

CHƯƠNG 1. TỔNG QUAN 1

1.1. DẪN NHẬP

1.2. CƠ SỞ TRI THỨC

1.3. ĐỘNG CƠ SUY DIỄN

1.4. CÁC HỆ CHUYÊN GIA

1.5. HỆ HỖ TRỢ RA QUYẾT ĐỊNH

1.6. HỆ GIẢI BÀI TOÁN

1.7. TIẾP THU TRI THỨC

1.8. TÍCH HỢP CÁC HỆ CSTT VÀ CÁC HỆ QUẢN TRỊ CSDL

1.9. HỆ THỐNG ĐIỀU KHIỂN MỜ

1.1. MỞ ĐẦU

- Khác biệt giữa các hệ cơ sở tri thức (CSTT) và các chương trình truyền thống **nằm ở cấu trúc.**
- **Trong các chương trình truyền thống:**
 - i. cách thức xử lý hay hành vi của chương trình đã được ấn định sẵn qua các dòng

lệnh của chương trình dựa trên một thuật giải đã định sẵn.

– Trong các hệ CSTT:

Có hai chức năng tách biệt nhau, trường hợp đơn giản có hai khối:

- i. khối tri thức hay còn được gọi là *cơ sở tri thức*,
- ii. và khối điều khiển hay còn được gọi là *động cơ suy diễn*.

Với các hệ thống phức tạp, bản thân động cơ suy diễn cũng có thể là một hệ CSTT chứa các siêu tri thức (tri thức về cách sử dụng tri thức khác).

Việc tách biệt giữa tri thức khỏi các cơ chế điều khiển giúp ta dễ dàng thêm vào các tri thức mới trong tiến trình phát triển một chương trình.

Đây là điểm tương tự của động cơ suy diễn trong một hệ CSTT và não bộ con người (điều khiển xử lý), là không đổi cho dù hành vi của cá nhân có thay đổi theo kinh nghiệm và kiến thức mới nhận được.

Giả sử một chuyên gia dùng các chương trình truyền thống để hỗ trợ công việc hàng ngày, sự thay đổi hành vi của chương trình yêu cầu họ phải biết cách cài đặt chương trình.

Nói cách khác, chuyên gia phải là một lập trình viên chuyên nghiệp. Hạn chế này được giải quyết khi các chuyên gia tiếp cận sử dụng các hệ CSTT.

Trong các hệ CSTT, tri thức được biểu diễn tường minh chứ không nằm ở dạng ẩn như trong các chương trình truyền thống.

Do vậy có thể thay đổi các CSTT, sau đó các động cơ suy diễn sẽ làm việc trên các tri thức mới được cập nhật nhằm thực hiện yêu cầu mới của chuyên gia.

1.2. CƠ SỞ TRI THỨC

Cơ sở tri thức có nhiều dạng khác nhau: trong chương 2, chúng ta sẽ tìm hiểu các dạng biểu diễn tri thức như mô hình *đối tượng-thuộc tính-giá trị*, *thuộc tính-luật dẫn*, *mạng ngữ nghĩa*, *frame*.

Tri thức cũng có thể ở dạng không chắc chắn, mập mờ. Trong chương 4, chúng ta sẽ thảo luận về hệ số chắc chắn trong các luật của hệ CSTT MYCIN, và chương 9 sẽ nghiên cứu cách áp dụng các luật mờ trong các *hệ thống mờ*.

1.3. ĐỘNG CƠ SUY DIỄN

Các CSTT đều có động cơ suy diễn để tiến hành các suy diễn nhằm tạo ra các tri thức mới dựa trên các sự kiện, tri thức cung cấp từ ngoài vào và tri thức có sẵn trong hệ CSTT.

Động cơ suy diễn thay đổi theo độ phức tạp của CSTT. Hai kiểu suy diễn chính trong động cơ suy diễn là *suy diễn tiến* và *suy diễn lùi*.

Các hệ CSTT làm việc theo cách được điều khiển bởi dữ liệu (data driven) sẽ dựa vào các thông tin sẵn có (các sự kiện cho trước) và tạo sinh ra các sự kiện mới được suy diễn. Do vậy không thể đoán được kết quả. Cách tiếp cận này được sử dụng cho các bài toán diễn dịch với mong muốn của người sử dụng là hệ CSTT sẽ cung cấp các sự kiện mới. Ngoài ra còn có cách điều khiển theo mục tiêu nhằm hướng đến các kết luận đã có và đi tìm các

dẫn chứng để kiểm định tính đúng đắn của kết luận đó. Các kiểu suy diễn này sẽ được thảo luận chi tiết trong chương 3.

1.4. CÁC HỆ CHUYÊN GIA

Các hệ chuyên gia là một loại hệ CSTT được thiết kế cho một lĩnh vực ứng dụng cụ thể. Ví dụ các hệ chuyên gia để cấu hình mạng máy tính, các hệ chẩn đoán hỏng hóc đường dây điện thoại,.... Hệ chuyên gia làm việc như một chuyên gia thực thụ và có thể cung cấp các ý kiến tư vấn hỏng hóc dựa trên kinh nghiệm của chuyên gia đã được đưa vào hệ chuyên gia. Hệ chuyên gia có các thành phần cơ bản sau:

- (1) Bộ giao tiếp ngôn ngữ tự nhiên
- (2) Động cơ suy diễn
- (3) Cơ sở tri thức
- (4) Cơ chế giải thích WHY-HOW
- (5) Bộ nhớ làm việc
- (6) Tiếp nhận tri thức

Hình 1.1. Các thành phần của hệ chuyên gia



Khái niệm hệ hỗ trợ ra quyết định được đề xuất bởi Michael S. Scott Morton vào những năm 1970. Hệ hỗ trợ ra quyết định có:

- Phần mềm máy tính
- Chức năng hỗ trợ ra quyết định.
- Làm việc với các bài toán có cấu trúc yếu
- Hoạt động theo cách tương tác với người dùng
- Được trang bị nhiều mô hình phân tích và mô hình dữ liệu

Hệ hỗ trợ quyết định có các tính chất:

- Hướng đến các quyết định cấp cao của các nhà lãnh đạo
- Tính uyển chuyển, thích ứng với hoàn cảnh và phản ứng nhanh
- Do người dùng khởi động và kiểm soát
- Ngoài việc cung cấp các dạng hỗ trợ quyết định thường gặp, hệ quyết định còn được trang bị khả năng trả lời các câu hỏi để giải

quyết các tính huống dưới dạng câu hỏi “if-then”

Trong chương 6, chúng ta sẽ tìm hiểu các hệ hỗ trợ ra quyết định.

1.6. HỆ GIẢI BÀI TOÁN

Mạng tính toán là một dạng biểu diễn tri thức, mỗi mạng tính toán là một mạng ngữ nghĩa chứa các biến và những quan hệ có thể cài đặt và sử dụng được cho việc tính toán. Mạng tính toán gồm một tập hợp các biến cùng với một tập các quan hệ (chẳng hạn các công thức) tính toán giữa các biến. Trong ứng dụng cụ thể mỗi biến và giá trị của nó thường gắn liền với một khái niệm cụ thể về sự vật, mỗi quan hệ thể hiện một sự tri thức về sự vật. Nhờ mạng tính toán có thể biểu diễn tri thức tính toán dưới dạng các đối tượng một cách tự nhiên và gần gũi đối với cách nhìn và nghĩ của con người khi giải quyết các vấn đề tính toán liên quan đến một số khái niệm về các đối tượng, chẳng hạn như các tam giác, tứ giác, hình bình hành, hình chữ nhật, v.v....Sau đó phát triển các thuật giải trên mạng tính toán để hỗ trợ tiến trình giải các bài toán.

1.7. TIẾP THU TRI THỨC

Nhu cầu tìm kiếm các tri thức từ dữ liệu của một lĩnh vực cụ thể là một nhu cầu bắt buộc khi xây dựng các hệ CSTT. Một số bài toán đã có sẵn tri thức, tuy vậy có nhiều lĩnh vực rất khó phát hiện các tri thức. Do vậy cần phát triển các kỹ thuật cho phép tiếp nhận tri thức từ dữ liệu. Máy học là một trong các nghiên cứu giúp tạo ra tri thức từ dữ liệu. Trong chương 7, một số thuật giải học trên cây định danh, thuật giải qui nạp ILA được trình bày nhằm hỗ trợ tiến trình phân tích dữ liệu và tạo ra tri thức.

1.8. TÍCH HỢP CÁC HỆ CSTT VÀ CÁC HỆ QUẢN TRỊ CSDL

Có thể áp dụng cơ chế CSTT và cơ chế lập luận để nâng cao các khả năng cung cấp thông tin của các CSDL hiện có. Một ví dụ tiêu biểu là trong CSDL về hành trình của các con tàu xuất phát từ cảng. Dựa trên các thông tin lưu trữ trong CSDL về giờ xuất phát và các qui luật hải hành có thể rút ra vị trí hiện tại của con tàu. Rõ ràng điều này không thể làm được với các câu lệnh SQL truyền thống. Tuy

vậy khi đưa các luật suy diễn vào CSDL, có thể dễ dàng tạo sinh thêm thông tin dựa trên các sự kiện cung cấp, các dữ liệu đang được lưu trữ trong CSDL và các luật, cơ chế suy diễn trong CSTT,

1.9. HỆ THỐNG ĐIỀU KHIỂN MỜ

Trong chương 9 sẽ trình bày các khái niệm liên quan đến tập mờ như khái niệm tập mờ và hàm thành viên, luật mờ và suy diễn mờ, các thành phần của một hệ thống mờ từ giai đoạn giải mờ, lập luận mờ, giai đoạn từ tập mờ chuyển sang trị rõ. Một số ứng dụng của các hệ thống điều khiển mờ được trình bày bao gồm tập các tập mờ, hàm thành viên, luật mờ và các tiến trình của hệ thống điều khiển mờ.

cuu duong than cong. com

CHƯƠNG 2. BIỂU DIỄN TRI THỨC 9

2.1. DẪN NHẬP

2.2. CÁC LOẠI TRI THỨC

2.3. CÁC KỸ THUẬT BIỂU DIỄN TRI THỨC

2.3.1. Bộ ba Đối tượng-Thuộc tính-Giá trị

2.3.2. Các luật dẫn

2.3.2.1. Các dạng luật cơ bản

2.3.2.2. Mở rộng cho các luật

2.3.3. Mạng ngữ nghĩa

2.3.4. Frame

2.3.5. Logic

2.3.5.1. Logic mệnh đề

2.3.5.2. Logic vị từ

2.1. MỞ ĐẦU

Việc biểu diễn tri thức đóng vai trò hết sức quan trọng trong việc khẳng định khả năng giải quyết vấn đề của một hệ cơ sở tri thức. Để hiểu rõ điều này, ta hãy tìm hiểu về mối liên hệ giữa *tri thức*, *lĩnh vực* và *biểu diễn tri thức*.

Tri thức là sự hiểu biết về một vấn đề nào đó, ví dụ hiểu biết về y khoa. Tuy nhiên, trong thực tế, tri thức của một hệ chuyên gia thường gắn liền với một lĩnh vực xác định, chẳng hạn như hiểu biết về các căn bệnh nhiễm trùng máu. Mức độ hỗ trợ (thành công) của một hệ chuyên gia phụ thuộc vào miền hoạt động của nó. Thế nhưng, cách thức tổ chức các tri thức như thế nào sẽ quyết định lĩnh vực hoạt động của chúng. Với cách biểu diễn hợp lý, ta có thể giải quyết các vấn đề đưa vào theo các đặc tính có liên quan đến tri thức đã có.

2.2. CÁC LOẠI TRI THỨC

Dựa vào cách thức con người giải quyết vấn đề, các nhà nghiên cứu đã xây dựng các kỹ thuật để biểu diễn các dạng tri thức khác nhau trên máy tính. Mặc dù vậy, không một kỹ thuật riêng lẻ nào có thể giải thích đầy đủ cơ chế tổ chức tri thức trong các chương trình máy tính. Để giải quyết vấn đề, chúng ta chỉ chọn dạng biểu diễn nào thích hợp nhất. Sau đây là các dạng biểu diễn tri thức thường gặp.

➤ **Tri thức thủ tục** mô tả cách thức giải quyết một vấn đề. Loại tri thức này đưa ra giải pháp để thực hiện một công việc nào đó. Các dạng tri thức thủ tục tiêu biểu thường là các luật, chiến lược, lịch trình, và thủ tục.

➤ **Tri thức khai báo** cho biết một vấn đề được thấy như thế nào. Loại tri thức này bao gồm các phát biểu đơn giản, dưới dạng các khẳng định logic đúng hoặc sai. Tri thức khai báo cũng có thể là một danh sách các khẳng định nhằm mô tả đầy đủ hơn về đối tượng hay một khái niệm khái niệm nào đó.

➤ **Siêu tri thức** mô tả *tri thức về tri thức*. Loại tri thức này giúp lựa chọn tri thức thích hợp nhất trong

số các tri thức khi giải quyết một vấn đề. Các chuyên gia sử dụng tri thức này để điều chỉnh hiệu quả giải quyết vấn đề bằng cách hướng các lập luận về miền tri thức có khả năng hơn cả.

➤ **Tri thức heuristic** mô tả các "*mẹo*" để dẫn dắt tiến trình lập luận. Tri thức heuristic còn được gọi là *tri thức nông cạn* do không đảm bảo hoàn toàn chính xác về kết quả giải quyết vấn đề. Các chuyên gia thường dùng các tri thức khoa học như sự kiện, luật, ... sau đó chuyển chúng thành các tri thức heuristic để thuận tiện hơn trong việc giải quyết một số bài toán.

➤ **Tri thức có cấu trúc** mô tả tri thức theo cấu trúc. Loại tri thức này mô tả mô hình tổng quan hệ thống theo quan điểm của chuyên gia, bao gồm khái niệm, khái niệm con, và các đối tượng; diễn tả chức năng và mối liên hệ giữa các tri thức dựa theo cấu trúc xác định.

2.3. CÁC KỸ THUẬT BIỂU DIỄN TRI THỨC

Phần này trình bày các kỹ thuật phổ biến nhất để biểu diễn tri thức, bao gồm:

Bộ ba Đối tượng-Thuộc tính-Giá trị.

Các luật dẫn.

Mạng ngữ nghĩa.

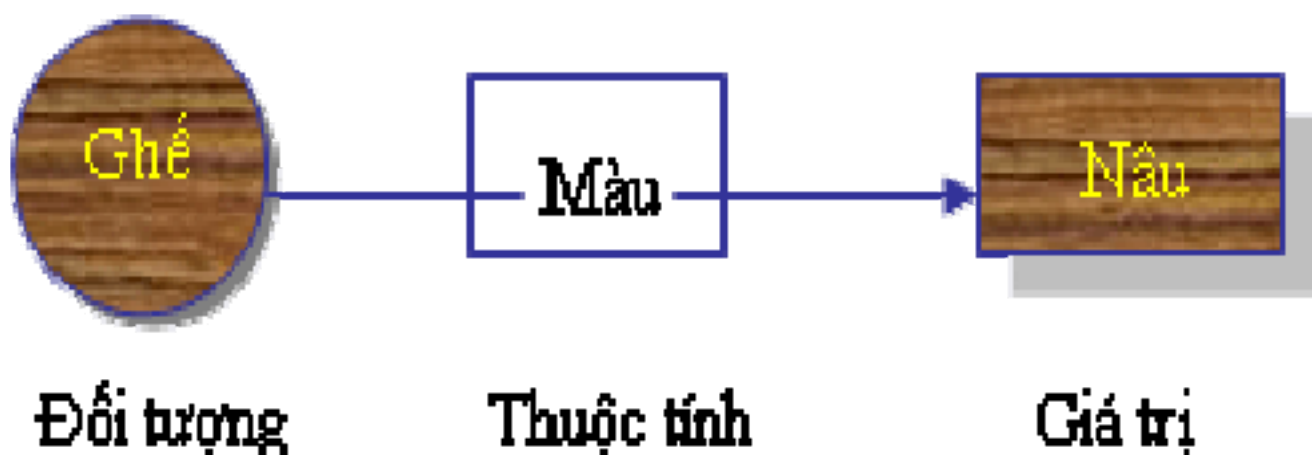
Frames.

Logic.

♦ 2.3.1. Bộ ba Đối tượng-Thuộc tính-Giá trị

Cơ chế tổ chức nhận thức của con người thường được xây dựng dựa trên các *sự kiện* (fact), xem như các đơn vị cơ bản nhất. Một sự kiện là một dạng tri thức khai báo. Nó cung cấp một số hiểu biết về một biến cố hay một vấn đề nào đó.

Một sự kiện có thể được dùng để xác nhận giá trị của một thuộc tính xác định của một vài đối tượng. Ví dụ, mệnh đề "quả bóng màu đỏ" xác nhận "đỏ" là giá trị thuộc tính "màu" của đối tượng "quả bóng". Kiểu sự kiện này được gọi là bộ ba Đối tượng-Thuộc tính-Giá trị (O-A-V – Object-Attribute-Value).



Hình 2.1. Biểu diễn tri thức theo bộ ba O-A-V

Một O-A-V là một loại mệnh đề phức tạp. Nó chia một phát biểu cho trước thành ba phần riêng biệt: đối tượng, thuộc tính, giá trị thuộc tính. Hình 0.1 minh họa cấu trúc bộ ba O-A-V.

Trong các sự kiện O-A-V, một đối tượng có thể có nhiều thuộc tính với các kiểu giá trị khác nhau. Hơn nữa một thuộc tính cũng có thể có một hay nhiều giá trị. Chúng được gọi là các sự kiện *đơn trị* (single-valued) hoặc *đa trị* (multi-valued). Điều này cho phép các hệ tri thức linh động trong việc biểu diễn các tri thức cần thiết.

Các sự kiện không phải lúc nào cũng bảo đảm là đúng hay sai với độ chắc chắn hoàn toàn. Ví thế, khi xem xét các sự kiện, người ta còn sử dụng thêm một khái niệm là *độ tin cậy*. Phương pháp truyền

thống để quản lý thông tin không chắc chắn là sử dụng nhân tố chắc chắn CF (certainly factor). Khái niệm này bắt đầu từ hệ thống MYCIN (khoảng năm 1975), dùng để trả lời cho các thông tin suy luận. Khi đó, trong sự kiện O-A-V sẽ có thêm một giá trị xác định độ tin cậy của nó là CF.

Ngoài ra, khi các sự kiện mang tính "nhập nhằng", việc biểu diễn tri thức cần dựa vào một kỹ thuật, gọi là logic mờ (do Zadeh đưa ra năm 1965). Các thuật ngữ nhập nhằng được thể hiện, lượng hoá trong *tập mờ*.

♦ 2.3.2. Các luật dẫn

Luật là cấu trúc tri thức dùng để liên kết thông tin đã biết với các thông tin khác giúp đưa ra các suy luận, kết luận từ những thông tin đã biết.

Trong hệ thống dựa trên các luật, người ta thu thập các tri thức lĩnh vực trong một tập và lưu chúng trong cơ sở tri thức của hệ thống. Hệ thống dùng các luật này cùng với các thông tin trong bộ nhớ để giải bài toán. Việc xử lý các luật trong hệ thống dựa trên các luật được quản lý bằng một module gọi là *bộ suy diễn*.

► 2.3.2.1. Các dạng luật cơ bản

Các luật thể hiện tri thức có thể được phân loại theo loại tri thức. Và như vậy, có các lớp luật tương ứng với dạng tri thức như *quan hệ*, *khuyến cáo*, *hướng dẫn*, *chiến lược*, và *heuristic*. Các ví dụ sau minh họa cho các loại luật.

• Quan hệ

IF Bình điện hỏng

THEN Xe sẽ không khởi động được

• Lời khuyên

IF Xe không khởi động được

THEN Đi bộ

• Hướng dẫn

IF Xe không khởi động được

AND Hệ thống nhiên liệu tốt

THEN Kiểm tra hệ thống điện

• Chiến lược

IF Xe không khởi động
được

THEN Đầu tiên hãy kiểm
tra hệ thống nhiên liệu, sau đó kiểm tra hệ thống
điện

Các luật cũng có thể được phân loại theo cách
thức giải quyết vấn đề. Diễn hình theo phân loại
này các luật theo cách thức diễn giải, chẩn
đoán, và thiết kế.

• **Diễn giải**

IF Cao 1m65

AND Nặng 65 kg

THEN Phát triển bình thường

• **Chẩn đoán**

IF Sốt cao

AND hay ho

AND Họng đỏ

THEN Viêm họng

• Thiết kế

IF Cao 1m75

AND Da sẫm

THEN Chọn áo vải sáng

AND Chọn tấm vải khổ 1m40

► 2.3.2.2. Mở rộng cho các luật

Trong một số áp dụng cần thực hiện cùng một phép toán trên một tập hay các đối tượng giống nhau. Lúc đó cần các *luật có biến*.

Ví dụ:

IF X là nhân viên

AND Tuổi của X > 65

THEN X xó thể nghỉ hưu

Khi mệnh đề phát biểu về sự kiện, hay bản thân sự kiện có thể không chắc chắn, người ta dùng hệ số chắc chắn CF. Luật thiết lập quan hệ không chính xác giữa các sự kiện giả thiết và kết luận được gọi là *luật không chắc chắn*.

Ví dụ:

IF Lạm phát CAO

THEN Hầu như chắc chắn lãi suất sẽ CAO

Luật này được viết lại với giá trị CF có thể như sau:

IF Lạm phát cao

THEN Lãi suất cao, $CF = 0.8$

Dạng luật tiếp theo là *siêu luật* - một luật với chức năng mô tả cách thức dùng các luật khác. Siêu luật sẽ đưa ra chiến lược sử dụng các luật theo lĩnh vực chuyên dụng, thay vì đưa ra thông tin mới.

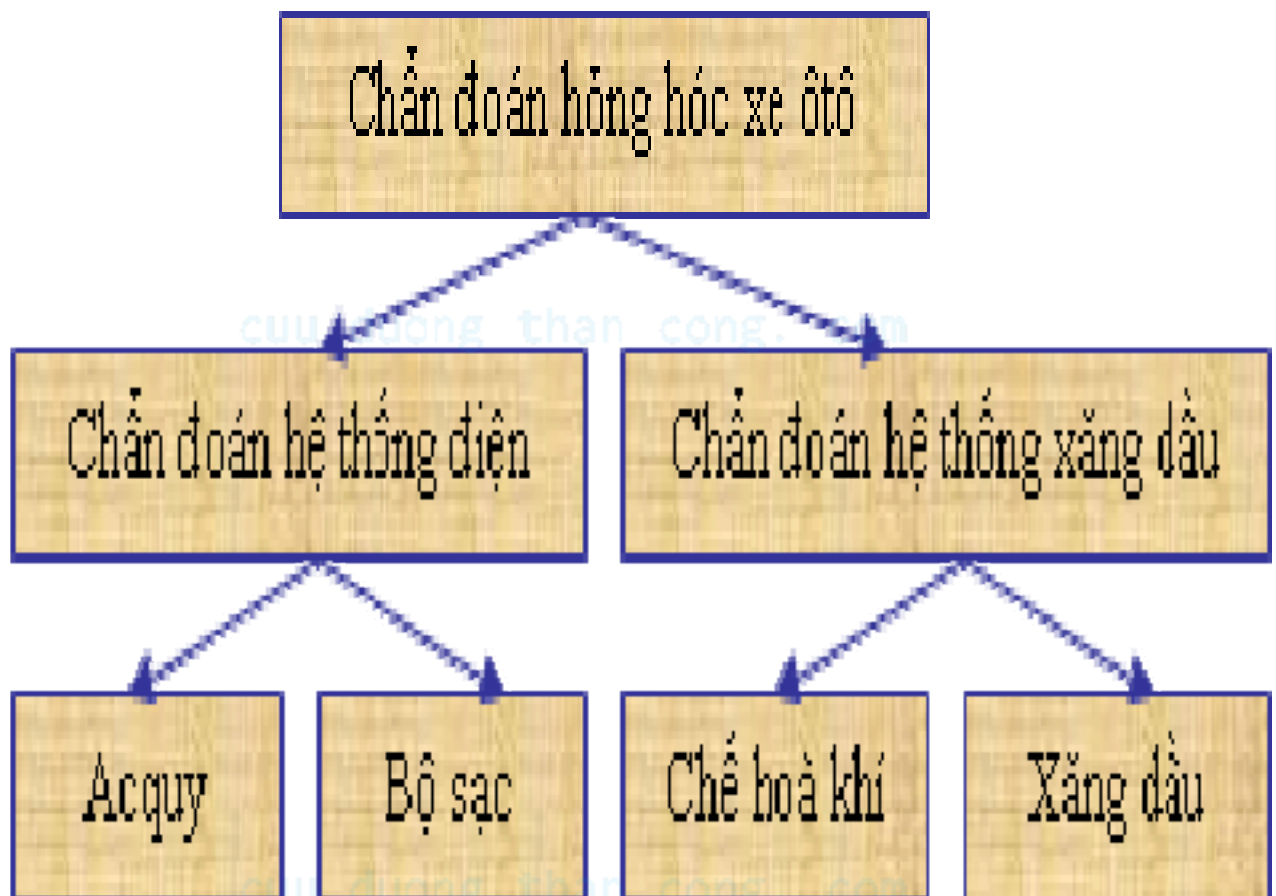
Ví dụ:

IF Xe không khởi động

AND Hệ thống điện làm việc bình thường

THEN Có thể sử dụng các luật liên quan đến hệ thống điện

Qua kinh nghiệm, các chuyên gia sẽ đề ra một *tập các luật* áp dụng cho một bài toán cho trước. Ví dụ tập luật trong hệ thống chẩn đoán hỏng hóc xe ô tô. Điều này giúp giải quyết các trường hợp mà khi chỉ với các luật riêng, ta không thể lập luận và giải quyết cho một vấn đề.

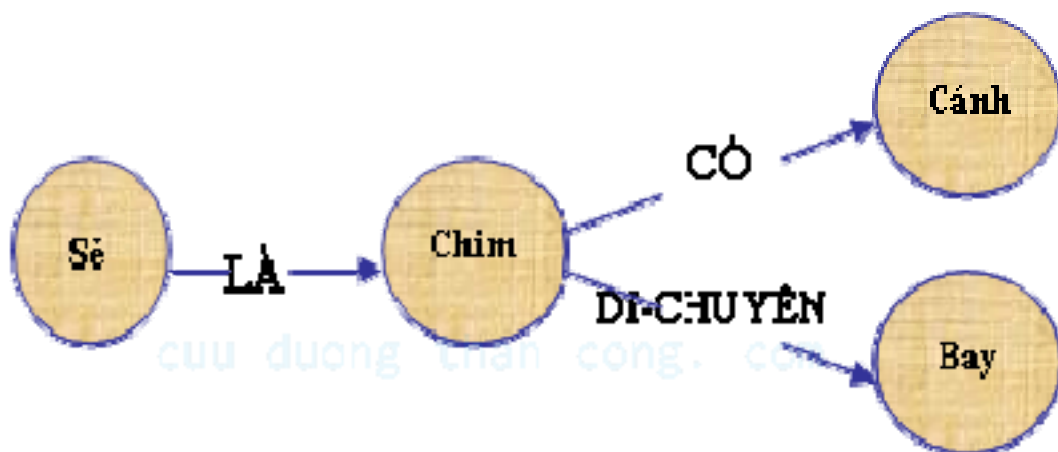


Hình 2.2. Tập các luật liên quan đến việc hỏng xe

Một nhu cầu đặt ra trong các hệ thống tri thức là sự hợp tác giữa các chuyên gia. Trên phương diện tổ chức hệ thống, ta có thể sử dụng một cấu trúc được gọi là *bảng đen*, dùng để liên kết thông tin giữa các luật tách biệt, thông qua các module với các nhiệm vụ tách biệt. Dạng hệ thống này được Erman đưa ra lần đầu tiên vào năm 1980 áp dụng cho hệ chuyên gia hiểu biết tiếng nói HEARSAY-II.

♦ 2.3.3. Mạng ngữ nghĩa

Mạng ngữ nghĩa là một phương pháp biểu diễn tri thức dùng đồ thị trong đó nút biểu diễn đối tượng và cung biểu diễn quan hệ giữa các đối tượng.



