ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

LƯU THANH SƠN

KHÓA LUẬN TỐT NGHIỆP

MỘT SỐ LỚP BÀI TOÁN VÀ PHƯƠNG PHÁP SUY DIỄN TRÊN CƠ SỞ TRI THỨC COKB

CỬ NHÂN NGÀNH KHOA HỌC MÁY TÍNH

TP. HỒ CHÍ MINH, 2018

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

LƯU THANH SƠN – 14520772

KHÓA LUẬN TỐT NGHIỆP

MỘT SỐ LỚP BÀI TOÁN VÀ PHƯƠNG PHÁP SUY DIỄN TRÊN CƠ SỞ TRI THỨC COKB

CỬ NHÂN NGÀNH KHOA HỌC MÁY TÍNH

GIẢNG VIÊN HƯỚNG DẪN

PGS.TS ĐỖ VĂN NHƠN

TP. HỒ CHÍ MINH, 2018

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số 413/QĐ-ĐHCNTT ngày 01/08/2018 của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. PGS.TS Nguyễn Đình Thuân – Chủ tịch.
2. ThS Phạm Nguyễn Trường An – Thư ký.
3. PGS.TS Vũ Thanh Nguyên – Ủy viên.

LỜI CẢM ƠN

Em xin chân thành gửi lời cảm ơn sâu sắc đến PGS.TS Đỗ Văn Nhơn, người đã tận tình chỉ dạy và hướng dẫn em trong suốt thời gian thực hiện Khoá luận tốt nghiệp cũng như khuyến khích động viên em khi gặp khó khăn trong quá trình thực hiện.

Em xin bày tỏ lòng biết ơn đến toàn thể quý Thầy cô trong trường Đại học Công nghệ thông tin vì đã truyền đạt cho em rất nhiều kiến thức quý báu trong suốt thời gian học tập tại trường.

Mặc dù đã đạt được một số kết quả nhất định, nhưng trong quá trình thực hiện khoá luận này không tránh khỏi các thiếu sót. Rất mong quý Thầy cô và độc giả thông cảm bỏ qua và góp ý để khoá luận được hoàn thiện hơn.

Xin chân thành cảm ơn.

Thành phố Hồ Chí Minh, tháng 7 năm 2018.

Sinh viên thực hiện

Lưu Thanh Sơn

MỤC LỤC

DANH MỤC BẢNG

DANH MỤC HÌNH VẼ

MỞ ĐẦU

Mục tiêu nghiên cứu chính của ngành khoa học Trí tuệ nhân tạo là làm sao để chế tạo các máy tính thông minh, có thể suy nghĩ như con người, nhằm tận dụng khả năng xử lý của máy tính vào giải quyết các vấn đề trong các lĩnh vực của cuộc sống. Một trong các ứng dụng đó là việc xây dựng các hệ giải vấn đề thông minh và hệ chuyên gia. Để các hệ giải quyết vấn đề thông minh có thể hoạt động được thì các hệ thống đó phải được trang bị các tri thức cần thiết nhất định trong một lĩnh vực. Tuy nhiên, các tri thức hiện nay của con người không thể đưa vào máy tính dưới dạng thô được mà phải thông qua các phương pháp và mô hình biểu diễn tri thức nhất định. Một trong các mô hình biểu diễn tri thức có hiệu quả để mô hình hoá các tri thức đó là mô hình tri thức các đối tượng tính toán – COKB. Mô hình này đã được ứng dụng để xây dựng các hệ giải vấn đề thông minh trên các miền tri thức khác nhau như Hình học phẳng, hình học giải tích, điện một chiều, … Lý thuyết về mô hình COKB đã được nhiều tác giả xây dựng tương đối đầy đủ về mô hình, cách lưu trữ và biểu diễn cũng như các thuật giải và phương pháp suy diễn. Tuy nhiên, các phương pháp suy diễn và kỹ thuật suy diễn trên mô hình COKB hiện tại vẫn chưa hoàn thiện để có thể áo dụng cho các miền tri thức khác nhau. Đề tài này sẽ hoàn thiện hơn động cơ suy diễn trên mô hình COKB, đồng thời sẽ giới thiệu thêm một số lớp bài toán cụ thể trên mô hình COKB để từ đó hoàn thiện hơn phương pháp suy luận của mô hình COKB.

CHƯƠNG 1: TỔNG QUAN

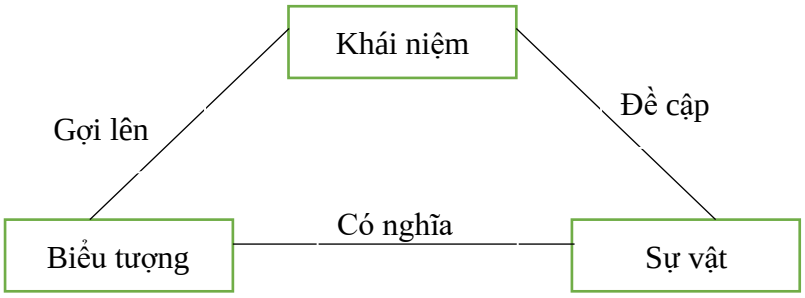
1.1. Các phương pháp biểu diễn tri thức

Các tri thức của con người thường được thể hiện ở dạng tự nhiên như ngôn ngữ, lời nói và ký hiệu. Tuy nhiên, các dạng biểu diễn tự nhiên đó không thể đưa vào máy tính để xử lý được mà phải thông qua các phương pháp biến đổi phù hợp để máy tính có thể đọc và hiểu được. Hiện nay chúng ta có một số phương pháp biểu diễn tri thức như sau:

- **Phương pháp biểu diễn theo logic:** Các phương pháp này sử dụng các biểu thức logic hình thức để diễn đạt các sự kiện và ngữ nghĩa trong câu. Phương pháp biểu diễn bằng Logic tương đối đơn giản và dễ sử dụng, tuy nhiên, đối với những tri thức với cấp độ trừu tượng cao thì các phương pháp biểu diễn bằng logic không khả thi. Các loại logic thường được sử dụng trong phương pháp biểu diễn bằng logic gồm: Logic mệnh đề, logic vị từ và logic mô tả.
- **Phương pháp biểu diễn theo hệ luật dẫn:** Phương pháp biểu diễn này dựa trên các luật có dạng *if – then,* là dạng luật khá thông dụng đối với con người, được sử dụng rộng rãi để mô tả các định lý, các điều kiện và các quan hệ trong thực thế cuộc sống. Tuy nhiên, cũng như phương pháp biểu diên bằng logic, hệ luật dẫn khó mô tả được các tri thức có mức độ trừu tượng cao cũng như việc kế thừa các tri thức.
- **Phương pháp mạng ngữ nghĩa:** Đây là phương pháp biểu diễn tri thức dưới dạng các đồ thị, do đó nó kế thừa được các công cụ và thuật toán đã được phát triển dùng cho lý thuyết đồ thị như: tìm đường đi ngắn nhất, tìm cây khung, tìm cây bao trùm nhỏ nhất hay kiểm tra tính liên thông, tìm chu trình Euler, … Hơn nữa, phương pháp này còn thể hiện được sự kế thừa giữa các tri thức với nhau. Điểm yếu lớn nhất của phương pháp mạng ngữ nghĩa là khi biểu diễn các miền tri thức lớn với số lượng tri thức khá nhiều thì kích thước mạng sẽ tăng dẫn đến chi phí cao khi thực thi việc tìm kiếm trên mạng ngữ nghĩa. Các

mô hình mạng ngữ nghĩa thường dùng là: Đồ thị khái niệm (conceptual graph) và Mạng tính toán (Computational Network).

- **Phương pháp biểu diễn bằng Frame và Script:** Frame là một cấu trúc dữ liệu chứa đựng tất cả những thứ liên quan đến một đối tượng cụ thể nào đó. Nói cách khác, frame đóng gói toàn bộ một đối tượng, một khái niệm hay một thực thể nào đó thành một thực thể có cấu trúc nhất định. Script tương tự như frame, nhưng thay vì mô tả các đối tượng, nó được dùng để đặc tả một chuỗi các sự kiện. Script sử dụng các Slot để chứa các thông tin liên quan đến đối tượng và hành động của chúng
- **Phương pháp ontology:** Đây là phương pháp biểu diễn tri thức dựa trên khái niệm chung về một đối tượng cụ thể. Theo [8], Ontology được định nghĩa là sự đặc tả tường minh của sự khái niệm hoá trong một lĩnh vực. Đây có thể coi là một phương pháp hiện đại để biểu diễn tri thức, nhất là các tri thức có tính trừu tượng cao. Tuỳ vào các miền tri thức cụ thể ta sẽ xây dựng các Ontology riêng cho các miền tri thức đó. Trong thực tế, để mô ta một sự vật, hiện tượng người ta thường dùng bộ ba mô tả thế giới: Biểu tượng, khái niệm và Sự vật. Theo cách biểu diễn này, Biểu tượng sẽ gợi nên Khái niệm và biểu diễn Sự vật còn Khái niệm sẽ đề cập tới sự vật.



Hình 1-1. Bộ ba mô tả thế giới của Ontology

Nhận xét: Không có một phương pháp nào gọi là tối ưu nhất cho biểu diễn tri thức do tri thức thực tế của con người rất đa dạng và rộng lớn trên nhiều lĩnh vực khác nhau. Mỗi tri thức sẽ có điểm mạng và điểm yếu riêng theo từng loại tri thức. Do đó, người ta thường có xu hướng kết hợp nhiều phương pháp lại với nhau để biểu diễn tri thức.

1.2. Các phương pháp suy diễn

Các phương pháp suy diễn là cơ sở cho bộ suy diễn hoạt động giải quyết vấn đề trên cơ sở tri thức có sẵn. Các phương pháp suy diễn thường được xây dựng mô phỏng theo cách giải quyết vấn đề của con người khi gặp một bài toán. Cụ thể là, khi con người gặp một bài toán về hình học, với các số liệu cho trước, con người sẽ sử dụng kiến thức của mình về hình học, cùng với các số liệu và tính chất cho trước của đề bài để tìm ra lời giải. Ở đây tôi sẽ giới thiệu một vài phương pháp và chiến lược suy diễn thường dùng.

- **Phương pháp suy diễn tiến:** Theo các tài liệu [9] và [12], Suy diễn tiến là phương pháp suy diễn hoạt động theo nguyên tắc dựa vào các sự kiện đã biết trước, sử dụng các luật mà phần giả thiết của luật khớp với các sự kiện đã cho trước, và tiếp tục như thế cho đến khi tìm ra kết quả mục tiêu hoặc không thể áp dụng luật nào khác. Phương pháp này khá phổ biến vì nó gần gũi với các mà con người giải quyết hầu hết các bài toán trong cuộc sống.
- **Phương pháp suy diễn lùi:** Phương pháp này hoạt động bằng cách xuất phát từ sự kiện cần chứng minh và thay vào đó là các sự kiện giả thiết của một luật có sự kiện kết luận là sự kiện cần chứng minh, và tiếp tục như thế cho đến khi tập sự kiện là tập con của tập sự kiện giả thiết. Nói các khác, phương pháp này truy ngược từ kết luận hay mục tiêu về giả thiết của bài toán dựa vào các luật trong cơ sở tri thức. Đây là cách phổ biến khi ta cần chẩn đoán một sự việc hay hiện tượng nào đó trong cuộc sống, chẳng hạn cần kiểm tra máy tính bị hỏng ở bộ phận nào khi biết tình trạng hiện tại của máy tính.

- **Phương pháp lập luận dựa trên tình huống:** Theo tài liệu [12], lập luận dựa trên tình huống là phương pháp hiệu chỉnh lời giải của các bài toán đã có trước đó để tìm ra lời giải cho bài toán mới. Khi gặp bài toán mới, hệ thống sẽ tìm kiếm lại một vài bài toán mẫu trong CSTT trước đó để tìm ra bài toán mẫu phù hợp nhất. Nếu như lời giải của bài toán mẫu tìm được phù hợp với bài toán mới thì hệ thống sẽ sử dụng lời giải của bài toán mẫu đó, ngược lại hệ thống sẽ tìm cách hiệu lời giải của bài toán mẫu để tạo ra lời giải mới phù hợp với bài toán đang xét.
- **Phương pháp suy diễn dựa trên bài toán mẫu:** Theo tài liệu [5] và [12], bài toán mẫu thể hiện những dạng bài toán mà khi sử dụng các phương pháp suy diễn chung sẽ rất tốn thời gian, những dạng này sẽ được ghi lại bước giải để có thể giải quyết các bài toán. Khi gặp bài toán mới tương tự, chúng ta sẽ dùng lại các bước giải này để giải quyết hoặc nếu bài toán mới có hơi khác đi một chút, ta sẽ cập nhật lời giải cho bài toán nhanh chóng bằng các bước giải có sẵn tương tự. Phương pháp này cũng khá thường gặp trong cuộc sống, nhất là đối với học sinh, khi gặp các bài toán mới, các học sinh thường có xu hướng xác định "dạng" của bài toán mới để có thể áp dụng các bước giải theo "dạng" của bài toán đã giải quyết trước đó để tìm ra lời giải nhanh chóng. "Dạng" đó chính là bài toán mẫu đã được học sinh ghi nhớ các bước giải trước đó.

1.3. Mô hình COKB và các nghiên cứu liên quan

Mô hình COKB (Computational Objects Knowledge base) hay còn gọi là Mô hình tri thức các đối tượng tính toán được giới thiệu lần đầu vào năm 2001 bởi tác giả Đỗ Văn Nhơn là một mô hình biểu diễn tri thức theo hướng tiếp cận Ontology. Từ khi ra đời tới nay, đã có nhiều tác giả đóng góp cho mô hình này hoàn thiện hơn. Dưới đây sẽ chỉ ra những nghiên cứu tiêu biểu về lý thuyết mô hình COKB:

- Bài báo [1] tác giả đã giới thiệu đầy đủ về mô hình COKB cùng các loại sự kiện cụ thể và phương pháp so khớp sự kiện. Tuy nhiên bài báo chưa đề cập cụ thể tới các quy tắc suy diễn.
- Bài báo [2] và [3] các tác giả đã giới thiệu một số phương pháp suy diễn trên một số miền tri thức cụ thể. Các phương pháp được đề cập ở trên chỉ đưa ra ý tưởng chứ chưa đi vào phương pháp thiết kế một quy tắc suy diễn cụ thể.
- Bài báo [4] các tác giả đã trình phương pháp suy diễn trên các thành phần Functions và Operators. Các tác giả cũng đã tiến hành đặc tả các thành phần tri thức Hàm và Toán tử. Tuy nhiên các bài báo chưa đưa ra các quy tắc suy diễn cụ thể để có thể áp dụng vào thiết kế động cơ suy diễn.
- Bài báo [5] các tác giả trình bày phương pháp suy diễn sử dụng mẫu bài toán (Sample Problem).
- Bài báo [6] các tác giả nghiên cứu về cách suy diễn trên thành phần Relations của mô hình COKB.
- Bài báo [6] các tác giả đã trình bày phương pháp suy diễn áp dụng mẫu bài toán trên COKB.
- Trong bài báo [7], các tác giả đã cơ bản hoàn thiện mô hình COKB, gọi là mô hình COKB đầy đủ (Perfect COKB). Ngoài ra, tác giả cũng trình bày các phương pháp suy diễn cụ thể trên các thành phần của mô hình COKB.
- Trong luận văn [10], tác giả đã hoàn thiện động cơ suy diễn trên mô hình COKB, tuy nhiên có hai thành phần mà tác giả chưa đề cập đến là thành phần tri thức Hàm (Functions) và Toán tử (Operators).

1.4. Mục tiêu của đề tài

Theo khảo sát trong phần 1.3, phần suy diễn trên tri thức về thành phần Hàm (Functions) và Toán tử (Operators) trên mô hình COKB còn hạn chế. Trong khi đó, các tri thức về thành phần Hàm và Toán tử trong thực tế lại đóng vai trò khá quan trọng trong việc thể

hiện các quan hệ tính toán giữa các đối tượng với nhau. Ví dụ như trong miền tri thức về Hình học phẳng, hình chiếu của một điểm lên một mặt phẳng là một Hàm, hay trong miền tri thức về Ma trận và hệ phương trình tuyến tính, phép cộng hay nhân giữa hai ma trận với nhau là một toán tử.

Trên cơ sở đó, đề tài này sẽ giải quyết các phương pháp suy diễn trên các thành phần tri thức Hàm và toán tử, cụ thể là đặc tả các thành phần hàm và toán tử, tổ chức lưu trữ và quy tắc suy diễn các thành phần hàm và toán tử cùng các thuật giải suy diễn liên quan. Từ đó, đề tài sẽ góp phần hoàn thiện động cơ suy diễn trên mô hình COKB.

Ngoài ra, đề tài này còn đề nghị một số lớp bài toán mới mà trong công trình [10] tác giả chưa đề cập cùng các thuật giải liên quan. Các lớp bài toán được đề nghị là các lớp bài toán tổng quát, có thể áp dụng trên nhiều miền tri thức thuộc các lĩnh vực khác nhau.

Đề tài chọn miền tri thức về Ma trận và hệ phương trình tuyến tính để làm minh hoạ cho các phương pháp và thuật giải suy diễn trên mô hình COKB trên các lớp bài toán khác nhau. Phần minh hoạ sẽ được xây dựng bằng ngôn ngữ Maple.

CHƯƠNG 2: MÔ HÌNH TRI THỨC COKB ĐẦY ĐỦ

2.1. Mô hình tri thức các đối tượng tính toán

Định nghĩa 2.1: Một mô hình tri thức các đối tượng tính toán (Computational Object Knowledge Base) gồm có 6 thành phần như sau (Tham khảo từ [7]):

(C, H. R. Ops, Funcs, Rules)

2.1.1. Tập C gồm các khái niệm về đối tượng tính toán.

C là tập hợp các khái niệm trong miền tri thức và mỗi khái niệm là một Lớp đối tượng tính toán. Mỗi lớp đối tượng tính toán được phân cấp dựa theo cấu trúc hay xác định (định nghĩa) của khái niệm trong miền tri thức.

Dựa vào sự phân cấp trên chúng ta có các cấp khái niệm sau:

- Các khái niệm nền: đây là các khái niệm cơ bản, được công nhận trước trong hầu hết các ngôn ngữ lập trình.
- Các khái niệm cấp 1: Đây là các khái niệm được xây dựng dựa trên các khái niệm đã được công nhận trước ở trên.
- Các khái niệm cấp 2: Đây là các khái niệm được xây dựng dựa trên các khái niệm cấp 1 và các khái niệm cơ bản.
- Các khái niệm cấp n: Đây là các khái niệm được xây dựng dựa trên các khái niệm cấp (n-1).

Định nghĩa 2.2: Một đối tượng tính toán (Computational Object hay C-Object), ký hiệu O, có những thành phần đặc trưng sau:

(Attr, Facts, Rules)

Trong đó:

Attrs: Tập các thuộc tính của đối tượng.

Facts: Tập các tính chất hay sự kiện vốn có của đối tượng.

Rules: Tập các luật.

