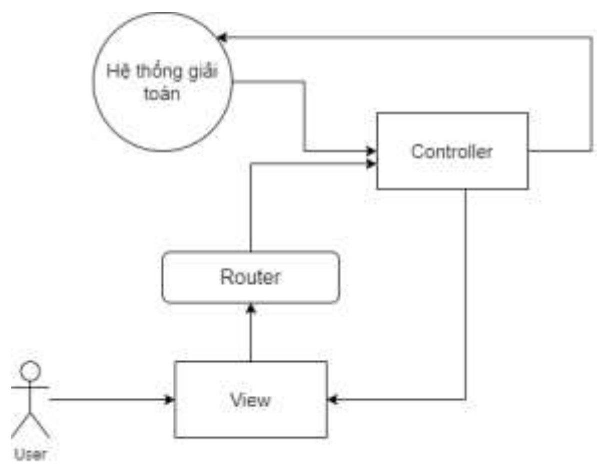
4.2. Cấu trúc tổng quát của hệ thống

Mô hình chuẩn của một hệ giải quyết vấn đề



Hình 4.1. Mô hình chuẩn hệ giải quyết vấn đề

Hệ thống được xây dựng thành website có cấu trúc như sau:

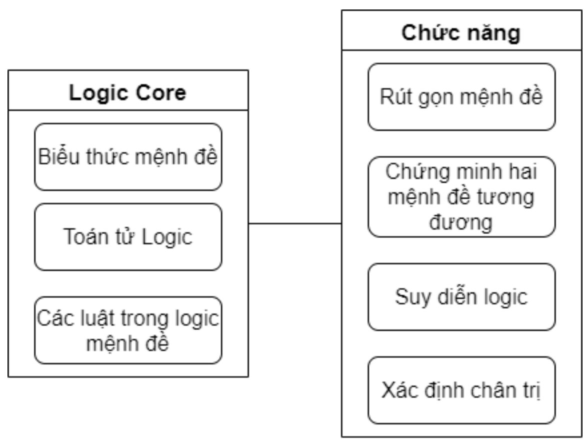


Sơ đồ 4.2 Cấu trúc website giải toán rời rạc

Trong đó cấu trúc cốt lõi, hệ thống giải toán bao gồm:

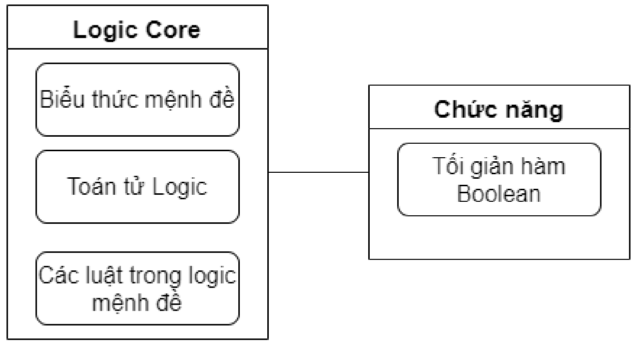
Sơ đồ 4.3 Cấu trúc hệ thống giải toán

**Cấu trúc bộ giải bài tập logic mệnh đề:* Cấu trúc này được ánh xạ từ cấu trúc của mô hình về miền tri thức Logic mệnh đề đã được đề cập trước đó:



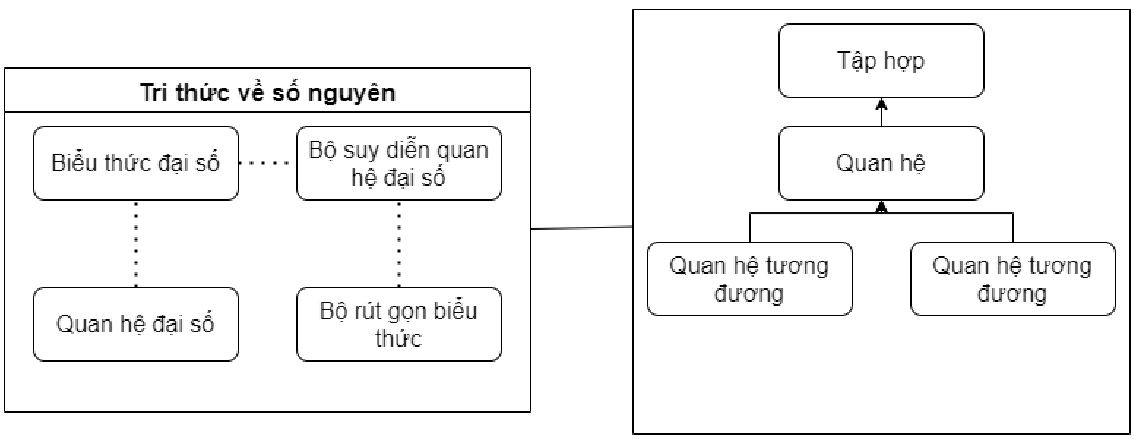
Sơ đồ 4.4 Cấu trúc bộ giải bài tập logic mệnh đề

**Cấu trúc bộ giải bài tập hàm Boolean:*



Sơ đồ 4.5 Cấu trúc bộ giải bài tập hàm Boolean

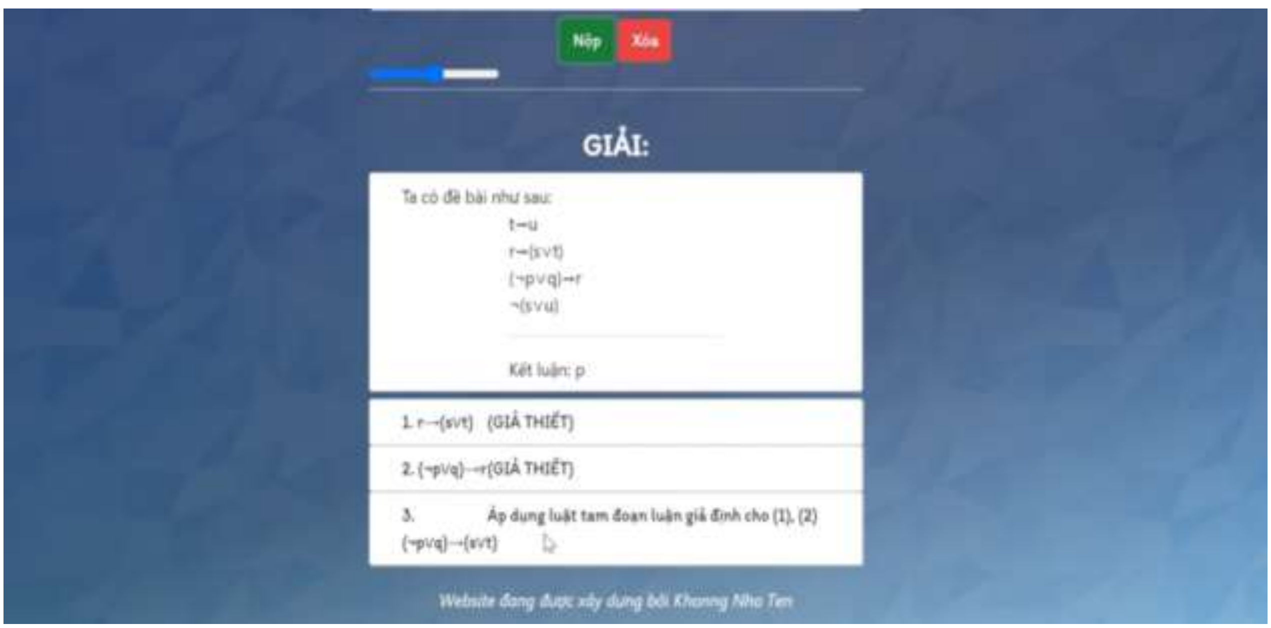
**Cấu trúc bộ giải bài tập quan hệ hai ngôi:*các thành phần chính bao gồm.