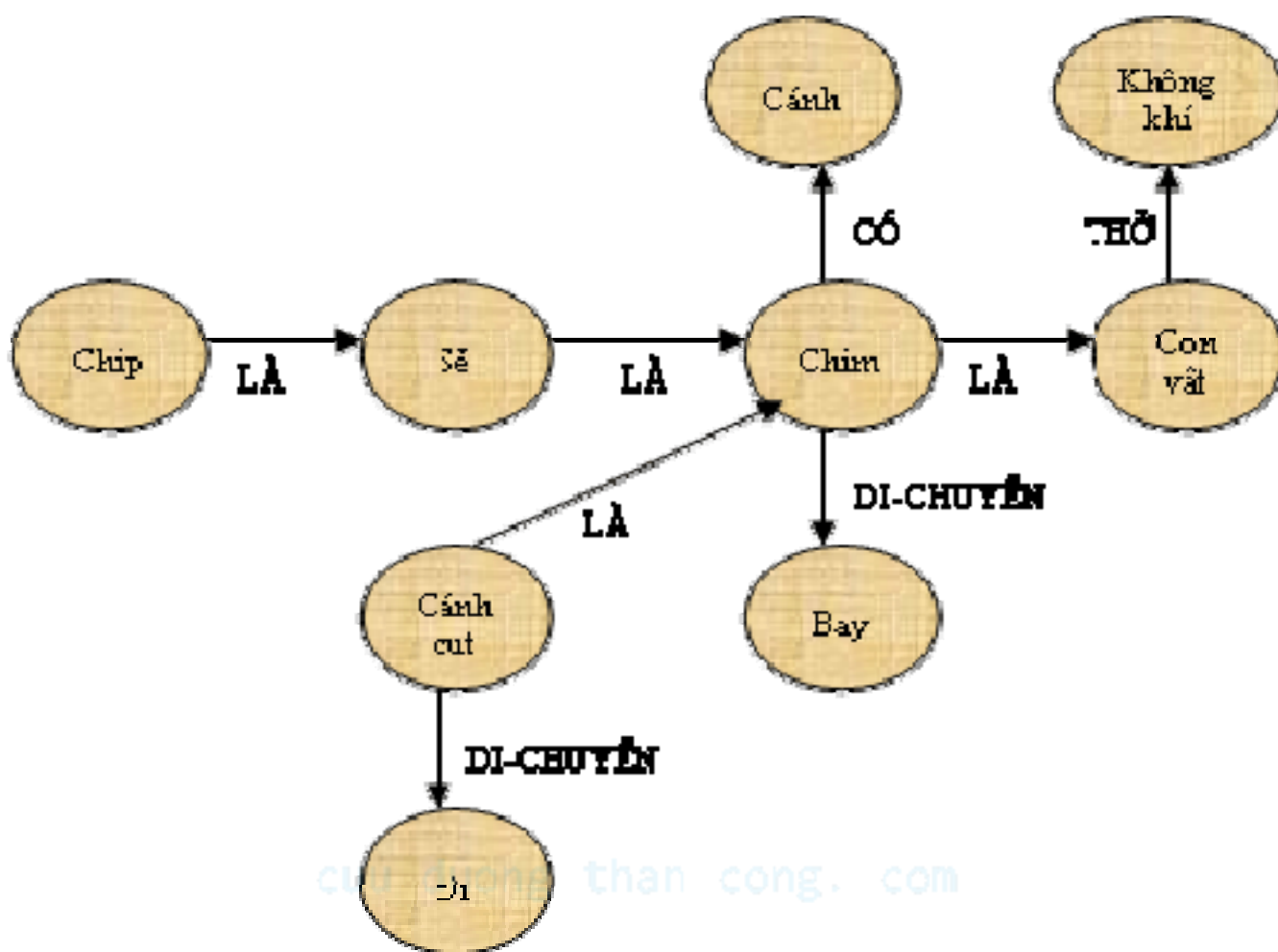
Hình 2.3. "Sẻ là Chim" thể hiện trên mạng ngữ nghĩa

Người ta có thể nói rộng mạng ngữ nghĩa bằng cách thêm các nút và nối chúng vào đồ thị. Các nút mới ứng với các đối tượng bổ sung. Thông thường có thể nói rộng mạng ngữ nghĩa theo ba cách:

- Thêm một đối tượng tương tự.
- Thêm một đối tượng đặc biệt hơn.
- Thêm một đối tượng tổng quát hơn

Thứ nhất, thêm "Cánh cụt" thể hiện một loại chim mới. *Thứ hai*, thêm "Chip" cũng có nghĩa nó là con "Sẻ" và đồng thời là "Chim". *Thứ ba*, có thể đưa ra đối tượng tổng quát như "Con vật". Lúc này, không những có thể biết được rằng "Chim là Con vật", mà còn biết "Chip thở bằng không khí".

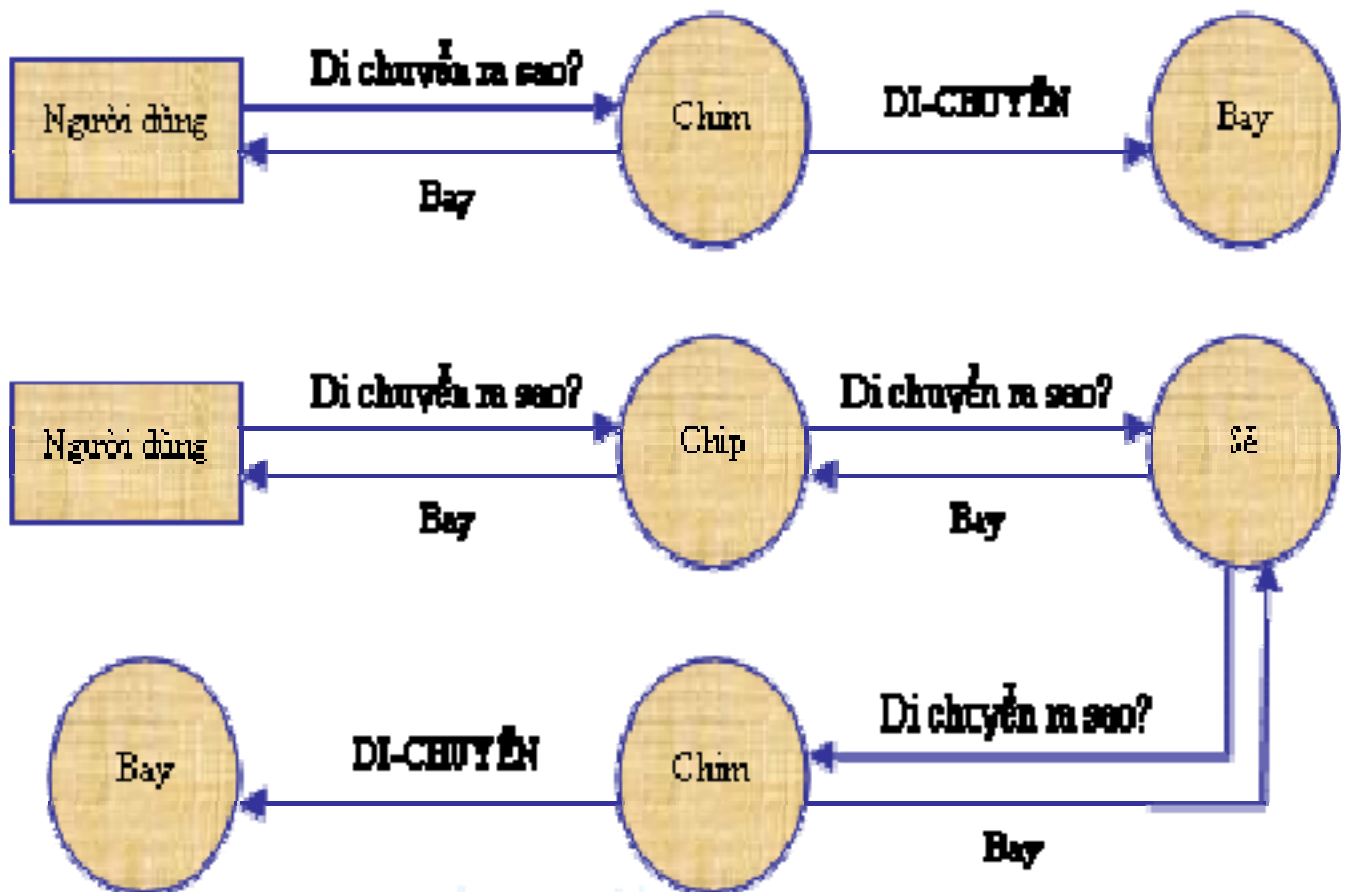
cuu duong than cong. com



Hình 2.4. Phát triển mạng ngữ nghĩa

Tính chất quan trọng của mạng ngữ nghĩa là tính kế thừa. Nó cho phép các nút được bổ sung sẽ nhận các thông tin của các nút đã có trước, và cho phép mã hóa tri thức một cách dễ dàng.

Để minh họa cho tính kế thừa của mạng ngữ nghĩa, hãy xét một câu hỏi trên đồ thị. Chẳng hạn tại nút "Chim", người ta muốn hỏi con "Chip" hoạt động như thế nào? Thông qua cung hoạt động người ta biết được nó bay.



Hình 2.5. Các bước thực hiện phép toán trên mạng ngữ nghĩa

♦ 2.3.4. Frame

Một trong các kỹ thuật biểu diễn tri thức là dùng frame, phát triển từ khái niệm *lược đồ*. Một lược đồ được coi là khối tri thức điển hình về khái niệm hay đối tượng nào đó, và gồm cả tri thức thủ tục lẫn tri thức mô tả.

Theo định nghĩa của Minsky (1975), thì frame là cấu trúc dữ liệu để thể hiện tri thức đa dạng về khái niệm hay đối tượng nào đó.

PHIẾU ĐIỂM	
Họ tên:	<input type="text"/>
Địa chỉ:	<input type="text"/>
Môn	Điểm
Vật lý	
Toán	
...	

Tên frame:	<input type="text"/>								
Lớp:	<input type="text"/>								
Thuộc tính:	<table border="1"><thead><tr><th>Thuộc tính 1</th><th>Giá trị 1</th></tr></thead><tbody><tr><td>Thuộc tính 2</td><td>Giá trị 2</td></tr><tr><td>...</td><td>...</td></tr><tr><td>...</td><td>...</td></tr></tbody></table>	Thuộc tính 1	Giá trị 1	Thuộc tính 2	Giá trị 2
Thuộc tính 1	Giá trị 1								
Thuộc tính 2	Giá trị 2								
...	...								
...	...								

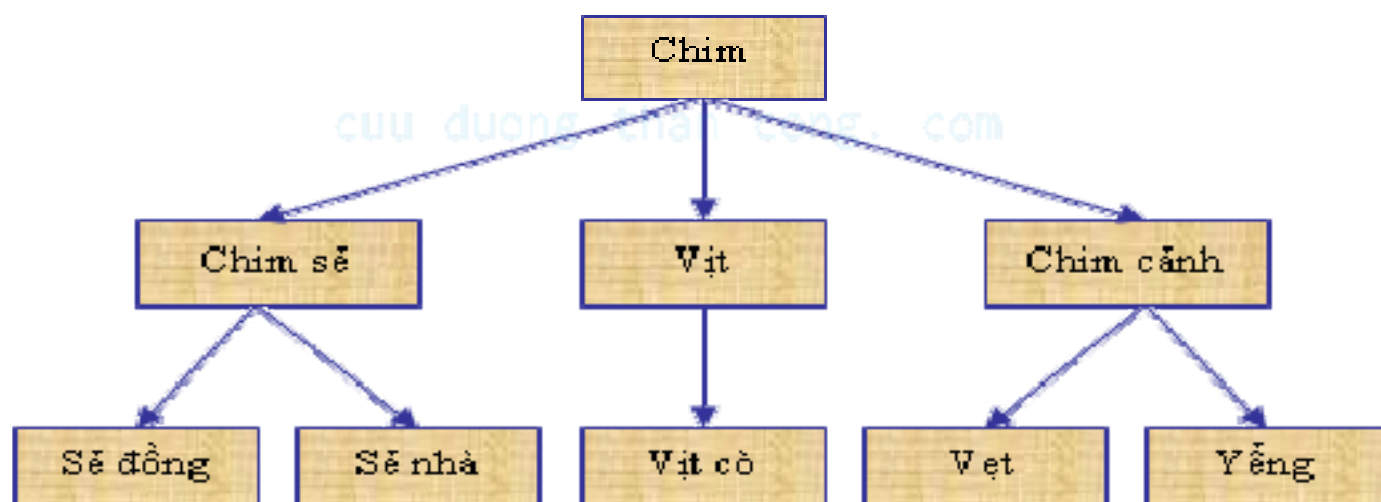
Hình 2.6. Cấu trúc frame

Một frame có hình thức như bảng mẫu, như tờ khai cho phép người ta điền các ô trống. Cấu trúc cơ bản của frame có *tên* đối tượng được thể hiện trong frame, có các trường thuộc tính của đối tượng. Mỗi thuộc tính có một ngăn để nhập dữ liệu riêng. Các thuộc tính và giá trị thuộc tính tạo nên danh sách các mệnh đề O-A-V, cho phép thể hiện đầy đủ về đối tượng.

Một *frame lớp* thể hiện các tính chất tổng quát của tập các đối tượng chung. Chẳng hạn người ta cần mô tả các tính chất tổng quát như bay, có cánh, sống tự do,... của cả loài chim.

Để mô tả một biểu diễn của frame lớp, ta dùng một dạng frame khác, gọi là *frame thể hiện*. Khi tạo ra thể hiện của một lớp, frame này kế thừa tính chất và giá trị của lớp. Có thể thay đổi giá trị để phù hợp với biểu diễn cụ thể. Thậm chí, ta cũng có thể thêm các tính chất khác đối với frame thể hiện.

Cũng như tính chất kế thừa giữa các đối tượng trong mạng ngữ nghĩa, frame thể hiện nhận giá trị kế thừa từ frame lớp. Khi tạo một frame thể hiện, người ta khẳng định frame đó là thể hiện của một frame lớp. Khẳng định này cho phép nó kế thừa các thông tin từ frame lớp.



Hình 2.7. Nhiều mức của frame mô tả quan hệ phức tạp hơn

Ngoài các frame lớp đơn giản và các thể hiện gắn với nó, người ta có thể tạo ra cấu trúc frame phức tạp. Ví dụ, dùng cấu trúc phân cấp các frame để mô tả thế giới loài chim. Cấu trúc này tổ chức khái niệm về chim theo các mức trừu tượng khác nhau. Frame ở mức cao mang thông tin chung về tất cả loài chim. Mức giữa có frame lớp con, mang thông tin đặc thù hơn của nhóm chim. Mức cuối cùng là frame thể hiện, ứng với đối tượng cụ thể.

♦2.3.5. Logic

Dạng biểu diễn tri thức cổ điển nhất trong máy tính là logic, với hai dạng phổ biến là logic mệnh đề và logic vị từ. Cả hai kỹ thuật này đều dùng *ký hiệu* để thể hiện tri thức và các *toán tử* áp lên các ký hiệu để suy luận logic. Logic đã cung cấp cho các nhà nghiên cứu một công cụ hình thức để biểu diễn và suy luận tri thức.

Phép toán	AND	OR	NOT	Kéo theo	Tương đương
Kí hiệu	\wedge , $\&$ \cap	\vee , \cup , $+$	\neg , \sim	\supset , \rightarrow	\equiv

Bảng 2.1. Các phép toán logic và các ký hiệu sử dụng

► 2.3.5.1. Logic mệnh đề

Logic mệnh đề biểu diễn và lập luận với các mệnh đề toán học. Mệnh đề là một câu nhận giá trị hoặc đúng hoặc sai. Giá trị này gọi là chân trị của mệnh đề. Logic mệnh đề gán một biến ký hiệu vào một mệnh đề, ví dụ $A = \text{"Xe sẽ khởi động"}$.

Khi cần kiểm tra trị chân trị của câu trên trong bài toán sử dụng logic mệnh đề, người ta kiểm tra giá trị của A . Nhiều bài toán sử dụng logic mệnh đề để thể hiện tri thức và giải vấn đề. Bài toán loại này được đưa về bài toán xử lý các luật, mỗi phần giả thiết và kết luận của luật có thể có nhiều mệnh đề.

Ví dụ:

IF Xe không khởi động được ; ; $\rightarrow A$

AND Khoảng cách từ nhà đến chỗ làm là xa $\rightarrow B$

THEN Sẽ trễ giờ làm ; ; ; $\rightarrow C$

Luật trên có thể biểu diễn lại như sau: $A \wedge B \rightarrow C$.

Các phép toán quen thuộc trên các mệnh đề được cho trong bảng 2.2.

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \equiv B$
T	T	F	T	T	T	T
F	T	T	F	T	T	F
T	F	F	F	T	F	F
F	F	T	F	F	T	T

Bảng 2.2. Bảng chân trị, với các giá trị Đúng (T), Sai (F)

► 2.3.5.2. Logic vị từ

Logic vị từ là sự mở rộng của logic mệnh đề nhằm cung cấp một cách biểu diễn rõ hơn về tri thức. Logic vị từ dùng ký hiệu để biểu diễn tri thức.

Logic vị từ, cũng giống như logic mệnh đề, dùng các ký hiệu để thể hiện tri thức. Những ký hiệu này gồm **hằng số**, **vị từ**, **biến** và **hàm**.

➤ **Hằng số:** Các hằng số dùng để đặt tên các đối tượng đặc biệt hay thuộc tính. Nhìn chung, các hằng số được ký hiệu bằng chữ viết thường, chẳng hạn **an**, **bình**, **nhật độ**. Hằng số **an** có thể được dùng để thể hiện đối tượng **An**, một người đang xét.

➤ **Vị từ:** Một mệnh đề hay sự kiện trong logic vị từ được chia thành 2 phần là *vị từ* và *tham số*. Tham số thể hiện một hay nhiều đối tượng của mệnh đề; còn mệnh đề dùng để khẳng định về đối tượng. Chẳng hạn mệnh đề "Nam thích Mai" viết theo vị từ sẽ có dạng:

thích(nam, mai)

Với cách thể hiện này, người ta dùng từ đầu tiên, tức "thích", làm vị từ. Vị từ cho biết quan hệ giữa các đối số đặt trong ngoặc. Đối số là các ký hiệu thay cho các đối tượng của bài toán. Theo quy ước chuẩn, người ta dùng các chữ thường để thể hiện các đối số.

➡ **Biến:** Các biến dùng để thể hiện các lớp tổng quát của các đối tượng hay thuộc tính. Biến được viết bằng các ký hiệu bắt đầu là chữ in hoa. Như vậy, có thể dùng vị từ có biến để thể hiện nhiều vị từ tương tự.

Ví dụ:

Có hai mệnh đề tương tự "Nam thích Mai" và "Bắc thích Cúc". Hai biến X, Y dùng trong mệnh đề **thích(X, Y)**.

Các biến nhận giá trị sẽ được thể hiện qua $X = \text{Nam, Bắc}$; $Y = \text{Mai, Cúc}$. Trong phép toán vị từ người ta dùng biến như đối số của biểu thức vị từ hay của hàm.

➡ **Hàm:** Logic vị từ cũng cho phép dùng ký hiệu để biểu diễn hàm. Hàm mô tả một ánh

xạ từ các thực thể hay một tập hợp đến một phần tử duy nhất của tập hợp khác. Ví dụ, các hàm sau đây được định nghĩa nhằm trả về một giá trị xác định:

$\text{cha}(\text{sơn}) = \text{Nam}$

$\text{mẹ}(\text{sơn}) = \text{Mai}$

• **Phép toán:** Logic vị từ cũng dùng các phép toán như logic mệnh đề.

Ví dụ:

$\text{thích}(X,Y) \text{ AND } \text{thích}(Z,Y) \rightarrow \neg \text{thích}(X,Z).$

Việc lập luận theo cách không hình thức đòi hỏi một khả năng rút ra được kết luận từ các sự kiện đã có. Việc lấy ra thông tin mới từ các thông tin đã biết và các luật là trọng tâm của lập luận trong các hệ chuyên gia. Quá trình lập luận được hình thức hoá trong bài toán suy luận.

CHƯƠNG 3. CÁC KỸ THUẬT SUY DIỄN VÀ LẬP LUẬN

3.3.1. Modus ponens

3.3.2. Suy diễn tiến/lùi

3.3.2.1. Giới thiệu

3.3.2.2. Suy diễn tiến

3.3.2.3. Suy diễn lùi

3.3.2.4. Ưu nhược điểm của các kỹ thuật suy diễn

3.4. MỘT CÀI ĐẶT CƠ CHẾ GIẢI THÍCH VỚI LẬP LUẬN SUY DIỄN LÙI

3.4.1. Xây dựng một Cơ sở tri thức

3.4.2. Cài đặt Động cơ suy diễn bằng cơ chế lập luận lùi

3.4.3. Cài đặt Cơ chế giải thích trong Suy diễn lùi

3.1. MỞ ĐẦU

Để giải bài toán trong trí tuệ nhân tạo, tối thiểu cần thiết việc thể hiện tri thức. Rồi cần có hệ thống suy lý trên các tri thức. Trong hệ thống như hệ chuyên gia, việc suy lý thể hiện thông qua kỹ thuật suy diễn và các chiến lược điều khiển. Các kỹ thuật suy diễn hướng dẫn hệ thống theo cách tổng hợp tri thức từ các tri thức đã có trong cơ sở tri thức và từ sự kiện ghi lại trong bộ nhớ. Các chiến lược điều khiển thiết lập đích cần đến và hướng dẫn hệ thống suy lý.

3.2. SUY LÝ

Con người giải bài toán bằng cách kết hợp các sự kiện với các tri thức. Họ dùng các sự kiện riêng về bài toán và dùng chúng trong ngữ cảnh hiểu tổng thể về lĩnh vực của bài toán để rút ra các kết luận logic. Quá trình này gọi là suy lý. Như vậy " Suy lý là quá trình làm việc với tri thức, sự kiện, và các chiến lược giải bài toán để rút ra kết luận".

Hiểu cách con người suy lý và cách họ làm việc với thông tin về loại bài toán đã cho, cộng với kiến thức của họ về lĩnh vực này sẽ đảm bảo hiểu rõ các

bước đi trong quá trình xử lý tri thức trong hệ thống tri thức nhân tạo.

♦3.2.1. Suy lý theo cách suy diễn

Con người suy lý suy diễn để rút ra thông tin mới từ các thông tin đã biết. Các thông tin này có quan hệ logic với nhau. Suy lý suy diễn dùng các sự kiện của bài toán gọi là các tiên đề và các kiến thức chung có liên quan ở dạng các luật gọi là các kéo theo.

Ví dụ : Kéo theo: Tôi sẽ ướt nếu tôi đứng dưới mưa.

Tiên đề: Tôi đứng dưới mưa.

Kết luận: Tôi sẽ ướt.

Suy lý suy diễn là một trong các kỹ thuật phổ biến nhất. Suy diễn là dùng modus ponens, là loại cơ bản của suy lý suy diễn. Khi có $A \rightarrow B$ và A đúng thì rút ra được B đúng.

♦3.2.2. Suy lý quy nạp

Con người dùng suy lý quy nạp để rút ra kết luận tổng quát từ một tập các sự kiện theo các tổng quát hóa.

Ví dụ: Giả thiết: Con khỉ ở vườn thú Hà Nội ăn chuối.

Giả thiết: Con khỉ ở vườn thú Cần Thơ ăn chuối.

Giả thiết: Nói chung, khỉ ăn chuối.

Qua suy lý này, người ta cho rằng kết luận sẽ đúng cho tất cả các trường hợp cùng loại, dựa trên một số hạn chế của các trường hợp. Thực chất của suy lý quy nạp là đem cái thiếu số áp dụng cho đa số. Năm 1988 Firebaugh mô tả quá trình như sau: "Cho tập các đối tượng $X = \{a, b, c, \dots\}$, nếu tính chất P đúng với a, và nếu tính chất P cũng đúng với b, và nếu tính chất P cũng đúng với c, thì tính chất này đúng với tất cả X.

♦ 3.2.3. Suy lý giả định

Suy diễn là suy lý chính xác từ các sự kiện và thông tin đã biết. Suy lý giả định (abductive) là một loại suy diễn có vẻ hợp lý. Điều này có nghĩa câu

kết luận có thể đúng, nhưng cũng có thể không đúng.

Ví dụ: Kéo theo: Đất ướt nếu trời mưa.

Tiên đề: Đất ướt.

Kết luận : Trời mưa?

Kết luận "trời mưa ?" cho rằng có thể trời mưa, cũng có thể không phải trời mưa mà "đã ướt" xảy ra vì lý do khác.

♦3.2.4. Suy lý tương tự, loại suy

Người ta tạo ra một mô hình của một vài khái niệm thông qua kinh nghiệm của họ. Họ dùng mô hình này để hiểu một vài hoàn cảnh và đối tượng tương tự, họ vạch ra điểm tương tự giữa hai vật đem ra so sánh, rút ra sự giống nhau và khác nhau nhằm hướng dẫn việc suy lý của họ.

Ví dụ: Khung: Con hổ

Chủng loại : thú vật

Ăn : thịt

Sống tại : Ấn Độ và Đông Nam Á

Màu: Vàng có vạch

Một khung cho biết thông tin đa dạng về đối tượng. Người ta có thể dùng khung để thể hiện những nét điển hình của các đối tượng tương tự. Nếu cho rằng Sư tử giống Hổ thì Sư tử cũng có nhiều tính chất như trên. Loại suy lý này dùng để hiểu biết về đối tượng mới và để hiểu rõ thêm bằng cách tra cứu đến những sự khác biệt giữa các đối tượng. Trong ví dụ này, Sư tử được phân biệt với Hổ do các nét khác nhau giữa chúng.

♦ 3.2.5. Suy lý theo lẽ thường

Nhờ kinh nghiệm, con người có thể giải quyết vấn đề một cách có hiệu quả. Họ sử dụng lẽ thông thường (common sense) để nhanh chóng rút ra kết luận. Suy hướng theo lẽ thường có khuynh hướng thiên về phán xét sự đúng đắn hơn là suy lý chính xác về logic.

Ví dụ: Vấn đề chẩn đoán hỏng hóc xe hơi : " xiết quạt lỏng thường gây ra tiếng ồn". Kết luận này có được do kinh nghiệm sửa nhiều xe hơi. Người ta đoán ngay “xiết quạt lỏng” khi thấy xe hơi sinh ra

tiếng ồn. Loại tri thức này coi như may rủi, cầu may.

Khi các may rủi dùng để hướng dẫn giải bài toán trong hệ thống trí tuệ nhân tạo, người ta có kiểu "tìm kiếm may rủi" hay "tìm phù hợp" (best-first). Loại tìm kiếm này phát hiện tại nơi có vẻ tốt nhất. Như vậy cũng chẳng đảm bảo nơi đó có lời giải. Tuy nhiên, cách tìm kiếm may rủi là thích hợp đối với những ứng dụng cần nhanh chóng thu được lời giải.

♦ 3.2.6. Suy lý không đơn điệu

Đối với nhiều trường hợp, người ta suy lý trên các thông tin tĩnh. Các thông tin này không thay đổi trạng thái trong quá trình giải bài toán. Loại suy lý này được gọi là suy lý đơn điệu.

Ví dụ: Trong bài toán trạng thái của các sự kiện thay đổi.

IF Gió thổi

THEN Cái nôi đung đưa

Nếu có sự kiện "con gió mạnh" thì trong câu **IF** đúng, tức là người ta đã sử dụng nếu "gió thổi mạnh" thì "gió thổi". Lúc này trong câu trên, người ta thu được kết luận trong câu **THEN**, tức là:

"Con gió mạnh" \rightarrow "gió thổi" \rightarrow "cái nôi đang đưa"

Sau khi gió mạnh hết, người ta muốn rằng cái nôi hết đang đưa. Tuy nhiên, hệ thống với cách suy lý đơn điệu sẽ “giữ” trạng thái đang đưa cái nôi.

Do việc theo dõi sự thay đổi của thông tin không mấy khó khăn, khi có sự kiện nào thay đổi người ta có thể dựa vào nhiều sự kiện phụ thuộc khác để thu được kết luận như mong muốn. Loại suy lý như vậy gọi là "suy lý không đơn điệu".

Hệ thống có thể suy lý không đơn điệu nếu nó có hệ thống quản lý giá trị chân lý. Hệ thống này quản lý dữ liệu về "nguyên nhân" để sự kiện được khẳng định. Do vậy, khi nguyên nhân thay đổi, sự kiện cũng thay đổi theo. Một hệ thống dùng suy lý không đơn điệu như ví dụ trên sẽ giữ được cái nôi đang đưa lại.

3.3. SUY DIỄN

Hệ thống trí tuệ nhân tạo mô hình hoá quá trình suy lý của con người nhờ kỹ thuật gọi là "suy diễn". Việc suy diễn là quen thuộc trong hệ chuyên gia. Như vậy: "Quá trình dùng trong hệ chuyên gia để rút ra thông tin mới từ các thông tin cũ được gọi là suy diễn".

Người ta quan tâm về một số khía cạnh của suy diễn, cũng như cách thức thực hiện của mô tơ suy diễn. Trong hệ thống, phần mô tơ suy diễn thường được coi là kín, ít thấy tường minh. Tuy nhiên cần biết:

- Câu hỏi nào sẽ được chọn để người sử dụng trả lời?
- Cách tìm kiếm trong cơ sở tri thức?
- Làm sao chọn được luật thực hiện trong số các luật có thể?

Lần lượt các vấn đề này sẽ được trả lời sau khi trình bày kỹ thuật suy diễn tiến và lùi. Cả hai kỹ thuật suy diễn này đều dựa trên suy diễn logic được xét dưới đây.

♦3.3.1. Modus ponens

Suy lý logic đã được giới thiệu qua các luật suy diễn đơn giản gọi là "modus ponens".

"Luật logic khẳng định rằng nếu biết **A** là đúng và **A** kéo theo **B** thì có thể cho là **B** đúng.".Modus ponens làm việc với các tiên đề (các câu đúng) để suy ra các sự kiện mới. Chẳng hạn có tiên đề với dạng **E1** \rightarrow **E2**, và tiên đề khác **E1** thì về logic suy ra **E2**, tức **E2** đúng. Các tiên đề này có thể dịch thành danh sách, trong đó tiên đề 3 có được do tiên đề 1 và tiên đề 2.

1. E1

2. E1 \rightarrow E2

3. E2

Nếu có tiên đề khác, có dạng **E2** \rightarrow **E3** thì **E3** được đưa vào danh sách.

Dựa trên các tập kéo theo, tức là các luật, và các dữ liệu ban đầu, luật modus ponens tạo nên một dãy các khẳng định. Quá trình suy diễn được tiến hành nhờ một dãy các thông tin đã được khẳng định.

Loại suy diễn này là cơ sở của suy diễn dữ liệu hay của hệ chuyên gia suy diễn tiến.