Trong thành phần Rules sẽ được chia ra làm 2 thành phần con gồm: ***(Rf,Rr)***

Rf: Tập các quan hệ suy diễn tính toán. Với mỗi rfi ∈ Rf một quan hệ dạng suy diễn tính toán có dạng như sau: f: e1 = e2 | var(e1) ⊆ Attr(O), var(e2) ⊆ Attr(O).

Rr: Tập các quan hệ dạng luật dẫn.

Ghi chú: Các luật suy diễn rule ∈ Rules là các luật suy diễn nội tại trên các thuộc tính và sự kiện của đối tượng

Trong miền tri thức về ma trận và Hệ phương trình tuyến tính, chúng ta có các khái niệm như sau:

- Các khái niệm cơ sở: Array, int, real.
- Các khái niệm cấp 1: ma trận, ma trận vuông, ma trận khả nghịch.
- Các khái niệm cấp 2: hệ phương trình tuyến tính.

Khái niệm về ma trận sẽ được mô hình hoá như sau:

MATRAN *= (Attr, Facts, Rules), với:*

Attr *= { m: int, n: int, K:array[m][n] }*

Trong đó:

m: là số dòng của ma trận, kiểu số nguyên.

n: là số cột của ma trận, kiểu số nguyên.

K: là giá trị của ma trận, kiểu array[m][n] (mảng 2 chiều). Với mỗi phần tử aij thuộc K, i <= m, j<=n, aij là kiểu số nguyên.

Fact*={}*

Rules *= { }*

Khái niệm về Ma trận vuông: Khái niệm về Ma trận vuông được mô tả như sau:

MATRANVUONG *= (Attr, Facts, Rules), với:*

Attr *= { m: int, n: int, K:array[m][n] }*

Trong đó:

m: là số dòng của ma trận, kiểu số nguyên.

n: là số cột của ma trận, kiểu số nguyên.

K: là giá trị của ma trận, kiểu array[m][n] (mảng 2 chiều). Với mỗi phần tử aij thuộc K, i <= m, j<=n, aij là kiểu số nguyên.

Fact *= { m = n } //Một ma trận vuông sẽ có số dòng bằng số cột.*

Rules *= { }*

Các hành vi trên đối tượng tính toán MATRANVUONG: Thực hiện biến đổi sơ cấp, tìm hạng của ma trận và quan trọng nhất là tìm giá trị định thức của ma trận vuông.

Khái niệm về Ma trận khả nghịch: Khái niệm về Ma trận khả nghịch được xây dựng như sau::

MATRANKHANGHICH *= (Attr, Facts, Rules), với:*

Attr *= { m: int, n: int, K:array[m][n] }*

Trong đó:

m: là số dòng của ma trận, kiểu số nguyên.

n: là số cột của ma trận, kiểu số nguyên.

K: là giá trị của ma trận, kiểu array[m][n] (mảng 2 chiều). Với mỗi phần tử aij thuộc K, i <= m, j<=n, aij là kiểu số nguyên.

Fact*={}*

Rules *= { }*

Các xác định một ma trận nghịch đảo của ma trận khả nghịch:

- Gọi Q = (A|I) là ma trận mở rộng của ma trận khả nghịch A với ma trận đơn vị B.
- Thực hiện biến đổi sơ cấp trên dòng của Q cho tới khi được dạng: $(I|A^{-1})$.

- Ma trận A^{-1} trên chính là ma trận nghịch đảo của ma trận A.

Từ cách xác định ma trận nghịch đảo của ma trận khả nghịch A ở trên, chúng ta suy ra các hành vi trên một đối tượng MATRANKHANGHICH sẽ bao gồm: Tìm ma trận Q = (A|I), thực hiện biến đổi sơ cấp ma trận Q và từ ma trận biến đổi sơ cấp của Q xác định được ma trận nghịch đảo của ma trận khả nghịch.

Khái niệm về Hệ phương trình tuyến tính: Hệ phương trình tuyến tính là tập hợp các phương trình tuyến tính có dạng m ẩn và n hệ số có dạng:

$$\begin{array}{rr} a11.x1 + a12.x2 + \cdots + a1n.xn & b1 \\ a21.x1 + a22.x2 + \cdots + a2n.xn & b2 \\ \cdots & \cdots \\ am1.x1 + am2.x2 + \cdots + amn.xn & bn \end{array}$$

Trong đó: aij và bi là các hệ số cho trước, x1 … xn là các ẩn.

Ta đặt A = (aij), B = (bi) và x = (xj) lần lượt là ma trận hệ số, cột các hệ số tự do và cột ẩn của hệ phương trình. Như thế, hệ phương trình tuyến tính sẽ có các dạng khác như sau:

- Dạng tích ma trận: A.X = B.
- Dạng ma trận hoá: ở đây ta hiểu ngầm cột ẩn X: Q = (A|B).

  Q ở đây được gọi là ma trận bổ sung (hay ma trận mở rộng của hệ phương trình tuyến tính.

Giải hệ phương trình tuyến tinh: Ta thực hiện **phép biến đổi Gauss-Jorrdan** để gỉai hệ phương trình tuyến tính. Các bước như sau:

Bước 1: Viết ra dạng Ma trận hoá: (A|B)

Bước 2: Thực hiện các phép biến đổi sơ cấp trên dòng. Nếu gặp trường hợp có một dòng có dạng (0 … 0 | a) (ứng với dạng phương trình tuyến tính 0x1 + … + 0xn = a) thì ngừng biến đổi và kết luận hệ phương trình vô nghiệm.

Bước 3: Kết luận:

- Hệ phương trình có nghiệm duy nhất khi thu được n dòng khác zero (hay còn gọi là Hạng của ma trận).
- Hệ phương trình có vô số nghiệm khi thu được k dòng khác zero với k < n.

Từ các bước trên, ta có **định lý Knronecker – Capelli** như sau: Đặt Q = (A|B) là ma trận mở rộng của hệ phương trình. Từ đó ta có r(Q) là hạng của ma trận mở rộng Q và r(A) là hạng của ma trận hệ số A. Ta có các trường hợp sau xảy ra:

Hệ phương trình có các trường hợp sau:

Có nghiệm duy nhất: khi r(Q) = r(A) = A.n;

Có vô số nghiệm khi: r(Q) = r(A) = k và k < A.n;

Vô nghiệm khi: r(Q) = r(A) + 1;

Dựa vào dạng Ma trận hoá của hệ phương trình, ta mô hình hoá đối tượng tính toán HEPHUONGTRINH như sau:

HEPHUONGTRINH[A,B] = (Attr, R, Rules)

Attr *= { A: MATRAN, B:MATRAN, Q:MATRAN }*

Trong đó:

A: là ma trận các hệ số của hệ phương trình, kiểu *MATRAN*

B: là ma trận các hệ số tự do của hệ phương trình, kiểu *MATRAN*.

Q: Ma trận mở rộng (bổ sung) của Hệ phương trình, kiểu MATRAN.

Fact *= { Q=<A|B> }*

Rules *= {*

[HANG(Q) = HANG(A), HANG(A) = A.n] => [NGHIEM(Q)],

[HANG(Q) = HANG(A) + 1] => [NGHIEM(Q) = [‘VO NGHIEM’]]

[HANG(Q) = HANG(A), HANG(A) < A.n] => [NGHIEM(Q) = [‘VO SO NGHIEM’]] }

Các hành vi có thể có trên đối tượng tính toán Hệ phương trình: Tìm ma trận mở rộng từ ma trận hệ số và ma trận hệ số tự do, xác định hạng của ma trận hệ mở rộng và thực hiện biến đổi sơ cấp trên ma trận mở rộng.

2.1.2. Tập H gồm các quan hệ phân cấp giữa các đối tượng

Đây là tập hợp các quan hệ phân cấp trên các đối tượng C. Các quan hệ phân cấp là các dạng quan hệ đặc biệt như: **IS_A**, **HAS_A**.

Cấu trúc của quan hệ phân cấp:

[<Tên đối tượng cấp cao>, <tên đối tượng cấp thấp>]

Ví dụ: **MATRANVUONG IS_A MATRAN** (ma trận vuông là một Ma trận) được cấu trúc như sau:

[MATRAN, MATRANVUONG].

2.1.3. Tập R gồm các quan hệ giữa các đối tượng tính toán

Đây là tập các quan hệ dựa trên các đối tượng tính toán. Mỗi quan hệ được xác định bởi tên qua hệ và các loại đối tượng của quan hệ. Một quan hệ tính toán R là một quan hệ 2 ngôi giữa các khái niệm trong tập C. Các quan hệ có thể có các tính chất sau: phản xạ, đối xứng, bắc cầu.

Cấu trúc một quan hệ:

[< Tên quan hệ>,< Loại đối tượng 1>,< Loại đối tượng 2>, ...] { tính chất 1, tính chất 2, ...}

2.1.4. Tập Ops gồm các toán tử

Một toán tử định nghĩa một ánh xạ 2 ngôi Ci x Ci → Ci, với Ci là một khái niệm trong tập C. Các toán tử thể hiện các quan hệ tính toán giữa các đối tượng với nhau và được biểu diễn thông qua một biểu thức tính toán.

expr ::= object | expr <operator> expr

Đặc tả một toán tử như sau (theo tài liệu [4]):

Operator-def ::= OPERATOR <name>

ARGUMENT: argument-def+

RETURN: return def+;

PROPERTY: prob-type+

[constraint]

[variables]

[statements]

END OPERATOR.

argument-def::= <name> : type

return-def::=name : type

prob-type ::= commutative | associative | identity

statements::= statement-def+

statement-def ::= assign-stmt | if-stmt | for-stmt

asign-stmt::= name := expr;

if-stmt ::= IF logic-expr THEN statements+ ENDIF; | IF logic-expr THEN statements+ ELSE statements+ ENDIF;

for-stmt::= FOR name IN [range] DO statements+ ENDFOR;

Giải thích về các đặc tả:

Operator-def: Đặc tả một thành phần toán tử. Một thành phần toán tử gồm: Tên toán tử, Tham số đầu vào, giá trị trả về, thuộc tính toán tử và các câu lệnh.

argument-def: Các tham số đầu vào, gồm có tên biến tham số và kiểu của nó.

return-def: Kiểu trả về, có thể trả về một kiểu dữ liệu cơ bản hoặc một kiểu định nghĩa.

prob-type: Tính chất của thành phần toán tử. Có 3 tính chất: kết hợp (associative), giao hoán (commutative) và hợp nhất (identity).

statement-def: Các câu lệnh thực thi. Các câu lệnh có 3 dạng: câu lệnh gán, câu lệnh điều kiện và câu lệnh lặp.

Câu lệnh gán: có dạng *<tên biến> := <biểu thức>*.

Câu lệnh điều kiện: có 2 dạng: *if <biểu thức logic> then <câu lệnh> end if;* hoặc *if <biểu thức logic> then <câu lệnh 1> else <câu lệnh 2> end if:*

Câu lệnh lặp: có dạng *for <biến chạy> in <khoảng biến chạy> do <câu lệnh> end for:*

2.1.5. Tập Funcs gồm các hàm

Một hàm được định nghĩa là một ánh xạ f: Ci → Ci, với Ci là một thành phần khái niệm trong tập khái niệm C.

Mỗi hàm gồm tên hàm và các loại đối tượng của Hàm. Hàm có một số tính chất: phản xạ, đối xứng, phản xứng, bắc cầu.

Theo tài liệu [4], có 2 dạng đặc tả một hàm:

Dạng 1:

function-def ::= FUNCTION name;
ARGUMENT: argument-def
RETURN: return-def;
[constraint]
[facts]
ENDFUNCTION;

Dạng 2:

function-def ::= FUNCTION name;
ARGUMENT: argument-def+

RETURN: return-def;
[constraint]
[variables]
[statements]
ENDFUNCTION;

argument-def::= <name> : type
return-def::=name : type
statement-def ::= assign-stmt | if-stmt | for-stmt
asign-stmt::= name := expr;
if-stmt ::= IF logic-expr THEN statements+ ENDIF; | IF logic-expr THEN statements+ ELSE statements+ ENDIF;
for-stmt::= FOR name IN [range] DO statements+ ENDFOR;

Giải thích về các đặc tả:

function-def: Đặc tả một thành phần hàm. Một thành phần hàm gồm: Tên hàm, tham số, kiểu trả về và các câu lệnh.

argument-def: Các tham số đầu vào, gồm có tên biến tham số và kiểu của nó.

return-def: Kiểu trả về, có thể trả về một kiểu dữ liệu cơ bản hoặc một kiểu định nghĩa.

statement-def: Các câu lệnh thực thi. Các câu lệnh có 3 dạng: câu lệnh gán, câu lệnh điều kiện và câu lệnh lặp.

Câu lệnh gán: có dạng *<tên biến> := <biểu thức>*.

Câu lệnh điều kiện: có 2 dạng: *if <biểu thức logic> then <câu lệnh> end if;* hoặc *if <biểu thức logic> then <câu lệnh 1> else <câu lệnh 2> end if:*

Câu lệnh lặp: có dạng *for <biến chạy> in <khoảng biến chạy> do <câu lệnh> end for:*

2.1.6. Tập Rules gồm các luật dẫn

Mỗi quy tắc suy luận từ các sự kiện biết trước suy ra được các sự kiện mới thông qua việc áp dụng định luật, định lý hay các quy tắc tính toán nào đó.

Cấu trúc một luật suy diễn như sau:

R: { kf1, kf2, …} => {nf1, nf2, …}.

Hay ta có thể rút gọn thành h(r) => g(r), với h(r) là tập sự kiện vế trái của luật, còn g(r) là tập sự kiện vế phải của luật.

Mối sự kiện kfi ∈ h(r) và nfi ∈ g(r) đều thuộc một trong 12 loại sự kiện theo như đã phân loại trong mô hình tri thức COKB.

Ví du: Quy tắc xác định ma trận vuông được phát biểu: "Một Ma trận có số dòng bằng số cột là một ma trận vuông" được mô hình hoá như sau:

{ [A, "MATRAN"], A.m = A.n } => { [A, "MATRANVUONG"] }.

2.2. Các loại sự kiên

Trọng tâm của mô hình COKB là quá trình so khớp các sự kiện để tìm ra sự hợp nhất, từ sự hợp nhất đó sẽ sinh ra các sự kiện mới phù hợp theo các quy tắc và tri thức đã định nghĩa trong mô hình. Để quá trình so khớp được thuận lợi, ta cần phân loại các sự kiện để từ đó có phương pháp so khớp thích hợp. Theo tài liệu [1] và [7], có 12 loại sự kiện được định nghĩa trong mô hình tri thức COKB.

2.2.1. Phân loại sự kiện

Sự kiện loại 1: Sự kiện thông tin về loại của một đối tượng. Cấu trúc như sau:

[<Tên đối tượng>, <loại đối tượng>]

Ví dụ: [A, "MATRAN"]

Sự kiện loại 2: Sự kiện về tính xác định của một đối tượng hay thuộc tính của đối tượng. Cấu trúc như sau:

<Đối tượng> | <Đối tượng>.<Thuộc tính>

Ví dụ: A, A.m, A.n

Sự kiện loại 3: Sự kiện xác định một đối tượng hay thuộc tính của đối tượng thông qua biểu thức hằng. Cấu trúc:

<Đối tượng> = <Biểu thức hằng>

<Đối tượng>.<thuộc tính> = <Biểu thức hằng>

Ví dụ: A.m = 4, A.n=5, với A là đối tượng tính toán biểu diễn khái niệm Ma trận, m và n lần lượt là số dòng và số cột.

Sự kiện loại 4: Sự kiện về sự bằng nhau giữa một đối tượng hay thuộc tính với một đối tượng hay thuộc tính khác. Sự bằng nhau giữa hai đối tượng cùng loại được hiểu là các thuộc tính của chúng tương ứng bằng nhau. Cấu trúc:

<Đối tượng> = <Đối tượng>

<Đối tượng>.<thuộc tính> = <Đối tượng>.<thuộc tính>

VÍ DỤ: A = B, A.m = B.m, A và B lần lượt là các đối tượng tính toán biểu diễn khái niệm Ma trận.

Sự kiện loại 5: Sự kiện về sự phụ thuộc của một đối tượng hay một thuộc tính của đối tượng thông qua một công thức tính toán hay đẳng thức theo các đối tượng hoặc thuộc tính. Cấu trúc:

<Đối tượng>|<Đối tượng>.<thuộc tính> = <biểu thức theo các đối tượng tính toán hay thuộc tính khác>

VÍ DỤ: A.m = A.n + 2, A.m = B.n + A.n, với A, B lần lượt là các đối tượng tính toán biểu diễn khái niệm Ma trận.

Sự kiện loại 6: Sự kiện về quan hệ trên các đối tượng hay các thuộc tính của đối tượng. Cấu trúc mô tả theo danh sách sau:

[<Tên quan hệ>, <đối tượng 1>, <đối tượng 2>, ...]

VÍ DỤ: [“BANGNHAU”, A, B], với A, B là các đối tượng tính toán biểu diễn khái niệm ma trận trong miền tri thức Ma trận và hệ phương trình tuyến tính.

Sự kiện loại 7: Sự kiện xác định một hàm. Cấu trúc như sau:

<Hàm>

VÍ DỤ: DINHTHUC(A), HANG(A).

Sự kiện loại 8: Sự kiện xác định một hàm thông qua một hằng hay biểu thức hằng. Cấu trúc:

<Hàm> = <Biểu thức hằng>

VÍ DỤ: DINHTHUC(A) = -18.

Sự kiện loại 9: Sự kiện về sự bằng nhau của một đối tượng và một hàm. Cấu trúc:

<Đối tượng> = <Hàm>

Sự kiện loại 10: Sự kiện về sự bằng nhau của một hàm và một hàm khác. Cấu trúc:

<Hàm> = <Hàm>

VÍ DỤ: DINHTHUC(A) = DINHTHUC(B), với A, B là đối tượng tính toán biểu diễn khái niệm về Ma trận.

Sự kiện loại 11: Sự phụ thuộc của một hàm vào một hàm khác thông qua biểu thức. Cấu trúc:

<Hàm> = <biểu thức theo các đối tượng hay hàm>

VÍ DỤ: HANG(A)= HANG(C) + 1, với A, C là các đối tượng tính toán biểu diễn khái niệm ma trận.

Sự kiện loại 12: Sự kiện về mối quan hệ giữa hàm và đối tượng khác. Cấu trúc:

[<Tên quan hệ>, <đối tượng 1>|<hàm 1>, <đối tượng 2>|<hàm 2>, ...]

VÍ DỤ: [“>”, HANG(A,C), B.n], với A, C, B lần lượt là đối tượng tính toán biểu diễn khái niệm Ma trận trong miền tri thức Ma trận và hệ phương trình tuyến tính.

2.2.2. Hợp nhất sự kiện

Định nghĩa 2.3: Cho một miền tri thức K được đặc tả theo mô hình COKB, cho trước 2 sự kiện là: f1, f2. Các sự kiện f1 và f2 thuộc 12 loại sự kiện như mô hình. Ta nói f1 và f2 “hợp nhất” (về ngữ nghĩa), ký hiệu ≅ khi:

Trường hợp 1: f1 ≡ f2.