Sơ đồ 4.6 Cấu trúc bộ giải bài tập quan hệ hai ngôi

4.3. Giao diện chương trình

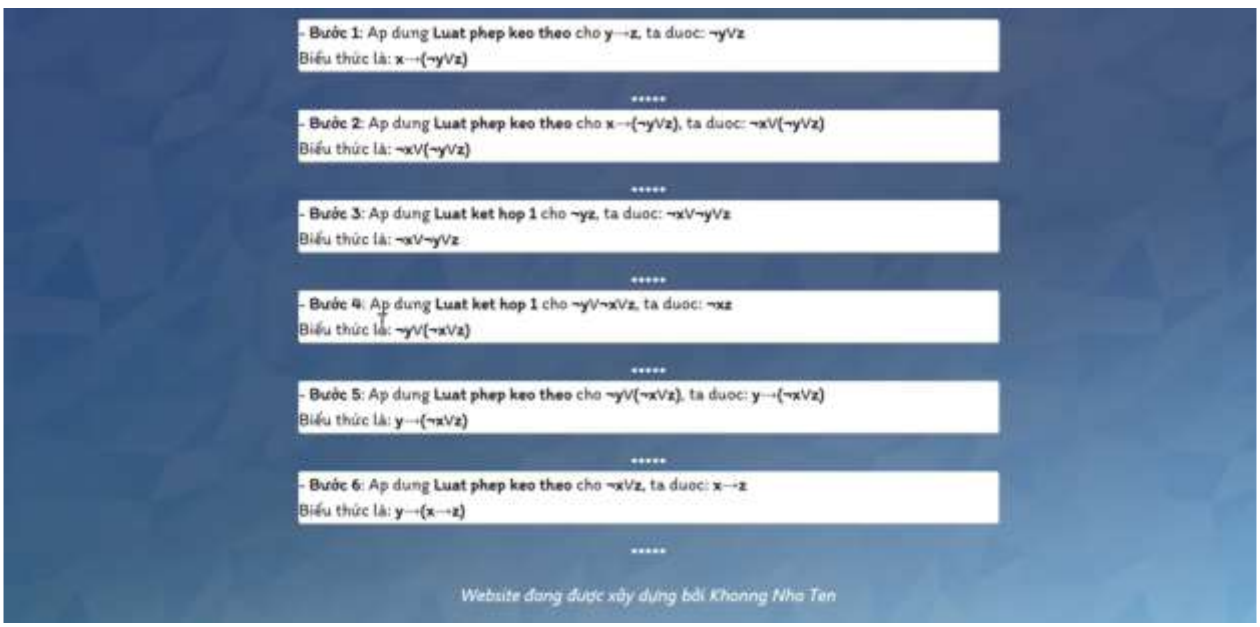
Hình 4.2 Màn hình nhập liệu



Hình 4.3 Màn hinh suy diễn

Hình 4.4 Màng hình xác định chân trị

-



Hình 4.5 Màn hình chứng minnh hai mệnh đề tương đương

Hình 4.6 Màn hình bài toán rút gọn hàm Boolea

-

-

Hình 4.7 Màn hình bài toán quan hệ hai ngôi

4.4. Công nghệ sử dụng

4.4.1. Nodejs

Hình 4.8 Logo nodejs

NodeJS là một nền tảng được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript, giúp xây dựng các ứng dụng web một cách đơn giản và dễ dàng mở rộng.

NodeJS được phát triển bởi Ryan Dahl vào năm 2009 và có thể chạy trên nhiều hệ điều hành khác nhau: OS X, Microsoft Windows, Linux.