♦ 3.3.2. Suy diễn tiến/lùi

▶ 3.3.2.1. Giới thiệu

Mục đích của quá trình tìm kiếm là phát hiện đường đi từ cấu hình hay trạng thái xuất phát đến trạng thái đích. Có hai hướng, tiến hay lùi, khi thực hiện phương pháp này. Hai phương pháp suy diễn là suy diễn tiến và suy diễn lùi sẽ được trình bày chi tiết ngay sau đây. Một cách chưa được hình thức, có thể coi suy diễn tiến là chiến lược giải bài toán xử lý dữ kiện hay dữ liệu; nó thiên về quá trình suy diễn lặp đi lặp lại từ tiên đề hay giả thiết di chuyển về phía trước, từ giả thiết về phía kết luận.

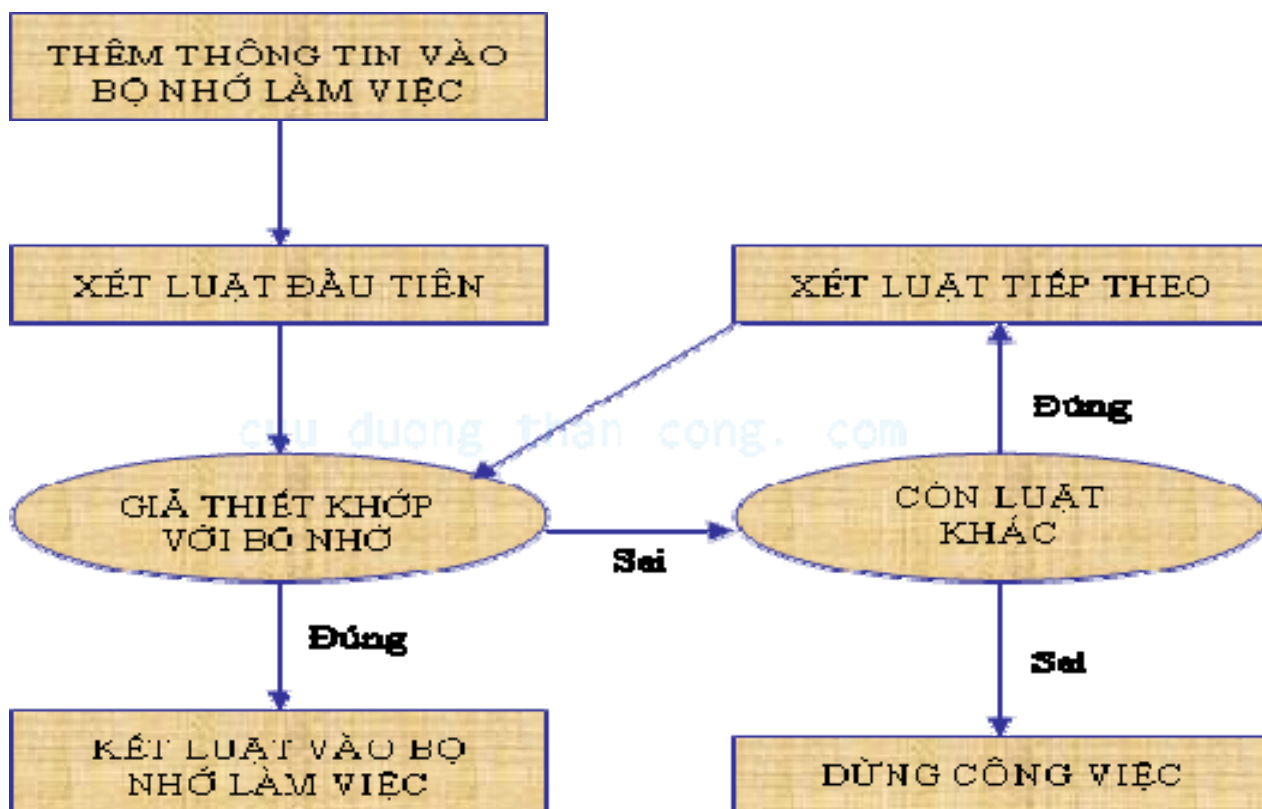
Suy diễn có mặt trái là khi các dữ liệu thừa cứ sinh ra có thể càng tiếp tục suy diễn càng không đi đến trạng thái đích mong muốn. Hướng tìm kiếm ngược với cách này suy diễn lùi.

Đối với các hệ dựa trên luật suy diễn lùi thiên về quá trình đặt điều kiện hay giả thiết đã có

đích bài toán rồi làm ngược với các luật nhằm thấy một tập các giả thiết thỏa mãn dùng cho đích này.

►3.3.2.2. Suy diễn tiến

Quá trình giải đối với vài vấn đề bắt đầu bằng việc thu thập thông tin. Thông tin này suy lý để suy ra kết luận. Điều này cũng như bác sĩ bắt đầu chuẩn đoán bằng một loạt các câu hỏi về triệu chứng của bệnh nhân. Loại suy diễn này được mô hình hóa trong hệ chuyên gia có tìm kiếm dữ liệu với tên là "suy diễn tiến". Suy diễn tương tự nh modus ponens đã trình bày.



Hình 3.1. Các Hoạt Động của Hệ Thống Suy Diễn Tiến

Chiến lược suy diễn bắt đầu bằng tập sự kiện đã biết, rút ra các sự kiện mới nhờ dùng các luật mà phần giả thiết khớp với sự kiện đã biết, và tiếp tục quá trình này cho đến khi thấy trạng thái đích, hoặc cho đến khi không còn luật nào khớp được các sự kiện đã biết hay được sự kiện suy diễn.

Ứng dụng đơn giản nhất của hệ thống suy diễn tiến hoạt động như sau:

- Trước tiên hệ thống này lấy các thông tin về bài toán từ người sử dụng và đặt chung vào bộ nhớ làm việc.
- Suy diễn quét các luật theo dãy xác định trước; xem phần giả thiết có trùng khớp với nội dung trong bộ nhớ.
- Nếu phát hiện một luật như mô tả trên, bổ sung kết luận của luật này vào bộ nhớ. Luật này gọi là cháy.

• Tiếp tục quá trình này, có thể bỏ qua các luật đã cháy. Quá trình tiếp tục cho đến khi không còn khớp được luật nào.

Lúc này bộ nhớ có các thông tin của người dùng và thông tin do hệ thống suy diễn.

Ví dụ: Giả sử có bệnh nhân đến thăm bệnh. Bác sỹ dùng kiến thức y học và thông do bệnh nhân khai để xem có bệnh gì không. Mô hình chẩn đoán theo suy diễn tiến. Thí dụ xét bệnh viêm họng.

• **Luật 1.**

IF Bệnh nhân rất họng

AND Nghi viêm nhiễm

THEN Tin rằng bệnh nhân viêm họng, đi chữa họng.

• **Luật 2.**

IF Nhiệt độ bệnh nhân quá 37 độ

THEN Bệnh nhân bị sốt

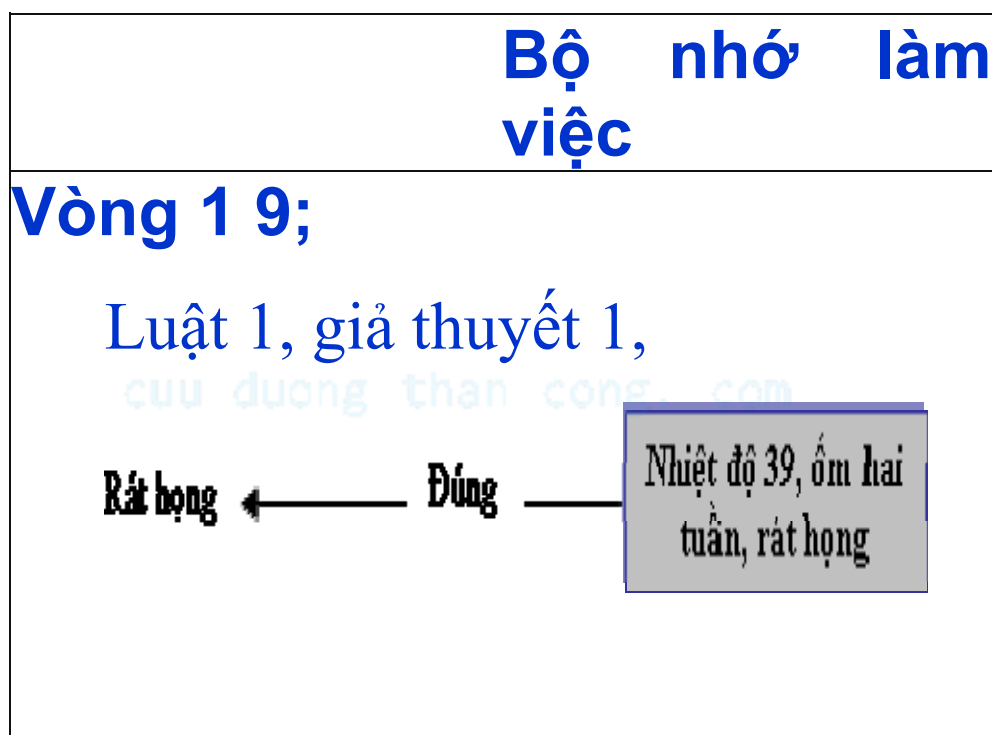
9; • **Luật 3.**

IF Bệnh nhân ốm trên 1 tuần
AND Bệnh nhân sốt
THEN Nghi bệnh nhân viêm nhiễm.

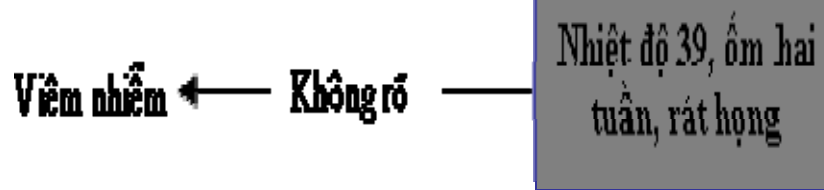
Thông tin từ nệnh nhân là:

- Bệnh nhân có nhiệt độ 39 độ
- Bệnh nhân đã ốm hai tuần
- Bệnh nhân họng rất

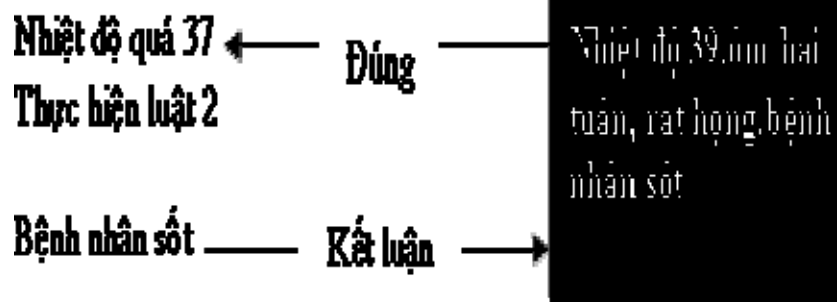
Khi hệ thống thấy giả thiết của luật khớp với thông tin trong bộ nhớ, câu kết luận của luật được bổ sung vào bộ nhớ.



Luật 2, giả thuyết 2,



Luật 2, giả thuyết 1,

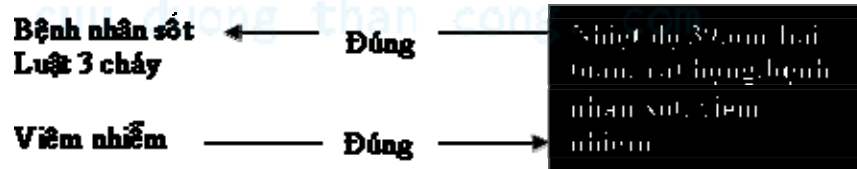


Vòng 2 : Luật 1 lại không khớp, luật 2 cháy

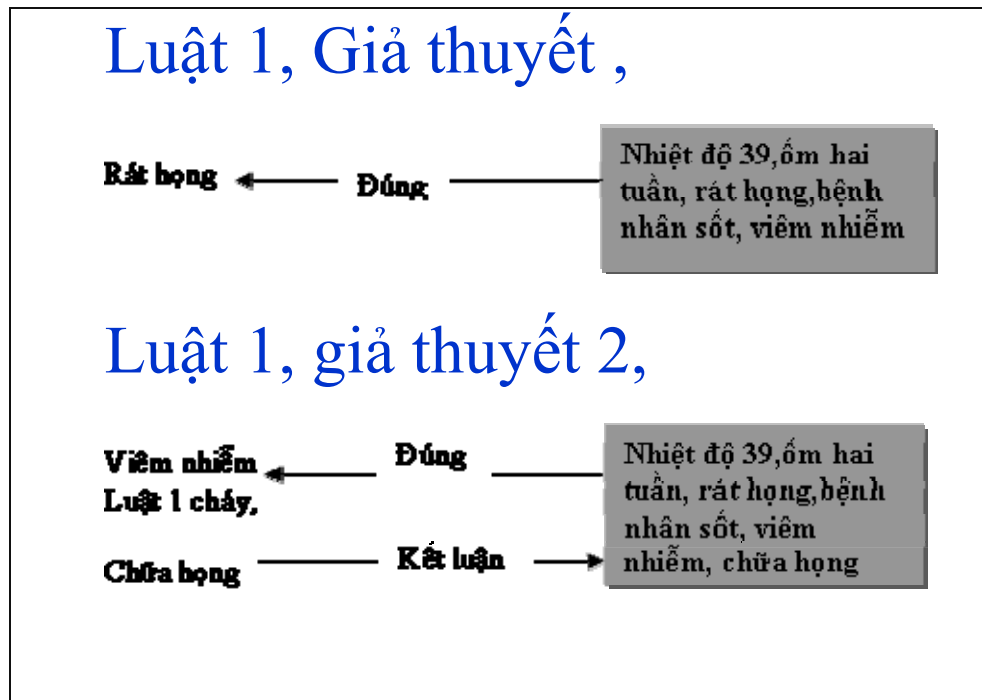
Luật 3, giả thuyết 1,



Luật 3, giả thuyết 2,



Vòng 3:



Hình 3.2. Các Luật Bị Cháy, Thông Tin Trong Bộ Nhớ Thay Đổi Trong Suy Diễn Tiến

Nhờ các thông tin , hệ thống kết luận được 3 thông tin mới:

- Bệnh nhân sốt
- Nghi viêm nhiễm
- Phải chữa họng bệnh nhân.

Hệ thống suy diễn kết luận mọi thứ có thể. Tiếp cận này phù hợp đối với một vài ứng dụng. Tuy nhiên, trong ứng dụng khác, tiếp cận này đưa ta

các thông tin không cần thiết. Giả sử cho thêm hai luật:

➡ **Luật 4.**

IF Bệnh nhân sốt

THEN Lúc ấy bệnh nhân không đi lại được

➡ **Luật 5.**

IF Bệnh nhân không đi lại được

THEN Bệnh nhân ở nhà và đọc sách.

Hệ thống dễ dàng suy được "bệnh nhân sốt". Thông tin làm cháy luật 4 và luật 5. Thông tin mới về việc họ ở nhà và đọc sách chẳng giúp ích gì cho bác sĩ. Đương nhiên làm sao hệ thống biết là thông tin đó quan trọng hay không.

Nói chung hệ thống suy diễn tiến không làm thế nào biết được là thông tin này quan trọng hơn hay thông tin kia. Do vậy nó tốn công sức vào việc phát hiện các sự kiện không cần thiết.

➡ **3.3.2.3. Suy diễn lùi**

Kỹ thuật suy diễn tiến là tốt khi làm việc với bài toán bắt đầu từ các thông tin và cần suy lý một cách logic đến các kết luận. Trong bài toán loại khác, người ta bắt đầu từ các giả thuyết định chứng minh rồi tiến hành thu thập thông tin. Chẳng hạn bác sĩ nghi người bệnh bị bệnh nào đó, ông ta tìm ra triệu chứng của bệnh đó. Loại suy lý này được mô hình hóa trong trí tuệ nhân tạo như hệ chuyên gia với tên là "Suy diễn lùi".

Suy diễn lùi là chiến lược suy diễn để chứng minh một giả thiết bằng cách thu thập thông tin hỗ trợ.

Hệ thống suy diễn lùi bắt đầu từ đích cần chứng minh:

- Trước hết nó kiểm tra trong bộ nhớ làm việc để xem đích này đã được bổ sung trước đó chưa. Bước này cần thiết vì cơ sở tri thức khác có thể đã chứng minh đích này.

- Nếu đích chưa hề được chứng minh, nó tìm các luật có phần **THEN** chứa đích. Luật này gọi là luật đích.

• Hệ thống xem phần giả thiết của các luật này có trong bộ nhớ làm việc không. Các giả thiết không được liệt kê trong bộ nhớ gọi là các đích mới, hoặc đích con, cần được chứng minh. Các đích con này được cung cấp, tức giải, nhờ các luật khác.

Quá trình này tiếp tục đệ qui cho đến khi hệ thống tìm thấy một giả thiết không được luật nào cung cấp. Đó là một "sơ khởi".

A. SƠ KHỞI (PRIMITIVE)

Là giả thiết của một luật mà không do luật nào kết luận.

Khi thấy sơ khởi hệ thống yêu cầu người sử dụng các thông tin về nó. Hệ thống dùng các thông tin này để giải đích con và đích ban đầu. Suy diễn lùi thực hiện tương tự như cách con người kiểm tra một giả thiết có đúng không.

Ví dụ: Giả sử sau khi tiếp chuyện bệnh nhân, bác sĩ nghĩ rằng người bệnh viêm họng. Công việc của ông ta là chứng tỏ khẳng định này. Thủ

tục chẩn đoán được mô hình hóa bằng hệ chuyên gia suy diễn lùi.

➡ **Luật 1.**

IF Có dấu hiệu viêm họng

AND Có cơ quan nội tạng bị viêm

THEN Bệnh nhân bị viêm họng.

➡ **Luật 2.**

IF Họng bệnh nhân đỏ

THEN Có dấu hiệu viêm nhiễm.

➡ **Luật 3**

IF Cơ quan bị thương tổn

AND Có khuẩn cầu

AND Có hạt

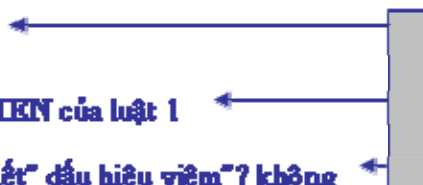
THEN Chắc chắn cơ quan nội tạng bị viêm.

Bước 1. Đích : Bệnh nhân bị viêm họng

Bước 2. Đã thấy đích : không

Bước 3. Tìm đích trong phần THEN của luật 1

Bước 4. Xem luật 1. Phần giả thiết "dấu hiệu viêm"? không



Bước 5. Tìm thấy các luật có giả thiết như phần **THEN** của luật 2?

Bước 6. Xem luật 2, giả thiết 1 đã biết "họng đỏ" ? không

Bước 7. Tìm thấy luật với giả thiết này trong phần **THEN** của luật nào đó không?

Bước 8. Giả thiết này là sơ khởi. Cần có thông tin này bằng hội thoại:



Bước 9. Xem luật 1, giả thiết 2 "có nội tạng viêm khiên" không?

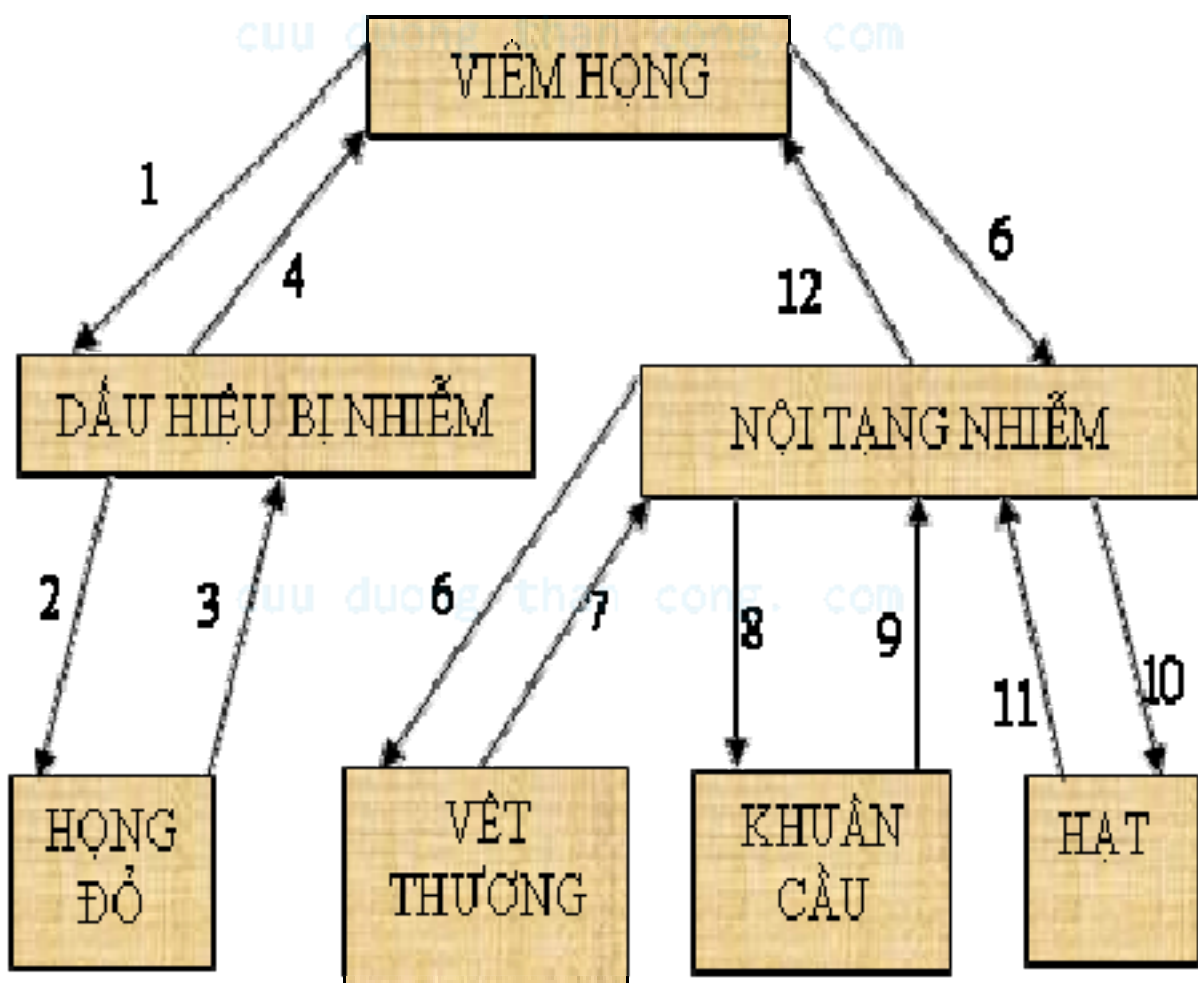
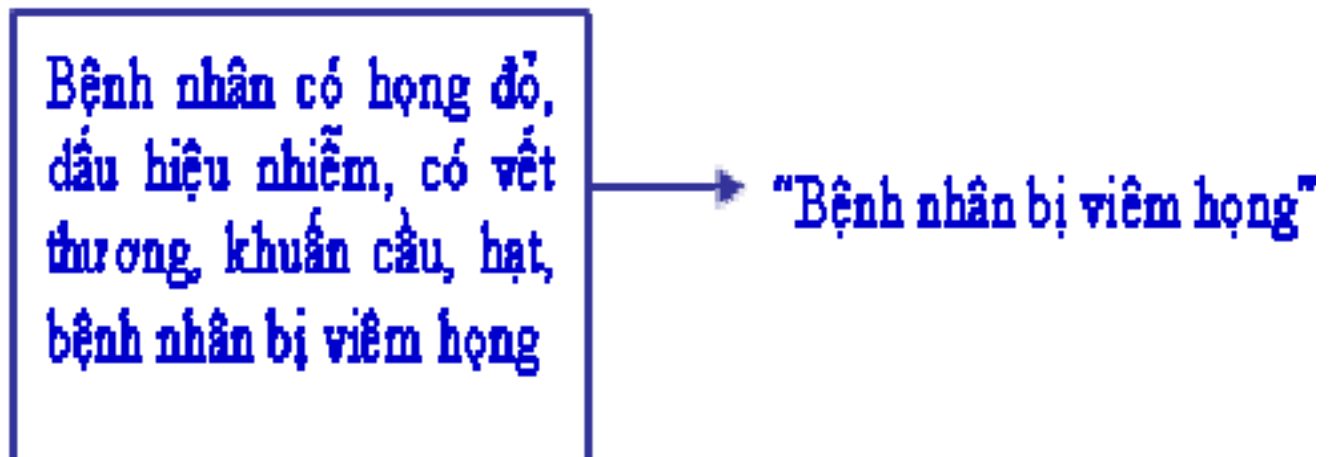
Bước 10. Tìm các luật có giả thiết trong phần **THEN** của luật 3

Bước 11. Tiếp theo là các suy diễn như trình bày. Tất cả 3 giả thiết của luật 3 là sơ khởi có được do trao đổi với người bệnh.

Giả sử thu được các câu trả lời đúng. Hệ thống bổ sung kết luận “chắc chắn có cơ

quan nội tạng bị viêm" vào bộ nhớ do luật 3 cháy.

Bước 12. Luật 1 cháy, thêm kết luận vào bộ nhớ.



Hình 3.3. Các Bước Suy Diễn Lùi

► 3.3.2.4. Ưu nhược điểm của các kỹ thuật suy diễn

Suy diễn tiến và suy diễn lùi là hai kỹ thuật suy diễn cơ bản trong hệ chuyên gia. Việc phân tích ưu nhược điểm của từng loại kỹ thuật nhằm sử dụng chính phù hợp trong các ứng dụng.

a) Ưu điểm:

• Suy diễn tiến:

- Ưu điểm chính của suy diễn tiến là làm việc tốt khi bài toán về bản chất đi thu thập thông tin rồi thấy điều cần suy diễn.
- Suy diễn tiến cho ra khối lượng lớn các thông tin từ một số thông tin ban đầu. Nó sinh ra nhiều thông tin mới.
- Suy diễn tiến là tiếp cận lý tưởng đối với loại bài toán cần giải quyết các nhiệm vụ như lập kế hoạch, điều hành điều khiển và diễn dịch.

• Suy diễn lùi

- Một trong các ưu điểm chính của suy diễn lùi là phù hợp với bài toán đưa ra giả thuyết rồi xem hiệu quả giả thiết đó có đúng không.
- Suy diễn lùi tập trung vào đích đã cho. Nó tạo ra một loạt câu hỏi chỉ liên quan đến vấn đề đang xét, đến hoàn cảnh thuận tiện đối với người dùng.
- Khi suy diễn lùi muốn suy diễn cái gì đó từ các thông tin đã biết, nó chỉ tìm trên một phần của cơ sở tri thức thích đáng đối với bài toán đang xét.

b) Nhược điểm

• Suy diễn tiến

- Một nhược điểm chính của hệ thống suy diễn tiến là không cảm nhận được rằng chỉ một vài thông tin là quan trọng. Hệ thống hỏi các câu hỏi có thể hỏi mà không biết rằng chỉ một ít câu đã đi đến kết luận được.

• Hệ thống có thể hỏi cả câu không liên quan. Có thể các câu trả lời cũng quan trọng, nhưng làm người dùng lúng túng khi phải trả lời các câu không dính đến chủ đề.

• Suy diễn lùi

• Nhược điểm cơ bản của suy diễn này là nó thường tiếp theo dòng suy diễn, thay vì đúng ra phải đúng ở đó mà sang nhánh khác. Tuy nhiên có thể dùng nhân tố tin cậy và các luật meta để khắc phục.

3.4. MỘT CÀI ĐẶT CƠ CHẾ GIẢI THÍCH VỚI LẬP LUẬN SUY DIỄN LÙI

Phần này đưa ra một cài đặt về cơ chế suy diễn lùi cho hệ chuyên gia. Bao gồm các phần sau:

• Xây dựng một Cơ sở Tri thức gồm tập các sự kiện và tập luật (Facts, Rules) đơn giản có tính chất minh họa.

• Cài đặt một Động cơ suy diễn (Inference Engine) theo cơ chế Suy diễn lùi.

Cài đặt Cơ chế giải thích (Explanation) các hoạt động của Hệ chuyên gia.

♦ 3.4.1. Xây dựng một Cơ sở tri thức

Để đơn giản trong phần trình bày, ta sẽ xây dựng một Cơ sở tri thức gồm tập các sự kiện và tập luật đơn giản có tính chất minh họa như sau:

a. Tập các sự kiện (Facts):

Giả sử các sự kiện về vi máy tính gồm có:

A: Khởi động được

B: Hoạt động bình thường

C: In được

D: Không hỏng

E: Hỏng phần in

F: Thông báo

G: Có âm thanh

H: Hỏng Ram

I: Thông báo đĩa

J: Hồng đĩa

Tập $F = \{A,B,C,D,E,F,G,H,I,J\}$ là tập các sự kiện.

b. Tập luật (Rules Bases)

Giả sử có một tập qui luật để chẩn đoán các trường hợp hỏng hóc của máy vi tính như sau:

R1: NẾU Khởi động được THÌ Hoạt động bình thường.

R2: NẾU Hoạt động bình thường VÀ In được THÌ Không hỏng.

R3: NẾU Hoạt động bình thường VÀ KHÔNG In được THÌ Hỏng phần in.

R4: NẾU KHÔNG Khởi động được VÀ KHÔNG Thông báo VÀ Có âm thanh THÌ Hỏng Ram.

R5: NẾU KHÔNG Khởi động được VÀ Thông báo đĩa THÌ Hỏng đĩa.