Trường hợp 2: f1 và f2 cùng loại **K** và,
lhs(f1) ≅ **lhs**(f2) và **compute**(rhs(f1)) = **compute**(**rhs**(f2)), nếu **K = 3**;

(**lhs**(f1) ≅ **lhs**(f2) và **rhs**(f1) ≅ **rhs**(f1)) hay (**lhs**(f1) ≅ **rhs**(f2) và **rhs**(f1) ≅ **lhs**(f2)), nếu **K = 4**;

simplify(**expand**(**lhs**(f1) - **rhs**(f1) - **lhs**(f2) + **rhs**(f2))) = 0 hay
simplify(**expand**(**lhs**(f1) - **rhs**(f1) + **lhs**(f2) - **rhs**(f2))) = 0, nếu **K = 5**;

NameofRelation(f1) ≡ **NameofRelation** (f2) and **Property**(f1) ≡ "**symmetric**" and **SetofObjects**(f1) ≅ **SetofObjects**(f2) or
NameofRelation(f1) ≡ **NameofRelation**(f2) or **ListofObjects**(f1) ≅ **ListofObjects**(f2), nếu **K = 6**;

NameofFunction(f1) ≡ **NameofFunction**(f2) và "symmetric" ∈ **Property**(f1) và **ArgumentsSetofFunction**(f1) ≅ **ArgumentsSetofFunction**(f2) hay
NameofFunction(f1) ≡ **NameofFunction**(f2) và **ArgumentsListofFunction**(f1) ≅ **ArgumentsListofFunction**(f2), nếu **K = 7**;

lhs(f1) ≅ **lhs**(f2) và **compute**(rhs(f1)) = **compute**(rhs(f2)), nếu **K = 8**;

(**lhs**(f1) ≅ **lhs**(f2) và **rhs**(f1) ≅ **rhs**(f2)) hay (**rhs**(f1) ≅ **lhs**(f2) và **lhs**(f1) ≅ **rhs**(f2)), nếu **K = 9**;

(**lhs**(f1) ≅ **lhs**(f2) và **rhs**(f1) ≅ **rhs**(f2)) hay (**rhs**(f1) ≅ **lhs**(f2) và **lhs**(f1) ≅ **rhs**(f2)), nếu **K = 10**;

simplify(**expand**(**lhs**(f1) - **rhs**(f1) - **lhs**(f2) + **rhs**(f2))) = 0 or
simplify(**expand**(**lhs**(f1) - **rhs**(f1) + **lhs**(f2) - **rhs**(f2))) = 0, nếu **K = 11**;

NameofRelation(f1) ≡ **NameofRelation**(f2) và "symmetric"∈**Property**(f1) và **SetofObjects(f1)** ∪ **SetofFunctions(f1)** ≅ **SetofObjects**(f2) ∪ **SetofFunctions**(f2)
or
NameofRelation(f1) ≡ **NameofRelation**(f2) và **ListofObjects**(f1) ∪ **ListofFunctions**(f1) ≅ **ListofObjects**(f2) ∪ **ListofFunctions**(f2), nếu **K = 12**;

Ký hiệu	Giải thích
=	Phép so sánh hai sự kiện về mặt giá trị
≡	Phép so sánh hai sự kiện về mặt ký tự
≅	Phép hợp nhất hai sự kiện.

Bảng 2-1: Ký hiệu so khớp các sự kiện

Tên thủ tục	Giải thích
lhs(f)	Lấy vế trái của một biểu thức. VÍ DỤ: f := a = 2. lhs(f) = a.
rhs(f)	Lấy vế phải của biêu thức. VÍ DỤ: f:=a = 4. rhs(f) = 4.
expand(f)	Khai triển biểu thức f VÍ DỤ: f := (x+1)*(x+2). expand(f) = x^2 + 3x + 2.
simplify(f)	Thu gọn biểu thức f. VÍ DỤ: f:=((x^2)+1)-(x+1)(x-1))

	simplify(f) = 2
compute(f)	Tính toán giá trị biểu thức. VÍ DỤ: f = 2+4+6. compute(f) = 12.
NameofRelation(f)	Trả về tên quan hệ của sự kiện f, với f là sự kiện loại 6. VÍ DỤ: f:=[“SONGSONG”, a,b] NameofRelation(f) = “SONGSONG”.
Property(f)	Trả về tập tính chất của sự kiện f, với f thuộc các loại sự kiện 6, 7, 12. VÍ DỤ: f:=[“SONGSONG”, a,b] Property(f) = { đối xứng }
NameofFunction(f)	Trả về tên hàm của sự kiện f, với f là sự kiện thuộc loại 7. VÍ DỤ: f:=DINHTHUC(A). NameofFunction(f) = DINHTHUC.
SetofObjects(f)	Trả về ***tập hợp*** các đối tượng trong sự kiện f, với f thuộc loại sự kiện 6 hoặc loại sự kiện 12. VÍ DỤ: f:=[“SONGSONG”, a,b] SetofObjects(f) = { a, b }
ListofObject(f)	Trả về ***danh sách*** các đối tượng trong sự kiện f, với f thuộc loại sự kiện 6 hoặc loại sự kiện 12 VÍ DỤ: f:=[“SONGSONG”, a,b] ListofObject(f) = [a, b].

SetofFunctions(f)	Trả về ***tập hợp*** các hàm trong f. f thuộc loại sự kiện 12. VÍ DỤ: f:=[“TRUNG”, GIAODIEM(A,B), C] SetofFunctions(f) = GIAODIEM(A,B).
ListofFunctions(f)	Trả về danh sach các hàm trong f. f thuộc loại sự kiện 12. VÍ DỤ: f:=[“TRUNG”, GIAODIEM(A,B), C] ListofFunctions(f) = GIAODIEM(A,B)
ArgumentsSetofFunction(f)	Trả về tập tham số trong hàm của sự kiện f. f thuộc loại sự kiện 7. VÍ DỤ: f:= DINHTHUC(A). ArgumentsSetofFunction(f) = {A}
ArgumentsListofFunction(f)	Trả về danh sach tham số trong hàm của sự kiện f. f thuộc loại sự kiện 7. VÍ DỤ: f:= DINHTHUC(A). ArgumentsSetofFunction(f) = [A]

Bảng 2-2: Giải thích các thủ tục so khớp

VÍ DỤ: Cho f1: = A.m = A.n + 4, f2:= A.m = A.n + 3 + 1.

Ta nói f1 và f2 hợp nhất.

Định nghĩa 2.4: Giả sử cho một miền tri thức K được mô hình hóa theo mô hình COKB, và x là một sự kiện, A là một tập các sự kiện, ta có: $x \in A \Leftrightarrow \exists y \mid y \in A, y \cong x$.Từ định nghĩ “$\in$” ta có các định nghĩa về các phép toán $\cup$, $\cap$, / (or -) và các quan hệ $\subseteq$, $\subset$, $\not\subset$, , $\supseteq$, $\supset$ cũng được định nghĩa như trên lý thuyết tập hợp.

Định nghĩa 2.5: Giả sử cho tập A = {a_1, a_2, ..., a_m} và B = {b_1, b_2, ..., b_n} (với mọi a_i, b_j thuộc 12 loại sự kiện theo mô hình COKB). Ta nói A hợp nhất với B, ký hiệu A $\cong$ B khi:

i) $\forall a_i$ (i=0..m), $\exists b_k$ (k=0..n) | $b_k \cong a_i$;

ii) $\forall b_i$ (i=0..n), $\exists a_k$ (k=0...n) | a $\cong$ b;

Dựa trên định nghĩa 2.4, các phép toán $\cup$, $\cap$, / và các quan hệ $\subseteq$, $\subset$, $\not\subset$, , $\supseteq$, $\supset$ trên các sự kiện trong mô hình COKB được hiểu theo nghĩa "hợp nhất".

Ví dụ:

A = { A.m, A.n, A.K = A.m }

B = { A.K = A.m }

Suy ra: B $\subset$ A.

2.3. Tổ chức lưu trữ

2.3.1. Cấu trúc tập tin

Ứng với các thành phần theo cơ sở tri thức COKB, chúng ta lưu thành dạng tập tin với các đặc tả và các từ khoá quy ước. Hệ thống tập tin được tổ chức như sau theo tài liệu [1]:

Tập tin CONCEPTS.TXT: Lưu trữ các định danh (hay tên gọi) cho các khái niệm về các loại đối tượng Đối tượng tính toán.

Tập tin RELATIONS.TXT: Lưu trữ thông tin về các loại quan hệ khác nhau trên các loại đối tương Đối tượng tính toán.

Tập tin HIERARCHY.TXT: Lưu lại các biểu đồ Hasse thể hiện quan hệ phân cấp đặc biệt hoá giữa các loại đối tượng.

Tập tin OPERATORS.TXT: Lưu trữ các thông tin, cơ sở tri thức của thành phần toán tử trên các đối tượng Đối tượng tính toán.

Tập tin OPERATORS_DEF.TXT: Lưu trữ định nghĩa về các loại toán tử hay định nghĩa của các thủ tục tính toán phục vụ toán tử.

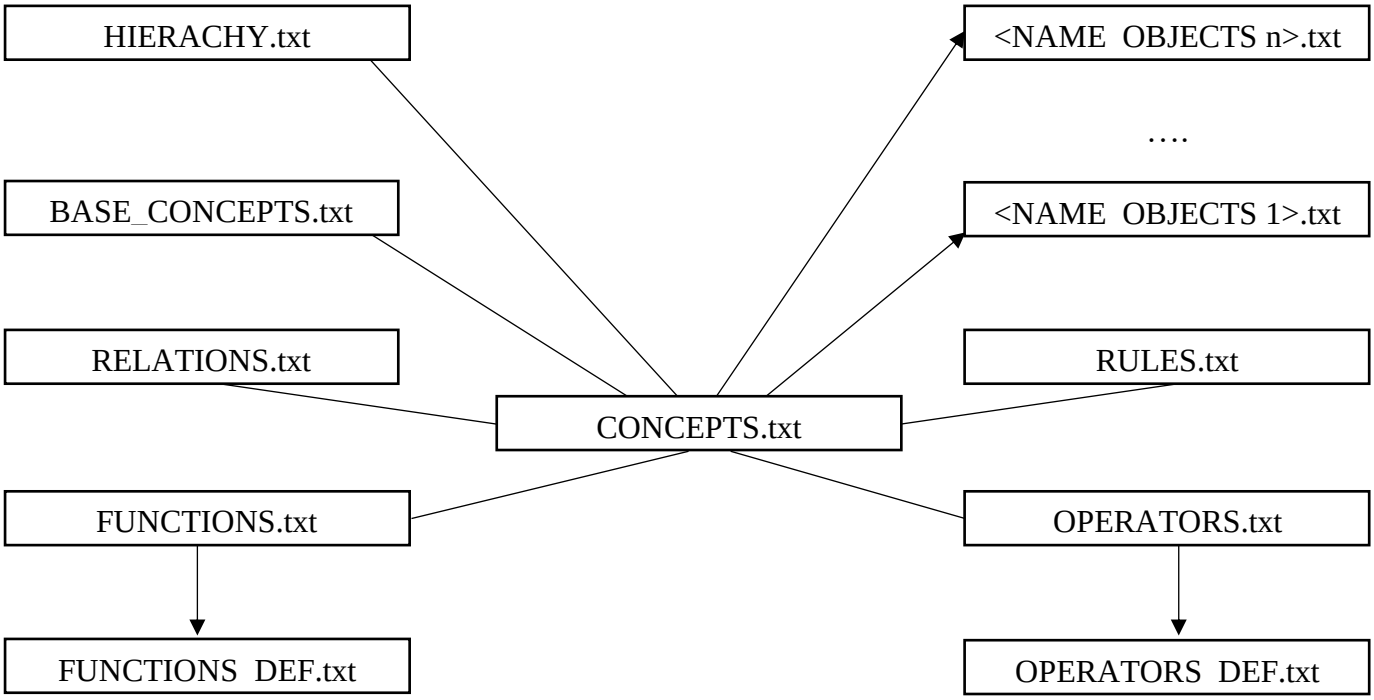
Tập tin RULES.TXT: Lưu trữ các hệ luật trên các loại đối tượng và các sự kiện (bao gồm cả sự kiện liên quan đến hàm) trong cơ sở tri thức.

Tập tin FUNCTIONS.TXT: Lưu trữ các khai báo hàm, thông tin về các hàm trên các loại đối tượng Đối tượng tính toán.

Tập tin FUNCTIONS_DEF.TXT: Lưu trữ định nghĩa về các hàm trên các đối tượng và các sự kiện.

Các tập tin có tên <TÊN KHÁI NIỆM>.TXT: Lưu trữ cấu trúc của loại đối tượng <tên khái niệm Đối tượng tính toán>.

Tập tin BASE_CONCEPTS.TXT: Lưu các khái niệm được công nhận trước trong miền tri thức.



Hình 2-1: Tổ chức lưu trữ các tập tin theo mô hình COKB

2.3.2. Đặc tả cho từng tập tin

Tập tin "**CONCEPTS.TXT**"

begin_concepts

<tên loại đối tượng 1>

<tên loại đối tượng 2>

...

end_concepts

Tập tin "**RELATIONS.TXT**"

begin_Relations

[<tên quan hệ 1>, <tên loại đối tượng 1>, <tên loại đối tượng 2>, ...]

{"<tính chất 1>","<tính chất 2>", ...}

[<tên quan hệ 2>, <tên loại đối tượng 1>, <tên loại đối tượng 2>, ...]

{"<tính chất 1>", "<tính chất 2>", ...}

...

end_Relations

Tập tin "**HIERARCHY.TXT***"*:

begin_Hierarchy

[<tên lớp đối tượng cấp cao>, <tên lớp đối tượng cấp thấp>]

[<tên lớp đối tượng cấp cao>, <tên lớp đối tượng cấp thấp>]

...

end_Hierarchy

Tập tin *"***RULES.TXT***"*

begin_rules

begin_rule

kind_rule= "<nội dung luật>"

<các tên đối tượng>: <kiểu đối tượng>;

<các tên đối tượng>: <kiểu đối tượng>;

...

hypothesis_part:

[các sự kiện giả thiết của luật]

end_hypothesis_part

goal_part:

[các sự kiện kết luận]

end_goal_part

end_rule

...

end_rules

Tập tin *"***OPERATORS.TXT***"*:

begin_operators

[<tên phép toán>, <kiểu trả về>[<tên operator 1>...] {"<tính chất 1>", "<tính chất 2>",...}

[<tên phép toán>, <kiểu trả về>[<tên operator 1>...] {"<tính chất 1>", "<tính chất 2>",...}

...

end_operators

Tập tin *"***OPERATORS_DEF.TXT***"*:

begin_operators

begin_operator: <ký hiệu toán tử>(đối tượng 1, đối tượng 2, ...)

< đối tượng 1>: <kiểu của đối số>

< đối tượng 2>: <kiểu của đối số>

...

result <đối tượng>: kiểu của đối tượng trả về

begin_proc

<statements>

end_proc

end_operator

...

end_operators

Tập tin *"***FUNCTIONS.TXT***"*:

begin_functions

<kiểu trả về của hàm 1> <tên hàm 1>(<loại của đối số 1>, <loại của đối số 2>, ...) {"<tính chất 1>", "<tính chất 2>", ...}

<kiểu trả về của hàm 2> <tên hàm 2<loại của đối số 1>, <loại của đối số 2>, ...) {"<tính chất 1>", "<tính chất 2>", ...}

...

end_functions

Tập tin *"***FUNCTIONS_DEF.TXT***"*:

begin_functions

begin_function: <tên hàm>(<đối số 1>, <đối số 2>, ...)

<các tên đối số>: <kiểu của đối số>

<các tên đối số>: <kiểu của đối số>

...

result <đối tượng>: kiểu của đối tượng trả về

begin_proc

<statements>

end_proc

properties

<fact 1>

<fact 2>

...

end_properties

end_function

...

end_functions

Các tập tin với tên tập tin có dạng "<**TÊN KHÁI NIỆM**>**.TXT**" :

begin_concept: <tên khái niệm>[<đối tượng nền 1>, <đối tượng nền 2>, ...]

<các đối tượng nền>: <kiểu đối tượng>

<các đối tượng nền>: <kiểu đối tượng>

...

begin_variables

<các tên thuộc tính>: <kiểu thuộc tính>

<các tên thuộc tính>: <kiểu thuộc tính>

...

end_variables

begin_constraints

<điều kiện ràng buộc>

<điều kiện ràng buộc>

...

end_constraints

begin_properties

<sự kiện tính chất>

<sự kiện tính chất>

...

end_properties

begin_computation_relations

begin_relation

flag = <0 hoặc 1>

Mf = {các thuộc tính}

rf = 1 // hạn của biểu thức

vf = {thuộc tính kết quả nếu flag = 0}

expr : ` biểu thức tính toán `

cost = <trọng số của sự tính toán>

end_relation

...

end_computation_relations

begin_rules

begin_rule

Kind_Rules = "<idRule>"

<tên đối tượng nền>: <kiểu đối tượng>;
<tên đối tượng nền>: <kiểu đối tượng>;
...
hypothesis_part:
<tập các sự kiện giả thiết của luật>
end_hypothesis_part
goal_part:
<tập các sự kiện kết luận>
end_goal_part
end_rule
...
end_rules
end_concept

Tập tin "**BASE_CONCEPTS.TXT**"

begin_concepts
<khái niệm công nhận trước>
<khái niệm công nhận trước>
...
end_concepts

Với các đặc tả về mô hình COKB như trong mục 2.3.2, bộ suy diễn sẽ dễ dàng đọc được các tri thức đã lưu trữ nhờ cấu trúc rõ ràng và được mô tả tường minh của nó.

CHƯƠNG 3: PHƯƠNG PHÁP SUY DIỄN VÀ CÁC LỚP BÀI TOÁN

Quá trình suy luận giải quyết vấn đề trên cơ sở tri thức COKB là quá trình suy luận sinh ra sự kiện mới dựa trên các sự kiện đã biết. Quá trình này lặp lại cho tới khi không thể sinh ra được thêm sự kiên nào nữa hoặc sự kiện đã biết chứa các sự kiện mục tiêu. Đây là cơ sở để giải quyết suy luận vấn đề trên mô hình COKB.

Để tiến hành quá trình suy diễn trên mô hình COKB, chúng ta cần phải khai thác các tính chất, sự kiện và hành vi của các đối tượng và các quan hệ giữa chúng. Có hai vấn đề chính cần giải quyết trên mô hình COKB:

Vấn đề 1: Các hành vi và quy tắc suy diễn trên một đối tượng tính toán.

Vấn đề 2: Các quy tắc suy diễn trên các thành phần của mô hình COKB.

3.1. Các thuật giải và quy tắc suy luận trên một đối tượng tính toán

3.1.1. Các quy tắc suy diễn.

Định nghĩa 3.1: Một quy tắc suy luận trên một đối tượng tính toán là cách phát sinh một sự kiện mới dựa trên tập sự kiện đã biết trước. Theo tài liệu [10], mỗi quy tắc suy luận thuộc một trong các dạng sau:

RC1*: Sinh ra các sự kiện mới từ các sự kiện đã biết trên đối tượng.*

VÍ DỤ: Khái niệm Ma trận vuông có số dòng m bằng số cột n.

RC2*: Các luật thay thế có thể sinh ra sự kiện mới bằng cách thay thế các sự kiện loại 3 trong một quan hệ tính toán*.