Nodejs được ứng dụng trong các việc xây dựng:

- Websocket server: Các máy chủ web socket như là Online Chat, Game Server…
- Fast File Upload Client: là các chương trình upload file tốc độ cao.
- Ad Server: Các máy chủ quảng cáo.
- Cloud Services: Các dịch vụ đám mây.
- RESTful API: đây là những ứng dụng mà được sử dụng cho các ứng dụng khác thông qua API.
- Any Real-time Data Application: bất kỳ một ứng dụng nào có yêu cầu về tốc độ thời gian thực.
- Micro Services: Ý tưởng của micro services là chia nhỏ một ứng dụng lớn thành các dịch vụ nhỏ và kết nối chúng lại với nhau.

4.4.2. Typescript

Hình 4.9 Logo typescript

Typescript là một dự án mã nguồn mở được Microsoft phát triển, được xem là một phiên bản nâng cao của Javascript. TypeScript là một ngôn ngữ giúp cung cấp quy mô lớn hơn so với JavaScript.

Vì sao lại được xem là phiên bản nâng cao của Javascript? Vì nó được bổ sung những tùy chọn kiểu tĩnh và các lớp hướng đối tượng, nó bao hàm luôn ES6(ECMAScript 6 2015) - phiên bản mới nhất của Javascript.

TypeScript thêm các namespace, class và module tùy chọn vào JavaScript. TypeScript hỗ trợ các công cụ cho các ứng dụng JavaScript quy mô lớn cho bất kỳ trình duyệt nào, cho bất kỳ máy chủ nào, trên bất kỳ hệ điều hành nào.

4.5. Kiểm thử

Sau khi đã xây dựng thành công website giải toán rời rạc. Em đã tiến hành giải một số bài toán và có được một số thống kê sau đây.

Yêu cầu nhập biểu thức mệnh đề thì:

+ Cần phải có dấu “( )” để phân tác các biểu thức con.

+ True là 1 và False là 0.

+ Không được để các ký tự đặc biệt.

+ Chấp nhận khoảng trắng.

Yêu cầu nhập quan hệ hai ngôi:

+ Chỉ cho phép trong phạm vi số nguyên.

+ Không được nhập sai biểu thức điều kiện

Loại bài	**Số lượng**	**Thành công**	**Độ khó**			**Tỷ lệ**	**Tốc độ**
			1	**2**	**3**		
Rút gọn biểu thức logic	27	20	17	6	4	74.1%	1.57s
Chứng minh hai biểu thức tương đương	34	25	18	10	6	73.35%	1.72s
Suy diễn logic	30	23	18	8	4	76.6%	1.95s
Xác định chân trị	21	20	14	6	1	95.2%	0.6s
Tối giản hàm Boolean	24	24	8	14	2	100%	0.72s
Xác định tính chất quan hệ	29	27	20	8	1	93.1%	0.31s
Chứng minh quan hệ tương đương	28	26	22	3	2	92.9%	0.27s
Chứng minh quan hệ thứ tự	22	19	16	4	2	86.3%	0.31s

Bảng 4.1 Thông kê kết quả

TỔNG KẾT

Kết quả đạt được

Trong quá trình hoàn thành đồ án này em đã có được các thành quả sau

A. Kiến thức đạt được

- Học được các kiến thức về toán rời rạc ba chương Logic mênh đề , đại số hàm Boolean và quan hệ hai ngôi. Nhờ vao quá trình khảo sát lý huyết.
- Đã tim hiểu được cấu trúc của mô hình COKB và biết vận dụng để xây dựng một mô hình cho một miền tri thức.
- Tìm hiểu và tiếp cận được các công nghệ đang nổi hiện nay bao gồm Nodejs và Expresss, ES6 của Javascript và ngôn ngữ TypeScript (cách tiếp cận khác của Javascript).
- Biết cách Deploy website lên heroku.
- Trao dồi thêm khả năng trình bày, viết báo cáo và thuyết trình.