Tập $R = \{ R1,R2,R3,R4,R5 \}$ là Tập các luật (RB)

Mô tả luật:

- Một luật bao gồm nhiều sự kiện, được chia làm hai phần: Giả thiết và Kết luận liên kết với nhau bằng từ khoá THÌ.
- NẾU: Từ khoá bắt đầu của phần giả thiết.
- THÌ: Từ khoá kết thúc phần giả thiết và bắt đầu phần kết luận.
- VÀ: Toán tử AND dùng nối các sự kiện của giả thiết.
- KHÔNG: Toán tử NOT dùng phủ định một sự kiện.
- Ri: Tên các luật (Mã luật).

❗**Chú ý:** Các toán tử logic khác đều có thể biểu diễn bằng toán tử NOT và AND. Nên để đơn giản, ta chỉ sử dụng các toán tử NOT và AND trong các luật làm ví dụ.

c. Đồ thị AND/OR:

Một Cơ sở tri thức bao gồm tập các sự kiện và tập các luật có thể được biểu diễn bằng một cấu trúc

Cây. Trong đó mỗi sự kiện là một Nút (Node), mỗi luật được biểu diễn bằng một hoặc nhiều cung (Arc). Một luật được biểu diễn dưới dạng:

VỀ TRÁI (Giả thiết)	→	VỀ PHẢI (Kết luận)
------------------------	---	-----------------------

R1: A → B

R2: B[^]C → D

R3: B[^] (~ C) → E

R4: (~ A)[^] (~ F → H)[^]G

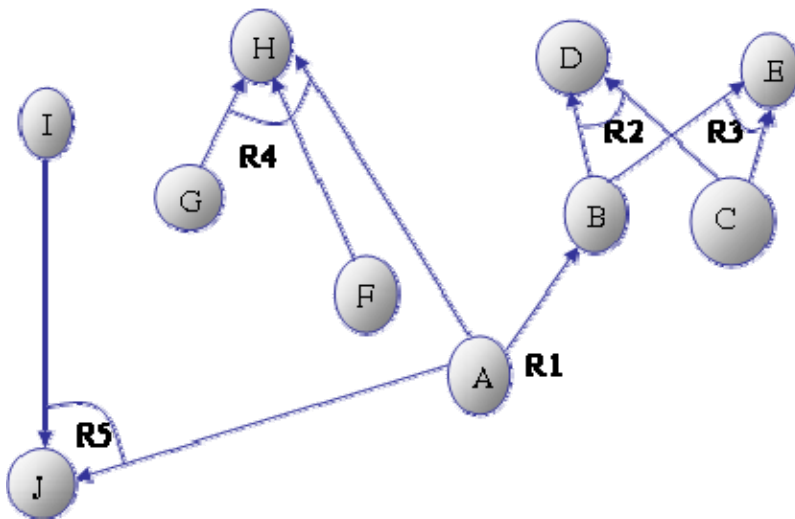
R5: (~ A)[^]I → J

Ký hiệu :

~: NOT (KHÔNG)

^: AND (VÀ)

Sau đây là đồ thị AND/OR của Cơ sở tri thức trong ví dụ trên:



Hình 3.4. Đồ thị thể hiện một Cơ sở tri thức

d. Cấu trúc dữ liệu dùng mô tả Cơ sở tri thức

Để biểu diễn Cơ sở tri thức, ta sử dụng các dữ liệu cấu trúc của C++. Sau đó chúng được lưu trên đĩa dưới dạng các tập tin nhị phân. Chương trình cũng được cài đặt bằng ngôn ngữ lập trình Borland C++.

• Biểu diễn nút:

Để biểu diễn một Nút, ta dùng các cấu trúc sau:

```
typedef struct
{
    int stt;
    char Ten[5];
```

```

int Loainut;
char Ynghia[40];
int Giatr;
}biennut;

```

Trong đó:

Mục	Kiểu dữ liệu	Chiều dài	Mô tả
Stt	int	2 bytes	Số thứ tự của Nút (1,2,3,4,...)
Ten	char	5 ký tự	Tên của Nút (A,B,C,...)
Loainut	int	2 bytes	Loại Nút (1,2,3)
Ynghia	char	40 ký tự	Ghi chú về Tên Nút (Khởi động

			được,...)
Cogt	int	2 bytes	Có hay không có giá trị? (1,0)
Giatri	int	2 bytes	Giá trị đúng hay sai? (1,0)

🔗 Biểu diễn luật:

Sử dụng cấu trúc dữ liệu cơ sở unsigned long (4 bytes*8 = 32 bits) để định nghĩa một kiểu dữ liệu mới là SET32, đồng thời sử dụng các toán tử xử lý bit của C++ (bitwise) để thao tác trên SET32:

```
typedef unsigned long SET32;
```

```
SET32 A, B;
```

Và cấu trúc biểu diễn một luật:

```
typedef struct
```

```
{
```

```
char Maluat[5];
```



```
SET32 vt;
SET32 vp;
}bienluat;
```

Trong đó:

Mục	Kiểu dữ liệu	Chiều dài	Mô tả
Maluat	Char	5 ký tự	Mã luật (R1,R2,R3,...)
vt	SET32	32 bit	Vế trái của luật (Giả thiết của Luật)
vp	SET32	32 bit	Vế phải của luật (Kết luận của Luật)

🚀 Mô tả biểu diễn chi tiết hai vế của một luật:

Giả sử Cơ sở tri thức của chúng ta trong ví dụ trên có 10 nút được đánh số thứ tự như hình dưới, thì các luật sẽ được biểu diễn bằng SET32 như sau:

Minh họa với luật R4: $(\sim A) \wedge (\sim F) \wedge G \rightarrow H$.

STT nút	Tên nút	Vé trái (vt)	Vé phải (vp)	Thứ tự bit
0	A	0	0	0
1	B	0	0	1
2	C	0	0	2
3	D	0	0	3
4	E	0	0	4
5	F	0	0	5
6	G	1	0	6
7	H	0	1	7
8	I	0	0	8
9	J	0	0	9
10		0	0	10
11		0	0	11
12		0	0	12
13		0	0	13
14		0	0	14
15		0	0	15
16	$\sim A$	1	0	16
17	$\sim B$	0	0	17
18	$\sim C$	0	0	18

19	~D	0	0	19
20	~E	0	0	20
21	~F	1	0	21
22	~G	0	0	22
23	~H	0	0	23
24	~I	0	0	24
25	~J	0	0	25
26		0	0	26
27		0	0	27
28		0	0	28
29		0	0	29
30		0	0	30
31		0	0	31

NHẬN XÉT:

Như vậy với SET32 các bit được dùng như sau:

Bit 0..15: biểu diễn cho 16 nút trên

Bit 16..31: biểu diễn phủ định (NOT) cho 16 nút trên.

Nghĩa là chỉ biểu diễn được các Cơ sở tri thức tối đa có 16 sự kiện, điều này không phù hợp và không có ý nghĩa thực tiễn.

Để khắc phục vấn đề này, ta có thể sử dụng một mảng bit gồm 518192 phần tử (thay vì chỉ có 32 phần tử như trong SET32), mỗi phần tử là một bit, chấp nhận các trị 0 hoặc 1. Và như vậy có thể mô tả đến $518.192/2=259.096$ nút hay sự kiện.

Sử dụng cấu trúc SET32 để đơn giản trong mô tả.

❖ Các phép toán trên SET32:

Để biểu diễn một luật bất kỳ bằng cấu trúc SET32 là một mảng có 32 bit phần tử, chúng ta cần thiết phải xây dựng các phép toán thao tác trên một mảng nhị phân. Chúng ta sẽ sử dụng các toán tử bitwise của C++ để làm điều này. Sau đây là các Prototype:

```
.void Include(int i, SET32 far & A)  
:
```

Đưa giá trị 1 vào bit thứ i của mảng A có kiểu SET32

```
.void Exclude(int i, SET32 far & A) :
```

Đưa giá trị 0 vào bit thứ i của mảng A có kiểu SET32

.SET32 Union (SET32 A, SET32 B) :

Phép hợp theo bit của hai mảng A và B

.SET32 Intersection (SET32 A, SET32 B) :

Phép giao theo bit của hai mảng A và B

.SET32 Difference (SET32 A, SET32 B) :

Phép hiệu theo bit của hai mảng A và B

.SET32 Complement (SET32 A) :

Lấy phần bù của hai mảng A

.int Equal (SET32 A, SET32B) :

Kiểm tra A có bằng B

.int In (SET32 A, SET32B) :

Kiểm tra B có chứa A

.int Card (SET32 A) :

Đếm số bit 1 của mảng A

Phần cài đặt cụ thể sẽ được trình bày trong phần III của tài liệu này.

🚀 **Các bước thiết lập Cơ sở tri thức bằng SET32**

Bước 1: Nhập các nút với các thông tin như Tên nút, Ý nghĩa các nút, lưu lại trên đĩa dưới dạng tập tin nhị phân của C++. Quá trình nhập có kiểm tra sự trùng nút.

Bước 2: Nhập các luật với các thông tin như Mã luật, Nút kết luận, Nút giả thiết lưu lại trên đĩa dưới dạng tập tin nhị phân của C++. Quá trình nhập có kiểm tra sự trùng luật, trùng nút trong luật, đồng thời kiểm tra tính dư thừa, tính mâu thuẫn và vòng lặp của tập luật ... nhờ vào tính toán các bao đóng trên SET32.

🔴 **3.4.2. Cài đặt Động cơ suy diễn bằng cơ chế lập luận lùi**

🚀 **Thế nào là lập luận suy diễn lùi:**

Xét luật R2:

R2: NẾU Hoạt động bình thường VÀ In được THÌ Không hỏng.

Để đi đến kết luận là *Không hỏng* thì phải chứng minh máy tính đúng là *Hoạt động bình thường* và đúng là *In được*.

Quá trình chứng minh sẽ là một chuỗi các suy diễn trên tập luật cộng với sự đối thoại giữa người với Hệ chuyên gia. Điều đầu tiên sẽ chứng minh là máy tính có đang *Hoạt động bình thường* không? Nếu đúng sẽ chứng minh tiếp điều thứ hai là máy tính có đang *In được* hay không? Nếu điều thứ hai cũng đúng thì kết luận *Không hỏng* là đúng. Nếu một trong hai điều trên là sai thì kết luận *Không hỏng* là sai. Và Hệ chuyên gia sẽ đi tìm một kết luận khác và công việc suy diễn sẽ lặp lại như trên.

Nếu xét trên cây đồ thị AND/OR biểu diễn cho cơ sở tri thức thì quá trình suy diễn sẽ lùi dần từ các nút lá đến nút trung gian đến nút gốc.

Vậy cơ chế Suy diễn lùi sẽ bắt đầu từ một kết luận và đi tìm một chuỗi các giả thiết để chứng minh cho kết luận đó. Nghĩa là ta sẽ phải xuất phát từ Tập nút kết luận. Để trình bày phần này, hãy trở lại Cơ sở tri thức ở ví dụ trên. Chúng ta sẽ liệt kê các nút ở mỗi vế như sau:

VỀ TRÁI = { A,B,C,F,G,I }

VỀ PHẢI = { B,D,E,H,J }

🔗Thuật toán phân loại nút:

Dễ dàng thấy rằng có những Nút chỉ xuất hiện ở về TRÁI, hoặc chỉ ở về PHẢI, đồng thời có những nút xuất hiện trong cả hai về. Dùng các phép toán đã được xây dựng trên SET32 vào thuật toán phân loại nút như sau:

Đọc tập nút vào mảng cấu trúc động

Đọc tập Luật vào mảng cấu trúc động

Input : Tên nút

Output: Loại nút

Với DOM (Loại nút)={ 1,2,3 } = { Nút kết luận, Nút trung gian, Nút tận cùng}.

Duyệt qua tập nút

Duyệt qua tập luật

Các trường hợp:

- + Nút chỉ ở VẾ PHẢI: Loại nút =1 // Tập nút kết luận
- + Nút chỉ ở VẾ TRÁI: Loại nút =3 // Tập nút tận cùng
- + Nút có ở VẾ TRÁI và VẾ PHẢI : Loại nút =2 // Tập nút trung gian

Hết tập Luật

Hết tập Nút

Như vậy sẽ phân loại thành 3 tập nút khác nhau (Loại 1,2,3):

- . Tập Nút kết luận = VẾ TRÁI \ VẾ PHẢI = { D,E,H,J } = { Các nút chỉ có trong VẾ PHẢI }
- . Tập Nút trung gian = VẾ TRÁI 1 VẾ PHẢI = { B } = { Các nút có trong VẾ TRÁI và VẾ PHẢI }
- . Tập Nút tận cùng = VẾ TRÁI \ VẾ PHẢI = { A,C,F,G,I } = { Các nút chỉ có trong VẾ TRÁI }

Kết quả của thuật toán phân loại nút áp dụng vào cơ sở tri thức trên sẽ có kết quả như sau:

Tên nút	Tên kiện sự	Loại nút số	Ý nghĩa
A	Khởi động được	3	Nút tận cùng
B	Hoạt động bình thường	2	Nút trung gian
C	In được	3	Nút tận cùng
D	Không hỏng	1	Nút kết luận
E	Hỏng phần in	1	Nút kết luận

F	Thông báo	3	Nút tận cùng
G	Có âm thanh	3	Nút tận cùng
H	Hồng Ram	1	Nút kết luận
I	Thông báo đĩa	3	Nút tận cùng
J	Hồng đĩa	1	Nút kết luận

Như vậy ta có tập nút Kết luận như sau:

Tập nút kết luận = $\{D, E, H, J\} = \{\text{Loại 1}\}$.

🔧 Cài đặt thuật toán Suy luận lùi

Như ví dụ trên thì các chẩn đoán về sự cố của máy tính chỉ thuộc 1 trong 4 sự cố nằm trong tập Nút kết luận trên:

D: Không hỏng

E: Hỏng phần in

H: Hỏng Ram

J: Hỏng đĩa

Chúng ta hãy hình dung công việc như sau: Khi máy vi tính của chúng ta bị một sự cố nào đó, và chúng ta muốn nhờ "Hệ chuyên gia chẩn đoán sự cố máy tính" giúp đỡ xem là máy bị hỏng gì. Hệ chuyên gia sẽ phỏng vấn chúng ta và cố gắng đưa sự cố hư hỏng của máy chúng ta vào một trong 4 trường hợp trên. Khi đã có một trong 4 kết luận thì Hệ chuyên gia kết thúc phỏng vấn và xuất ra kết luận. Nếu sự cố hư hỏng của máy chúng ta không ở vào một trong 4 trường hợp trên thì Hệ chuyên gia cũng đành chào thua và sẽ trả lời là "Không giải đáp được".

Thuật toán suy diễn lùi như sau:

Input: Tập các Nút kết luận

Output: Một kết luận đúng hoặc "Không giải đáp được"

Đọc tập nút vào mảng cấu trúc động

Đọc tập Luật vào mảng cấu trúc động

Áp dụng thuật toán phân loại nút

Với DOM (Kết luận) ' { D,E,H,J, "Không giải đáp được" }

Có kết luận = 0

Duyệt qua tập Nút kết luận

Lấy ra một nút kết luận

Tìm giá trị nút 0;

Nếu giá trị nút kết luận đang xét = 1

Xuất ra kết luận đúng

Chấm dứt chuỗi suy
diễn

Có kết luận = 1;

Hết tập Nút kết luận

Nếu có kết luận = 0

Xuất ra "Không giải đáp
được"

Sau đây là thủ tục Tìm giá trị nút 0

Duyệt qua tập luật

Giá trị luật = 1;

+ Nếu Nút đang xét là kết luận
của một luật

+ Lấy ra vế trái của Luật đó
// Các giả thiết

**Lặp lại khi hết các nút
trong vế trái của
Luật**

+ Nếu một nút đã có

giá trị thì

giá trị luật = giá trị luật

AND giá trị nút

+ Ngược lại:

+ Trường hợp

Loại nút = 2 // Nút
trung gian

Tìm giá trị nút
0;

+ Trường hợp

Loại nút = 3 // Nút
tận cùng

Yêu cầu nhập
dữ liệu cho nút
này

giá trị luật = giá trị luật

AND giá trị nút

**Hết các nút trong vế
trái của Luật**

+ Giá trị của nút đang xét = giá trị luật

Hết tập Luật

❗ NHẬN XÉT:

- Quá trình suy diễn lùi chính là quá trình đi tìm giá trị của các Nút kết luận. Khi một Nút kết luận đầu tiên có giá trị đúng thì lập tức quá trình suy luận chấm dứt với kết quả là thành công. Nếu không có Nút kết luận nào có giá trị đúng thì quá trình suy luận cũng chấm dứt với kết quả là không thành công
- Kỹ thuật áp dụng để suy diễn lùi là vòng lặp kết hợp với gọi là Độ quy hàm Tìm giá trị nút 0.

❖ 3.4.3. Cài đặt Cơ chế giải thích trong Suy diễn lùi

➡ Thế nào là Cơ chế giải thích

Như đã khảo sát, quá trình suy diễn lùi cũng là quá trình đối thoại giữa người dùng và Hệ chuyên gia. Đó là khi Hệ chuyên gia cần nhập dữ liệu cho các sự kiện yêu cầu (Là các nút tận cùng – Loại nút số 3). Ở đây người ta có thể có quyền đặt ra những

câu hỏi nghi vấn như Tại sao? WHY phải cung cấp số liệu này? Hoặc khi đã tìm ra kết luận và xuất kết luận cho người dùng, họ cũng có thể đặt nghi vấn như làm Thế nào? HOW mà có kết quả như vậy?

Trong cả hai trường hợp trên, để khẳng định niềm tin, Hệ chuyên gia phải trả lời được cho người dùng các câu hỏi Why, How. Đó chính là Cơ chế giải thích của Hệ chuyên gia. Rõ ràng là cơ chế giải thích phải được cài đặt song song với cơ chế suy diễn lùi.

➡ **Cài đặt Cơ chế giải thích câu hỏi Why?**

Như đã trình bày, thời điểm để Hệ chuyên gia trả lời câu hỏi Why? là lúc Hệ chuyên gia yêu cầu cung cấp dữ liệu cho các sự kiện là các nút tận cùng. Theo như kết quả của thuật toán phân loại nút thì đó là loại nút số 3.

Chúng ta hãy trở lại xem xét luật R2 với loại nút đã tính toán được:

NẾU Hoạt động bình thường **VÀ** In được **THÌ** Không hỏng

Loại nút số (2) (3)

Theo như thuật toán Suy diễn lùi áp dụng vào luật R2 thì các bước trải qua như sau:

Bước 1: Phát hiện thấy *Không hỏng* có loại nút số 1 là nút kết luận của luật R2.

Bước 2: Rút ra giả thiết của luật R2 là *Hoạt động bình thường VÀ In được*.

Bước 2.1: Phát hiện thấy *Hoạt động bình thường* có loại nút số 2, nên tiếp tục đi truy tìm và lại phát hiện *Hoạt động bình thường* là nút kết luận của luật R1.

R1: NẾU Khởi động được THÌ Hoạt động bình thường

Loại nút số
(3) (1)

Bước 2.1.1: Rút ra giả thiết của luật

R1 là *Khởi động được*.

Bước 2.1.2: Thấy rằng *Khởi động được* có loại nút số 3, nên yêu cầu nhập dữ liệu:

Máy tính có *Khởi động được* không? (YES/NO)

Giả sử người dùng nhập vào YES.

Kết quả: *Khởi động được* có giá trị 1 (đúng)

Hệ quả: *Hoạt động bình thường* có giá trị 1 (đúng)

Bước 2.2: Thấy rằng *Khởi động được* có loại nút số 3, nên yêu cầu nhập dữ liệu:

Máy tính có *In được* không ? (YES/NO)

Giả sử người dùng nhập vào YES.

Kết quả: *In được* có giá trị 1 (đúng)

Hệ quả: *Hoạt động bình thường* có

giá trị 1 (đúng)

Hệ quả: *Không hỏng* có giá trị 1 (đúng)

Vì không hỏng là nút có loại nút số 1 (Nút kết luận), nên xuất ra kết quả:

Máy tính của bạn không hỏng.

Giả sử tại bước (2.1.2) khi Hệ chuyên gia yêu cầu nhập dữ liệu, người dùng có thể đặt câu hỏi Tại sao (Why)? Lúc ấy Hệ chuyên gia phải xuất ra chuỗi luật nhằm giải thích lý do tại sao cần phải nhập dữ liệu:

R1: NẾU Khởi động được THÌ Hoạt động bình thường

R2: NẾU Hoạt động bình thường VÀ In được THÌ Không hỏng

Tương tự như trên cho bước (2.2)....

Như vậy để cài đặt cơ chế giải thích cho câu hỏi Why thì tại mỗi lúc phát hiện một luật mới tham gia vào quá trình suy diễn ta phải lưu lại dấu vết

của luật đó. Kỹ thuật Stack là thích hợp trong trường hợp này. Chúng ta sẽ dùng hai stack tên là stackw1 và stackw2:

Stackw1: chứa số thứ tự nút kết luận mỗi khi có một luật mới được phát hiện.

Stackw2: bản copy của stackw1 sau khi một câu hỏi Why được trả lời để chuẩn bị cho câu hỏi Why kế tiếp.

Hoạt động của 2 stack:

+ Khi phát hiện một luật mới:

Push (stackw1, stt nút kết luận)

Push (stackw2, stt nút kết luận)

+ Khi có câu hỏi Why xuất hiện: gọi thủ tục Giải thích Why 0.

Thủ tục Giải thích Why 0

Lặp đến khi stack1 rỗng

Pop (stackw1, int biến stt)

Nối chuỗi các Luật có Nút kết

luận mang số thứ tự là biến stt

Hết stackw1

Xuất chuỗi các luật để trả lời

Sau khi giải thích xong:

Stackw1 = Stackw2

Thuật toán Suy diễn lùi có câu hỏi Why

Input: Tập các Nút kết luận

Output: Một kết luận đúng hoặc "Không giải đáp được"

Với DOM (Kết luận) $\ni \{ D, E, H, J, \text{"Không giải đáp được"} \}$

Đọc tập Nút vào mảng cấu trúc động

Đọc tập Luật vào mảng cấu trúc động

Áp dụng thuật toán phân loại nút

Có kết luận = 0;

Duyệt qua tập Nút kết luận

Lấy ra một nút kết luận

Tìm giá trị nút 0;

Nếu giá trị nút kết luận
đang xét = 1

Xuất ra kết luận đúng

Chấm dứt chuỗi suy
diễn

Có kết luận = 1

Hết tập Nút kết luận

Nếu có kết luận = 0

Xuất ra "Không giải đáp
được"

Sau đây là thủ tục Tìm giá trị nút 0;

Duyệt qua tập Luật

Giá trị luật = 1;

+ Nếu Nút đang xét là kết luận

của một luật

Push (stackw1, Số thứ tự
của Nút kết luận đang xét)

Push (stackw2, Số thứ tự
của Nút kết luận đang xét)

+ Lấy ra về trái của Luật đó //

Các giả thiết

**Lặp đến khi hết các nút
trong về trái của Luật**

+ Nếu một nút đã có
giá trị thì

giá trị luật =
giá trị luật AND
giá trị nút

+ Ngược lại :

+ Trường hợp Loại
nút = 2 // Nút trung
gian

Tìm giá trị nút 0;

+ Trường hợp Loại nút = 3 // Nút tận cùng

yêu cầu nhập dữ liệu cho nút này

giá trị luật = giá trị luật AND giá trị nút

Câu hỏi Why? Xuất hiện

Gọi thủ tục Giải thích Why 0;

Hết các nút trong vế trái của Luật

+ Giá trị của nút đang xét = giá trị luật.

Hết tập Luật

👉 Cài đặt cơ chế giải thích câu hỏi HOW?

Cùng với câu hỏi Why, hệ chuyên gia đồng thời cũng phải trả lời câu hỏi How. Thời điểm để Hệ

chuyên gia có thể trả lời câu hỏi How là lúc Hệ chuyên gia tìm được kết luận và trả lời cho người dùng. Câu hỏi How là một phương tiện giúp cho Hệ chuyên gia khẳng định niềm tin đối với người dùng. Đó là làm cách nào mà Hệ chuyên gia có thể đi đến một kết luận như vậy. Sau khi trả lời câu hỏi How kết thúc cũng là lúc quá trình suy diễn lùi chấm dứt.

Chúng ta hãy trở lại ví dụ trên. Sau bước (2.2) Hệ chuyên gia tìm được kết quả và xuất ra câu trả lời:

Máy tính của bạn không hỏng.

Là thời điểm mà người dùng có thể đặt ra câu hỏi How. Nghĩa là làm thế nào mà Hệ chuyên gia kết luận là: Máy tính của bạn không hỏng. Như vậy câu hỏi How quan tâm đến chuỗi suy diễn là dẫn đến kết luận đúng.

Như vậy để cài đặt cơ chế giải thích cho câu hỏi How thì tại mỗi lúc một luật được thẩm định là đúng sẽ phải được lưu lại dấu vết trong stack. Chúng ta sẽ dùng một stack tên là stackh để thực hiện công việc này.

Hoạt động của How stackh:

+ Khi một luật được thẩm định là đúng:

Push(stackh, stt nút kết luận)

+ Khi có câu hỏi How xuất hiện:

Gọi thủ tục Giải thích How 0

Thủ tục Giải thích How 0:

Lặp đến khi stackh rỗng

Pop(stackh, int biếnstt)

Nối chuỗi các Luật có Nút
kết luận mang số thứ tự là
biến stt

Hết stackh

Xuất chuỗi các luật để trả lời

Kết thúc

Thuật toán Suy diễn lùi có câu hỏi WHY và HOW

Input: Tập các Nút kết luận

Output: Một kết luận đúng hoặc
'Không giải đáp được'

Với DOM (Kết luận) ' { D,E,H,J,
"Không giải đáp được"}

Đọc tập Nút vào mảng cấu trúc
động

Đọc tập Luật vào mảng cấu trúc
động

Áp dụng thuật toán phân loại nút

Có kết luận = 0;

Duyệt qua tập Nút kết luận

Lấy ra một nút kết luận

Tìm giá trị nút 0;

Nếu giá trị nút kết luận đang
xét = 1

Xuất ra kết luận đúng

Xuất hiện câu hỏi
How?

Gọi thủ tục Giải thích
How0;

Chấm dứt chuỗi suy
diễn

Có kết luận =1;

Hết tập Nút kết luận

Nếu có kết luận = 1

Xuất ra "Không giải đáp
được"

Sau đây là thủ tục Tìm giá trị nút 0

Duyệt qua tập Luật

Giá trị luật = 1;

+ Nếu Nút đang xét là kết
luận của một luật

Push (stackw1, Số thứ tự
của Nút kết luận đang xét)

Push (stackw2, Số thứ tự
của Nút kết luận đang xét)

+ Lấy ra về trái của Luật đó //
Các giả thiết

**Lặp đến khi hết các nút
trong về trái của Luật**

+ Nếu một nút đã có
giá trị thì

$giá trị luật = giá trị luật$
 AND $giá trị nút$

+ Ngược lại :

+ Trường hợp Loại
nút = 2 // Nút trung
gian

Tìm giá trị nút 0;

+ Trường hợp Loại
nút = 3 // Nút tận
cùng

yêu cầu nhập dữ
liệu cho nút này

$giá trị luật =$

giá trị luật AND
giá trị nút

Câu hỏi Why?
Xuất hiện

Gọi thủ tục Giải
thích Why 0;

**Hết các nút trong vế trái
của Luật**

+ Giá trị của nút đang xét =
giá trị luật.

+ Nếu giá trị của nút đang
xét = 1

Push (stack, Số thứ tự
của nút kết luật đang
xét)

Hết tập Luật

CHƯƠNG 4. HỆ HỖ TRỢ QUYẾT ĐỊNH

4.1. DẪN NHẬP

4.2. HỆ HỖ TRỢ RA QUYẾT ĐỊNH VÀ HỆ THỐNG THÔNG TIN

4.3. CÁC THÀNH PHẦN CỦA MỘT HỆ HỖ TRỢ RA QUYẾT ĐỊNH

4.3.1. Thành phần đối thoại

4.3.1.1. Xem xét chung

4.3.1.2. Cơ sở tri thức (knowledge base)

4.3.1.3. Ngôn ngữ hành động (action language)

4.3.1.4. Ngôn ngữ trình bày

4.3.1.5. Các kiểu (style) thành phần đối thoại

4.3.2. Thành phần dữ liệu

4.3.3. Thành phần mô hình

4.3.3.1. Các loại mô hình

4.3.3.2. Các lớp mô hình

4.3.3.3. Các vấn đề thường gặp với mô hình

4.3.4. Thành phần đối thoại

4.4. CÂY QUYẾT ĐỊNH

4.4.1. Giới thiệu cây quyết định

4.4.2. Suy diễn trên cây quyết định

4. 1. MỞ ĐẦU

Khái niệm hệ hỗ trợ ra quyết định được đề xuất bởi Michael S. Scott Morton vào những năm 1970. Hệ hỗ trợ ra quyết định có là:

- Phần mềm máy tính.
- Chức năng hỗ trợ ra quyết định.
- Làm việc với các bài toán có cấu trúc yếu.
- Hoạt động theo cách tương tác với người dùng.

- Được trang bị nhiều mô hình phân tích và mô hình dữ liệu.

Một số hệ hỗ trợ ra quyết định thể hiện trong bảng 4.1.

4.2. HỆ HỖ TRỢ RA QUYẾT ĐỊNH VÀ HỆ THỐNG THÔNG TIN

Các hệ thống thông tin quản lý tập trung vào các hoạt động của hệ thống thông tin.

Hệ thống thông tin quản lý có các tính chất:

- Tập trung vào thông tin, hướng đến các nhà quản lý cấp điều hành.
- Làm việc với dòng thông tin có cấu trúc.