Chúng ta có các luật thay thế như sau:

RC2.1*: Phát sinh sự kiện loại 2 từ sự kiện loại 3.*

VÍ DỤ: m = 2 → m.

RC2.2: *Phát sinh sự kiện loại 4 bằng cách kết hợp các sự kiện loại 3 lại với nhau.*

VÍ DỤ: m = 2, n = 2 → m = n.

RC2.3: *Phát sinh sự kiện loại 3 bằng cách thay thế sự kiện loại 3 vào một sự kiện loại 4.*

VÍ DỤ: m = 4, m = n → m = 4.

RC2.4: *Phát sinh sự kiện loại 5 bằng phép thay thế sự kiện loại 3 vào sự kiện loại 5.*

VÍ DỤ: m = 3, n = m + 1 → m = 4.

RC2.5: *Phát sinh sự kiện loại 5 bằng cách thay thế cùng lúc nhiều sự kiện loại 3 vào sự kiện loại 5.*

VÍ DỤ: a = 3, c = 4, d = a + c → d = a + c = 7.

RC2.6: *Phát sinh sự kiện loại 5 bằng phép thay thế sự kiện loại 4 vào sự kiện loại 5.*

VÍ DỤ: n = k, m = 2*n + 1 → m = 2*k + 1.

RC2.7: *Phát sinh sự kiện loại 2 từ sự kiện loại 2 và sự kiện loại 4.*

VÍ DỤ: m, n = m → n.

RC2.8: *Phát sinh sự kiện loại 2 từ sự kiện loại 2 và sự kiện loại 5.*

VÍ DỤ: m, n, c = m + n → c.

RC3*: Sinh ra sự kiện mới bằng cách sử dụng luật:*

RC3.1*: Sinh ra sự kiện mới bằng cách thay thế các sự kiện loại 3 vào luật có dạng quan hệ tính toán (Luật dạng Rr).*

VÍ DỤ: Cho luật dạng quan hệ tính toán: rr: A + B + C = 180.

KnowFact = { A = 30, C = 30 }

Sau khi áp dụng luật rr, ta có: KnowFact = {A =30,C=30,*B=120*}

RC3.2: *Sinh ra sự kiện mới bằng cách áp dụng luật dẫn (Luật dạng Rf).*

RC4*: Sinh ra sự kiện mới bằng cách giải hệ phương trình.*

VÍ DỤ: f1 := a = b + 2, f2 := b = 2*a + 1. KnowFact = {}.

Sau khi giải phương trình bằng cách kết hợp f1 và f2, ta được KnowFact = { a = -3, b = -5 }

***RC5**: Sinh ra sự kiện mới dựa trên hành vi của đối tượng là thuộc tính bên trong bản thân đối tượng và các sự kiện liên quan tới nó.*

3.1.2. Các hành vi và thuật giải trên đối tượng

Dựa vào các sự kiện trong không gian sự kiện hiện tại, một đối tượng sẽ tìm cách sinh ra các sự kiện mới dựa vào hành vi và các thuộc tính bên trong nó bằng cách áp dụng các quy tắc suy diễn trong định nghĩa 3.1. Bước này gọi là bước tìm bao đóng của một đối tượng. Theo tài liệu [10] và [12], bao đóng của đối tượng được định nghĩa như sau:

Định nghĩa 3.2: Giả sử cho một đối tượng tinh toán O, và cho trước một tập sự kiện GT, GT $\subseteq$ FactSpace(O). Bao đóng tập sự kiện là sự mở rộng tối đa tập sự kiện GT dựa trên việc sử dụng các quy tắc suy luận trên đối tượng tính toán O, ký hiệu Fclosure(GT) gọi là bao đóng tập sự kiện GT. Bài toán tìm bao đóng từ một tập sự kiện GT được mô hình hóa như sau: GT $\rightarrow$ Fclosure(GT).

Từ định nghĩa về bao đóng của một đối tượng, ta xây dựng thuật giải tìm bao đóng như sau:

Thuật giải 3.1: Tìm bao đóng của đối tượng (mã giả).

KnowFact = F $\cup$ *Facts*

Flag = true;

While (Flag) do

Produce newf from KnowFact (RC1).

Find a rule rf in Rf to produce new fact (RC2)

Find a rule rr in Rr to produce new fact (RC3)

Find a system of equations eqs from the set of computational relation (RC4).

Find an object O with has level(O) > 0. (RC5).

if all step from RC1 to RC5 failed then

flag = False

else

update KnowFact

end if:

End do:

VÍ DỤ 3.1: Cho ma trận A gồm 2 dòng 2 cột có giá trị là $\begin{pmatrix} -2 & 2 \\ 3 & 4 \end{pmatrix}$. Hãy tìm bao đóng của A.

Ta có không gian sự kiện cho trước như sau:

Fact = { A.K = [[-2,2], [3,4]], A.m=2, A.n=2 }

Các bước tìm bao đóng:

Bước 1:

Từ: { A.K = [[-2,2], [3,4]], A.m=2, A.n = 2 }

Tìm được: { A.K }

(Áp dụng quy tắc sinh ra sự kiện loại 2 từ sự kiện loại 3).

Bước 2:

Từ { A.K = [[-2,2], [3,4]], A.m=2, A.n = 2, A.K }

Tìm được: { A.m }

(Áp dụng quy tắc sinh ra sự kiện loại 2 từ sự kiện loại 3).

Bước 3:

Từ { A.K = [[-2,2], [3,4]], A.m=2, A.n = 2, A.K, A.m }

Tìm được: { A.n }

(Áp dụng quy tắc sinh ra sự kiện loại 2 từ sự kiện loại 3).

Bước 4:

Từ { A.K = [[-2,2], [3,4]], A.m=2, A.n = 2, A.K, A.m, A.n }

Tìm được: { A.m = A.n }

(Áp dụng quy tắc sinh ra sự kiện loại 4 từ sự kiện loại 3).

Bao đóng cho ta các sự kiện sau: **ClosureA** = { A.K = [[-2,2], [3,4]], A.m=2, A.n = 2, A.K, A.m, A.n, A.m = A.n }.

Định nghĩa 3.3: Theo tài liệu [10] và [12], ta gọi một bước suy luận trên một Đối tượng tính toán là một bộ ba:

(rf, nf, kf)

Trong đó:

rf là một luật suy luận thuộc tập các quy tắc suy luận (RC1 ∪ RC2 ∪ RC3 ∪ RC4 ∪ RC5)

nf là tập sự kiện mới được sinh ra dựa vào rf và các sự kiện kf.

kf là tập các sự kiện cần để có thể áp dụng để áp dụng được rf;

3.2. Các lớp bài toán và phương pháp suy diễn trên mô hình COKB

3.2.1. Mô hình bài toán tổng quát và các quy tắc suy luận

Định nghĩa 3.4: Theo tài liệu [7], mô hình bài toán tổng quát trên cơ sở tri thức COKB là một bộ ba (O, F) → G. Trong đó:

O : Tập các đối tượng tính toán

F : Tập các sự kiện cho trước

G: Tập sự kiện mục tiêu.

Định nghĩa 3.5: Một quy tắc suy luận trên COKB là cách phát sinh ra một sự kiện mới từ tập sự kiện đã biết trước. Mỗi quy tắc suy luận thuộc một trong các loại suy luận sau (Theo tài liệu [7] và [10]):

RCN1*: Quy tắc tự động sinh ra các sự kiện mặc nhiên từ các sự kiện đã biết.*

Ở đây, chúng ta kế thừa các quy tắc suy luận RC2 trên mô hình đối tượng tính toán và thêm một số quy tắc như sau:

RCN1.1*: Phát sinh sự kiện loại 7 từ sự kiện loại 8.*

VÍ DỤ: DINHTHUC(A) = -4 → DINHTHUC(A).

RCN1.2: *Phát sinh sự kiện loại 3 bằng cách thay thế sự kiện loại 8 vào sự kiện loại 9.*

VÍ DỤ: DINHTHUC(A) = K, DINHTHUC(A) = -4 → K = -4.

RCN1.3: *Phát sinh sự kiện loại 8 bằng phép thay thế sự kiện loại 8 vào sự kiện loại 10.*

VÍ DỤ: DINHTHUC(A) = 4, DINHTHUC(A) = DINHTHUC(C) → DINHTHUC(C) = 4

RCN1.4: *Phát sinh sự kiện loại 11 bằng cách thế sự kiện loại 8 vào sự kiện loại 11.*

VÍ DỤ: DINHTHUC(A) = DINHTHUC(B) + 2, DINHTHUC(B) = 5 → DINHTHUC(A) = 7.

RCN1.5: *Phát sinh sự kiện loại 8 bằng cách thế cùng lúc nhiều sự kiện loại 8 vào sự kiện loại 11.*

VÍ DỤ: DINTHUC(A) = 3, DINHTHUC(B) = 4, DINHTHUC(C) = DINHTHUC(A) + DINHTHUC(B) → DINHTHUC(C) = 7.

RCN1.6: *Phát sinh sự kiện loại 8 bằng cách thay thế cùng lúc nhiều sự kiện loại 8 và loại 3 vào sự kiện loại 11.*

VÍ DỤ: DINHTHUC(A) = A.m + 2*DINHTHUC(B), A.m = 2, DINHTHUC(B) = 3 → DINHTHUC(A) = 8.

RCN1.7: *Phát sinh sự kiện loại 7 từ sự kiện loại 7 khác và sự kiện loại 10.*

RCN1.8: *Phát sinh sự kiện loại 2 (hoặc sự kiện loại 7) từ các sự kiện loại 2, 7 và 11.*

RCN2*: Quy tắc sinh ra sự kiện mới dựa trên việc áp dụng luật dẫn.*

RCN3*: Quy tắc sinh ra sự kiện mới dựa trên hành vi đối tượng.*

RCN4*.: Quy tắc sinh ra sự kiện mới dựa trên việc thực hiện các hàm*

RCN5*: Quy tắc sinh ra sự kiện mới dựa trên việc giải một hệ phương trình.*

RCN6*: Quy tắc sinh ra sự kiện mới dựa trên việc áp dụng một luật dẫn có phát sinh đối tượng.*

3.2.2. Các lớp bài toán trên mô hình COKB

Theo mô hình bài toán tổng quát như trong định nghĩa 3.3, nếu các sự kiện G trong lớp bài toán trên thuộc loại 1 hoặc 2, chúng ta có lớp bài toán xác định một đối tượng hoặc một thuộc tính của đối tượng. Nếu các sự kiện trong G thuộc loại 4 hoặc 6, chúng ta có lớp bài toán chứng minh một quan hệ tính toán giữa các đối tượng hay thuộc tính với nhau. Hai lớp bài toán trên đã được giải quyết thành công trong luận văn [10] đối với miền tri thức Hình học phẳng. Thuật giải giải quyết lớp bài toán trên sẽ được trình bày dưới đây (thuật giải này còn gọi là thuật giải suy diễn tổng quát):

Thuật giải 3.2: Thuật giải suy diễn tổng quát:

KnowFact := F;

Sol:= []

Flag := true;

While (Flag and not G ⊆ KnowFacts) do

Produce newf from kfacts ⊆ KnowFacts (RCN1)

If (newf != empty) then

KnowFacts := KnowFacts ∪ newf

Add solving step ["Auto deduce facts", facts, newf] to Sol

Find a rule rr in Rr to produce new fact (RCN2)

If (rr found) then

KnowFacts : = KnowFacts ∪ gr(rr)

Add solving step [rr, hr(rr), gr(rr)] to Sol

Find an object o which level(o) > 0 (RCN3)

If (o found) then

Find a solution for problem H→G reduced to object o

Update KnowFacts and Sol

Find a function f from the set of functions (RCN4)

If (f found) then

Solve problem H→G reduced to applying fucntion f

Update KnowFacts and Sol

Find a system of equation eqs from the set of computational relations (RCN5)

If (eqs found) then

kfacts := setvars(eqs) ∩ KnowFacts

varnews := solve(eqs, newfacts);

KnowFacts := KnowFacts ∪ varnews

Add solving step [eqs, kfacts, varnews] to Sol

Find a rules to create new objects and relates facts (RCN6)

If (r found) then

KnowFacts := KnowFacts ∪ gr(r)

Add solving step [r, hr(r), gr(r)] to Sol

If (RCN1 to RCN6) are failed then

Flag := false;

End do

VÍ DỤ 3.2: Cho ma trận A gồm 4 dòng và 4 cột như sau: $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}$. Hãy chứng minh Ma trận A vuông.

Mô hình bài toán:

O = { [A, "MATRAN"] }

F = $\left\{ \mathrm{A.m} = 4, \mathrm{A.n} = 4, \mathrm{A.K} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix} \right\}$

G = { [A, "MATRANVUONG"] }

Các bước giải:

Bước 1:

Từ: $\left\{ \mathrm{A.m} = 4, \mathrm{A.n} = 4, \mathrm{A.K} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix} \right\}$

Tìm được: {A.m, A.n, A.K }

(Áp dụng quy tắc RCN1)

Bước 2:

Từ: { A.m = 4, A.n = 4, A.m, A.n }

Tìm được: { A.m = A.n }

(Áp dụng quy tắc RCN3 trên đối tượng [A,"MATRAN"])

Bước 3:

Từ: { A.m = A.n, A.m, A.n }

Tìm được: { [A, "MATRANVUONG"] }

(Áp dụng quy tắc RCN6 trên luật xác định ma trận vuông)

Đối với lớp bài toán trên, trong một số trường hợp ta thường gặp các sự kiện có dạng: *<đối tượng 1> = <đối tượng 2> + <đối tượng 3>*. Để sinh ra các sự kiện mới từ sự kiện có dạng như trên, ta phải dùng tới các toán tử đã được đặc tả và lưu trữ trong Cơ

sở tri thức lên các đối tượng để thực thi sinh ra sự kiện mới. Tuy nhiên, các quy tắc suy diễn hiện có theo định nghĩa 3.5 chưa có quy tắc nào về suy diễn trên thành phần toán tử. Do đó để thực thi giải quyết bài toán với các sự kiện như trên, ta bổ sung vô một quy tắc mới như sau:

*Quy tắc **RCN7**: Quy tắc sinh ra một sự kiện mới dựa trên việc áp dụng toán tử.*

Đối với quy tắc *RCN7,* chúng ta có các bước thực hiện việc áp dụng một toán tử như sau:

- *Bước 1:* Xét các sự kiện loại 5 trong tập sự kiện, vì đây là các sự kiện mô tả mối quan hệ giữa các đối tượng với nhau thông qua biểu thức gồm các toán hạng liên kết bởi các toán tử.
- *Bước 2:* Tìm trong cơ sở tri thức các lưu trữ về đặc tả toán tử.
- *Bước 3:* Xét xem các tham số truyền vào toán tử có thoả mãn như đặc tả toán tử trong cơ sở tri thức hay không.
- *Bước 4:* Nếu tham số truyền vào thoả mãn, tiến hành thực thi toán tử dựa vào các sự kiện đã có trong tập sự kiện.
- *Bước 5:* Trả về gía trị thực thi của toán tử trên các đối tượng tính toán.

Thuật giải 3.3: Thuật giải áp dụng một toán tử

```
KnowFact := F;
Sol := [];
Foreach f in KnowFact do
    If (kind of Fact f is 5) then
            kfact = KnowFact
            Find the operator o in f
            If (o found in Knowledge Base and set_of_argument (o) are satisfied) then
                    Executing the statements that has defined in o by facts in kfact
```

End if

If (o was executed) then

Produce newf from executing o

Update newf to KnowFact;

Add solving step ["Applying operator o: ", newf, kfact] to Sol;

End if

End if;

End foreach

Sau khi bổ sung quy tắc suy diễn RCN7, thuật giải suy diễn tổng quát (thuật giải 3.2) sẽ được bổ sung và cập nhật lại như sau:

Thuật giải 3.4: Thuật giải suy diễn tổng quát cập nhật:

KnowFact := F;

Sol:= []

Flag := true;

While (Flag and not G ⊆ KnowFacts) do

Produce newf from kfacts ⊆ KnowFacts (RCN1)

If (newf != empty) then

KnowFacts := KnowFacts ∪ newf

Add solving step ["Auto deduce facts", facts, newf] to Sol

Find a rule rr in Rr to produce new fact (RCN2)

If (rr found) then

KnowFacts : = KnowFacts ∪ gr(rr)

Add solving step [rr, hr(rr), gr(rr)] to Sol

Find an object o which level(o) > 0 (RCN3)

If (o found) then

Find a solution for problem H→G reduced to object o

Update KnowFacts and Sol

Find a function f from the set of functions (RCN4)

If (f found) then

Solve problem H→G reduced to applying a fucntion f

Update KnowFacts and Sol

Find a system of equation eqs from the set of computational relations (RCN5)

If (eqs found) then

kfacts := setvars(eqs) ∩ KnowFacts

varnews := solve(eqs, newfacts);

KnowFacts := KnowFacts ∪ varnews

Add solving step [eqs, kfacts, varnews] to Sol

Find a rules to create new objects and relates facts (RCN6)

If (r found) then

KnowFacts := KnowFacts ∪ gr(r)

Add solving step [r, hr(r), gr(r)] to Sol

Find an operator op from the set of operators (RCN7)

If (op found) then

Solve problem H→G reduced to applying an operator op

Update KnowFacts and Sol

If (RCN1 to RCN7) are failed then

Flag := false;

End do

VÍ DỤ 3.3: Cho ma trận A = $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}$ và ma trận B = $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}$.

Biết C = A + B và E = A + C. Tìm giá trị của E.

Mô hình hoá bài toán:

O = { [A, “MATRAN”], [B, “MATRAN”], [C, “MATRAN”], [D, “MATRAN”] }

$$F = \left\{ A.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, B.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, C = A + B, E = A + C \right\}$$

G = { E.K }

Lời giải:

Bước 1:

Từ: $\left\{ A.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, B.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix} \right\}$

Tìm được: { A.m = 3, A.n = 4, B.m = 3, B.n = 4 }

(Áp dụng quy tắc RCN3 trên các đối tượng A, B)

Bước 2:

Từ: $\left\{ A.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, A.m = 3, A.n = 4, B.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, B.m = 3, B.n = 4, C = A + B \right\}$

Tìm được: $\left\{ C.K = \begin{pmatrix} 2 & 4 & 6 & 8 \\ 8 & 6 & 10 & 14 \\ 10 & 12 & 16 & 12 \end{pmatrix}, C.m = 3, C.n = 4 \right\}$

(Áp dụng quy tắc RCN7 trên toán tử + giữa A và B trên sự kiện C = A+B)

Bước 3:

Từ: $\left\{ A.K = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 5 & 7 \\ 5 & 6 & 8 & 6 \end{pmatrix}, A.m = 3, A.n = 4, C.K = \begin{pmatrix} 2 & 4 & 6 & 8 \\ 8 & 6 & 10 & 14 \\ 10 & 12 & 16 & 12 \end{pmatrix}, C.m = 3, C.n = 4, E = A + C \right\}$

Tìm được: $\left\{ \mathrm{E.K} = \begin{pmatrix} 3 & 6 & 9 & 12 \\ 12 & 9 & 15 & 21 \\ 15 & 18 & 24 & 18 \end{pmatrix}, \mathrm{E.m} = 3, \mathrm{E.n} = 4 \right\}$

(Áp dụng quy tắc RCN7 trên toán tử + giữa A và C trên sự kiện E = A + C)

Mặt khác, đối với các bài toán có dạng (O,F) → G trong đó G thuộc sự kiện loại 7 – sự kiện xác định hàm, chúng ta có lớp bài toán xác định giá trị một hàm. Với lớp bài toán này, chúng ta cần áp dụng quy tắc suy diễn *RCN4* (quy tắc suy diễn áp dụng một hàm) để sinh ra các sự kiện mới bằng cách sử dụng các hàm được đặc tả trong cơ sở tri thức. Tuy nhiên, quy tắc *RCN4* trong các đề tài trước đó, cụ thể là luận văn [10] chưa được giải quyết trọn vẹn. Để giải được lớp bài toán trên ta cần định nghĩa các bước xử lý cho quy tắc suy diễn *RCN4*

Khi gặp lớp bài toán xác định giá trị một hàm có 2 trường hợp xảy ra như sau:

- Hàm được xác định và được thực thi, trả về kết quả là giá trị của hàm (có thể phát sinh thêm các sự kiện).
- Hàm được xác định nhưng chưa thực thi do không đủ các điều kiện cần thiết.