B. Kỹ năng khác

Trong quá trình hoàn thanh đồ án. Em đã có cơ hội để trtao dồi thêm những kỹ năng mềm bao gồm tìm kiếm, khả năng viết báo cáo, thuyết trình, cách xây dựng ý tưởng và phát triển thuật toán.

Vấn đề tồn tại

Trong thời gian gấp rút, cùng với khả năng của tác giả, chương trình có nhiều sai xót cần phải khắc phục:

- Giao diện còn xấu
- Chưa kiểm tra dữ liệu đầu vào
- Hiệu quả giải toán chưa cao
- Chưa xây dựng được toàn bộ miền tri thức Toán rời rạc

Kiến nghị

- Hoàn thiện mô hình biểu diễn.
- Khắc phục các lỗi của hệ thống web
- Cải thiện giao diện người dùng

- Nâng cấp lên các nền tảng khác nhất là di động.
- Thêm chức năng tra cứu kiến thức.

PHỤ LỤC

Phụ lục 1: hướng dẫn sử dụng

Demo: https://toanroiracnhanh.herokuapp.com/

Source code: https://github.com/KhongTaiKhoan/toanroiracnhanh

Hoặc truy cập https://www.youtube.com/channel/UCl5h9rYAGlEFKXMbeLKJX-g để xem demo qua video.

Phụ lục 2: yêu cầu nhập liệu

Yêu cầu nhập biểu thức mệnh đề thì:

+ Cần phải có dấu "( )" để phân tác các biểu thức con.

+ True là 1 và False là 0.

+ Không được để các ký tự đặc biệt.

+ Chấp nhận khoảng trắng.

Yêu cầu nhập quan hệ hai ngôi:

+ Chỉ cho phép trong phạm vi số nguyên.

+ Không được nhập sai biểu thức điều kiện

TÀI LIỆU THAM KHẢO

[1] Kenneth H. Rosen, *Discrete Mathematics and Its applications*, 7th edition, McGraw-Hill (2012).

[2] C. Li, K. Mehrotra, *Problems on Discrete Mathematics*, Syracuse University, New York (2011).

[3] "*Final Report on the 2013 NSF Workshop on Research Challenges and Opportunities in Knowledge Representation*" Natasha Noy and Deborah McGuinness (Eds). National Science Foundation Workshop Report, August 2013

[4] William F. Klostermeyer, *Discrete Mathematics Problems, University of North Florida* (2012)

[5] Do, V.N., 2012. *Intelligent Problem Solvers in Education: Design Method and Applications, Intelligent Systems, In: Vladimir M. Koleshko (Ed.), Intelligent systems, InTech*, pp. 121-148

[6] Aladova, E., Plotkin, T., 2017. *Logically automorphically equivalent knowledge bases,* arXiv:1707.01027v1

[7] Do, V.N., 2015. *Ontology COKB for knowledge representation and reasoning in designing knowledge-based systems. Communications in Computer and Information Science* (CCIS), vol. 513, Springer, pp. 101-118

[8] Do, V.N., Nguyen, D.H., Mai, T.T., 2015. *Reasoning Method on Knowledge about Functions and Operators, International Journal of Advanced Computer Science and Applications* (IJACSA), 6(6): 156 – 168

[9] Hien D. Nguyen and Nhon V. Do, *Intelligent Problem Solver in Education for Discrete Mathematics University of Information Technology, New Trends in Intelligent Software Methodologies, Tools and Techniques* (SoMeT_17)

[10] *Tổng quan về Nodejs* https://trungquandev.com/mot-cai-nhin-tong-quan-nhat-ve-nodejs/ (truy cập ngày 27-5-2021)

[11] *Giới thiệu về typescript* https://viblo.asia/p/gioi-thieu-typescript-su-khac-nhau-giua-typescript-va-javascript-LzD5dDn05jY (truy cập 11-06-2021)