Các hệ hỗ trợ quyết định có các tính chất:

- Hướng đến các quyết định, các nhà lãnh đạo
- Tính uyển chuyển, thích ứng với hoàn cảnh và phản ứng nhanh
- Do người dùng khởi động và kiểm soát

- Hỗ trợ các quyết định các nhân của nhà lãnh đạo

Tên	Lĩnh vực ứng dụng
GADS Geodata Analysis Display System	Phân tích và cung cấp tài nguyên địa lý.
PMS Portfolio Management System	Tư vấn và quản trị đầu tư
IRIS Industrial Relations Information	Phân tích chất lượng và bố trí nhân lực trong sản xuất
PROJECTOR	Hoạch định

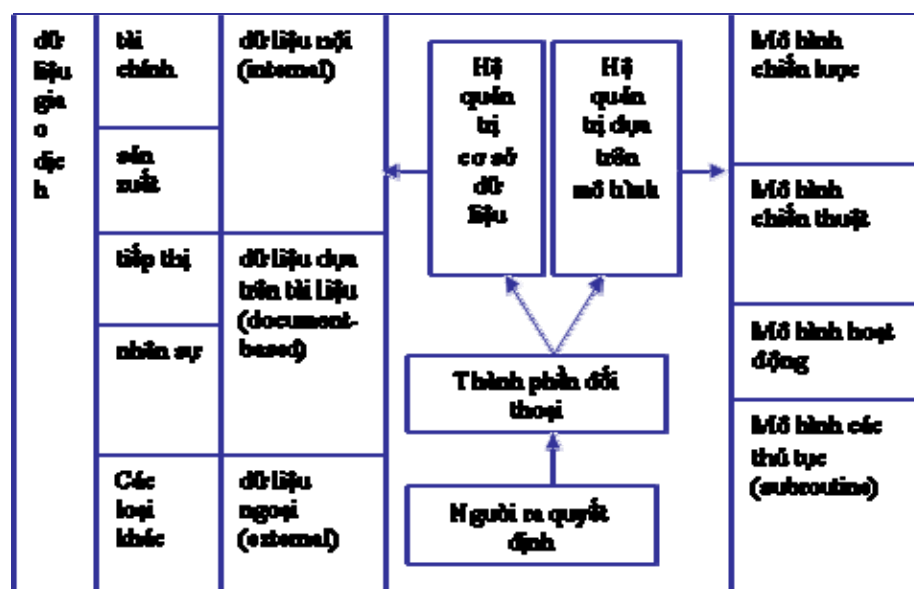
	kế hoạch tài chính
IFPS Interactive Financial Planning System	Phân tích tài chính, giá thành, sản phẩm
BRANDAID	Phân tích thị trường, ngân sách, quảng cáo

Bảng 4.1. Một số hệ hỗ trợ ra quyết định.

4.3. CÁC THÀNH PHẦN CỦA MỘT HỆ HỖ TRỢ RA QUYẾT ĐỊNH

Một cách hình dung về các thành phần của một hệ hỗ trợ ra quyết định (DDS – decision support system) và quan hệ giữa chúng là sử dụng các khái niệm đối thoại (dialog), dữ liệu (data) và mô hình (model). Đối với những người thiết kế hệ thống DDS cũng như những người sử dụng hệ thống, điều quan trọng là hiểu được các thành phần này được

thiết kế như thế nào. Người sử dụng cần phải biết có thể yêu cầu cái gì ở DDS. Người thiết kế phải biết được DDS có thể cung cấp cái gì.



Các kỹ thuật mới có nhiều ảnh hưởng đến các thành phần đối thoại, dữ liệu, và mô hình; ví dụ như giao diện đồ họa hay cơ sở dữ liệu quan hệ. Ngoài ra trí tuệ nhân tạo cũng cung cấp các khả năng biểu diễn và sử dụng mô hình trong những hình thức mới.

◆4.3.1. Thành phần đối thoại

Từ cách nhìn của người sử dụng, thành phần đối thoại là toàn bộ hệ thống. Cách dùng hệ thống, hướng dẫn cách vận hành của hệ thống và thể hiện các trả lời của hệ thống đều thông qua thành phần

đối thoại. Bennett gọi các yếu tố này bằng các khái niệm: cơ sở tri thức (knowledge base), ngôn ngữ hành động (action language), và ngôn ngữ trình bày (representation language). Các yếu tố khác như phần cứng và phần mềm, cách thức lưu trữ dữ liệu, các thuật toán được dùng thường không được nhận thức bởi người dùng.

4.3.1.1. Xem xét chung

Khi thiết kế thành phần đối thoại của một DDS, điều quan trọng là nhận ra ai là người dùng của nó. Một DDS có thể chỉ có một người dùng, nhưng cũng có thể có nhiều người dùng. Một số người dùng chỉ quan tâm đến khía cạnh hỗ trợ quyết định có tính bề mặt của DDS, một số khác lại có thể dùng DDS một cách rất thành thục. Đôi khi người ra quyết định dùng DDS một cách trực tiếp, nhưng đôi lúc họ ra quyết định dựa trên một ban cố vấn và ban cố vấn lại sử dụng DDS. Như vậy ban quyết định có thể được xem là phần mở rộng của DDS.

Thiết kế thành phần đối thoại của DDS phải cân bằng giữa tính dễ sử dụng và tính mềm dẻo

(flexibility). Ví dụ cơ chế hỏi-đáp thì dễ sử dụng nhưng không mềm dẻo vì hệ thống chỉ bao gồm các câu hỏi đã được lập trình sẵn. Ngược lại ngôn ngữ lệnh (command language) cung cấp cho người dùng nhiều chức năng hơn, nhưng lại đòi hỏi người dùng phải am hiểu về các lệnh đó. Phần nhiều các DDS sử dụng ngôn ngữ lệnh.

4.3.1.2. Cơ sở tri thức (knowledge base)

Cơ sở tri thức bao gồm những gì người dùng biết về cách thức hệ thống vận hành cũng như cách dùng hệ thống đó. Thường thì các tri thức xung quanh bài toán cần được giải phải được cung cấp cho DDS, sau đó thì DDS mới có thể ra quyết định. Một ngoại lệ là trường hợp DDS được dùng để huấn luyện người ra quyết định. Lúc này DDS là một phương tiện giáo dục.

Người dùng có thể được huấn luyện sử dụng DDS theo nhiều cách khác nhau. Có thể học sử dụng DDS theo cách một truyền một (one to one), nhưng khi có nhiều người cần được huấn luyện thì phải sử dụng đến các lớp hay khoá

học. Thêm vào đó, có thể tìm kiếm sự trợ giúp từ một chuyên gia (con người) hay từ những lệnh giúp đỡ đã được chuẩn bị kèm theo DDS.

DDS có thể dễ sử dụng hơn bằng cách công bố các tài liệu hướng dẫn (manuals) trên mạng. Hệ thống trợ giúp cảm ngữ cảnh (context sensitive), được kích hoạt khi người dùng nhấn một phím nào đó.

Tập tin lệnh cũng có thể được dùng. Tập tin lệnh chứa các lệnh cần được thực thi trong một tập tin, và các lệnh này được thực hiện tuần tự khi tập tin lệnh được thi hành. Một vài DDS cung cấp cơ chế lưu lại các lệnh: một chuỗi các lệnh đã được thực thi bởi người dùng có thể được lưu lại trong một tập tin và được thực hiện lại trong những lần sau bằng cách thực thi tập tin lệnh.

4.3.1.3. Ngôn ngữ hành động (action language)

Có nhiều loại ngôn ngữ hành động khác nhau, hiểu theo nghĩa ngôn ngữ dùng để điều hành DDS. Hỏi-đáp, dùng menu, hay ngôn ngữ lệnh

đã được giải thích ở trên. Ngoài ra còn có một số "ngôn ngữ" khác như sau.

Một vài DDS sử dụng form để nhập/xuất dữ liệu. Người dùng điền dữ liệu đầu vào (input) dùng form và nhận dữ liệu đầu ra (output) cũng trên form.

Giao diện đồ họa cung cấp một phương pháp tiếp cận khác. Các biểu tượng (icon), ảnh được dùng để đại diện cho các đối tượng như tài liệu, tập tin..., người dùng sử dụng con chuột để tác động lên các đối tượng đó (như di chuyển, chọn menu...).

Giọng nói cũng là một loại ngôn ngữ hành động, và yêu cầu công nghệ nhận dạng giọng nói (speech recognition). Với sự phát triển của công nghệ này, chúng ta có thể trông đợi nhiều DDS sử dụng giọng nói làm ngôn ngữ hành động hơn.

Tóm lại, bàn phím không phải là sự lựa chọn duy nhất, có thể kể đến các lựa chọn khác như chuột, các thiết bị trợ dùng trực tiếp trên màn hình hay là micro.

4.3.1.4. Ngôn ngữ trình bày

Ngày trước, máy in là một nguồn xuất dữ liệu. Khả năng đồ họa của màn hình cung cấp nhiều cách thể hiện mới. Màn hình có thể thể hiện các hình ảnh, đồ thị, các động ảnh. Ngoài ra âm thanh cũng được xem xét như một khả năng mới.

4.3.1.5. Các kiểu (style) thành phần đối thoại

Tổ hợp các kiểu thực hiện các thành phần con như cơ sở tri thức, ngôn ngữ hành động và ngôn ngữ trình bày, ta được nhiều kiểu thành phần hội thoại khác nhau. Một số DDS thiên về bàn phím và buộc người dùng phải nhớ các tổ hợp phím để thực thi các lệnh. Một số DDS trực quan hơn thì cho phép người dùng dùng chuột để tác động lên các đại diện của các đối tượng cần thao tác.

◆ 4.3.2. Thành phần dữ liệu

DDS không dùng các dạng dữ liệu thô thu được trong các quá trình giao dịch của các tổ chức. Dữ liệu thường phải được tóm tắt, cô đọng trước khi

được sử dụng bởi DDS. Lý tưởng nhất là công việc này cũng được tự động bằng máy tính. Nhưng đôi lúc cũng được thực hiện bằng tay khi không tốn quá nhiều công sức hay công việc đòi hỏi việc xử lý tinh tế của con người. Thông thường cần phải dùng một hệ quản trị cơ sở dữ liệu (DBMS).

Các dữ liệu nội (internal data) cũng được cần đến. Ví dụ như loại dữ liệu liên quan đến các lĩnh vực của kỹ sư hay của nhà quản lý. Các dữ liệu này thường không thể có được qua các quá trình xử lý dữ liệu thông thường được. Chúng phải được thu thập, nhập liệu và lưu trữ và cập nhật thông qua các phương pháp và tiến trình đặc biệt. Loại dữ liệu này cũng cần dùng đến hệ quản trị cơ sở dữ liệu (DBMS).

Các dữ liệu ngoại (external data): như thông tin thương mại, tài chính của một nền kinh tế, các số liệu công nghiệp cũng đòi hỏi nhiều nỗ lực đặc biệt để có được. Nhưng khác với dữ liệu nội, dữ liệu ngoại có thể mua được từ các công ty, tổ chức. Loại dữ liệu này được rút trích từ các cơ sở dữ liệu thương mại...

◆ 4.3.3. Thành phần mô hình

4.3.3.1. Các loại mô hình

Có nhiều loại mô hình khác nhau được phân chia dựa trên mục đích sử dụng, cách xử lý với tính tình cờ (randomness), tính tổng quát của ứng dụng...

Mục đích của mô hình là tối ưu hoá hay để mô tả. Một mô hình dùng để tối ưu hoá là một mô hình trong đó một đại lượng nào đó cần phải được cực tiểu hoá hay cực đại hoá. Ví dụ như cực đại hoá lợi nhuận hay cực tiểu hoá chi phí. Nói chung loại mô hình dùng để mô tả cho người dùng một hình dung đúng về thực tế, còn theo nghĩa hẹp nó mô tả về cách vận hành của hệ thống và không thực hiện một phép tối ưu nào.

Nói về tính tình cờ, hầu hết các hệ thống đều mang tính xác suất, nghĩa là hành vi của hệ thống không thể được đoán trước một cách chính xác, các dữ liệu nhập vào đều mang tính xác suất thống kê và các dữ liệu xuất ra cũng vậy. Tuy vậy, đa số các mô hình toán học đều là

mô hình tiên định (deterministic). Các mô hình tiên định thường dễ xây dựng hơn, ít tốn kém về thời gian và tiền bạc hơn.

Về tính tổng quát, có mô hình có thể chỉ được dùng với một hệ thống (custom-built model), nhưng cũng có những mô hình được xây dựng chung cho nhiều hệ thống khác nhau (ready-built model). Nói chung, custom-built model cung cấp một cái nhìn kỹ hơn về một hệ thống cụ thể, tuy nhiên thường tốn kém hơn để xây dựng vì phải làm từ những việc nhỏ nhất.

4.3.3.2. Các lớp mô hình

Thông thường các mô hình được phân thành các lớp sau:

- Mô hình chiến lược: được dùng cho công việc quản lý ở tầm cao, dùng để hỗ trợ xác định mục đích của tổ chức, các tài nguyên cần có để thực thi các mục đích này
- Mô hình chiến thuật: được dùng quản lý ở mức trung cấp, để giúp cấp phát và sử dụng tài nguyên của tổ chức.

- Mô hình hoạt động: dùng để hỗ trợ để ra những quyết định ngắn hạn (hàng ngày, hàng tuần).

4.3.3.3. Các vấn đề thường gặp với mô hình

- Khó khăn trong việc tìm dữ liệu nhập cho mô hình
- Khó khăn trong việc sử dụng dữ liệu xuất ra từ mô hình
- Khó khăn trong việc cập nhật hoá mô hình
- Sự thiếu tin cậy đối với mô hình của người dùng
- Ít có sự hợp nhất, tích hợp giữa các mô hình
- Sự tương tác yếu (nghèo nàn) giữa mô hình và người dùng
- Người dùng khó mà tạo mô hình của riêng họ
- Các mô hình thường ít đưa ra giải thích về dữ liệu xuất (output)

◆ 4.3.4. Thành phần đối thoại

Các khái niệm thành phần dữ liệu, thành phần đối thoại và thành phần mô hình cung cấp một phương pháp hữu hiệu để hiểu các thành phần của một DDS và các tương tác giữa chúng với nhau.

Thành phần dữ liệu cung cấp dữ liệu để xây dựng, kiểm tra và "bảo dưỡng" mô hình. Kết xuất của mô hình lại được lưu trong cơ sở dữ liệu nên có thể làm dữ liệu nhập cho các mô hình khác, do đó có thể tích hợp nhiều mô hình lại với nhau.

Thành phần đối thoại không chỉ giúp cho người dùng sử dụng tốt mô hình, sử dụng một DDS có hiệu quả để ra quyết định mà còn giúp người dùng xây dựng mô hình của riêng họ, cho những nhu cầu của riêng họ.

4.4. CÂY QUYẾT ĐỊNH

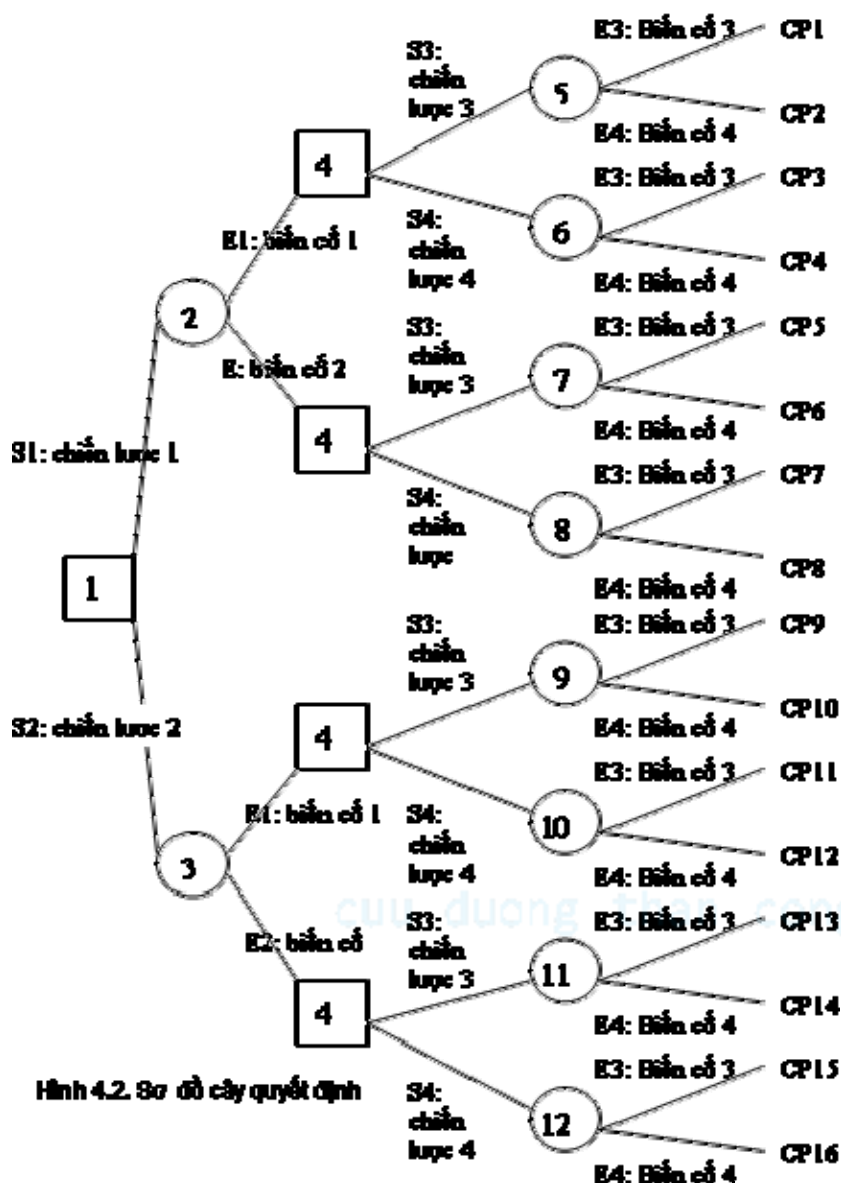
◆ 4.4.1. Giới thiệu cây quyết định

Cây quyết định bao gồm bốn thành phần: nhánh, nút quyết định, nút biến cố và kết quả. Nhánh là một biến cố hay chiến lược nối hai nút hay một nút

và kết quả. Nút quyết định là một điểm trên cây được biểu diễn bằng hình vuông và từ đó sẽ phát xuất nhiều nhánh. Mỗi nhánh từ nút quyết định là một chiến lược khả dĩ sẽ được người ra quyết định xem xét. Nút biến cố là một điểm trên cây quyết định được biểu diễn bằng hình tròn và từ đó cũng sẽ phát xuất nhiều nhánh , mỗi nhánh là một biến cố có thể xảy ra. Kết quả là một chuỗi chiến lược và biến cố tạo thành một con đường duy nhất trên cây quyết định từ điểm đầu cho đến điểm cuối.

cuu duong than cong. com

cuu duong than cong. com



Hình 4.2 là một cây quyết định tiêu biểu. Nút đầu tiên của cây sẽ bắt đầu bằng quyết định thứ nhất, một sự chọn lựa giữa chiến lược 1 hay chiến lược 2 sẽ xảy ra. Theo sau sự chọn chiến lược là một biến cố ngẫu nhiên: biến cố 1 hoặc biến cố 2. Lúc này người ra quyết định sẽ đứng giữa một trong 4 nút quyết định và phải thực hiện quyết định thứ 2 giữa chiến lược 3 và chiến lược 4. Theo sau quyết định

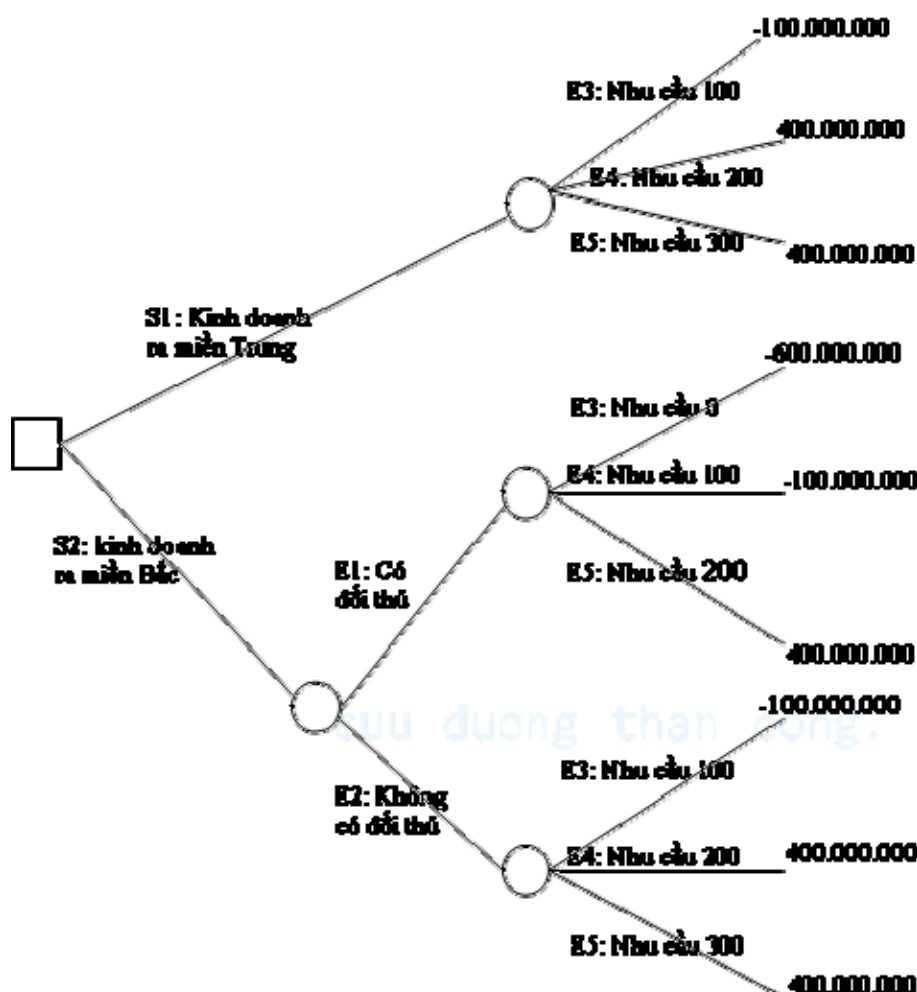
này là một biến cố ngẫu nhiên thứ 2: biến cố 3 và biến cố 4. Tùy theo con đường đã chọn, một trong 16 kết quả sẽ được xem là kết cuộc (từ CP1 đến CP16). Ví dụ: con đường gồm chiến lược 1, biến cố 2, chiến lược 3, biến cố 4 sẽ dẫn đến kết quả CP6.

Quyết định tối ưu cho loại bài toán này là chọn một bộ chiến lược duy nhất cho giá trị kỳ vọng tốt nhất ứng với nút đầu tiên. Lời giải này giả định có thể ấn định giá trị kỳ vọng ở từng nút biến cố và người ra quyết định sẽ thực hiện một quyết định phức tạp dựa trên nhiều biến cố ngẫu nhiên.

◆4.4.2. Suy diễn trên cây quyết định

Để trình bày cách giải các bài toán quyết định dựa trên sơ đồ cây, chúng ta hãy khảo sát bài toán sau: giả sử một công ty có trụ sở đặt tại thành phố Hồ Chí Minh muốn kinh doanh máy vi tính ra miền Bắc hoặc miền Trung. Nếu kinh doanh ra miền Trung, công ty sẽ không có đối thủ cạnh tranh và nhu cầu cho thị trường này khoảng 100, 200, 300 bộ/tháng. Nếu kinh doanh ra miền Bắc thì có thể bị cạnh tranh và nhu cầu cho thị trường này chỉ có thể

là 0, 100, 200 bộ/tháng. Hình 4.3 là sơ đồ cây quyết định của bài toán.



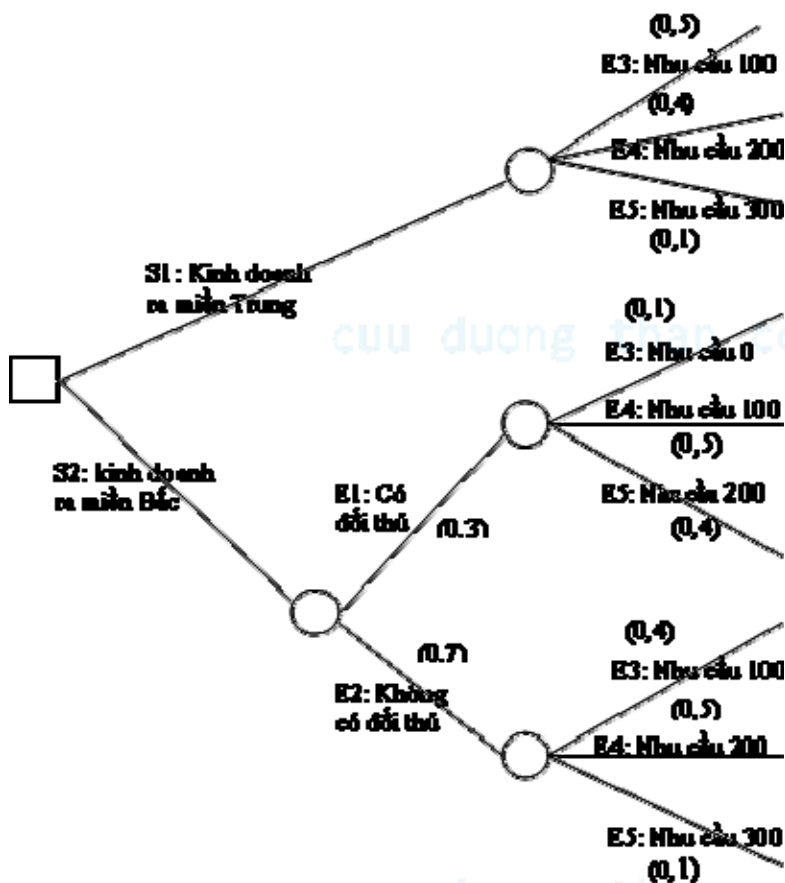
Hình 4.3. Sơ đồ cây quyết định của bài toán kinh doanh máy vi tính.

Số lượng máy vi tính dự định kinh doanh là 200 bộ/tháng. Bước đầu tiên để giải bài toán là ấn định kết quả thu được ứng với từng con đường trên cây. Giả định giá mua của một bộ máy vi tính là 3.000.000đ, tổng giá mua của 200 bộ sẽ là $200 \times 3.000.000 = 600.000.000$ đ. Giá bán dự kiến

cho mỗi bộ là 5.000.000 đ, chúng ta có kết cuộc CP1 và CP2 tương ứng:

$$CP1 = 100 \times 5.000.000 - 600.000.000 = -100.000.000 \text{ đ}$$

$$CP2 = 200 \times 5.000.000 - 600.000.000 = 400.000.000 \text{ đ}$$



Hình 4.4. Sơ đồ cây quyết định của bài toán kinh doanh máy tính.

Giá trị của kết quả sẽ nằm ở các điểm cuối của hình 4.4. Qua kinh nghiệm nhiều năm kinh doanh ở thị

trường này, người ra quyết định sẽ ra một số xác suất cho từng biến cố khả dĩ. Giá trị xác suất là con số được đặt trong cặp dấu ngoặc nằm phía trên các nhánh (xem Hình 4.4).

Người ra quyết định sẽ dùng giá trị kỳ vọng (**EMV**) làm tiêu chuẩn quyết định, do vậy chúng ta cần tính giá trị kỳ vọng của hai chiến lược khả dĩ là kinh doanh máy tính ra miền Bắc hay ra miền Trung, chúng ta có :

$$\text{EMV}(\text{S1: Kinh doanh ra miền Trung}) = 0,5(-100.000.000) + 0,4(400.000.000) + 0,1(400.000.000) = 150.000.000đ$$

EMV : Giá trị kỳ vọng

Đối với kinh doanh ra miền Bắc, đầu tiên chúng ta tính **EMV** của hai biến cố "có đối thủ" và "không có đối thủ" như sau:

$$\text{EMV}(\text{E1: Có đối thủ}) = 0,1(-600.000.000) + 0,5(-100.000.000) + 0,4(400.000.000) = 50.000.000đ$$

$$\begin{aligned} \text{EMV}(\text{E2: không có đối thủ}) &= \\ 0,4(100.000.000) &+ 0,5(400.000.000) + \\ 0,1(400.000.000) &= 200.000.000\text{đ} \end{aligned}$$

Do vậy:

$$\begin{aligned} \text{EMV}(\text{S2: Kinh doanh ra miền Bắc}) &= \\ 0,3(50.000.000) &+ \\ 0,7(200.000.000) &= 155.000.000\text{đ} \end{aligned}$$

Quyết định tối ưu sẽ theo hướng S2 vì mang lại kết quả cao hơn S1.

Phương pháp phân tích sử dụng trong bài toán cây quyết định là phương pháp "suy diễn lùi". Phương pháp này cho rằng để thẩm định một chiến lược nhất thiết phải khảo sát tất cả chiến lược và biến cố đi sau và cùng xuất phát từ chiến lược đó. Do vậy, các biến cố khả dĩ và nút quyết định sau cùng nhất sẽ được phân tích trước nhất. Kế đó sẽ lần ngược lên các nút trước để hướng về nút đầu tiên. Dùng kỹ thuật này, ta sẽ thiết lập các động tác tối ưu cho từng kết quả bằng cách duyệt trên sơ đồ cây.