Đối với lớp bài toán trên, ta cần thêm một thủ tục là xác định mục tiêu bài toán có thỏa hay không nhằm tránh trường hợp Hàm có xác định nhưng không được thi thi. Cách xác định mục tiêu như sau:

- *Bước 1:* Xét các sự kiện loại 7 trong mục tiêu.
- *Bước 2:* Đối với mỗi sự kiện mục tiêu loại 7, tìm xem trong tập không gian trạng thái có sự kiện mục tiêu loại 7 đó hay không. Nếu không thì kết luận mục tiêu không thỏa mãn.
- *Bước 3:* Nếu xuất hiện sự kiện mục tiêu loại 7 trong không gian trạng thái thì tìm trong không gian trạng thái đó xem có sự kiện loại 8 mà vế trái của nó

"hợp nhất" sự kiện mục tiêu loại 7 đang xét hay không. Nếu hợp nhất thì kết luận mục tiêu thỏa mãn, không thì kết luận mục tiêu không thỏa mãn.

Từ các bước xác định mục tiêu cho lớp bài toán xác định giá trị một hàm ở trên, thuật giải xác định mục tiêu cho lớp bài toán xác định giá trị một hàm được trình bày như sau:

Thuật giải 3.5: Thuật giải xác định mục tiêu cho lớp bài toán xác định hàm

KnowFact := <FactSpace>;

G := goal_facts_type_7;

For each fact in KnowFact do

If (fact is kind 8 and ***Unify****(lhs(fact), G) = true) then*

Return true;

End if

End foreach

Return false;

Ghi chú: **Unify** là hàm kiểm tra xem 2 sự kiện có hợp nhất hay không. Các kỹ thuật kiểm tra hợp nhất giữa 2 sự kiện được trình bày trong mục 2.2.2

Để giải quyết lớp bài toán trên, chúng ta cần bổ sung một số quy tắc suy luận mới trong định nghĩa 3.7 như sau:

Đối với quy tắc suy luận *RCN1* trong thuật giải suy diễn tổng quát, ta bổ sung thêm các quy tắc suy diễn cụ thể như sau:

RCN1.9*: Phát sinh sự kiện loại 9 từ các sự kiện loại 3 và sự kiện loại 8.*

VÍ DỤ: HANG(A) = 3, A.n = 3 → HANG(A)=A.n.

RCN1.10: *Phát sinh sự kiện loại 10 từ sự kiện loại 9*

VÍ DỤ: HANG(A) = 3, HANG(B) = 3 → HANG(A) = HANG(B).

RCN1.11: *Phát sinh sự kiện loại 6 từ các sự kiện loại 3.*

VÍ DỤ: A.n = 3, B.n = 4 → [“<”, A.n, B.n]

RCN1.12: *Phát sinh sự kiện loại 12 từ sự kiện loại 3 và sự kiện loại 8.*

VÍ DỤ: HANG(A) = 3, A.n = 4 → [“<”, HANG(A), A.n]

RCN1.13: *Phát sinh sự kiện loại 8 từ sự kiện loại 9 và sự kiện loại 8 khác*

VÍ DỤ: HANG(A) = HANG(B), HANG(A) = 3 → HANG(B) = 3.

Đối với quy tắc RCN4: Quy tắc RCN4 sẽ thực thi một hàm, do đó chúng ta sẽ có các bước cụ thể khi thực thi một hàm:

- *Bước 1:* Xét các sự kiện loại 7 trong tập sự kiện, vì đây là các sự kiện phát sinh hàm.
- *Bước 2:* Tìm trong Cơ sở tri thức các lưu trữ về đặc tả hàm.
- *Bước 3:* Xét xem các tham số truyền vào hàm có thoả mãn như đặc tả hàm trong cơ sở tri thức hay không.
- *Bước 4:* Nếu tham số truyền vào thoả mãn, tiến hành thực thi hàm dựa vào các sự kiện đã có trong tập sự kiện.
- *Bước 5:* Trả về giá trị hoặc phát sinh sự kiện mới và lưu vào tập sự kiện đã biết.

Thuật giải 3.6: Thuật giải áp dụng một hàm (áp dụng cho quy tắc suy diễn RCN4)

```
KnowFact := F;
Sol := [];
Foreach f in KnowFact do
    If (kind of Fact f is 7 and f was found in the Knowledge Base) then
        kfact = KnowFact
        If (set_of_argument (f) are satisfied) then
            Executing the statements that has defined in f by facts in kfact
        End if
```

If (f was executed) then

Produce newf from executing f

Update newf to KnowFact;

Add solving step ["Applying function f", newf, kfact]

End if

End if;

End foreach

VÍ DỤ 3.4: Cho ma trận A gồm 4 dòng và 4 cột như sau:

A = $\begin{pmatrix} 1 & -1 & 5 & -1 \\ 1 & 1 & -2 & 3 \\ 3 & -1 & 8 & 1 \\ 1 & 3 & -9 & 7 \end{pmatrix}$. Hãy tìm hạng của ma trận A.

Ta mô hình hoá bài toán trên như sau:

O = { [A, "MATRAN"]}

F = $\left\{ \text{A.K} = \begin{pmatrix} 1 & -1 & 5 & -1 \\ 1 & 1 & -2 & 3 \\ 3 & -1 & 8 & 1 \\ 1 & 3 & -9 & 7 \end{pmatrix}, \text{A.m} = 4, \text{A.n} = 4 \right\}$

G = { HANG(A) }

Các bước giải:

Bước 1:

Từ: $\left\{ \text{A.K} = \begin{pmatrix} 1 & -1 & 5 & -1 \\ 1 & 1 & -2 & 3 \\ 3 & -1 & 8 & 1 \\ 1 & 3 & -9 & 7 \end{pmatrix}, \text{A.m} = 4, \text{A.n} = 4 \right\}$

Tìm được: { A.K, A.m, A.n }

(Áp dụng quy tắc RCN1)

Bước 2:

Từ: { A.K, A.m, A.n, HANG(A) }

Tìm được: { BDSOCAP(A) }

(Áp dụng quy tắc RCN4 trên hàm HANG(A) trên đối tượng tính toán A)

Bước 3:

$$\text{Từ: } \left\{ \text{A.m} = 4, \text{A.n} = 4, \text{A.K} = \begin{pmatrix} 1 & -1 & 5 & -1 \\ 1 & 1 & -2 & 3 \\ 3 & -1 & 8 & 1 \\ 1 & 3 & -9 & 7 \end{pmatrix}, \text{BDSOCAP(A)} \right\}$$

$$\text{Tìm được: } \left\{ \text{BDSOCAP(A)} = \begin{pmatrix} 1 & 0 & 3/2 & 1 \\ 0 & 1 & -7/2 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right\}$$

(Áp dụng quy tắc RCN4 trên hàm BDSOCAP(A) trên đối tượng tính toán A)

Bước 4:

$$\text{Từ: } \left\{ \text{A.m} = 4, \text{A.n} = 4, \text{BDSOCAP(A)} = \begin{pmatrix} 1 & 0 & 3/2 & 1 \\ 0 & 1 & -7/2 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right\}$$

Tìm được: { HANG(A) = 2 }

(Áp dụng quy tắc RCN4 trên hàm HANG(A) trên đối tượng tính toán A)

Ngoài ra, trong thực tế ta còn gặp một số lớp bài toán có dạng biện luận tham số, đó là các bài toán mà giá trị của sự kiện mục tiêu sẽ khác nhau tùy theo giá trị của tham số. Do đó, với lớp bài toán này, mô hình bài toán sẽ được mở rộng từ mô hình bài toán tổng quát như sau: **(O, P, F) → G**. Trong đó:

O : Tập các đối tượng tính toán.

P: Tập các tham số.

F : Tập các sự kiện cho trước.

G: Tập sự kiện mục tiêu.

Giá trị của sự kiện mục tiêu trong G sẽ thay đổi tùy theo giá trị của tham số truyền vào. Do đó, lớp bài toán này cần thêm một thủ tục nhằm xác định các giá trị khả dụng của các tham số. Tuy nhiên, không có một quy tắc nào chung đối với việc xác định một giá trị khả dụng của tham số trong P. Đối với mỗi bài toán khác nhau ta cần định nghĩa cách xác định tham số khác nhau.

Một các tổng quát, để giải lớp bài toán biện luận tham số, ta thực hiện như sau:

- *Bước 1:* Tìm sự kiện sinh ra giá trị tham số cần thiết.
- *Bước 2:* Tiến hành tìm các giá trị khả dụng của tham số bằng cách giải các phương trình, bất phương trình hay hệ phương trình.
- *Bước 3:* Với mỗi giá trị v tìm được của các tham số:
    - *Bước 3.1:* Thay thế tham số trong các sự kiện cho trước ban đầu bởi giá trị v. Ta thu được tập sự kiện F' là các sự kiện ban đầu đã được thay thế tham số bằng giá trị v.
    - *Bước 3.2:* Giải bài toán ***(O,F')→G***, ứng với mỗi giá trị v của tham số ta tìm được giá trị của mục tiêu G khác nhau.

Thuật giải 3.7: Thuật giải tổng quát giải bài toán biện luận tham số (O,P,F) → G.

KnowFact:= F;
P = { set_of_parameter };
Param_value := []; *//Danh sách các giá trị khả dụng của các tham số*
Flag:= true;
While (Flag) do
    Using RCN1 to RCN7 to create fact f that contain the param p ∈ P.
    Solving the equation or inequaltion to find the value vp of p from f.
    Push vp to Param_value;

If (RCN1 to RCN7 fail or not create any new vp)

Flag:=false;

End if

End do:

For each v in Param_value do

NewFact := substitute v to facts in KnowFact

Solve problem ***(O,NewFact) → G.***

End do:

VÍ DỤ 3.5: Cho hệ phương trình J như sau $\left(\begin{array}{ccc|c} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{array}\right)$. Hãy biện luận nghiệm của hệ phương trình theo m.

Mô hình hoá bài toán:

O = { [A, "MATRAN"], [B, "MATRAN"], [J[A,B], "HEPHUONGTRINH"]}

P = { m }

F = { A.K = $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & m \\ 1 & m & 1 \\ m & 1 & 1 \end{pmatrix}$, B.K = (m,1,1,1) }

G = { NGHIEM(J) }

Lời giải:

Bước 1:

Từ: $\left\{ \text{A.K} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & m \\ 1 & m & 1 \\ m & 1 & 1 \end{pmatrix}, \text{B.K} = (\text{m}, 1,1,1) \right\}$

Tìm được: { A.m = 4, A.n = 3, B.m = 1, B.n = 4, HANG(A }

(Áp dụng quy tắc RCN3 trên các đối tượng A, B)

Bước 2:

Từ: {HANG(A), A.m = 4, A.n = 3 }

Tìm được: {BDSOCAP(A)}

(Áp dụng quy tắc RCN4 trên hàm HANG(A))

Bước 3:

$$\text{Từ:}\left\{\begin{matrix}\mathrm{A.K} = \begin{pmatrix}1 & 1 & 1\\ 1 & 1 & m\\ 1 & m & 1\\ m & 1 & 1\end{pmatrix}, \mathrm{B.K} = (\mathrm{m},1,1,1), \mathrm{A.m} = 4, \mathrm{A.n} = 3, \mathrm{B.m} = \\ 1, \mathrm{B.n} = 4\end{matrix}\right\}$$

Tìm được: { J.Q.K = MORONG(A,B), HANG(J.Q.K) }

(Áp dụng quy tắc RCN3 trên đối tượng J[A,B])

Bước 4:

Từ: { J.Q.K = MORONG(A,B) }

Tìm được: { MORONG(A,B) }

(Áp dụng quy tắc RCN1)

Bước 5:

$$\text{Từ:}\left\{\begin{matrix}\mathrm{MORONG(A,B)}, \mathrm{A.K} = \begin{pmatrix}1 & 1 & 1\\ 1 & 1 & m\\ 1 & m & 1\\ m & 1 & 1\end{pmatrix}, \mathrm{B.K} = (\mathrm{m},1,1,1), \mathrm{A.m} = \\ 4, \mathrm{A.n} = 3, \mathrm{B.m} = 1, \mathrm{B.n} = 4,\end{matrix}\right\}$$

Tìm được: $\left\{\text{MORONG(A, B)} = \begin{pmatrix} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{pmatrix}\right\}$

(Áp dụng quy tắc RCN4 trên hàm MORONG(A,B) trên 2 đối tượng tính toán A, B).

Bước 6:

Từ: $\left\{\text{MORONG(A, B)} = \begin{pmatrix} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{pmatrix}, \text{J. Q. K} = \text{MORONG(A, B)}\right\}$

Tìm được: $\left\{\text{J. Q. K} = \begin{pmatrix} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{pmatrix}, J.Q.n = 4, J.Q.m = 4\right\}$

(Áp dụng quy tắc RCN1)

Bước 7:

Từ: $\left\{\text{J. Q. K} = \begin{pmatrix} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{pmatrix}, \text{HANG(J. Q)}\right\}$

Tìm được: {BDSOCAP(J.Q)}

(Áp dụng quy tắc RCN4 trên hàm HANG(J.Q) trên đối tượng J.Q)

Bước 8:

Từ: $\left\{\text{J. Q. K} = \begin{pmatrix} 1 & 1 & 1 & m \\ 1 & 1 & m & 1 \\ 1 & m & 1 & 1 \\ m & 1 & 1 & 1 \end{pmatrix}, J.Q.n = 4, J.Q.m = 4\right\}$

Tìm được: $\left\{\text{BDSOCAP(J. Q)} = \begin{pmatrix} 1 & 1 & 1 & m \\ 0 & m-1 & 0 & 1-m \\ 0 & 0 & m-1 & 1-m \\ 0 & 0 & 0 & -m\text{\^{}}2-2m+3 \end{pmatrix}\right\}$

(Áp dụng quy tắc RCN4 trên hàm BDSOCAP(J.Q) trên đối tượng J.Q)

Bước 9: Với giá trị thu được từ BDSOCAP(W.Q), ta tiến hành biện luận giá trị nghiệm của m bằng cách xét hai trường hợp:

$-m^2 - 2m + 3 = 0$ (1)

$-m^2 - 2m + 3 \neq 0$ (2)

Từ (1) và (2), ta kết luận các giá trị khả dụng của biến tham số m là: [m=-3, m=1, $-3 \neq m \neq 1$].

Với $m = -3$, F'_1 = { A.K = $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -3 \\ 1 & -3 & 1 \\ -3 & 1 & 1 \end{pmatrix}$, B.K = (-3,1,1,1) }. Khi giải bài toán $(O,F'_1) \rightarrow G$ ta được kết quả NGHIEM(J) = [-1, -1, -1, 0].

Với $m = -3$, F'_2 = { A.K = $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$, B.K = (1,1,1,1) }. Khi giải bài toán $(O,F'_2) \rightarrow G$ ta được kết quả NGHIEM(J) =”Vo so nghiem”.

Với $-3 \neq m \neq 1$, ta chọn $m = -1 \neq \{-3, 1\}$, F'_3 = { A.K = $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{pmatrix}$, B.K = (-1,1,1,1) }. Khi giải bài toán $(O,F'_3) \rightarrow G$ ta được kết quả NGHIEM(J) = “Vo nghiem”.

3.2.3. Lời giải cho bài toán trên mô hình COKB

Một lời giải cho bài toán $H \rightarrow G$ trên mô hình tri thức COKB là một tập hợp $D = [d_1, d_2, ..., d_m]$ gồm các bước suy luận sinh ra các sự kiện mới bằng cách áp dụng các quy tắc suy luận. Từ đó ta có định nghĩa về lời giải cho bài toán $H \rightarrow G$ trên mô hình COKB như sau:

Định nghĩa 3.6: Một bước suy luận sinh ra sự kiện mới $d_i \in D$ (i=0…m) là một bộ ba như sau:

(rf, nf, kf)

Trong đó:

rf là một dãy các quy tắc suy luận (*RCN1* ∪ *RCN2* ∪ *RCN3* ∪ *RCN4* ∪ *RCN5* ∪ *RCN6* ∪ *RCN7*)

nf là tập sự kiện mới được sinh ra dựa vào rf và các sự kiện kf.

kf là tập các sự kiện cần để có thể áp dụng để áp dụng được rf

Trong quá trình thực hiện suy diễn giải bài toán bởi thuật giải suy diễn tổng quát không tránh khỏi việc sinh ra các bước giải thừa. Do đó, để lời giải của bài toán gọn và tối ưu, ta phải xóa bỏ các bước suy diễn thừa, không cần thiết. Theo các tài liệu [7] và [10] ta có định nghĩa về một lời giải tốt hay lời giải không có bước thừa như sau:

Định nghĩa 3.7: Ta giả sử $D = [d_1, d_2, ..., d_m]$ là một lời giải của bài toán $H \rightarrow G$, ta nói D là một lời giải không có bước thừa khi và chỉ khi:

D là một lời giải của bài toán $H \rightarrow G$.

Not $\exists d_i$ (i=0..m) sao cho D loại bỏ đi là một lời giải của bài toán $H \rightarrow G$.

Từ định nghĩa trên, chúng ta xây dựng một thuật giải nhằm loại bỏ các bước giải thừa không cần thiết nhằm tối ưu lời giải.

Thuật giải 3.8: Loại bỏ lời giải thừa (theo [7]):

NewVar := G;

D':=[]

For i = m downto 1 do

If nf(d[i]) ∩ NewVar != empty then

NewVar := (NewVar – nf(d[i])) ∪ kf(d[i])

Insert d[i] at the first of D'

End do:

Sau khi áp dụng thuật giải 3.4 trên lời giải D = [d_1, d_2, ..., d_m], ta thu được lời giải D' ⊆ D, với D' = [d'_1, d'_2, … d'_k], với k ≤ m là lời giải tối ưu của bài toán H → G.