CHƯƠNG 5. HỆ MYCIN 81

5.1. DẪN NHẬP

5.2. LÝ THUYẾT VỀ SỰ CHẮC CHẮN

5.2.1. Luật đơn giản

5.2.2. Luật phức tạp

5.2.3. Kết hợp nhiều luật có cùng kết luận

5.3. CHUỖI LẬP LUẬN

5.3.1. Mạng suy diễn

5.3.2. Lập luận trên mạng suy diễn

5.1. MỞ ĐẦU

MYCIN là một hệ lập luận trong y học được hoàn tất vào năm 1970 tại đại học Stanford, Hoa Kỳ. Đây là một hệ chuyên gia dựa trên luật và sự kiện. MYCIN sử dụng cơ chế lập luật gần đúng để xử lý các luật suy diễn dựa trên độ đo chắc chắn. Tiếp theo sau MYCIN, hệ EMYCIN ra đời. EMYCIN là một hệ chuyên gia tổng quát được tạo lập bằng cách loại phân cơ sở tri thức ra khỏi hệ MYCIN. EMYCIN cung cấp một cơ chế lập luận và tùy theo

bài toán cụ thể sẽ bổ sung tri thức riêng của bài toán đó để tạo thành hệ chuyên gia.

5.2. LÝ THUYẾT VỀ SỰ CHẮC CHẮN

Lý thuyết về sự chắc chắn dựa trên số lần quan sát. Đầu tiên theo lý thuyết xác suất cổ điển thì tổng số của sự tin tưởng và sự phản bác một quan hệ phải là 1. Tuy vậy trong thực tế các chuyên gia lại gán cho kết luận của họ những mệnh đề đại loại như “có vẻ đúng”, “gần đúng”, “đúng khoảng 70%” ...

Lý thuyết về sự chắc chắn dùng độ đo chắc chắn để lượng định những mệnh đề trên và cung cấp một số luật nhằm kết hợp các độ đo chắc chắn để dẫn đến kết luận. Trước khi tìm hiểu độ đo chắc chắn, chúng ta xét “sự tin cậy” và “sự phản bác” một quan hệ:

Gọi:

MB(H/E) là độ đo sự tin cậy của giả thuyết khi có chứng cứ E.

MD(H/E) là độ đo sự không tin cậy và giả thuyết khi có chứng cứ E.

Thế thì:

$0 < \mathbf{MB}(H/E) < 1$ trong khi $\mathbf{MD}(H/E) = 0$

$0 < \mathbf{MD}(H/E) < 1$ trong khi $\mathbf{MB}(H/E) = 0$

Độ đo chắc chắn $\mathbf{CF}(H/E)$ được tính bằng công thức:

$$\mathbf{CF}(H/E) = \mathbf{MB}(H/E) - \mathbf{MD}(H/E)$$

Khi giá trị của độ đo chắc chắn tiến dần về 1 thì chúng có biện minh cho giả thuyết nhiều hơn

Khi giá trị của độ đo chắc chắn tiến dần về -1 thì chúng có phản bác giả thuyết nhiều hơn.

Khi \mathbf{CF} có giá trị 0 có nghĩa là có rất ít chứng cứ để biện minh hay phản bác giả thuyết.

Khi các chuyên gia tạo ra các luật suy diễn, họ phải cung cấp độ đo chắc chắn của luật. Trong quá trình lập luận, chúng ta sẽ thu nhận được độ đo chắc chắn của chứng cứ và dựa vào hai độ đo chắc chắn trên để tính được độ đo chắc chắn của giả thuyết (còn được gọi là kết luận).

◆ 5.2.1. Luật đơn giản

Luật đơn giản có dạng sau:

If(e) then (c)

Gọi:

CF(e) là độ đo chắc chắn của chứng cứ.

CF(r) là độ đo chắc chắn của luật suy diễn.

Thế thì **CF(c)** là độ đo chắc chắn của kết luận sẽ được tính bằng công thức:

$$\mathbf{CF(c) = CF(e) * CF(r)}$$

Công thức này chính là nền tảng cho cơ chế lập luận của MYCIN.

♦5.2.2. Luật phức tạp

Trong thực tế chúng ta có thể gặp các luật phức tạp như sau:

If(e1 AND e2) then (c)

Toán tử **AND** được dùng để liên kết chứng cứ e1 và e2. Lúc bấy giờ ta có:

$$\mathbf{CF(e1 \text{ AND } e2) = MIN(CF(e1), CF(e2))}$$

Với luật có dạng **OR** như sau:

if (e1 OR e2) then (c)

Thì **CF (e1 OR e2) = MAX(CF(e1), CF(e2))**

Với luật có dạng **AND** và **OR** như sau:

if ((e1 AND e2) OR e3) then (c)

Thì **CF ((e1 AND e2) OR e3) = MAX(MIN(CF(e1), CF(e2)), CF(e3))**

Ngoài ra độ đo chắc chắn có dạng **NOT** được tính như sau:

CF(NOT e) = - CF(e)

Sau khi tính được độ đo chắc chắn của chúng có liên kết, ta dùng công thức nêu trong mục **5.2.1** để tính **CF** của kết luận.

♦ **5.2.3. Kết hợp nhiều luật có cùng kết luận**

Ví dụ: bạn có 2 luật có cùng kết luận như sau:

Luật 1: **If(e1) then (c)** với **CF(r1)**: độ đo chắc chắn của luật 1

Luật 2: If(e2) then (c) với CF(r2): độ đo chắc chắn của luật 2

Trong lý thuyết xác suất cổ điển ta có thủ tục nhân các độ đo xác suất để kết hợp các chứng cứ độc lập. Có thể dùng thủ tục này để kết hợp các kết luận của một số tùy ý các luật. Với **CF(t1)**, **CF(t2)** là **CF** của kết luận của luật 1 và 2, khi **CF(t1)** và **Cf(t2)** đều dương thì:

$$\text{Ctổng} = \text{CF}(t1) + \text{CF}(t2) - \text{CF}(t1) * \text{CF}(t2)$$

Khi **CF(t1)** và **Cf(t2)** đều âm thì:

$$\text{Ctổng} = \text{CF}(t1) + \text{CF}(t2) + \text{CF}(t1) * \text{CF}(t2)$$

Nếu **CF(t1)** khác dấu với **CF(t2)** thì:

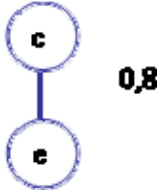
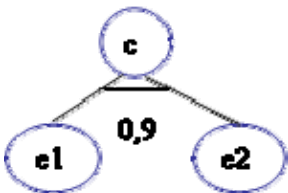
$$\text{Ctổng} = (\text{CF}(t1) + \text{CF}(t2)) / (1 - \text{MIN}(\text{ABS}(\text{CF}(t1)), \text{ABS}(\text{CF}(t2))))$$

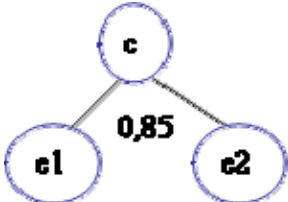
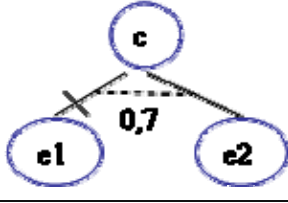
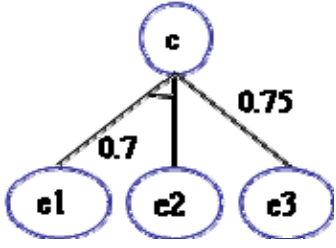
5.3. CHUỖI LẬP LUẬN

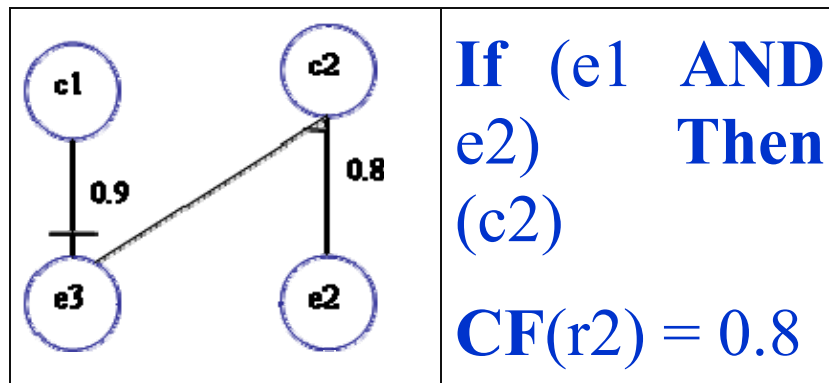
◆5.3.1. Mạng suy diễn

Cho đến lúc này, chúng ta chỉ mới xem xét các tình huống lập luận đơn giản theo đó kết luận cuối cùng được suy từ chứng cứ bằng một bước lập luận duy

nhất. Thực tế chúng ta có một mạng mà kết luận cuối cùng được suy từ một loạt chứng cứ qua nhiều kết luận trung gian. Trường hợp này được gọi là lập luận qua nhiều giai đoạn. Đối với loại lập luận này ta dùng một đồ thị dạng mạng suy diễn (trong đó các chứng cứ đơn và tiên đề là các nút) để biểu diễn mối liên hệ giữa các luật. Sau đây là một số dạng cơ bản trên mạng suy diễn.

Dạng biểu diễn trên mạng	Dạng luật
<p>a. Suy diễn đơn giản</p> 	<p>If(e) Then (c)</p> <p>$CF(r) = 0,8$</p>
<p>b. Suy diễn có AND</p> 	<p>If (e1 AND e2) Then (c)</p> <p>$CF(r) = 0,9$</p>

<p>c. Suy diễn OR</p> 	<p>If (e1 OR e2) Then (c)</p> <p>$CF(R) = 0.85$</p>
<p>d. Suy diễn có NOT</p> 	<p>If ((NOT e1) OR e2) Then (c)</p> <p>$CF(r) = 0.7$</p>
<p>e. Nhiều luật cho cùng kết luận</p> 	<p>If (e1 AND e2) Then (c)</p> <p>$CF(r1) = 0.7$</p> <p>If (e3) Then (c)</p> <p>$CF(r2) = 0.75$</p>
<p>f. Một chứng cứ được dùng trong hai luật</p>	<p>If (NOT (e1) Then (c1)</p> <p>$CF(r1) = 0.9$</p>



Ví dụ: chúng ta có 7 luật sau đây:

r1: If(e1) Then (c1) CF(r1) = 0,8

r2: If (e2) Then (c2) CF(r2) = 0,9

r3: If (e3) Then (c2) CF(r3) = 0,7

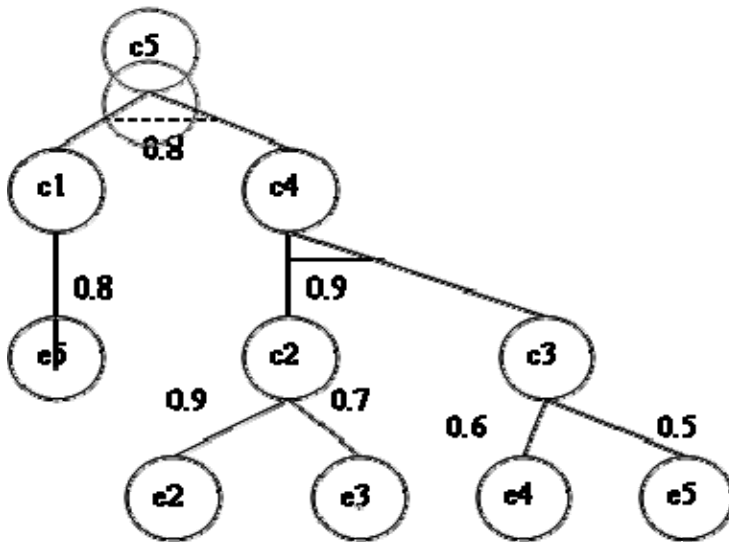
r4: If (e4) Then (c3) CF(r4) = 0,6

r5: If (NOT e5) Then (c3) CF(r5) = 0,5

r6: If (c2 AND c3) Then (c4) CF(r6) = 0,9

r7: If (c1 OR c4) Then (c5) CF(r7) = 0,8

Bảng luật này tạo thành mạng suy diễn ở hình 5.1 với c5 là giả thuyết cần hướng đến.



Hình 5.1. Mạng suy diễn

♦ 5.3.2. Lập luận trên mạng suy diễn

Giả sử các chứng cứ $e1, e2, e3, e4, e5$ có độ đo chắc chắn như sau:

$$CF(e1) = 0,9$$

$$CF(e2) = 0,9$$

$$CF(e3) = -0,3$$

$$CF(e4) = 0,4$$

$$CF(e5) = -0,3$$

Chúng ta sẽ lập luận từ các **CF** của chứng cứ dần lên giả thuyết $c5$ như sau:

Dựa vào luật **r1** tính được **CF(c1)**:

$$\mathbf{CF(c1) = CF(e1) * CF(r1) = 0,8*0,9 = 0,72}$$

Dựa vào luật **r2, r3** tính được **CF(c2)**

$$\text{Với luật } \mathbf{r2: CF(c2) = CF(e2) * CF(r2) = 0,9 * 0,9 = 0,81}$$

$$\text{Với luật } \mathbf{r3: CF(c2) = CF(e3) * CF(r3) = -0,3 * 0,7 = -0,21}$$

$$\text{Do } \mathbf{CF(c2)} \text{ của } \mathbf{r2} \text{ trái dấu với } \mathbf{CF(c2)} \text{ của } \mathbf{r3}, \text{ nên: } \mathbf{CF(c2)_{tổng} = (0,81 + (-0,21)) / (1 - \text{MIN}(0,81, 0,21)) = 0,74}$$

Dựa vào luật **r4, r5** ta tính được **CF(c3)**

Với luật **r4**:

$$\mathbf{F(c3) = CF(e4) * CF(r4) = 0,4 * 0,6 = 0,24}$$

Với luật **r5**:

$$\mathbf{F(c3) = CF(\text{NOT } e5) * CF(r5) = -CF(e5) * CF(r5) = 0,3 * 0,5 = 0,15}$$

$$\text{Do } \mathbf{CF(c3)} \text{ của } \mathbf{r4} \text{ và } \mathbf{CF(c3)} \text{ của } \mathbf{r5} \text{ cùng dương nên } \mathbf{CF(c3)_{tổng} = 0,24 + 0,15 - 0,24 * 0,15 = 0,324}$$

Dựa vào luật **r6** ta tính được **CF(c4)**:

$$\text{CF(c4)} = \text{MIN}(\text{CF(c2)}, \text{CF(c3)}) * \text{CF(r6)} = \text{MIN}(0,74, 0,324) * 0,9 = 0,324 * 0,9 = 0,292$$

Dựa vào luật **r7** ta tính được **CF(c5)**

$$\text{CF(c5)} = \text{MAX}(\text{CF(c1)}, \text{CF(c2)}) * \text{CF(r7)} = \text{MAX}(0,72, 0,292) * 0,8 = 0,58$$

Như thế độ chắc chắn của giả thuyết **c5** là 0,58.

CHƯƠNG 6. MẠNG TÍNH TOÁN

6.1. DẪN NHẬP

6.2. MẠNG TÍNH TOÁN

6.2.1. Các quan hệ

6.2.2. Mạng tính toán và các ký hiệu

6.3. VẤN ĐỀ TRÊN MẠNG TÍNH TOÁN

6.4. GIẢI QUYẾT VẤN ĐỀ

6.4.1. Tính giải được của bài toán

6.4.2. Lời giải của bài toán

6.4.3. Lời giải tối ưu của bài toán

6.4.4. Kiểm định giả thiết cho bài toán

6.4.5. Định lý về sự phân tích quá trình giải

6.5. ỨNG DỤNG TRONG CÁC PHẢN ỨNG HÓA HỌC 116

6.1. MỞ ĐẦU

Mạng tính toán là một dạng biểu diễn tri thức có thể dùng biểu diễn các tri thức về các vấn đề tính toán và được áp dụng một cách có hiệu quả để giải quyết các vấn đề này. *Mỗi mạng tính toán là một mạng ngữ nghĩa chứa các biến và những quan hệ có thể cài đặt và sử dụng được cho việc tính toán.* Có thể nói rằng mạng tính toán là một sự tổng quát hoá của kiểu dữ liệu trừu tượng có khả năng tự xây dựng các hàm dùng cho việc tổng hợp thành các chương trình.

Trong chương này chúng ta xét một mạng tính toán gồm một tập hợp các biến cùng với một tập các quan hệ (chẳng hạn các công thức) tính toán giữa các biến. Trong ứng dụng cụ thể mỗi biến và giá trị của nó thường gắn liền với một khái niệm cụ thể về sự vật, mỗi quan hệ thể hiện một sự tri thức về sự vật.

Cách biểu diễn tri thức tính toán dưới dạng các đối tượng này rất tự nhiên và gần gũi đối với cách nhìn và nghĩ của con người khi giải quyết các vấn đề tính toán liên quan đến một số khái niệm về các đối tượng, chẳng hạn như các tam giác, tứ giác, hình bình hành, hình chữ nhật, v.v....

6.2. MẠNG TÍNH TOÁN

♦ 6.2.1. Các quan hệ

Cho $M = \{x_1, x_2, \dots, x_m\}$ là một tập hợp các biến có thể lấy giá trị trong các miền xác định tương ứng D_1, D_2, \dots, D_m . Đối với mỗi quan hệ $R \subseteq D_1 \times D_2 \times \dots \times D_m$ trên các tập hợp D_1, D_2, \dots, D_m ta nói rằng quan hệ này liên kết các biến x_1, x_2, \dots, x_m , và ký hiệu là $R(x_1, x_2, \dots, x_m)$ hay vắn tắt là $R(x)$

(ký hiệu x dùng để chỉ bộ biến $\langle x_1, x_2, \dots, x_m \rangle$).
Quan hệ $R(x)$ xác định một (hay một số) ánh xạ :

$$fR_{u,v} : D_u \rightarrow D_v,$$

trong đó u, v là các bộ biến và $u \subseteq x, v \subseteq x$; D_u và D_v là tích của các miền xác định tương ứng của các biến trong u và trong v .

Ta có thể thấy rằng quan hệ $R(x)$ có thể được biểu diễn bởi một ánh xạ $fR_{u,v}$ với $u \cup v = x$, và ta viết :

$$fR_{u,v} : u \rightarrow v,$$

hay vắn tắt là:

$$f : u \rightarrow v.$$

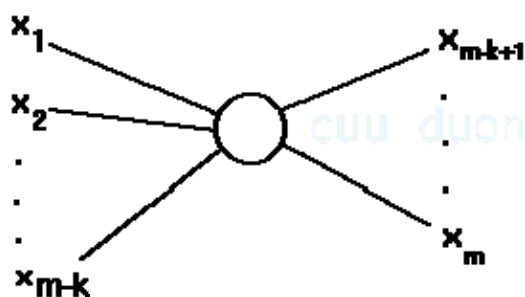
Đối với các quan hệ dùng cho việc tính toán, cách ký hiệu trên bao hàm ý nghĩa như là một hàm: ta có thể tính được giá trị của các biến thuộc v khi biết được giá trị của các biến thuộc u .

Trong phần sau ta xét các quan hệ xác định bởi các hàm có dạng:

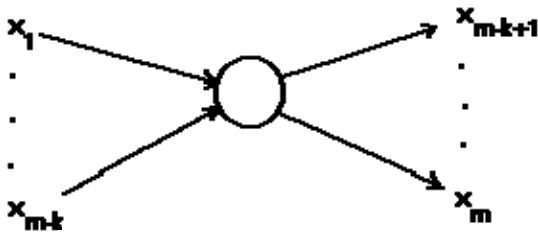
$$f : u \rightarrow v,$$

trong đó $u \cap v = \emptyset$ (tập rỗng). Đặc biệt là các **quan hệ đối xứng** có hạng (rank) bằng một số nguyên dương k . Đó là các quan hệ mà ta có thể tính được k biến bất kỳ từ $m-k$ biến kia (ở đây x là bộ gồm m biến $\langle x_1, x_2, \dots, x_m \rangle$). Ngoài ra, trong trường hợp cần nói rõ ta viết $u(f)$ thay cho u , $v(f)$ thay cho v . Đối với các quan hệ không phải là đối xứng có hạng k , không làm mất tính tổng quát, ta có thể giả sử quan hệ xác định duy nhất một hàm f với tập biến vào là $u(f)$ và tập biến ra là $v(f)$; ta gọi loại quan hệ này là quan hệ không đối xứng xác định một hàm, hay gọi vắn tắt là **quan hệ không đối xứng**.

Ta có thể vẽ hình biểu diễn cho các quan hệ đối xứng và các quan hệ không đối xứng (xác định một hàm) như trong hình 6.1 và 6.2.



Hình 6.1. Quan hệ đối xứng có hạng k



Hình 6.2. Quan hệ không đối xứng có hạng k

Nhận xét:

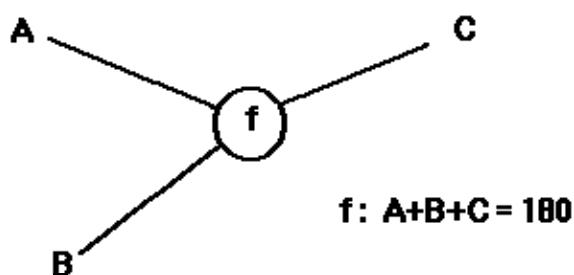
1/ Một quan hệ không đối xứng hạng k có thể được viết thành k quan hệ không đối xứng có hạng 1.

2/ Nếu biểu diễn một quan hệ đối xứng có hạng k thành các quan hệ đối xứng có hạng là 1 thì số quan hệ có hạng 1 bằng :

$$C_n^{k-1} = C_n^{k-1}$$

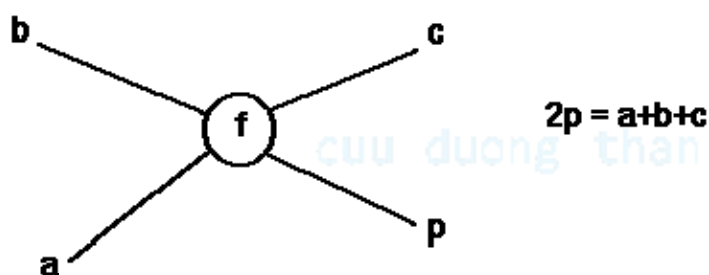
Dưới đây là một vài ví dụ về các quan hệ (tính toán) và mô hình biểu diễn tương ứng.

Ví dụ 1: Quan hệ f giữa 3 góc A, B, C trong tam giác ABC cho bởi hệ thức: $A+B+C = 180$ (đơn vị: độ).



Quan hệ f giữa 3 góc trong một tam giác trên đây là một quan hệ đối xứng có hạng 1.

Ví dụ 2: quan hệ f giữ a nửa chu vi p với các độ dài của 3 cạnh a, b, c :



Ví dụ 3: Quan hệ f giữ a n biến x_1, x_2, \dots, x_n được cho dưới dạng một hệ phương trình tuyến tính có nghiệm. Trong trường hợp này f là một quan hệ có hạng k bằng hạng của ma trận hệ số của hệ phương trình.

♦ 6.2.2. Mạng tính toán và các ký hiệu

Như đã nói ở trên, ta sẽ xem xét các mạng tính toán bao gồm một tập hợp các biến M và một tập hợp các quan hệ (tính toán) F trên các biến. Ta gọi một

mạng tính toán một cách vắn tắt là một *MTT*, và trong trường hợp tổng quát có thể viết:

$$M = \{ x_1, x_2, \dots, x_n \} ,$$

$$F = \{ f_1, f_2, \dots, f_m \} .$$

Đối với mỗi $f \in F$, ta ký hiệu $M(f)$ là tập các biến có liên hệ trong quan hệ f . Dĩ nhiên $M(f)$ là một tập con của M : $M(f) \subseteq M$. Nếu viết f dưới dạng:

$$f : u(f) \rightarrow v(f)$$

thì ta có $M(f) = u(f) \cup v(f)$.

Ví dụ 4:

Trong ví dụ 1 ở trên, ta có $M(f) = \{ A, B, C \} .$

Trong ví dụ 2 ở trên, ta có $M(f) = \{ a, b, c, p \} .$

Trong ví dụ 3 ở trên, ta có $M(f) = \{ x_1, x_2, \dots, x_n \} .$

Ví dụ 5 : Mạng tính toán cho một hình chữ nhật.

Việc tính toán trên một hình chữ nhật liên quan đến một số giá trị của hình chữ nhật như sau :

b_1, b_2 : hai cạnh của hình chữ nhật;

d : đường chéo của hình chữ nhật;

s : diện tích của hình chữ nhật;

p : chu vi của hình chữ nhật;

trong đó mỗi biến đều có giá trị là thuộc tập các số thực dương. Giữa các biến ta đã biết có các quan hệ sau đây:

$$f_1 : s = b_1 * b_2;$$

$$f_2 : p = 2 * b_1 + 2 * b_2;$$

$$f_3 : d^2 = b_1^2 + b_2^2;$$

các quan hệ này đều là các quan hệ đối xứng có hạng 1.

Như vậy tập biến và tập quan hệ của mạng tính toán này là :

$$M = \{ b_1, b_2, d, s, p \} ,$$

$$F = \{ f_1, f_2, f_3 \} .$$

6.3. VẤN ĐỀ TRÊN MẠNG TÍNH TOÁN

Cho một mạng tính toán (M, F) , M là tập các biến và F là tập các quan hệ. Giả sử có một tập biến $A \subseteq M$ đã được xác định (tức là tập gồm các biến đã biết trước giá trị), và B là một tập biến bất kỳ trong M .

🔗 Các vấn đề đặt ra là:

1. Có thể xác định được tập B từ tập A nhờ các quan hệ trong F hay không? Nói cách khác, ta có thể tính được giá trị của các biến thuộc B với giả thiết đã biết giá trị của các biến thuộc A hay không? CuuDangThanCong.com
2. Nếu có thể xác định được B từ A thì quá trình tính toán giá trị của các biến thuộc B như thế nào?
3. Trong trường hợp không thể xác định được B , thì cần cho thêm điều kiện gì để có thể xác định được B . CuuDangThanCong.com

Bài toán xác định B từ A trên mạng tính toán (M, F) được viết dưới dạng:

$$A \rightarrow B,$$

trong đó A được gọi là giả thiết, B được gọi là mục tiêu tính toán (hay tập biến cần tính) của vấn đề. Trường hợp tập B chỉ gồm có một phần tử b , ta viết vắn tắt bài toán trên là $A \rightarrow b$.

🔗 Định nghĩa 2.1:

Bài toán $A \rightarrow B$ được gọi là **giải được** khi có thể tính toán được giá trị các biến thuộc B xuất phát từ giả thiết A . Ta nói rằng một dãy các quan hệ $\{f_1, f_2, \dots, f_k\} \subseteq F$ là một **lời giải** của bài toán $A \rightarrow B$ nếu như ta lần lượt áp dụng các quan hệ f_i ($i=1, \dots, k$) xuất phát từ giả thiết A thì sẽ tính được các biến thuộc B . Lời giải $\{f_1, f_2, \dots, f_k\}$ được gọi là **lời giải tốt** nếu không thể bỏ bớt một số bước tính toán trong quá trình giải, tức là không thể bỏ bớt một số quan hệ trong lời giải. Lời giải được gọi là **lời giải tối ưu** khi nó có số bước tính toán ít nhất, tức là số quan hệ áp dụng trong tính toán là ít nhất.

Việc tìm lời giải cho bài toán là việc tìm ra một dãy quan hệ để có thể áp dụng tính ra được B từ A . Điều này cũng có nghĩa là tìm ra được một quá trình tính toán để giải bài toán.

Trong quá trình tìm lời giải cho bài toán chúng ta cần xét một dãy quan hệ nào đó xem có thể tính thêm được các biến từ một tập biến cho trước nhờ dãy quan hệ này hay không. Do đó chúng ta đưa thêm định nghĩa sau đây.

➤ Định nghĩa 2.2 :

Cho $D = \{ f_1, f_2, \dots, f_k \}$ là một dãy quan hệ của mạng tính toán (M, F) , A là một tập con của M . Ta nói dãy quan hệ D là **áp dụng được** trên tập A khi và chỉ khi ta có thể lần lượt áp dụng được các quan hệ f_1, f_2, \dots, f_k xuất phát từ giả thiết A .

📌 **Nhận xét** : Trong định nghĩa trên, nếu đặt : $A_0 = A$, $A_1 = A_0 \cup M(f_1)$, \dots , $A_k = A_{k-1} \cup M(f_k)$, và ký hiệu A_k là **$D(A)$** , thì ta có D là một lời giải của bài toán $A \rightarrow D(A)$. Trong trường hợp D là một dãy quan hệ bất kỳ (không nhất thiết là áp dụng được trên A), ta vẫn ký hiệu $D(A)$ là tập biến đạt được khi lần lượt áp dụng các quan hệ trong dãy D (nếu được). Chúng ta có thể nói rằng $D(A)$ là sự mở rộng của tập A nhờ áp dụng dãy quan hệ D .

➤ Thuật toán tính $D(A)$:

Nhập : Mạng tính toán (M, F) ,

$$A \subseteq M,$$

dãy các quan hệ $D = \{ f_1, f_2, \dots, f_m \}$.

Xuất : $D(A)$.

Thuật toán :

1. $A' \leftarrow A$;

2. for $i=1$ to m do

if f_i áp dụng được
trên A' then

$$A' \leftarrow A' \cup M(f_i);$$

3. $D(A) \leftarrow A'$

6.4. GIẢI QUYẾT VẤN ĐỀ

◆ 6.4.1. Tính giải được của bài toán

Trong mục này chúng ta nêu lên một khái niệm có liên quan đến tính giải được của vấn đề trên một mạng tính toán : bao đóng của một tập hợp biến trên một mạng tính toán.

🔴➡️ Định nghĩa 3.1:

Cho mạng tính toán (M, F) , và A là một tập con của M . Ta có thể thấy rằng có duy nhất một tập hợp B lớn nhất $\subseteq M$ sao cho bài toán $A \rightarrow B$ là giải được, và tập hợp B này được gọi là ***bao đóng*** của A trên mô hình (M, F) . Một cách trực quan, có thể nói bao đóng của A là sự mở rộng tối đa của A trên mô hình (M, F) . Ký hiệu bao đóng của A là \bar{A} , chúng ta có kiểm tra dễ dàng các tính chất liên quan đến bao đóng trong mệnh đề dưới đây.

🔴➡️ **Mệnh đề 3.1:** Cho A và B là hai tập con của M . Ta có:

$$(1) \bar{A} \supseteq A.$$

$$(2) \bar{\bar{A}} = A.$$

$$(3) A \subseteq B \Rightarrow \bar{A} \subseteq \bar{B}$$

$$(4) \overline{A \cap B} \subseteq \bar{A} \cap \bar{B}$$

$$(5) \overline{A \cup B} \supseteq \bar{A} \cup \bar{B}$$

Đối với tính giải được của bài toán, ta có thể dễ dàng kiểm chứng mệnh đề sau:

➤ **Mệnh đề 3.2.**

(1) Bài toán $A \rightarrow B$ là giải được khi và chỉ khi các bài toán $A \rightarrow b$ là giải được với mọi $b \in B$.

(2) Nếu $A \rightarrow B$ và $B \rightarrow C$ là các bài toán giải được thì bài toán $A \rightarrow C$ cũng giải được. Hơn nữa, nếu $\{f_1, f_2, \dots, f_m\}$ và $\{g_1, g_2, \dots, g_p\}$ lần lượt là lời giải của bài toán $A \rightarrow B$ và bài toán $B \rightarrow C$ thì $\{f_1, f_2, \dots, f_m, g_1, g_2, \dots, g_p\}$ là một lời giải của bài toán $A \rightarrow C$.

(3) Nếu bài toán $A \rightarrow B$ là giải được và B' là một tập con của B thì $A \rightarrow B'$ cũng là một bài toán giải được. Hơn nữa, nếu $\{f_1, f_2, \dots, f_m\}$ là một lời giải của bài toán $A \rightarrow B$ thì đó cũng là một lời giải của bài toán $A \rightarrow B'$.

Từ khái niệm bao đóng đã nói ở trên ta cũng có các định lý sau:

➤ **Định lý 3.1.** Trên một mạng tính toán (M, F) , bài toán $A \rightarrow B$ là giải được khi và chỉ khi $B \subseteq \bar{A}$

Từ định lý này, ta có thể kiểm tra tính giải được của bài toán $A \rightarrow B$ bằng cách tính bao đóng của tập A rồi xét xem B có bao hàm trong \bar{A} hay không.

• **Mệnh đề 3.3:** Cho một dãy quan hệ $D = \{ f_1, f_2, \dots, f_k \} \subseteq F$, $A \subseteq M$. Đặt :

$A_0 = A$, $A_1 = A_0 \cup M(f_1)$, ..., $A_k = A_{k-1} \cup M(f_k)$.
Ta có các điều sau đây là tương đương :

(1) Dãy D áp được trên A .

(2) Với mọi $i=1, \dots, k$ ta có:

$\text{Card} (M(f_i) \setminus A_{i-1}) \leq r(f_i)$ nếu f_i là quan hệ đối xứng,

$M(f_i) \setminus A_{i-1} \subseteq v(f_i)$ nếu f_i là quan hệ không đối xứng.

(ký hiệu $\text{Card} (X)$ chỉ số phần tử của tập X).

• **Ghi chú :** Dựa vào mệnh đề 3.3 ta có một thuật toán để kiểm tra tính áp dụng được của một dãy quan hệ D trên một tập biến A .

• **Định lý 3.2.** Cho một mạng tính toán (M, F) , A, B là hai tập con của M . Ta có các điều sau đây là tương đương:

(1) $B \subseteq \bar{A}$.

(2) Có một dãy quan hệ $D = \{ f_1, f_2, \dots, f_k \} \subseteq F$ thỏa các điều kiện :

(a) D áp được trên A .

(b) $D(A) \supseteq B$.

Chứng minh : Giả sử có (1), tức là $B \subseteq \bar{A}$. Khi đó bài toán $A \rightarrow B$ là giải được. Do đó có một dãy quan hệ $\{ f_1, f_2, \dots, f_k \} \subseteq F$ sao cho khi ta lần lượt áp dụng các quan hệ f_i ($i=1, \dots, k$) xuất phát từ giả thiết A thì sẽ tính được các biến thuộc B . Dễ dàng thấy rằng dãy $\{ f_1, f_2, \dots, f_k \}$ này thỏa các điều kiện (2).

Đảo lại, giả sử có (2). Với các điều kiện có được bởi (2) ta thấy $\{ f_i \}$ là lời giải của vấn đề $A_{i-1} \rightarrow A_i$, với mọi $i = 1, 2, \dots, k$. Từ mệnh đề 3.2 suy ra bài toán $A_0 \rightarrow A_k$ là giải được. Do đó bài toán $A \rightarrow B$ cũng giải được, suy ra $B \subseteq \bar{A}$ theo định lý 3.1.

Nhận xét :

1. dãy quan hệ $\{ f_1, f_2, \dots, f_k \}$ trong định lý trên là một lời giải của vấn đề $A \rightarrow B$ trên mạng tính toán (M, F) .
2. Trong lời giải $\{ f_1, f_2, \dots, f_k \}$ ta có thể bỏ bớt những f_i nào mà $M(f_i) \subseteq D_{i-1}(A)$, với $D_{i-1} = \{ f_1, f_2, \dots, f_{i-1} \}$.

Qua các định lý trên, ta thấy rằng việc xác định bao đóng của một tập biến trên mô hình tính toán là cần thiết. Dưới đây là thuật toán cho phép xác định bao đóng của tập hợp $A \subseteq M$. Trong thuật toán này chúng ta thử áp dụng các quan hệ $f \in F$ để tìm dần những biến thuộc M có thể tính được từ A ; cuối cùng sẽ được bao đóng của A .

Thuật toán 3.1. tìm bao đóng của tập $A \subseteq M$:

Nhập : Mạng tính toán
 (M, F) ,

$A \subseteq M$.

Xuất : \bar{A}

Thuật toán :

1. $B \leftarrow A;$

2. **Repeat**

$B1 \leftarrow B;$

for $f \in F$ **do**

if (f đối xứng **and**
 $\text{Card } (M(f) \setminus B) \leq$
 $r(f)$) **or** (f không đối
xứng **and** $M(f) \setminus B \subseteq$
 $v(f)$) **then**

begin

$B \leftarrow B \cup M(f);$

$F \leftarrow F \setminus \{ f \} ;$ // loại f
khỏi lần xem xét
sau

end;

Until $B = B1;$

3. $\bar{A} \leftarrow B;$

Ghi chú : Trên đây ta đã nêu lên đặc trưng cho tính giải được của bài toán trên một mạng tính toán và chỉ ra thuật toán để kiểm tra khi nào bài toán là giải được. Ngoài ra chúng ta sẽ còn nêu lên cách để kiểm định giả thiết của bài toán; và trong trường hợp bài toán chưa đủ giả thiết có thể bổ sung thêm nếu được.

♦6.4.2. Lời giải của bài toán

Ở trên ta đã nêu lên cách xác định tính giải được của bài toán. Tiếp theo, ta sẽ trình bày cách tìm ra lời giải cho bài toán $A \rightarrow B$ trên mạng tính toán (M, F) . Trước hết từ nhận xét sau định lý 3.2 ta có mệnh đề sau đây:

• **Mệnh đề 3.4:** Dãy quan hệ D là một lời giải của bài toán $A \rightarrow B$ khi và chỉ khi D áp dụng được trên A và $D(A) \supseteq B$.

Do mệnh đề trên, để tìm một lời giải ta có thể làm như sau: Xuất phát từ giả thiết A , ta thử áp dụng các quan hệ để mở rộng dần tập các biến có giá trị được xác định; và quá trình này tạo ra một sự lan truyền tính xác định trên tập các biến cho đến khi đạt đến tập biến B . Dưới đây là thuật toán tìm một

lời giải cho bài toán $A \rightarrow B$ trên mạng tính toán (M, F) .

➤ **Thuật toán 3.2:** Tìm một lời giải cho bài toán $A \rightarrow B$:

Nhập : Mạng tính toán (M, F) ,

tập giả thiết $A \subseteq M$,

tập biến cần tính $B \subseteq M$.

Xuất : lời giải cho bài toán $A \rightarrow B$

Thuật toán :

1. Solution \leftarrow empty; //
Solution là dãy các
quan hệ sẽ

//áp
dụng

2. if $B \subseteq A$ then

begin

Solution_found \leftarrow true;

```
// biến Solution_found  
=
```

```
// true khi bài toán là  
giải được
```

```
goto 4;
```

```
end
```

```
else
```

```
Solution_found ←  
false;
```

cuuduongthancong.com

```
3. Repeat
```

```
Aold ← A;
```

```
Chọn ra một  $f \in F$   
chưa xem xét;
```

```
while not
```

cuuduongthancong.com

```
Solution_found and  
(chọn được f) do
```

```
begin
```

```
if ( f đối xứng
```

and $0 < \text{Card}$
 $(M(f) \setminus A) \leq r(f)$)
or(f không đối
xứng **and** $\emptyset \neq$
 $M(f) \setminus A \subseteq v(f)$)

then

begin

$A \leftarrow A \cup M(f);$

$\text{Solution} \leftarrow$
 $\text{Solution} \cup \{ f \} ;$

end;

if $B \subseteq A$ **then**

Solution_found
 $\leftarrow \text{true};$

$\text{Chọn ra một } f \in$
 $F \text{ chưa xem xét};$

end; { while }

Until Solution_found

or ($A = Aold$);

4. **if** **not**
Solution_found **then**

Bài toán không có
lời giải;

else

Solution là một lời
giải;

Ghi chú :

1. Về sau, khi cần trình bày quá trình giải (hay bài giải) ta có thể xuất phát từ lời giải tìm được dưới dạng một dãy các quan hệ để xây dựng bài giải.

2. Lời giải (nếu có) tìm được trong thuật toán trên chưa chắc là một lời giải tốt. Ta có thể bổ sung thêm cho thuật toán ở trên thuật toán để tìm một lời giải tốt từ một lời giải đã biết nhưng chưa chắc là tốt. Thuật toán sẽ dựa trên định lý được trình bày tiếp theo đây.

• **Định lý 3.3.** Cho $D = \{ f_1, f_2, \dots, f_m \}$ là một lời giải của bài toán $A \rightarrow B$. Với mỗi $i=1, \dots, m$ đặt $D_i = \{ f_1, f_2, \dots, f_i \}$, $D_0 = \emptyset$. Ta xây dựng một họ các dãy con $S_m, S_{m-1}, \dots, S_2, S_1$ của dãy D như sau :

$S_m = \emptyset$ nếu D_{m-1} là một lời giải,

$S_m = \{ f_m \}$ nếu D_{m-1} không là một lời giải,

$S_i = S_{i+1}$ nếu $D_{i-1} \cup S_{i+1}$ là một lời giải,

$S_i = \{ f_i \} \cup S_{i+1}$ nếu $D_{i-1} \cup S_{i+1}$ không là một lời giải, với mọi $i = m-1, m-2, \dots, 2, 1$.

Khi đó ta có:

(1) $S_m \subseteq S_{m-1} \subseteq \dots \subseteq S_2 \subseteq S_1$.

(2) $D_{i-1} \cup S_i$ là một lời giải của bài toán $A \rightarrow B$ với mọi $i=m, \dots, 2, 1$.

(3) Nếu S'_i là một dãy con thật sự của S_i thì $D_{i-1} \cup S'_i$ không phải là một lời giải của bài toán $A \rightarrow B$ với mọi i .

(4) S_1 là một lời giải tốt của bài toán $A \rightarrow B$.

Từ định lý 3.3 trên ta có một thuật toán tìm lời giải tốt từ một lời giải đã biết sau đây:

➤ **Thuật toán 3.3.** tìm một lời giải tốt từ một lời giải đã biết.

Nhập : Mạng tính toán (M, F) ,
lời giải $\{ f_1, f_2, \dots, f_m \}$
của bài toán $A \rightarrow B$.

Xuất : lời giải tốt cho bài toán
 $A \rightarrow B$

Thuật toán :

1. $D \leftarrow \{ f_1, f_2, \dots, f_m \}$;

2. **for** $i=m$ **downto** 1 **do**

if $D \setminus \{ f_i \}$ là một lời giải
 then

$D \leftarrow D \setminus \{ f_i \}$;

3. D là một lời giải tốt.

Trong thuật toán 3.3 có sử dụng việc kiểm tra một dãy quan hệ có phải là lời giải hay không. Việc

kiểm tra này có thể được thực hiện nhờ thuật toán sau đây:

➤ **Thuật toán kiểm tra lời giải cho bài toán:**

Nhập : Mạng tính toán (M, F) ,

bài toán $A \rightarrow B$,

dãy các quan hệ $\{ f_1, f_2, \dots, f_m \}$.

Xuất : thông tin cho biết $\{ f_1, f_2, \dots, f_m \}$ có phải là lời giải của bài toán $A \rightarrow B$ hay không.

Thuật toán :

1. for $i=1$ to m do

if (f_i đối xứng and $\text{Card}(M(f_i) \setminus A) \leq r(f_i)$) or

(f_i không đối xứng and $M(f_i) \setminus A \subseteq v(f_i)$) then

$$A \leftarrow A \cup M(f_i);$$

2. if $A \supseteq B$ then

$\{f_1, f_2, \dots, f_m\}$ là lời giải

else

$\{f_1, f_2, \dots, f_m\}$ không là lời giải;

Ở trên ta đã có một thuật toán tổng quát để tìm lời giải tốt cho bài toán khi đã biết trước một lời giải. Tuy nhiên trong ứng dụng cụ thể ta thường gặp các quan hệ đối xứng có hạng một hơn là các quan hệ đối xứng có hạng lớn hơn 1. Trong trường hợp như thế ta có thể áp dụng một thuật toán khác để tìm một lời giải tốt từ một lời giải biết trước với mức độ tính toán ít hơn. Theo thuật toán này, ta lần lượt xem xét các quan hệ trong tập lời giải đã biết và chọn ra các quan hệ để đưa vào một lời giải mới sao cho trong lời giải mới này không thể bớt ra bất kỳ một quan hệ nào.

➡ **Thuật toán 3.4.** Tìm một lời giải tốt từ một lời giải đã biết không chứa quan hệ đối xứng hạng > 1 .

Nhập : Mạng tính toán (M, F) ,

Lời giải $\{ f_1, f_2, \dots, f_m \}$ của
bài toán $A \rightarrow B$,

Điều kiện : f_i không phải là
quan hệ đối xứng có hạng
lớn hơn 1.

Xuất : lời giải tốt cho bài toán
 $A \rightarrow B$

Thuật toán :

1. NewSolution $\leftarrow \emptyset$; //
đầu tiên tập lời giải mới

//
*chưa
có
quan
hệ
nào.*

$A_0 \leftarrow A$;

for $i=1$ **to** m **do** $A_i =$

$A_{i-1} \cup M(f_i);$

2. // *Dò theo chỉ số i từ 0 tìm i đầu tiên sao cho $A_i \supseteq B$.*

$i \leftarrow 0;$

while not ($A_i \supseteq B$)
do

$i \leftarrow i + 1;$

3. **if** $i = 0$ **then goto** 8;

4. $m \leftarrow i;$

5. // *Ghi nhận fm trong lời giải mới.*

$\text{NewSolution} \leftarrow \{ fm \}$
 $\cup \text{NewSolution};$

6. // *Dò theo chỉ số i từ 1 đến $m-1$ tìm i đầu tiên (nếu có) sao cho ta có thể áp dụng fm trên A_i để tính*

ra được // B.

$i_found \leftarrow \text{false};$

$i \leftarrow 1;$

while not i_found and $(i \leq m-1)$ do

if ((fm đối xứng **and**
 $\text{Card } (M(fm) \setminus A_i) \leq r(fm)$)

or (fm không đối xứng
and $M(fm) \setminus A_i \subseteq v(fm)$)

and ($B \subseteq M(fm) \cup A_i$)
then

$i_found \leftarrow \text{true}$

else

$i \leftarrow i + 1;$

7. if i_found then

begin

$$B \leftarrow (B \cup M(fm)) \cap A_i;$$

goto 2;

end;

8. NewSolution là một lời giải tốt của bài toán $A \rightarrow B$.

Ví dụ : Bây giờ ta xét một ví dụ cụ thể để minh họa cho các thuật toán trên.

Cho tam giác ABC có cạnh a và 2 góc kề là β , γ được cho trước.

Tính diện tích S của tam giác.

Để tìm ra lời giải cho bài toán trước hết ta xét mạng tính toán của tam giác. Mạng tính toán này gồm :

1. Tập biến $M = \{ a, b, c, \alpha, \beta, \gamma, h_a, h_b, h_c, S, p, R, r, \dots \}$,

trong đó a,b,c là 3 cạnh; α, β, γ là 3 góc tương ứng với 3 cạnh; h_a, h_b, h_c là 3 đường cao; S là diện tích tam giác; p là nửa chu vi; R là bán

kính đường tròn ngoại tiếp tam giác; r là bán kính đường tròn nội tiếp tam giác, v.v...

2. Các quan hệ:

$$f_1 : \alpha + \beta + \gamma = 180$$

$$f_2 : \frac{a}{\sin \alpha} = \frac{b}{\sin \beta}$$

$$f_3 : \frac{c}{\sin \gamma} = \frac{b}{\sin \beta}$$

$$f_4 : \frac{a}{\sin \alpha} = \frac{c}{\sin \gamma}$$

$$f_5 : p = (a+b+c) / 2$$

$$f_6 : S = a.h_a / 2$$

$$f_7 : S = b.h_b / 2$$

$$f_8 : S = c.h_c / 2$$

$$f_9 : S = a.b.\sin \gamma / 2$$

$$f_{10} : S = b.c.\sin \alpha / 2$$

$$f_{11} : S = c.a.\sin \beta / 2$$

$$f_{12} : S = \sqrt{p(p-a)(p-b)(p-c)}$$

V.V ...

3. Yêu cầu tính : S (diện tích của tam giác).

Theo đề bài ta có giả thiết là : $A = \{ a, \beta, \gamma \}$,
và tập biến cần tính là $B = \{ S \}$.

Áp dụng thuật toán tìm lời giải (thuật toán 3.2) ta có một lời giải cho bài tính là dãy quan hệ sau:

$\{ f_1, f_2, f_3, f_5, f_9 \}$.

Xuất phát từ tập biến A, lần lượt áp dụng các quan hệ trong lời giải ta có tập các biến được xác định mở rộng dần đến khi S được xác định :

$$\begin{aligned} \{ a, \beta, \gamma \} &\xrightarrow{f_1} \{ a, \beta, \gamma, \alpha \} \xrightarrow{f_4} \{ a, \beta, \gamma, \alpha, b \} \\ &\xrightarrow{f_3} \{ a, \beta, \gamma, \alpha, b, c \} \xrightarrow{f_2} \{ a, \beta, \gamma, \alpha, b, c, p \} \\ &\xrightarrow{f_5} \{ a, \beta, \gamma, \alpha, b, c, p, S \} . \end{aligned}$$

Có thể nhận thấy rằng lời giải này không phải là lời giải tốt vì có bước tính toán thừa, chẳng hạn là f_5 . Thuật toán 3.3 hay thuật toán 3.4 sẽ lọc ra từ lời giải trên một lời giải tốt là $\{ f_1, f_2, f_9 \}$:

$$\begin{aligned} \{ a, \beta, \gamma \} &\xrightarrow{f_1} \{ a, \beta, \gamma, \alpha \} \xrightarrow{f_4} \{ a, \beta, \gamma, \alpha, b \} \\ &\xrightarrow{f_9} \{ a, \beta, \gamma, \alpha, b, S \} . \end{aligned}$$

Theo lời giải này, ta có quá trình tính toán như sau :

bước 1: tính α (áp dụng f_1).

bước 2: tính b (áp dụng f_2).

bước 3: tính S (áp dụng f_9).

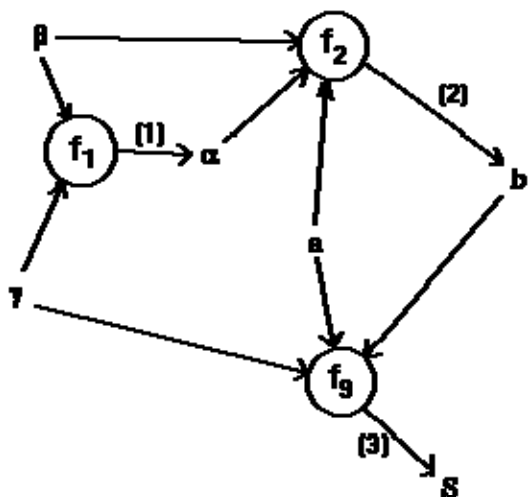
Quá trình tính toán (gồm 3 bước) này có thể được diễn đạt một cách rõ ràng trên sơ đồ mạng hình 6.3.

◆ 6.4.3. Lời giải tối ưu của bài toán

Liên quan đến lời giải tối ưu cho bài toán, ta có thể dễ dàng chứng minh mệnh đề dưới đây dựa vào tính thứ tự tốt của tập hợp các số tự nhiên.

➤ **Mệnh đề 3.3.** Nếu bài toán $A \rightarrow B$ là giải được thì sẽ tồn tại một lời giải tối ưu cho bài toán.

Ngoài ra, ta có thể áp dụng thuật toán A^* (thuật toán heuristic) để tìm ra một lời giải tối ưu trong trường hợp bài toán là giải được.



Hình 6.3. Sơ đồ thể hiện một mạng tính toán.

♦ 6.4.4. Kiểm định giả thiết cho bài toán

Xét bài toán $A \rightarrow B$ trên mạng tính toán (M, F) . Trong mục này chúng ta xem xét giả thiết A của bài toán xem thừa hay thiếu, và trong trường hợp cần thiết ta tìm cách điều chỉnh giả thiết A .

Trước hết ta cần xét xem bài toán có giải được hay không. Trường hợp bài toán giải được thì giả thiết là đủ. Tuy nhiên có thể xảy ra tình trạng thừa giả thiết. Để biết được bài toán có thật sự thừa giả thiết hay không, ta có thể dựa vào thuật toán tìm sự thu gọn giả thiết sau đây:

Thuật toán 3.5. Tìm một sự thu gọn giả thiết của bài toán.

Nhập : Mạng tính toán (M, F) ,

Bài toán $A \rightarrow B$ giải được,

Xuất : tập giả thiết mới $A' \subseteq A$ tối tiểu theo thứ tự \subseteq .

Thuật toán :

Repeat

$A' \leftarrow A;$

for $x \in A$ **do**

if $A \setminus \{x\} \rightarrow B$ giải được **then**

$A \leftarrow A \setminus \{x\};$

Until $A = A';$

Ghi chú : Trong thuật toán trên nếu tập giả thiết mới A' thật sự bao hàm trong A thì bài toán bị thừa giả thiết và ta có thể bớt ra từ giả thiết A tập hợp các biến không thuộc A' , coi như là giả thiết thừa.

Trường hợp bài toán $A \rightarrow B$ là không giải được thì ta nói giả thiết A thiếu. Khi đó có thể điều chỉnh bài toán bằng nhiều cách khác nhau để cho bài toán là giải được. Chẳng hạn ta có thể sử dụng một số phương án sau đây:

➤ **Phương án 1** : Tìm một $A' \subseteq M \setminus (\bar{A} \cup B)$ tối tiểu sao cho bao đóng của tập hợp $A' \cup A$ chứa B .

➤ **Phương án 2** : Khi phương án 1 không thể thực hiện được thì ta không thể chỉ điều chỉnh giả thiết để cho bài toán là giải được. Trong tình huống này, ta phải bỏ bớt kết luận hoặc chuyển bớt một phần kết luận sang giả thiết để xem xét lại bài toán theo phương án 1.

♦ 6.4.5. Định lý về sự phân tích quá trình giải

Xét bài toán $A \rightarrow B$ trên mạng tính toán (M, F) . Trong các mục trên chúng ta đã trình bày một số phương pháp để xác định tính giải được của bài toán, tìm ra một lời giải tốt cho bài toán.

Trong mục này ta nêu lên một cách xây dựng quá trình giải từ một lời giải đã biết. Đối với một lời

giải, rất có khả năng một quan hệ nào đó dẫn tới việc tính toán một số biến thừa, tức là các biến tính ra mà không có sử dụng cho các bước tính phía sau. Do đó, chúng ta cần xem xét quá trình áp dụng các quan hệ trong lời giải và chỉ tính toán các biến thật sự cần thiết cho quá trình giải theo lời giải. Định lý sau đây cho ta một sự phân tích tập các biến được xác định theo lời giải và trên cơ sở đó có thể xây dựng quá trình tính toán các biến để giải quyết bài toán.

➤ **Định lý 3.4.** Cho $\{ f_1, f_2, \dots, f_m \}$ là một lời giải tốt cho bài toán $A \rightarrow B$ trên một mạng tính toán (M, F) . Đặt :

$$A_0 = A, A_i = \{ f_1, f_2, \dots, f_i \} (A), \text{ với mọi } i=1, \dots, m.$$

Khi đó có một dãy $\{ B_0, B_1, \dots, B_{m-1}, B_m \}$, thỏa các điều kiện sau đây:

$$(1) B_m = B.$$

$$(2) B_i \subseteq A_i, \text{ với mọi } i=0, 1, \dots, m.$$

(3) Với mọi $i=1,...,m$, $\{ f_i \}$ là lời giải của bài toán $B_{i-1} \rightarrow B_i$ nhưng không phải là lời giải của bài toán $G \rightarrow B_i$, trong đó G là một tập con thật sự tùy ý của B_{i-1} .

Chứng minh : Ta xây dựng dãy $\{ B_0, B_1, ..., B_{m-1}, B_m \}$ bằng cách đặt: $B_m = B$, và ứng với mỗi $i < m$, đặt:

$$B_i = (B_{i+1} \cap A_i) \cup A_i',$$

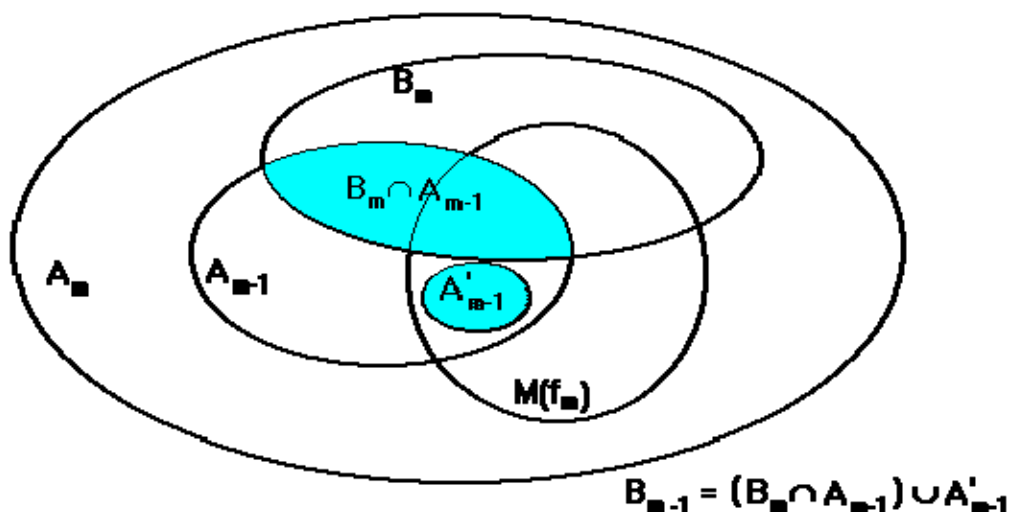
với A_i' là tập có ít phần tử nhất trong $A_i \setminus B_{i+1}$ sao cho f_{i+1} áp dụng được trên tập hợp $(B_{i+1} \cap A_i) \cup A_i'$. Thật ra, A_i' có được xác định như sau:

$A_i' = u(f_{i+1}) \setminus B_{i+1}$ nếu f_{i+1} không đối xứng,

và nếu f_{i+1} đối xứng thì

$A_i' =$ một tập con gồm $\max(0, t_i)$ phần tử của tập hợp $(M(f_{i+1}) \setminus B_{i+1}) \cap A_i$

trong đó $t_i = \text{card}(M(f_{i+1})) - r(f_{i+1}) - \text{card}(M(f_{i+1}) \cap B_{i+1} \cap A_i)$.



Với cách xây dựng này ta có thể kiểm tra được rằng dãy $\{ B_0, B_1, \dots, B_{m-1}, B_m \}$ thỏa mãn các điều kiện ghi trong định lý.

Ghi chú: [cuu duong than cong. com](http://cuuduongthancong.com)

(1) Từ định lý trên ta có quá trình tính toán các biến để giải bài toán $A \rightarrow B$ như sau:

bước 1: tính các biến trong tập $B_1 \setminus B_0$ (áp dụng f_1).

bước 2: tính các biến trong tập $B_2 \setminus B_1$ (áp dụng f_2). [cuu duong than cong. com](http://cuuduongthancong.com)

V.V...

bước m: tính các biến trong tập $B_m \setminus B_{m-1}$ (áp dụng f_m).

(2) Từ chứng minh của định lý trên, ta có thể ghi ra một thuật toán để xây dựng dãy các tập biến $\{ B_1', \dots, B_{m-1}', B_m' \}$ rời nhau cần lần lượt tính toán trong quá trình giải bài toán ($B_i' = B_i \setminus B_{i-1}$) gồm các bước chính như sau:

- xác định các tập A_0, A_1, \dots, A_m .
- xác định các tập $B_m, B_{m-1}, \dots, B_1, B_0$.
- xác định các tập B_1', B_2', \dots, B_m' .