3.2.4. Một số quy tắc heuristics

Sau khi bổ sung thêm một vài quy tắc suy diễn mới, tổng kết lại chúng ta có các quy tắc suy diễn trên mô hình tri thức COKB như sau:

RCN1*: Quy tắc tự động sinh ra các sự kiện mặc nhiên từ các sự kiện đã biết.*

Ở đây, chúng ta kế thừa các quy tắc suy luận RC2 trên mô hình đối tượng tính toán và thêm một số quy tắc như sau:

RCN1.1*: Phát sinh sự kiện loại 7 từ sự kiện loại 8.*

RCN1.2: *Phát sinh sự kiện loại 3 bằng cách thay thế sự kiện loại 8 vào sự kiện loại 9.*

RCN1.3: *Phát sinh sự kiện loại 8 bằng phép thay thế sự kiện loại 8 vào sự kiện loại 10.*

RCN1.4: *Phát sinh sự kiện loại 11 bằng cách thế sự kiện loại 8 vào sự kiện loại 11.*

RCN1.5: *Phát sinh sự kiện loại 8 bằng cách thế cùng lúc nhiều sự kiện loại 8 vào sự kiện loại 11.*

RCN1.6: *Phát sinh sự kiện loại 8 bằng cách thay thế cùng lúc nhiều sự kiện loại 8 và loại 3 vào sự kiện loại 11.*

RCN1.7: *Phát sinh sự kiện loại 7 từ sự kiện loại 7 khác và sự kiện loại 10.*

RCN1.8: *Phát sinh sự kiện loại 2 (hoặc sự kiện loại 7) từ các sự kiện loại 2, 7 và 11.*

RCN1.9*: Phát sinh sự kiện loại 9 từ các sự kiện loại 3 và sự kiện loại 8.*

RCN1.10: *Phát sinh sự kiện loại 10 từ sự kiện loại 9*

RCN1.11: *Phát sinh sự kiện loại 6 từ các sự kiện loại 3.*

RCN1.12: *Phát sinh sự kiện loại 12 từ sự kiện loại 3 và sự kiện loại 8.*

RCN1.13: *Phát sinh sự kiện loại 8 từ sự kiện loại 9 và sự kiện loại 8 khác*

RCN2*: Quy tắc sinh ra sự kiện mới dựa trên việc áp dụng luật dẫn.*

RCN3*: Quy tắc sinh ra sự kiện mới dựa trên hành vi đối tượng.*

RCN4*.: Quy tắc sinh ra sự kiện mới dựa trên việc thực hiện các hàm*

RCN5*: Quy tắc sinh ra sự kiện mới dựa trên việc giải một hệ phương trình.*

RCN6*: Quy tắc sinh ra sự kiện mới dựa trên việc áp dụng một luật dẫn có phát sinh đối tượng.*

RCN7*: Quy tắc sinh ra một sự kiện mới dựa trên việc áp dụng toán tử.*

Thuật giải suy diễn tổng quát sẽ lần lượt áp dụng các quy tắc trên để sinh ra sự kiện mới từ các sự kiện đã biết. Về cơ bản thuật giải suy diễn tổng quát đã giải quyết được các lớp bài toán được đưa vào, tuy nhiên trong một số trường hợp sẽ có các quy tắc suy diễn không cần thiết được thực thi làm cho lời giải bài toán bị dư thừa và quá trình thực thi bị kéo dài hơn. Do đó ta cần định nghĩa một số quy tắc heuristic nhằm làm giảm đi các bước suy diễn không cần thiết. Các quy tắc này được trình bày như sau:

Quy tắc 1: Luôn ưu tiên quy tắc RCN1 và RCN3 trên mọi lớp bài toán vì đây là 2 quy tắc sinh ra các sự kiện mặc nhiên đã biết tự động.

Quy tắc 2: Đối với lớp bài toán mà các sự kiện mục tiêu thuộc loại 2, loại 4, ta ưu tiên sử dụng các quy tắc suy diễn RCN4 trên các sự kiện loại 8 vì đây là các sự kiện

giúp ta phát sinh ra các sự kiện loại 3, từ đó chúng ta sẽ tìm được các sự kiện loại 2 và 4.

Quy tắc 3: Đối với lớp bài toán mà các sự kiện mục tiêu thuộc loại 7, ta ưu tiên áp dụng quy tắc suy diễn RCN4 trước vì đây là các sự kiện phát sinh một hàm.

Quy tắc 4: Đối với quy tắc suy diễn RCN6, ta chỉ áp dụng khi cần phát sinh một đối tượng (phát sinh sự kiện loại 1).

Quy tắc 5: Đối với các bài toán biện luận tham số, ta sẽ sử dụng lại các sự kiện đã tìm được trước đó trong quá trình đi tìm giá trị của tham số. Các sự kiện sẽ được sử dụng lại bao gồm các sự kiện loại 2 và loại 7, các sự kiện loại 3 và loại 8 với điều kiện không phải là giá trị hằng (có tham số trong giá trị).

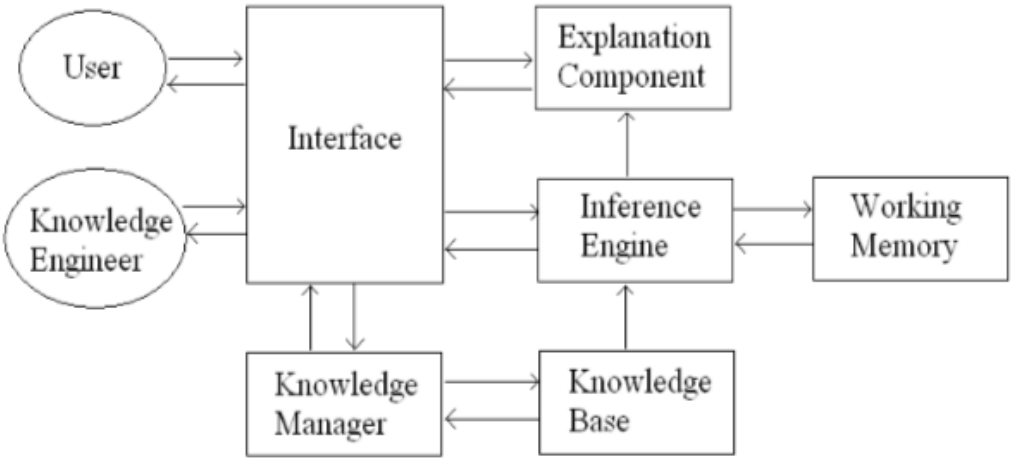
Việc áp dụng một số quy tắc heuristic ở trên vào thuật giải suy diễn sẽ giúp thuật giải loại bỏ được những bước thừa không cần thiết và làm tăng tốc quá trình giải quyết bài toán.

CHƯƠNG 4: CÀI ĐẶT THỬ NGHIỆM TRÊN MỘT MIỀN TRI THỨC

4.1. Cấu trúc của một hệ thống giải vấn đề thông minh

Hệ giải vấn đề thông minh (Intelligient Problem Solver - IPS) là một dạng của Hệ cơ sở tri thức. Theo tài liệu [13], Các thành phần của một hệ giải vấn đề thông minh như sau:

- Cơ sở tri thức (Knowledge base).
- Bộ suy diễn (Inference Engine).
- Bộ giải thích (Explanation Component).
- Vùng nhớ làm việc (Working memory).
- Bộ quản lý tri thức (Knowledge manager).
- Giao diện (Interfacre).



Hình 4-1: Cấu trúc một hệ giải vấn đề thông minh

Trong một hệ thống giải vấn đề thông minh (IPS), cơ sở tri thức và bộ suy diễn là hai thành phần then chốt, được coi như là trái tim của hệ thống thông minh. Cơ sở tri thức lưu trữ các tri thức đã được đặc tả và mô hình hoá, trong khi đó bộ suy diễn sẽ thực hiện

việc suy luận tìm lời giải của bài toán dựa trên các tri thức được lưu trữ trong cơ sở tri thức. Nếu thiếu một trong hai thành phần trên thì hệ thống sẽ không thể hoạt động được.

4.2. Tổ chức lưu trữ các tri thức và bộ suy diễn

Các tri thức sau khi đã được phân loại và mô hình hoá theo mô hình COKB cần được tổ chức lưu trữ thành các tập tin theo đặc tả trong mục 2.3. Các tri thức là độc lập với mô hình COKB. Một miền tri thức cụ thể bất kỳ nếu được lưu trữ theo cấu trúc đã đặc tả thì bộ suy diễn sẽ có thể đọc và xử lý các tri thức.

Các tri thức về các đối tượng tính toán thuộc miền tri thức Ma trận và Hệ phương trình tuyến tính được tổ chức lưu trữ như sau:

- Các khái niệm: Gồm các khái niệm về Ma trận, ma trận vuông, ma trận khả nghịch và hệ phương trình, được lưu dưới dạng các file .txt lần luợt là: *MATRAN.txt, MATRANVUONG.txt, MATRANKHANGHICH.txt, HEPHUONGTRINH.txt*.
- Các quan hệ phân cấp: Gồm các quan hệ có dạng IS_A như: MATRANVUONG IS_A MATRAN, MATRANKHANGHICH IS_A MATRAN. Các quan hệ trên được lưu trữ trong file *HIERACHY.txt*
- Các quan hệ tính toán: Gồm quan hệ bằng nhau giữa 2 ma trận, được lưu trữ trong file *RELATIONS.txt.*
- Các hàm trên các đối tượng tính toán: Gồm hàm HANG tính hạng của một ma trận và trả về giá trị số nguyên, hàm BDSOCAP thực hiện biến đổi sơ cấp trên một ma trận, giá trị trả về là một MATRAN, hàm NGHIEM tính toán nghiệm của một hệ phương trình chỉ trong trường hợp đã xác định hệ phương trình trên đã có nghiệm duy nhất, hàm SODONG tính số dòng của một ma trận và hàm SOCOT tính sô cột của một ma trận. Các đặc tả về hàm được lưu trữ trong file *FUNCTIONS.txt* và định nghĩa về hàm được lưu trữ trong file *FUNCTIONS_DEF.txt.*

- Các toán tử trên các đối tượng tính toán: Gồm toán tử cộng hai ma trận, nhân một ma trận với một ma trận và nhân một số với một ma trận. Các đặc tả về toán tử được lưu trong file *OPERATORS.txt* và định nghĩa về toán tử được lưu trong file *OPERATORS_DEF.txt*.
- Các luật suy diễn trên các đối tượng tính toán: Gồm luật xác định một ma trận vuông và luật xác định một ma trận khả nghịch. Các luật được lưu trong file *RULES.txt*.

Phần tổ chức lưu trữ các cơ sở tri thức trên miền tri thức Ma trận và Hệ phương trình tuyến tính sẽ được trình bày chi tiết ở phụ lục A.

Bộ suy diễn được xây dựng bằng ngôn ngữ Maple dùng để xử lý suy diễn tự động các tri thức đã được tổ chức lưu trữ trên máy tính. Bộ suy diễn có các chức năng sau:

- Đọc cơ sở tri thức: Sẽ tiến hành khởi tạo các biến cần thiết và đọc các tri thức đã lưu trữ trên máy tính.
- Thành phần xử lý sự kiện: Thực hiện phân loại sự kiện, so khớp và hợp nhất các sự kiện cùng các xử lý khác liên quan đến tập hợp các sự kiện như đã mô tả trong mục 2.2.
- Thành phần suy diễn: Gồm các quy tắc suy diễn (5 quy tắc) trên đối tượng và trên mô hình COKB (7 quy tắc) như đã định nghĩa trong mục 3.2. Thuật giải suy diễn tổng quát trên các lớp bài toán được thiết kế theo các thuật giải trong mục 3.2
- Thành phần giải quyết bài toán: Thực hiện suy diễn bài toán đưa vào và tối ưu lời giải, sau đó hiển thị lời giải từng bước ra màn hình.

4.3. Thử nghiệm và đánh giá

Dựa trên miền tri thức cụ thể về Ma trận và hệ phương trình tuyến tính, đề tài đã thu thập được 10 bài toán mẫu dùng để thử nghiệm bộ suy diễn. Các bài toán này được thu thập dựa vào tài liệu [15] và [16] và được trình bày chi tiết trong Phụ lục B.

Dưới đây sẽ trình bày kết quả một vài ví dụ trích ra từ các bài toán mẫu dùng để thử nghiệm ở trên. Các ví dụ này là các lớp bài toán mới mà trong các đề tài hay luận văn trước đó chưa đề cập.

Mẫu ví dụ	**Lời giải của bộ suy diễn**
Ví dụ 1	Cho hệ phương trình W tuyến tính có dạng: $\begin{pmatrix} 1 & 1 & 2 & 3 \\ 2 & 3 & -1 & -1 \\ 3 & -1 & -1 & -2 \\ 1 & 2 & 3 & -1 \end{pmatrix} X = \begin{bmatrix} 1 \\ -6 \\ -4 \\ -4 \end{bmatrix}.$ Hãy tính nghiệm của hệ phương trình trên.
	Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]} -------------- Buoc 2 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {B . m = SODONG(B), B . n = SOCOT(B)} Tu: {[B, MATRAN]} -------------- Buoc 3 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {HANG(W . Q), W . a = HANG(A), W . Q . K = MORONG(A,B), W . Q . m = SODONG(W . Q), W . Q . n = SOCOT(W . Q)} Tu: {[W[A,B], HEPHUONGTRINH]} -------------- Buoc 4 Thao tac: Ap dung quy tac RCN4 ham: HANG(W . Q) Tim duoc: {BDSOCAP(W . Q), HANG(W . Q)} Tu: {HANG(W . Q)} -------------- Buoc 5 Thao tac: Ap dung quy tac RCN4 ham: SODONG(A) Tim duoc: {SODONG(A), SODONG(A) = 4} Tu: {SODONG(A), A . K = (Array(1..4, 1..4, [[1,1,2,3],[2,3,-1,-1],[3,-1,-1,-2],[1,2,3,-1]]))} -------------- Buoc 6 Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A) Tim duoc: {SOCOT(A), SOCOT(A) = 4}

```
Tu: {SOCOT(A), A . K = (Array(1..4, 1..4, [[1,1,2,3],[2,3,-1,-
1],[3,-1,-1,-2],[1,2,3,-1]]))}
--------------
Buoc 7
Thao tac: Ap dung quy tac RCN4 ham: SODONG(B)
Tim duoc: {SODONG(B), SODONG(B) = 1}
Tu: {SODONG(B), B . K = (Array(1..4, [1,-6,-4,-4]))}
--------------
Buoc 8
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(B)
Tim duoc: {SOCOT(B), SOCOT(B) = 4}
Tu: {SOCOT(B), B . K = (Array(1..4, [1,-6,-4,-4]))}
--------------
Buoc 9
Thao tac: Ap dung quy tac RCN4 ham: HANG(A)
Tim duoc: {BDSOCAP(A), HANG(A)}
Tu: {HANG(A)}
--------------
Buoc 10
Thao tac: Ap dung quy tac RCN4 ham: MORONG(A,B)
Tim duoc: {MORONG(A,B), MORONG(A,B) = (Array(1..4, 1..5,
[[1,1,2,3,1],[2,3,-1,-1,-6],[3,-1,-1,-2,-4],[1,2,3,-1,-4]]))}
Tu: {MORONG(A,B), A . K = (Array(1..4, 1..4, [[1,1,2,3],[2,3,-1,-
1],[3,-1,-1,-2],[1,2,3,-1]])), B . K = (Array(1..4, [1,-6,-4,-4]))}
--------------
Buoc 11
Thao tac: Ap dung quy tac RCN4 ham: BDSOCAP(A)
Tim duoc: {BDSOCAP(A), BDSOCAP(A) = (Array(1..4, 1..4,
[[1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]))}
Tu: {BDSOCAP(A), A . K = (Array(1..4, 1..4, [[1,1,2,3],[2,3,-1,-
1],[3,-1,-1,-2],[1,2,3,-1]])), A . m = 4, A . n = 4}
--------------
Buoc 12
Thao tac: Ap dung quy tac RCN4 ham: HANG(A)
Tim duoc: {HANG(A), HANG(A) = 4}
Tu: {HANG(A), A . m = 4, A . n = 4, BDSOCAP(A) = (Array(1..4, 1..4,
[[1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]))}
--------------
Buoc 13
Thao tac: Ap dung quy tac RCN4 ham: SODONG(W . Q)
Tim duoc: {SODONG(W . Q), SODONG(W . Q) = 4}
Tu: {SODONG(W . Q), W . Q . K = (Array(1..4, 1..5,
[[1,1,2,3,1],[2,3,-1,-1,-6],[3,-1,-1,-2,-4],[1,2,3,-1,-4]]))}
--------------
Buoc 14
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(W . Q)
Tim duoc: {SOCOT(W . Q), SOCOT(W . Q) = 5}
Tu: {SOCOT(W . Q), W . Q . K = (Array(1..4, 1..5,
[[1,1,2,3,1],[2,3,-1,-1,-6],[3,-1,-1,-2,-4],[1,2,3,-1,-4]]))}
--------------
Buoc 15
Thao tac: Ap dung quy tac RCN4 ham: BDSOCAP(W . Q)
Tim duoc: {BDSOCAP(W . Q), BDSOCAP(W . Q) = (Array(1..4, 1..5,
[[1,0,0,0,-1],[0,1,0,0,-1],[0,0,1,0,0],[0,0,0,1,1]]))}
Tu: {BDSOCAP(W . Q), W . Q . K = (Array(1..4, 1..5,
[[1,1,2,3,1],[2,3,-1,-1,-6],[3,-1,-1,-2,-4],[1,2,3,-1,-4]])), W . Q
. m = 4, W . Q . n = 5}
--------------
Buoc 16
```

	Thao tac: Ap dung quy tac RCN4 ham: HANG(W . Q) Tim duoc: {HANG(W . Q), HANG(W . Q) = 4} Tu: {HANG(W . Q), W . Q . m = 4, W . Q . n = 5, BDSOCAP(W . Q) = (Array(1..4, 1..5, [[1,0,0,0,-1],[0,1,0,0,-1],[0,0,1,0,0],[0,0,0,1,1]]))} -------------- Buoc 17 Thao tac: Ap dung quy tac RCN4 ham: NGHIEM(W) Tim duoc: {NGHIEM(W), NGHIEM(W) = [-1, -1, 0, 1]} Tu: {NGHIEM(W), W . a = 4, BDSOCAP(W . Q) = (Array(1..4, 1..5, [[1,0,0,0,-1],[0,1,0,0,-1],[0,0,1,0,0],[0,0,0,1,1]])), HANG(W . Q) = 4} --------------
Ví dụ 2	Tính tích của 2 ma trận sau: $\begin{pmatrix} 1 & -3 & 2 \\ 3 & -4 & 1 \\ 2 & -5 & 3 \end{pmatrix}$ và $\begin{pmatrix} 2 & 5 & 6 \\ 1 & 2 & 5 \\ 1 & 3 & 2 \end{pmatrix}$
	Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]} -------------- Buoc 2 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {B . m = SODONG(B), B . n = SOCOT(B)} Tu: {[B, MATRAN]} -------------- Buoc 3 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {C . m = SODONG(C), C . n = SOCOT(C)} Tu: {[C, MATRAN]} -------------- Buoc 4 Thao tac: Ap dung quy tac RCN4 ham: SODONG(A) Tim duoc: {SODONG(A), SODONG(A) = 3} Tu: {SODONG(A), A . K = (Array(1..3, 1..3, [[1,-3,2],[3,-4,1],[2,-5,3]]))} -------------- Buoc 5 Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A) Tim duoc: {SOCOT(A), SOCOT(A) = 3} Tu: {SOCOT(A), A . K = (Array(1..3, 1..3, [[1,-3,2],[3,-4,1],[2,-5,3]]))} -------------- Buoc 6 Thao tac: Ap dung quy tac RCN4 ham: SODONG(B) Tim duoc: {SODONG(B), SODONG(B) = 3} Tu: {SODONG(B), B . K = (Array(1..3, 1..3, [[2,5,6],[1,2,5],[1,3,2]]))} -------------- Buoc 7 Thao tac: Ap dung quy tac RCN4 ham: SOCOT(B) Tim duoc: {SOCOT(B), SOCOT(B) = 3} Tu: {SOCOT(B), B . K = (Array(1..3, 1..3, [[2,5,6],[1,2,5],[1,3,2]]))} -------------- Buoc 8 Thao tac: Ap dung quy tac RCN7 tren toan tu: *(A,B)

	Tim duoc: {C . K = (Array(1..3, 1..3, [[1,5,-5],[3,10,0],[2,9,-7]])), C . m = 3, C . n = 3} Tu: {A . K = (Array(1..3, 1..3, [[1,-3,2],[3,-4,1],[2,-5,3]])), A . m = 3, A . n = 3, B . K = (Array(1..3, 1..3, [[2,5,6],[1,2,5],[1,3,2]])), B . n = 3} ----------
Ví dụ 3	Cho ma trận A = $\begin{pmatrix} m & 5m & -m \\ 2m & m & 10m \\ -m & -2m & -3m \end{pmatrix}$ với tham số m. Hãy biện luận hạng của ma trận theo m.
	Thao tac: Ap dung bien doi so cap tren dong Tim duoc: {BDSOCAP(A) = (Array(1..3, 1..3, [[m,5*m,-m],[0,-9*m,12*m],[0,0,0]]))} Thao tac: Giai cac phuong trinh: Tim duoc: {[m = 0], [m <> 0]} ---------- Truong hop[m = 0] -- Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]} ---------- Buoc 2 Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A) Tim duoc: {SOCOT(A), SOCOT(A) = 3} Tu: {SOCOT(A), A . K = (Array(1..3, 1..3, [[0,0,0],[0,0,0],[0,0,0]]))} ---------- Buoc 3 Thao tac: Ap dung quy tac RCN4 ham: SODONG(A) Tim duoc: {SODONG(A), SODONG(A) = 3} Tu: {SODONG(A), A . K = (Array(1..3, 1..3, [[0,0,0],[0,0,0],[0,0,0]]))} ---------- Buoc 4 Thao tac: Ap dung quy tac RCN4 ham: BDSOCAP(A) Tim duoc: {BDSOCAP(A), BDSOCAP(A) = (Array(1..3, 1..3, [[0,0,0],[0,0,0],[0,0,0]]))} Tu: {BDSOCAP(A), A . K = (Array(1..3, 1..3, [[0,0,0],[0,0,0],[0,0,0]])), A . m = 3, A . n = 3} ---------- Buoc 5 Thao tac: Ap dung quy tac RCN4 ham: HANG(A) Tim duoc: {HANG(A), HANG(A) = 0} Tu: {HANG(A), A . m = 3, A . n = 3, BDSOCAP(A) = (Array(1..3, 1..3, [[0,0,0],[0,0,0],[0,0,0]]))} ---------- ----------++++----------- Truong hop[m <> 0] -- Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]}

	---------- Buoc 2 Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A) Tim duoc: {SOCOT(A), SOCOT(A) = 3} Tu: {SOCOT(A), A . K = (Array(1..3, 1..3, [[1,5,-1],[2,1,10],[-1,-2,-3]]))} ---------- Buoc 3 Thao tac: Ap dung quy tac RCN4 ham: SODONG(A) Tim duoc: {SODONG(A), SODONG(A) = 3} Tu: {SODONG(A), A . K = (Array(1..3, 1..3, [[1,5,-1],[2,1,10],[-1,-2,-3]]))} ---------- Buoc 4 Thao tac: Ap dung quy tac RCN4 ham: BDSOCAP(A) Tim duoc: {BDSOCAP(A), BDSOCAP(A) = (Array(1..3, 1..3, [[1,0,17/3],[0,1,-4/3],[0,0,0]]))} Tu: {BDSOCAP(A), A . K = (Array(1..3, 1..3, [[1,5,-1],[2,1,10],[-1,-2,-3]])), A . m = 3, A . n = 3} ---------- Buoc 5 Thao tac: Ap dung quy tac RCN4 ham: HANG(A) Tim duoc: {HANG(A), HANG(A) = 2} Tu: {HANG(A), A . m = 3, A . n = 3, BDSOCAP(A) = (Array(1..3, 1..3, [[1,0,17/3],[0,1,-4/3],[0,0,0]]))} ---------- ----------+++++----------

Bảng 4-1: Kết quả thực hiện một số ví dụ

Ngoài ra, đề tài cũng tiến hành thử nghiệm để so sánh tính hiệu quả khi áp dụng một số quy tắc heuristic vào thuật giải giải quyết bài toán. Ví dụ về việc áp dụng quy tắc heuristic như sau:

Xét **Ví dụ 2** trong bảng 4.1, ta so sánh lời giải với 2 trường hợp có heuristic và không có heuristic:

Không có heuristic	**Có heuristic**
Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]} ---------- Buoc 2 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {B . m = SODONG(B), B . n = SOCOT(B)} Tu: {[B, MATRAN]} ----------	Buoc 1 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {A . m = SODONG(A), A . n = SOCOT(A)} Tu: {[A, MATRAN]} ---------- Buoc 2 Thao tac: Ap dung quy tac RC1 tren doi tuong: Tim duoc: {B . m = SODONG(B), B . n = SOCOT(B)} Tu: {[B, MATRAN]} ----------

```
Buoc 3
Thao tac: Ap dung quy tac RC1 tren doi tuong:
Tim duoc: {C . m = SODONG(C), C . n = SOCOT(C)}
Tu: {[C, MATRAN]}
--------------
Buoc 4
Thao tac: Ap dung quy tac RC1 tren doi tuong:
Tim duoc: {E . m = SODONG(E), E . n = SOCOT(E)}
Tu: {[E, MATRAN]}
--------------
Buoc 5
Thao tac: Ap dung quy tac RCN4 ham: SODONG(A)
Tim duoc: {SODONG(A), SODONG(A) = 3}
Tu: {SODONG(A), A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]]))}
--------------
Buoc 6
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A)
Tim duoc: {SOCOT(A), SOCOT(A) = 3}
Tu: {SOCOT(A), A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]]))}
--------------
Buoc 7
Thao tac: Ap dung quy tac RCN4 ham: SODONG(B)
Tim duoc: {SODONG(B), SODONG(B) = 3}
Tu: {SODONG(B), B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]]))}
--------------
Buoc 8
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(B)
Tim duoc: {SOCOT(B), SOCOT(B) = 3}
Tu: {SOCOT(B), B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]]))}
--------------
Buoc 9
Thao tac: Ap dung quy tac RCN6 tren luat phat sinh doi tuong:  XAC DINH MA TRAN VUONG
Tim duoc: {[A, MATRANVUONG]}
Tu: {A . m = A . n, A . n = A . m}
--------------
Buoc 10
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(A,B)
Tim duoc: {C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
Tu:  {A  .  K  =  (Array(1..3,  1..3, [[1,2,3],[4,3,5],[5,6,8]])), A . m = 3, A . n = 3, B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3}
--------------
Buoc 11
```

```
Buoc 3
Thao tac: Ap dung quy tac RC1 tren doi tuong:
Tim duoc: {C . m = SODONG(C), C . n = SOCOT(C)}
Tu: {[C, MATRAN]}
--------------
Buoc 4
Thao tac: Ap dung quy tac RC1 tren doi tuong:
Tim duoc: {E . m = SODONG(E), E . n = SOCOT(E)}
Tu: {[E, MATRAN]}
--------------
Buoc 5
Thao tac: Ap dung quy tac RCN4 ham: SODONG(A)
Tim duoc: {SODONG(A), SODONG(A) = 3}
Tu: {SODONG(A), A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]]))}
--------------
Buoc 6
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(A)
Tim duoc: {SOCOT(A), SOCOT(A) = 3}
Tu: {SOCOT(A), A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]]))}
--------------
Buoc 7
Thao tac: Ap dung quy tac RCN4 ham: SODONG(B)
Tim duoc: {SODONG(B), SODONG(B) = 3}
Tu: {SODONG(B), B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]]))}
--------------
Buoc 8
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(B)
Tim duoc: {SOCOT(B), SOCOT(B) = 3}
Tu: {SOCOT(B), B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]]))}
--------------
Buoc 9
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(A,B)
Tim duoc: {C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
Tu:  {A  .  K  =  (Array(1..3,  1..3, [[1,2,3],[4,3,5],[5,6,8]])), A . m = 3, A . n = 3, B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3}
--------------
Buoc 10
Thao tac:  Ap  dung  quy  tac  RCN4  ham: SODONG(C)
Tim duoc: {SODONG(C), SODONG(C) = 3}
Tu: {SODONG(C), C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]]))}
--------------
```

```
Thao tac: Ap dung quy tac RCN4 ham: SODONG(C)
Tim duoc: {SODONG(C), SODONG(C) = 3}
Tu: {SODONG(C), C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]]))}
--------------
Buoc 12
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(C)
Tim duoc: {SOCOT(C), SOCOT(C) = 3}
Tu: {SOCOT(C), C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]]))}
--------------
Buoc 13
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(A,B)
Tim duoc: {C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
Tu: {A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]])), A . m = 3, A . n = 3, B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3}
--------------
Buoc 14
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(B,C)
Tim duoc: {E . K = (Array(1..3, 1..3, [[3,-4,9],[12,11,-5],[3,18,-2]])), E . m = 3, E . n = 3}
Tu: {B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3, C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
--------------
```

```
Buoc 11
Thao tac: Ap dung quy tac RCN4 ham: SOCOT(C)
Tim duoc: {SOCOT(C), SOCOT(C) = 3}
Tu: {SOCOT(C), C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]]))}
--------------
Buoc 12
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(A,B)
Tim duoc: {C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
Tu: {A . K = (Array(1..3, 1..3, [[1,2,3],[4,3,5],[5,6,8]])), A . m = 3, A . n = 3, B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3}
--------------
Buoc 13
Thao tac: Ap dung quy tac RCN7 tren toan tu: +(B,C)
Tim duoc: {E . K = (Array(1..3, 1..3, [[3,-4,9],[12,11,-5],[3,18,-2]])), E . m = 3, E . n = 3}
Tu: {B . K = (Array(1..3, 1..3, [[1,-3,3],[4,4,-5],[-1,6,-5]])), B . m = 3, B . n = 3, C . K = (Array(1..3, 1..3, [[2,-1,6],[8,7,0],[4,12,3]])), C . m = 3, C . n = 3}
--------------
```

Bảng 4-2: So sánh giữa việc thực hiện có heuristic và không có heuristic

Bảng sau sẽ trình bày các kết quả thu được khi bộ suy diễn thực hiện giải 10 bài toán mẫu được đề nghị. Trong bảng này cũng có phần so sánh tính hiệu quả giữa thuật giải tổng quát và thuật giải có áp dụng các quy tắc heuristic:

Bài toán cụ thể	**Số đối tượng (đối tượng)**	**Tính đúng (Đúng/ Sai)**	**Thời gian không có heuristic (s)**	**Thời gian có heuristic (s)**	**Số bước giải không có heuristic (bước)**	**Số bước giải có heuristic (bước)**
BÀI TOÁN 1	3	Đúng	9.451	11.537	9	8
BÀI TOÁN 2	4	Đúng	39.657	49.508	14	13

BÀI TOÁN 3	1	Đúng	0.358	0.408	6	6
BÀI TOÁN 4	1	Đúng	2.779	2.886	10	10
BÀI TOÁN 5	3	Đúng	23.125	21.469	18	17
BÀI TOÁN 6	3	Đúng	31.000	31.406	17	17
BÀI TOÁN 7	3	Đúng	29.328	28.406	18	17
BÀI TOÁN 8	1	Đúng (2TH)	1.097	1.090	2 TH: TH1: 6 TH2: 6	2 TH: TH1: 5 TH2: 5
BÀI TOÁN 9	1	Đúng (2TH)	1.082	1.072	2 TH: TH1: 6 TH2: 6	2 TH: TH1: 6 TH2: 6
BÀI TOÁN 10	3	Đúng (3TH)	102.172	87.922	2 TH: TH1: 17 TH2: 17 TH3: 17	2 TH: TH1: 14 TH2: 13 TH3: 13

Bảng 4-3: Thống kê kết quả thực hiện một số bài toán mẫu (*)

(*) Thời gian đo được của các lần chạy thử nghiệm khác nhau sẽ có sự chênh lệch nhất định

Các sô liệu ở trên thu được khi chạy thử bộ suy diễn trên các bài toán với máy tính có cấu hình như sau: CPU Intel Core i5 1.6 Ghz, Memory 8GB. Thời gian thu được bằng cách sử dụng hàm thời gian *time()* của Maple và được tính từ lúc bắt đầu thực hiện giải bài toán cho đến lúc kết thúc quá trình giải, không tính thời gian thực hiện xuất lời giải ra màn hình.

Nhìn chung, đối với các bài toán cụ thể được đưa ra, bộ suy diễn có thể giải và đưa ra các bước giải chi tiết đối với từng bài toán cụ thể. Với các quy tắc heuristic được thêm

vào, bộ suy diễn có thể lược bỏ một số bước suy diễn thừa không cần thiết và giảm thời gian thực thi.

So sánh với một số công cụ hỗ trợ giải bài toán về Ma trận và hệ phương trình tuyến tính như gói *LinearAlgebra* trong phần mềm Maple thì bộ suy diễn giải bài toán về Ma trận và hệ phương trình tuyến tính có ưu điểm hơn về mặt suy luận giải quyết vấn đề. Đối với các bài toán cần yếu tố suy luận như chứng minh Hệ phương trình có nghiệm, vô nghiệm hay vô số nghiệm hoặc lớp bài toán như biện luận hạng của ma trận theo tham số, biện luận nghiệm của hệ phương trình theo tham số thì gói LinearAlgebra không thể giải quyết được trong khi bộ suy diễn được xây dựng dựa theo mô hình COKB có thể giải quyết hoàn chỉnh và đưa ra lời giải cùng các bước suy luận tìm ra vấn đề.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết quả đạt được

Về phương diện lý thuyết, đề tài này đã đạt được một số kết quả như sau:

- Thực hiện khảo sát 2 lớp bài toán mới: lớp bài toán xác định một hàm và lớp bài toán biện luận tham số.
- Định nghĩa quy tắc suy diễn RCN4 để thực hiện suy diễn trên các sự kiện hàm mà trong luận văn [10] chưa giải quyết.
- Đề xuất thêm một quy tắc suy diễn mới (quy tắc RCN7) trên các thành phần toán tử cho mô hình COKB, cùng các thuật giải suy diễn và thủ tục cần thiết để suy luận tìm ra sự kiện mới dựa trên việc thực thi một toán tử trên các đối tượng.
- Đưa ra mô hình bài toán tổng quát cho lớp bài toán biện luận tham số cùng thuật giải suy diễn tổng quát cho lớp bài toán này.
- Đề xuất một số quy tắc heuristic nhằm hỗ trợ cho thuật giải suy diễn tổng quát.

Tổng kết lại, mô hình COKB có tất cả là 6 thành phần cùng 7 quy tắc suy diễn cụ thể trên các thành phần. Mô hình COKB trên cùng với các quy tắc suy diễn có thể được áp dụng cho các miền tri thức khác nhau. Ngoài ra, đề tài đề xuất một số quy tắc heuristic hỗ trợ cho thuật giải suy diễn tổng quát hoạt động hiệu quả hơn.

Về phương diện kỹ thuật, đề tài này đã sử dụng ngôn ngữ Maple cài đặt thành công một module suy diễn trên các tri thức thuộc miền tri thức Ma trận và hệ phương trình tuyến tính. Module suy diễn này đã thực hiện giải thành công một số bài toán mẫu trên miền tri thức Ma trận và hệ phương trình tuyến tính, trong đó có các bài toán thuộc lớp bài toán mới là lớp bài toán xác định một hàm và lớp bài toán biện luận tham số. Ngoài ra

module suy diễn cũng thực hiện suy diễn được đối với các bài toán xác định giá trị của một biểu thức gồm các đối tượng liên kết với nhau bởi các toán tử.

5.2. Hạn chế và hướng phát triển trong tương lai

5.2.1. Hạn chế

Ngoài các kết quả đạt được như trên, đề tài cũng có những mặt hạn chế như sau:

- Thời gian thực thi của một số lớp bài toán còn khá lâu, nhất là đối với các bài toán có nhiều đối tượng tính toán tham gia do thuật giải suy diễn giải quyết bài toán chưa thực sự tối ưu.
- Lời giải của bài toán do bộ suy diễn đưa ra chưa thực sự tốt và gần với cách giải quyết của con người.
- Miền tri thức còn hạn chế, chưa cập nhật đầy đủ các tri thức cần thiết để thiết kế thành một hệ thống giải quyết vấn đề thông minh hoàn chỉnh.
- Chương trình thực thi còn khá đơn sơ, chưa đủ để cấu tạo thành một hệ giải bài toán thông minh.

Do chỉ giới hạn về mặt demo cho mô hình nên đề tài này chỉ xây dựng một moudle suy diễn cho miền tri thức giới hạn là Ma trận và hệ phương trình tuyến tính. Để xây dựng thành một hệ thống giải vấn đề hoàn chỉnh thì ngoài module suy diễn ra cần phải cập nhật bổ sung một số thành phần khác như: Bộ giải thích, giao diện và bộ quản lý tri thức. Bản thân module suy diễn cũng còn một vài hạn chế nên cần được cải tiến trong tương lai.

5.2.2. Hướng phát triển trong tương lai

Cải tiến bộ suy diễn để thời gian thực thi tốt hơn. Đối với một số lớp bài toán có cách giải tương tự nhau thì có thể kết hợp sử dụng phương pháp Bài toán mẫu để tái sử dụng lại lời giải của các bài toán trước đó và giảm tải việc suy luận cho bộ suy diễn.

Thiết kế một bộ công cụ hoặc một framework hỗ trợ các nhà thiết kế hệ thống thông minh xây dựng và phát triển một hệ cơ sở tri thức COKB trên một miền tri thức cụ thể.

Nghiên cứu các dạng Ontology đơn giản hơn dựa trên mô hình COKB và cách tích hợp các dạng Ontology đó lại với nhau thành một hệ thống hoàn chỉnh thay vì mở rộng mô hình COKB đầy đủ có sẵn.

Áp dụng mô hình COKB vào xây dựng các hệ giải vấn đề thông minh phục vụ nhu cầu học tập của học sinh và sinh viên trên các lĩnh vực: Đại số tuyến tính, Toán rời rạc, Xác suất Thống kê, Hóa hữu cơ, Hóa vô cơ.

Xây dựng các hệ chuyên gia, hệ hỗ trợ quyết định trợ giúp cho con người trong các hoạt động thực tế trên các lĩnh vực của cuộc sống.

TÀI LIỆU THAM KHẢO

Tiếng Anh

[1] Nhon V. Do (2014), "Ontology COKB for Designing Knowledge-based Systems", *New Trends in Intelligent Software Methodologies, Tools and Techniquess, H. Fujira et al (Eds).*

[2] Nhon V. Do, Thanh T. Mai (2015), "Intelligent Problem Solving based on COKB Model", *2015 Seventh International Conference on Knowledge and Systems Engineering*.

[3] Nhon D. Van, Thanh M. Trung (2016), "Development of Reasoning Techniques on Knowledge Representation Model COKB and Applications", *KSE 2016, pp.29*.

[4] Nhon V. Do, Hien D. Nguyen, Thanh M. Trung (2015), "Reasoning Method on Knowledge about Functions and Operators", *IJACSA International Journal of Advanced Computer Science and Applications, Vol. 6, No. 6, 2015*.

[5] Nhon Do, Hien Nguyen (2010), "Model for Knowledge Representation using Sample Problems and Designing a Program for Automatically Solving Algebraic Problems", *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering Vol:4, No:6.*

[6] Nhon V. Do, Hien D. Nguyen (2012), "A Knowledge Model about Relations and Application", *6th International Conference on New Trends in Information Science, Service Science and Data Mining (ISSDM2012).*

[7] Nhon V. Do, Thanh T. Mai (2017), "Perfect COKB Model and Reasoning methods for the Design of Intelligent Problem Solvers", *New Trends in Intelligent Software Methodologies, Tools and Techniquess, H. Fujira et al (Eds).*

[8] T.R Grubber (1993), "A Translation Approach to Portable Ontologies", *Knowledge Aquisition, Vol.5(2), pp.199 – 200.*

[9] Stuart Russel, Peter Norvig (2010), "Artificial Intelligent – A modern approach, 3rd edition", Prentice Hall.

Tiếng Việt

[10] Mai Trung Thành (2017), "Phương pháp suy luận giải quyết vấn đề trên mô hình COKB", Luận văn Thạc sĩ ngành Khoa học máy tính, Trường Đại học Công nghệ Thông tin, ĐHQG TPHCM.

[11] Nguyễn Đình Hiển (2012), "Phương pháp suy diễn trên mô hình COKB dựa trên tri thức bài toán mẫu", Luận văn Thạc sĩ Toán - Tin, Trường Đại học Khoa Học Tự Nhiên – ĐHQG Thành phố Hồ Chí Minh.

[12] Đỗ Văn Nhơn (2016), "Biểu diễn tri thức và suy luận", NXB ĐHQG TPHCM.

[13] Đỗ Văn Nhơn, Nguyễn Đình Hiển (2017), "Các hệ cơ sở tri thức", NXB ĐHQG TPHCM.

[14] Hoàng Kiếm, Đinh Nguyễn Anh Dũng (2010), "Giáo trình Nhập môn trí tuệ nhân tạo", NXB ĐHQG TPHCM.

[15] Đỗ Văn Nhơn (2008), "Toán cao cấp A3", NXB ĐHQG TPHCM.

[16] Bùi Xuân Hải (2011), "Đại số tuyến tính và ứng dụng – Tập 1", NXB ĐHQG TPHCM.

PHỤ LỤC A

Tổ chức lưu trữ tập tin cho miền tri thức Ma trận và hệ phương trình tuyến tính theo COKB.

Tập tin CONCEPTS.txt

begin_concepts
MATRAN
MATRANVUONG
MATRANKHANGHICH
HEPHUONGTRINH
end_concepts

Tập tin BASE_CONCEPTS.txt

begin_concepts
INT
REAL
Array
list
set
end_concepts

Tập tin RELATIONS.txt

begin_relations
[BANGNHAU,MATRAN,MATRAN]{doixung}
end_relations

Tập tin FUNCTIONS.txt

begin_functions
REAL DINHTHUC (MATRANVUONG)
INT HANG (MATRAN)
Array NGHIEM (HEPHUONGTRINH)
INT SODONG (MATRAN)
INT SOCOT (MATRAN)
Array BDSOCAP (MATRAN)
Array MORONG (MATRAN,MATRAN)
Array NGHICHDAO (MATRANKHANGHICH)

Array MORONGNGHICHDAO (MATRANKHANGHICH)
end_functions

Tập tin FUNCTIONS_DEF.txt

begin_functions
begin_function:DINHTHUC(_A)
_A: MATRANVUONG
result: REAL
begin_proc
LinearAlgebra:-Determinant(convert(_A.K,'Matrix'));
end_proc
properties
end_properties
end_function
begin_function:HANG(_A)
_A: MATRAN
result: INT
begin_proc
rAnK := 0: VALuE := convert(BDSOCAP(_A), 'Matrix'): COlMATriX:=_A.n: RoWMAtrIx:=_A.m: for i to RoWMAtrIx do if not LinearAlgebra:-Equal(LinearAlgebra:-Row(VALuE, i), LinearAlgebra:-ZeroVector[row](COlMATriX)) then rAnK := rAnK+1: else rAnK: end if: end do: rAnK;
end_proc
properties
BDSOCAP(_A)
end_properties
end_function
begin_function:NGHIEM(_W)
_W: HEPHUONGTRINH
result: Array
begin_proc
MaTRAnMoRONg:=convert(BDSOCAP(_W.Q),'Matrix'): HaNgMr:=HANG(_W.Q): HaNgHs:=_W.a: if (HaNgMr=HaNgHs and HaNgMr=(LinearAlgebra:-ColumnDimension(MaTRAnMoRONg))-1) then REsUlT:=LinearAlgebra:-Column(MaTRAnMoRONg, LinearAlgebra:-ColumnDimension(MaTRAnMoRONg)): convert(REsUlT,'list'); else `` end if:
end_proc
properties
end_properties
end_function
begin_function:SODONG(_Q)
_Q: MATRAN

```
result: INT
begin_proc
ArrayTools:-Size(_Q.K)[1];
end_proc
properties
end_properties
end_function
begin_function:SOCOT(_Q)
_Q: MATRAN
result: INT
begin_proc
ArrayTools:-Size(_Q.K)[2];
end_proc
properties
end_properties
end_function
begin_function:MORONG(_A,_B)
_A: MATRAN
_B: MATRAN
result: Array
begin_proc
convert((<convert(_A.K,'Matrix')|convert(_B.K,'Vector[column]')>),'Array');
end_proc
properties
end_properties
end_function
begin_function:BDSOCAP(_A)
_A: MATRAN
result: Array
begin_proc
ChECk:=true: MAtriX:=_A.K: for i from 1 to _A.m do for j from 1 to _A.n do if (not
type(MAtriX[i][j],'numeric') and type(MAtriX[i][j],'symbol')) then ChECk:=false: break: end
if end do: if (ChECk=false) then break: end if: end do: if (ChECk=true) then
convert(LinearAlgebra:-ReducedRowEchelonForm(convert(MAtriX,'Matrix')),'Array'); else
convert(LinearAlgebra:-GaussianElimination(convert(MAtriX,'Matrix')),'Array'); end if:
end_proc
properties
end_properties
end_function
begin_function:NGHICHDAO(_A)
_A: MATRANKHANGHICH
result: Array
begin_proc
```

*MOrONg:=BDSOCAP(_A.Q): if (not type(MOrONg,'function')) then MOrONg(1 .. _A.m, _A.n+1 .. _A.n*2) else `` end if:*
end_proc
properties
BDSOCAP(_A.Q)
end_properties
end_function
begin_function:MORONGNGHICHDAO(_A)
_A: MATRANKHANGHICH
result: Array
begin_proc
convert((<convert(_A.K,'Matrix')|LinearAlgebra:-IdentityMatrix(_A.n)>),'Array');
end_proc
properties
_A.Q.n=SOCOT(_A.Q)
_A.Q.m=SODONG(_A.Q)
end_properties
end_function
end_functions

Tập tin OPERATORS.txt

begin_operators
[+,MATRAN,MATRAN,MATRAN]
[,MATRAN,MATRAN,MATRAN]*
[,MATRAN,INT,MATRAN]*
end_operators

Tập tin OPERATORS_DEF.txt

begin_operators
begin_operator:(_A,_B)*
_A:MATRAN
_B:MATRAN
result: MATRAN
begin_proc
*REsult:=Array(1.._A.m,1.._B.n): PARAm1:=_A.K: PARAm2:=_B.K: for i from 1 to _A.m do for j from 1 to _B.n do for k from 1 to _A.n do REsult[i,j]:= REsult[i,j] + PARAm1[i][k]*PARAm2[k][j]: end do: end do: end do: {_.K= REsult, _.m=_A.m, _.n=_B.n};*
end_proc
properties
end_properties
end_operator

begin_operator:+(_A,_B)
_A:MATRAN
_B:MATRAN
result: MATRAN
begin_proc
REsult:=Array(1.._A.m,1.._A.n): PARAm1:=_A.K: PARAm2:=_B.K: if (_A.m = _B.m and _A.n = _B.n) then for i from 1 to _A.m do for j from 1 to _A.n do REsult[i,j]:= PARAm1[i][j]+ PARAm2[i][j]: end do: end do: {_.K= REsult, _.m=_A.m, _.n=_A.n} else `` end if:
end_proc
properties
end_properties
end_operator
begin_operator:(_a,_B)*
_a:INT
_B:MATRAN
result: MATRAN
begin_proc
*REsult:=Array(1.._B.m,1.._B.n): PARAm:=_B.K: for i from 1 to _B.m do for j from 1 to _B.n do REsult[i,j]:= _a * PARAm[i][j]: end do: end do: {_.K= REsult, _.m=_B.m, _.n=_B.n};*
end_proc
properties
end_properties
end_operator
end_operators

Tập tin RULES.txt

begin_rules
begin_rule
kind_rule = XAC DINH MA TRAN VUONG
_A : MATRAN
hypothesis_part:
_A.m = _A.n
_A.n = _A.m
end_hypothesis_part
goal_part:
[_A, "MATRANVUONG"]
end_goal_part
end_rule
begin_rule
kind_rule = XAC DINH MATRAN KHA NGHICH
_A : MATRANVUONG

hypothesis_part:
HANG(_A)=_A.n
end_hypothesis_part
goal_part:
[_A, "MATRANKHANGHICH"]
end_goal_part
end_rule
begin_rule
kind_rule = TINH CHAT HAI MA TRAN BANG NHAU
_A : MATRAN
_B : MATRAN
hypothesis_part:
["BANGNHAU", _A, _B]
end_hypothesis_part
goal_part:
_A.m = _B.m
_A.n = _B.m
_A.K = _B.K
end_goal_part
end_rule
end_rules

Tập tin HIERACHY.txt

begin_hierachy
[MATRAN,MATRANVUONG]
[MATRAN,MATRANKHANGHICH]
[MATRANVUONG,MATRANKHANGHICH]
end_hierachy

Tập tin MATRAN.txt

begin_concept:MATRAN
begin_variables
m: INT
n: INT
K: Array
end_variables
begin_constraints
end_constraints
begin_properties
m=SODONG(A)
n=SOCOT(A)

end_properties
begin_computation_relations
end_computation_relations
begin_rules
end_rules
end_concept

Tập tin MATRANVUONG.txt

begin_concept:MATRANVUONG
begin_variables
m: INT
n: INT
K: Array
end_variables
begin_constraints
end_constraints
begin_properties
n=SODONG(A)
m=n
DINHTHUC(A)
end_properties
begin_computation_relations
end_computation_relations
begin_rules
end_rules
end_concept

Tập tin MATRANKHANGHICH.txt

begin_concept:MATRANKHANGHICH
begin_variables
m: INT
n: INT
K: Array
Q: MATRAN
end_variables
begin_constraints
end_constraints
begin_properties
m=SODONG(A)
n=SOCOT(A)
Q.K=MORONGNGHICHDAO(A)
end_properties

begin_computation_relations
end_computation_relations
begin_rules
end_rules
end_concept

Tập tin HEPHUONGTRINH.txt

begin_concept:HEPHUONGTRINH[_A,_B]
_A:MATRAN
_B:MATRAN
begin_variables
Q:MATRAN *//Ma trận mở rộng*
a:INT *//Hạng của ma trận hệ số*
end_variables
begin_constraints
end_constraints
begin_properties
Q.K=MORONG(_A,_B)
Q.m=SODONG(Q)
Q.n=SOCOT(Q)
a=HANG(_A)
HANG(Q)
end_properties
begin_computation_relations
end_computation_relations
begin_rules
begin_rule
kind_rules = nghiem duy nhat
hypothesis_part:
_A.n=HANG(_A)
HANG(Q)=HANG(_A)
end_hypothesis_part
goal_part:
NGHIEM(W)
end_goal_part
end_rule
begin_rule
kind_rules = Vo nghiem
hypothesis_part:
[">",HANG(Q),HANG(_A)]
end_hypothesis_part
goal_part:

NGHIEM(W)
NGHIEM(W)=['Vo_nghiem']
end_goal_part
end_rule
begin_rule
kind_rules = Vo so nghiem
hypothesis_part:
HANG(Q)=HANG(_A)
[">",_A.n,HANG(Q)]
end_hypothesis_part
goal_part:
NGHIEM(W)
NGHIEM(W)=['Vo_so_nghiem']
end_goal_part
end_rule
end_rules
end_concept

Đối với trường hợp Hệ phương trình tuyến tính ở dạng các tích các hệ số, ta lưu trữ như sau:

Tập tin HEPHUONGTRINH.txt

begin_concept:HEPHUONGTRINH
begin_variables
Q:MATRAN
A:MATRAN //Ma trận hệ sô
B:MATRAN //Ma trận hê số tự do
a:INT //Hạng của ma trận hệ số
q:INT //Hạng của ma trận mở rộng
K:list //Danh sách gồm các hệ phương trình tuyến tính
p:set //Tập hợp các biến số trong hệ phương trình
end_variables
begin_constraints
end_constraints
begin_properties
Q.K=MORONG(A,B)
Q.m=SODONG(Q)
Q.n=SOCOT(Q)
A.m=SODONG(A)
A.n=SOCOT(A)
B.m=SODONG(B)
B.n=SOCOT(B)
a=HANG(A)

q=HANG(Q)
A.K=HESO(W)
B.K=HESOTUDO(W)
end_properties
begin_computation_relations
end_computation_relations
begin_rules
begin_rule
kind_rules = nghiem duy nhat
hypothesis_part:
A.n=HANG(A)
HANG(Q)=HANG(A)
end_hypothesis_part
goal_part:
NGHIEM(W)
end_goal_part
end_rule
begin_rule
kind_rules = Vo nghiem
hypothesis_part:
[">",HANG(Q),HANG(A)]
end_hypothesis_part
goal_part:
NGHIEM(W)
NGHIEM(W)=['Vo_nghiem']
end_goal_part
end_rule
begin_rule
kind_rules = Vo so nghiem
hypothesis_part:
HANG(Q)=HANG(A)
[">",A.n,HANG(Q)]
end_hypothesis_part
goal_part:
NGHIEM(W)
NGHIEM(W)=['Vo_so_nghiem']
end_goal_part
end_rule
end_rules
end_concept

Ta phải định nghĩa thêm các hàm *HESO* dùng để tìm ma trận hệ số và hàm *HESOTUDO* dùng để tìm ma trận hệ số tự do cho hệ phương trình tuyến tính

PHỤ LỤC B

Trong phần này sẽ trình bày 10 bài toán mẫu được dùng để thử nghiệm. Các bài toán mẫu được thu thập từ các tài liệu tham khảo [15] và [16].

BÀI TOÁN 1: Tính tích của 2 ma trận sau: $\begin{pmatrix}1 & -3 & 2\\ 3 & -4 & 1\\ 2 & -5 & 3\end{pmatrix}$ và $\begin{pmatrix}2 & 5 & 6\\ 1 & 2 & 5\\ 1 & 3 & 2\end{pmatrix}$

(Thời gian giải: 11.537s)

BÀI TOÁN 2: Cho ma trận A = $\begin{pmatrix}1 & 2 & 3\\ 4 & 3 & 5\\ 5 & 6 & 8\end{pmatrix}$ và B = $\begin{pmatrix}1 & 3 & -3\\ 4 & 4 & -5\\ -1 & 6 & -5\end{pmatrix}$. Biết C = A+B và E = C + B. hãy tính giá trị của ma trận E.

(Thời gian giải: 49.508s)

BÀI TOÁN 3: Cho ma trận A = : $\begin{pmatrix}1 & -1 & 5 & -1\\ 1 & 1 & -2 & 3\\ 3 & -1 & 8 & 1\\ 1 & 3 & -9 & 7\end{pmatrix}$. Hãy tìm hạng của A.

(Thời gian giải: 0.408s)

BÀI TOÁN 4: Cho ma trận A = $\begin{pmatrix}2 & 5 & -1\\ 6 & -1 & 2\\ 6 & 4 & 2\end{pmatrix}$. Biết A khả nghịch, hãy tìm nghịch đảo của A.

(Thời gian giải: 2.886s)

BÀI TOÁN 5: Giải hệ phương trình tuyến tính sau: $\begin{pmatrix}1 & 1 & 2 & 3\\ 2 & 3 & -1 & -1\\ 3 & -1 & -1 & -2\\ 1 & 2 & 3 & -1\end{pmatrix}$X=$\begin{bmatrix}1\\ -6\\ -4\\ -4\end{bmatrix}$.

(Thời gian giải: 21.469s)

BÀI TOÁN 6: Giải hệ phương trình tuyến tính sau: $\begin{pmatrix}1 & 1 & -3\\ 2 & 1 & -2\\ 1 & 1 & 1\\ 1 & 2 & -3\end{pmatrix}$X=$\begin{bmatrix}-1\\ 1\\ 3\\ 1\end{bmatrix}$.

(Thời gian giải: 31.406s)

BÀI TOÁN 7: Giải hệ phương trình tuyến tính sau: $\begin{pmatrix}3 & 4 & -5 & 7\\ 4 & 11 & -13 & 16\\ 2 & -3 & 3 & -2\\ 7 & -2 & 1 & 3\end{pmatrix}$X=$\begin{bmatrix}0\\ 0\\ 0\\ 0\end{bmatrix}$.

(Thời gian giải: 28.406s)

BÀI TOÁN 8: Cho ma trận A = $\begin{pmatrix}m & 5m & -m\\ 2m & m & 10m\\ -m & -2m & -3m\end{pmatrix}$ với tham số m. Hãy biện luận hạng của ma trận theo m.

(Thời gian giải: 1.090s)

BÀI TOÁN 9: Cho ma trận A = $\begin{pmatrix}1 & 2 & 0 & -1\\ 2 & 1 & 3 & 0\\ 0 & 3 & a & b\\ 3 & 3 & 3 & -1\end{pmatrix}$ với tham số a,b. Hãy biện luận hạng của ma trận theo a,b.

(Thời gian giải: 1.072s)

BÀI TOÁN 10: Giải hệ phương trình tuyến tính sau với tham số m:
$\begin{pmatrix}1 & 1 & 1\\ 1 & 1 & m\\ 1 & m & 1\\ m & 1 & 1\end{pmatrix}$X=$\begin{bmatrix}m\\ 1\\ 1\\ 1\end{bmatrix}$.
(Thời gian giải: 87.922s)