Ví dụ :

Giả sử trên một mạng tính toán ta có $\{ f_1, f_2, f_3 \}$ là một lời giải tốt cho bài toán $A \rightarrow B$, trong đó :

$$A = \{ a_1, b_1, b_2 \},$$

$$B = \{ b_3 \},$$

f_1, f_2 là các quan hệ đối xứng có hạng 2, f_3 là quan hệ đối xứng có hạng 1, và

$$M(f_1) = \{ a_1, b_1, c_1, d_1 \},$$

$$M(f_2) = \{ a_1, c_1, b_2, d_2, e_2 \},$$

$$M(f_3) = \{ b_1, e_2, b_3 \}.$$

Dựa theo sự phân tích quá trình giải trong định lý trên ta có :

$$A_0 = A,$$

$$A_1 = \{ a_1, b_1, b_2, c_1, d_1 \} ,$$

$$A_2 = \{ a_1, b_1, b_2, c_1, d_1, d_2, e_2 \} ,$$

$$A_3 = \{ a_1, b_1, b_2, c_1, d_1, d_2, e_2, b_3 \} ,$$

$$B_3 = B,$$

$$B_2 = \{ b_1, e_2 \} ,$$

$$B_1 = \{ b_1, a_1, c_1, b_2 \} ,$$

$$B_0 = \{ a_1, b_1, b_2 \} ,$$

và các tập biến cần lần lượt tính toán trong bài giải cho bài toán là :

$$B_1' = \{ c_1 \} ,$$

$$B_2' = \{ e_2 \} ,$$

$$B_3' = \{ b_3 \} .$$

Từ đó ta có quá trình tính toán theo lời giải trên như sau:

tính c_1 (áp dụng f_1),

tính e_2 (áp dụng f_2),

tính b_3 (áp dụng f_3).

6.5. ỨNG DỤNG TRONG CÁC PHẢN ỨNG HÓA HỌC

Ngoài ứng dụng đã nêu trong các ví dụ ở các mục trên về việc giải các bài toán về tam giác và tứ giác, mạng tính toán có thể được áp dụng trong việc biểu diễn và giải một số bài toán trên các phản ứng hóa học. Chúng ta biết rằng trong hóa học, việc xem xét các phản ứng hóa học là một trong những vấn đề quan trọng. Về mặt tri thức người ta đã biết được nhiều chất và các phản ứng hóa học có thể chuyển hóa từ một số chất này thành các chất khác. Tại thời bỏ qua một số điều kiện phản ứng, ta có thể xem tri thức đó như một mạng tính toán mà mỗi phản ứng là một quan hệ của mạng. Ví dụ như phản ứng điều chế Clo từ axit Clohidric và đioxit mangan :



Phản ứng trên có thể được xem như một quan hệ cho chúng ta có được các chất Cl_2 , MnCl_2 , H_2O từ các chất MnO_2 , HCl .

Trong mục này ta dùng mạng tính toán để giải 2 bài toán sau:

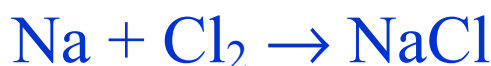
1. Cho một số chất, hỏi có điều chế được một vài chất nào đó không?
2. Tìm các phương trình phản ứng để biểu diễn dãy các biến hóa, chẳng hạn như các dãy :



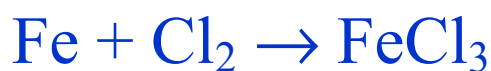
Kiến thức ta cần có đối với 2 bài toán trên là tập hợp tất cả các chất cùng với các phản ứng hóa học được phân loại thành các nhóm phản ứng khác nhau. Dưới đây chúng ta liệt kê một số nhóm phản ứng hóa học được lưu trữ trong phần kiến thức của chương trình.

➤ Một số phản ứng liên quan đến khí Clo:

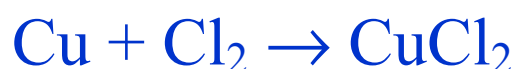
1/ Natri nóng chảy cháy trong Clo cho phản ứng tạo thành natri clorua:



2/ Bột sắt nóng chảy trong Clo cho phản ứng:



3/ Nung đỏ dây đồng cho vào khí Clo, ta có phản ứng:



4/ Clo tác dụng với nước:



5/ Điều chế Clo từ axit Clohidric và đioxit mangan:



6/ Điều chế Clo bằng axit clohidric và Kali pemanganat:



7/ Điện phân dung dịch đậm đặc muối ăn trong nước:

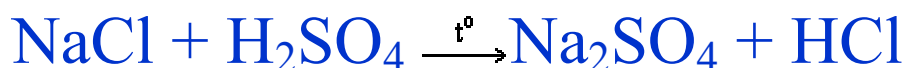
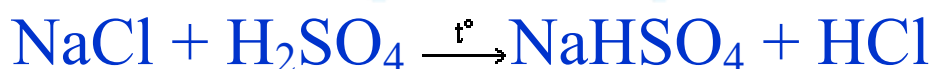


• Một số phản ứng khác :



• Các phản ứng liên quan đến Hidro Clorua HCl :

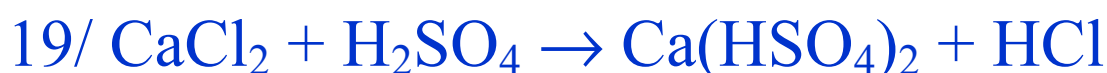
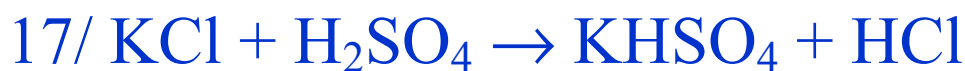
15/ Cho Natri Clorua tinh thể tác dụng với axit sunfuric đậm đặc, đun nóng (phương pháp sunfat), tùy theo nhiệt độ ta có các phản ứng sau đây :



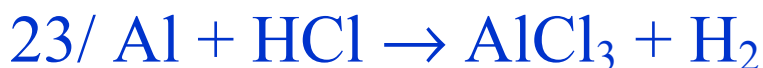
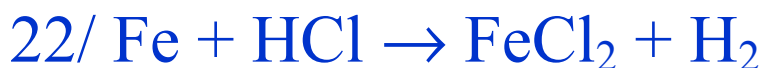
16/ Phản ứng điều chế HCl bằng phương pháp tổng hợp :

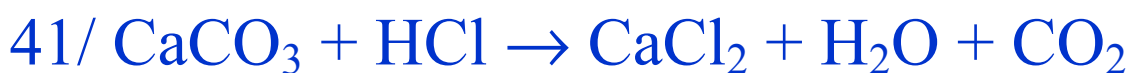


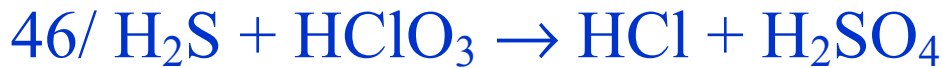
➤ **Một số phản ứng khác :**



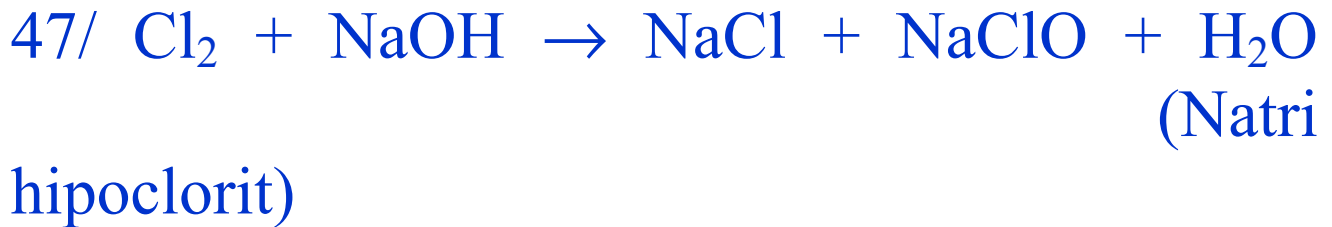
➤ **Các phản ứng của axit clohidric và muối clorua:**







•➡ **Nước Javen** : Dẫn Clo vào dung dịch NaOH:



tương tự ta cũng có :

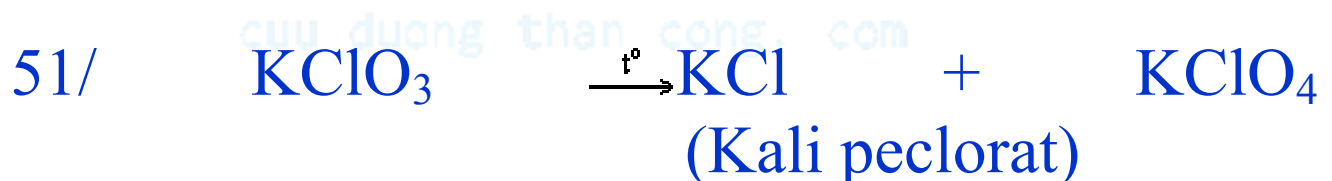
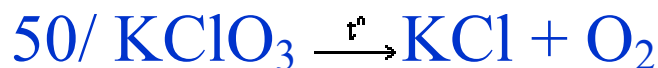


•➡ **Kali Clorat KClO_3**

Clo đi vào dung dịch kiềm đun nóng đến 100°C sẽ cho phản ứng :



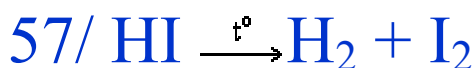
•➡ **Kali Clorat** bị phân hủy khi đun nóng theo các phương trình :



•➡ **Clorua vôi** : Clo tác dụng với vôi:



🌐 Các phản ứng của Brom và Iot :

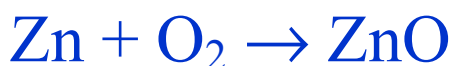


Dựa trên các thuật toán tìm lời giải trên mạng tính toán ta có thể giải được các bài toán như trong các ví dụ sau:

Ví dụ 1: Viết phương trình phản ứng biểu diễn các biến hóa sau :

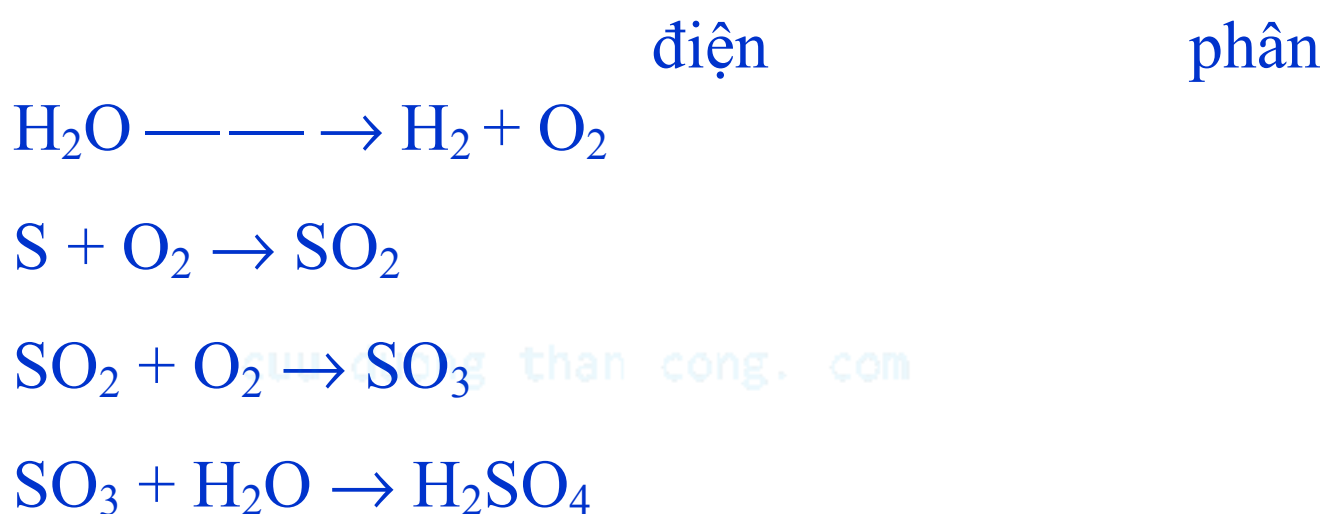


Giải : Trên cơ sở dò tìm các phản ứng (xem là các quan hệ của mạng tính toán các chất hóa học) đã biết ta có thể tìm thấy được các phản ứng sau đây :



Ví dụ 2 : Từ lưu huỳnh (S) và nước (H₂O) ta có thể điều chế được axit sunfuaric (H₂SO₄) không ?

Giải : Áp dụng các thuật toán tìm lời giải cho mạng tính toán các chất hóa học, dò theo các phản ứng liên quan đến lưu huỳnh và nước ta tìm ra được quá trình điều chế như sau :



Tương tự như 2 ví dụ trên, với tri thức gồm các phản ứng hóa học đã biết dưới dạng một mạng các chất hóa học chúng ta cũng có thể dễ dàng giải các bài toán sau đây:

Ví dụ 3: Viết các phương trình phản ứng để thực hiện các biến hóa theo các sơ đồ sau đây :



Ví dụ 4: Hoàn thành các phương trình phản ứng sau đây:



Ví dụ 5: Viết phương trình phản ứng theo các sơ đồ sau:



Ví dụ 6: Từ muối NaCl và nước (H₂O) ta có thể điều chế được axit clohidric (HCl) và NaOH không?

CHƯƠNG 7. HỆ HỌC

7.1. DẪN NHẬP

7.2. CÁC HÌNH THỨC HỌC

7.2.1. Học vẹt

7.2.2. Học bằng cách chỉ dẫn

7.2.3. Học bằng qui nạp

7.2.4. Học bằng tương tự

7.2.6. Học dựa trên tình huống

7.2.7. Khám phá hay học không giám sát

7.3. CÂY ĐỊNH DANH

7.3.1. Thí dụ về thế giới thực thu gọn

7.3.2. Phân loại đối tượng theo các thuộc tính

7.3.3. Độ lộn xộn của tập hợp

7.3.4. Chuyển cây sang luật

7.3.4.1. Lược bỏ giả thiết không cần thiết trong luật

7.3.4.2. Lược bỏ luật thừa

7.4. THUẬT GIẢI ILA

7.4.1. Xác định dữ liệu

7.4.2. Thuật giải ILA

7.4.3. Mô tả thuật giải ILA

7.4.4. Đánh giá thuật giải

7.1. MỞ ĐẦU

Các chương trước đã thảo luận về biểu diễn tri thức và các kỹ thuật suy diễn. Trong trường hợp này giả định đã có sẵn tri thức và có thể biểu diễn tường minh tri thức. Tuy vậy trong nhiều tình huống, sẽ không có sẵn tri thức như:

- Kỹ sư phần mềm cần thu nhận tri thức từ chuyên gia lĩnh vực.
- Cần biết các luật mô tả lĩnh vực cụ thể.
- Bài toán không được biểu diễn tường minh theo luật, sự kiện hay các quan hệ.

Do vậy cần phát triển các hệ thống có thể học từ tập các ví dụ. Có hai tiếp cận cho hệ thống học là học từ ký hiệu và học từ dữ liệu số. Học từ ký hiệu bao gồm việc hình thức hóa, sửa chữa các luật

tường minh, sự kiện và các quan hệ. Học từ dữ liệu số được áp dụng cho những hệ thống được mô hình dưới dạng số liên quan đến các kỹ thuật nhằm tối ưu các tham số. Học theo dạng số bao gồm mạng neural nhân tạo, thuật giải di truyền, bài toán tối ưu truyền thống. Các kỹ thuật học theo số không tạo ra CSTT tường minh.

7.2. CÁC HÌNH THỨC HỌC

Có thể phân chia các loại học như sau:

♦ 7.2.1. Học vẹt

Hệ tiếp nhận các khẳng định của các quyết định đúng. Khi hệ tạo ra một quyết định không đúng, hệ sẽ đưa ra các luật hay quan hệ đúng mà hệ đã sử dụng. Hình thức học vẹt nhằm cho phép chuyên gia cung cấp tri thức theo kiểu tương tác.

♦ 7.2.2. Học bằng cách chỉ dẫn

Thay vì đưa ra một luật cụ thể cần áp dụng vào tình huống cho trước, hệ thống sẽ được cung cấp bằng các chỉ dẫn tổng quát. Ví dụ: "gas hầu như bị thoát ra từ van thay vì thoát ra từ ống dẫn". Hệ thống

phải tự mình đề ra cách biến đổi từ trừu tượng đến các luật khả dụng.

♦7.2.3. Học bằng qui nạp

Hệ thống được cung cấp một tập các ví dụ và kết luận được rút ra từ từng ví dụ. Hệ liên tục lọc các luật và quan hệ nhằm xử lý từng ví dụ mới.

♦7.2.4. Học bằng tương tự

Hệ thống được cung cấp đáp ứng đúng cho các tác vụ tương tự nhưng không giống nhau. Hệ thống cần làm thích ứng đáp ứng trước đó nhằm tạo ra một luật mới có khả năng áp dụng cho tình huống mới.

♦7.2.5. Học dựa trên giải thích

Hệ thống phân tích tập các lời giải ví dụ (và kết quả) nhằm ấn định khả năng đúng hoặc sai và tạo ra các giải thích dùng để hướng dẫn cách giải bài toán trong tương lai.

♦7.2.6. Học dựa trên tình huống

Bấy kỳ tính huống nào được hệ thống lập luận đều được lưu trữ cùng với kết quả cho dù đúng hay sai.

Khi gặp tình huống mới, hệ thống sẽ làm thích nghi hành vi đã lưu trữ với tình huống mới.

♦ 7.2.7. Khám phá hay học không giám sát

Thay vì có mục tiêu tường minh, hệ khám phá liên tục tìm kiếm các mẫu và quan hệ trong dữ liệu nhập. Các ví dụ về học không giám sát bao gồm gom cụm dữ liệu, học để nhận dạng các đặc tính cơ bản như cạnh từ các điểm ảnh.

7.3. CÂY ĐỊNH DANH

Cây định danh là một công cụ khá phổ biến trong nhiều dạng ứng dụng, với cơ chế rút trích các luật nhân quả xác định các mẫu dữ liệu.

♦ 7.3.1. Thí dụ về thế giới thực thu gọn

Tưởng tượng có các dữ liệu về bài toán đánh giá độ rám nắng tại một nơi nghỉ mát. Có người vui vì ngấm đen, nhưng cũng có người rát vì rộp da. Dữ liệu quan sát trên tám người được ghi lại theo bảng sau đây.

TT	Tên người	Màu tóc	Chiều cao	Cân nặng	Dùng thuốc?	Kết quả
----	-----------	---------	-----------	----------	-------------	---------

1	Hoa	Đen	Tầm thuốc	Nhẹ	Không	Bị rám
2	Lan	Đen	Cao	Vừa phải	Có	Không
3	Xuân	Râm	Thấp	Vừa phải	Có	Không
4	Hạ	Đen	Thấp	Vừa phải	Không	Bị rám
5	Thu	Bạc	Tầm thuốc	Nặng	Không	Bị rám
6	Đông	Râm	Cao	Nặng	Không	Không
7	Mơ	Râm	Tầm thuốc	Nặng	Không	Không
8	Đào	Đen	Thấp	Nhẹ	Có	Không

Bảng 7.1. Số liệu quan sát về hiện tượng rám nắng.

Nếu có ba dữ liệu khác nhau về màu tóc, cân nặng, chiều cao và người ta có thể dùng thuốc hoặc không thì sẽ có $3 \times 3 \times 3 \times 2 = 54$ tổ hợp. Một người

mới được chọn thì xác suất khớp được với mẫu là $8/54 = 0.15$. Xác suất này cao hơn thực tế do còn nhiều thuộc tính và nhiều giá trị khác mà thuộc tính có thể nhận.

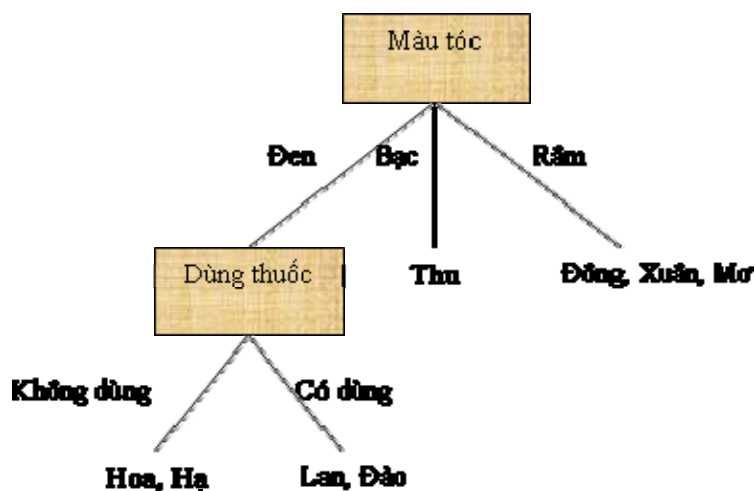
Do vậy nếu so sánh các thuộc tính của đối tượng chưa biết với thuộc tính của các đối tượng thống kê thì không chính xác. Một số nhận xét về việc giải bài toán này:

- Người ta có thể xử lý dữ liệu như không gian các đặc tính, nhưng nếu không rõ thuộc tính nào là quan trọng hơn thì cũng khó tìm thấy đối tượng khớp nhất.

- Có thể dùng không gian các thể hệ để cô lập các thuộc tính liên quan và thuộc tính không liên quan. Tuy nhiên thường không tìm thấy lý do thuyết phục để tin được mô hình phân loại có thể được diễn tả như tổ hợp các giá trị của tập các thuộc tính. Mặt khác cũng có thể các mẫu bị nhiễu, không thực chính xác như thế giới thực.

- Người ta dùng thủ tục phân loại chính xác các mẫu. Khi thủ tục đã học mẫu với số lượng mẫu

"khá đủ", thủ tục có thể nhận ra đối tượng chưa biết.



Hình 7.1. Cây định danh (*Người có tên ghi đậm là người bị râm nắng*).

Một cách phù hợp cho phép thực hiện các thủ tục thử nghiệm các thuộc tính là sắp xếp các thử nghiệm trên **cây định danh**. Do cây định danh thuộc loại cây quyết định, đặc tả của nó như đặc tả cây quyết định.

➤ **Định nghĩa 7.1 : Cây định danh (Identification tree)**

Cây định danh có thể hiện như cây quyết định, trong đó mỗi tập các kết luận được thiết lập ngầm định bởi một danh sách đã biết.

Cây định danh dùng để xác định người bị râm nắng với kiểm tra đầu tiên là màu tóc. Nếu thấy tóc đen người ta kiểm tra có dùng thuốc không? Ngược lại, nếu tóc bạc hay râu, người ta không cần kiểm tra. Nói chung việc chọn tiến hành loại kiểm tra nào là phụ thuộc vào kết quả của lần kiểm tra trước. Xem ví dụ thể hiện trong hình 7.1.

Mỗi đối tượng đưa vào định danh đi xuống theo một nhánh cây, tùy theo các giá trị thuộc tính của nó. Cây định danh như hình vẽ có thể phân loại người trong cơ sở dữ liệu "râm nắng" vì mỗi người ứng với một nút lá trên cây định danh. Nút này dùng cho một hay nhiều người. Trong hình cho thấy người bị râm nắng được đánh dấu bằng tên in đậm hơn.

Người ta có thể đưa ra một cây, có các nút lá ứng với mỗi người, không liên quan gì đến thuộc tính râm nắng. Liệu nó được dùng phân loại không?

So sánh cây định danh và cây tổng quát, người ta thấy cây thứ nhất là cây định danh, có vẻ liên quan đến sự râm nắng nhiều hơn cây thứ hai. Cây định

danh tỏ ra đã biết rằng màu tóc và phần lộ ra ánh nắng liên quan trực tiếp đến tính rám nắng.

Làm sao có thể chương trình hóa để đến được cùng một kết luận mà không cần biết trước màu tóc và việc dùng thuốc liên quan đến đặc tính của da? Một trong các giải đáp là phát biểu của Occam.

• **Phát biểu Occam dùng cho các cây định danh:**

Thế giới vốn đơn giản. Do vậy cây định danh gồm các mẫu là cái thích hợp nhất để định danh các đối tượng chưa biết một cách chính xác.

Theo phát biểu này, cây định danh đầu tiên nhỏ hơn cây sau nên nó phù hợp hơn.

♦ **7.3.2. Phân loại đối tượng theo các thuộc tính**

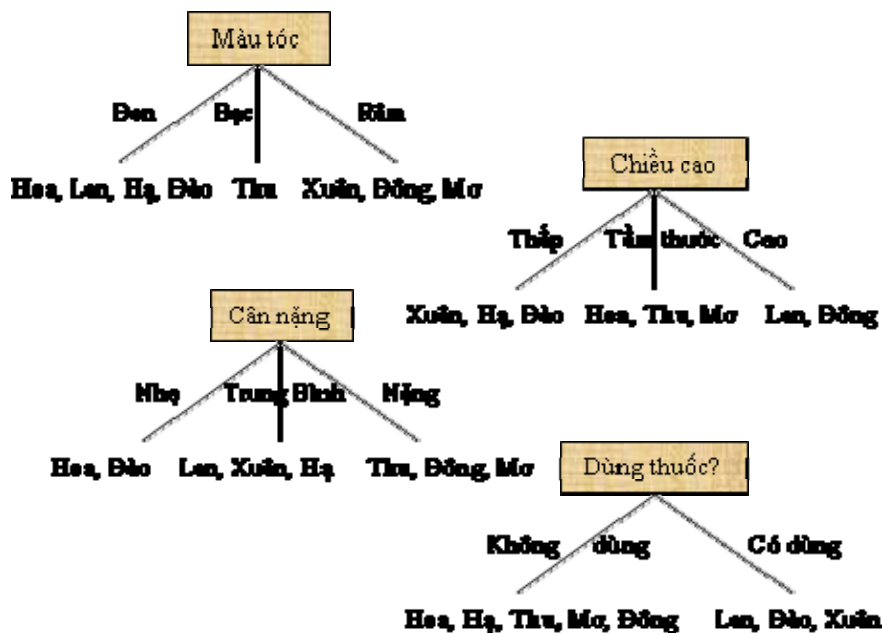
Nếu như ta tìm kiếm cây định danh nhỏ nhất khi cần có rất nhiều thử nghiệm thì thực là không thực tế. Chính vì vậy mà cũng nên dừng lại ở thử tục xây dựng những cây định danh nhỏ, dù rằng nó không phải là nhỏ nhất. Người ta chọn thử nghiệm cho phép chia cơ sở dữ liệu các mẫu thành các tập con.

Trong đó nhiều mẫu cùng chung một loại. Đối với mỗi tập có nhiều loại mẫu, dùng thử nghiệm khác để chia các đối tượng không đồng nhất thành các tập chỉ gồm đối tượng đồng nhất.

Xét ví dụ thể hiện ở hình 7.2. Cơ sở dữ liệu "rám nắng" có thể được chia nhỏ theo bốn thử nghiệm ứng với bốn thuộc tính:

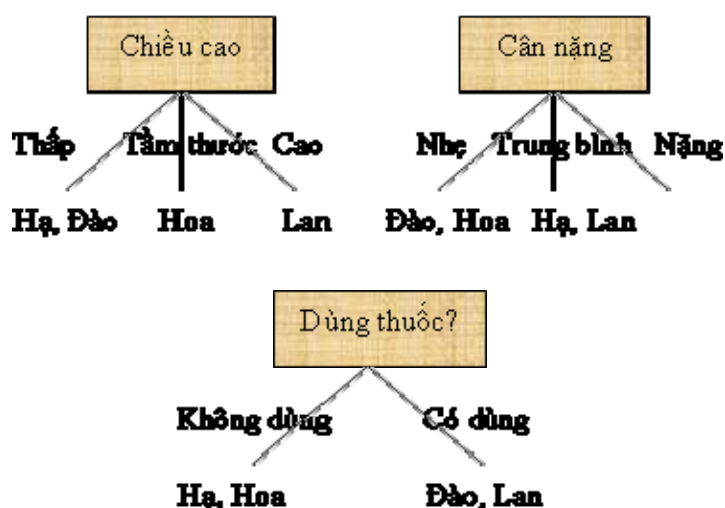
- Thử nghiệm theo cân nặng là tồi nhất nếu người ta đánh giá thử nghiệm theo các tập đồng nhất, có cùng tính chất rám nắng. Sau khi dùng thử nghiệm này, những mẫu rám nắng nằm đều ở các tập.
- Thử nghiệm theo chiều cao có vẻ tốt hơn. Có hai người trong một tập đồng nhất. Hai tập kia có lần cả người rám và không rám nắng.
- Thử nghiệm về việc dùng thuốc thu được ba đối tượng trong một tập đồng nhất gồm những người không rám nắng.
- Thử nghiệm theo màu tóc là tốt nhất. Trong tập đồng nhất rám nắng có một người là Thu, và tập

đồng nhất không râm nắng có ba người là Đông, Mơ và Xuân.



Hình 7.2. Bốn cách phân chia cơ sở dữ liệu theo bốn thuộc tính khác nhau

Theo các thử nghiệm này người ta sử dụng trước tiên thử nghiệm về màu tóc. Thử nghiệm này có một tập không đồng nhất ứng với màu tóc, lẫn lộn người râm nắng và không râm nắng. Bốn người Hoa, Lan, Hạ và Đào được chia nhỏ ra.



Hình 7.3. Ba cách phân chia tiếp theo đối với bốn người thuộc tập không đồng nhất

Sau lần chia này người ta nhận thấy trong ba cách chia, cách chia theo việc dùng thuốc cho phép tách bốn đối tượng thành hai tập đồng nhất.

♦ 7.3.3. Độ lộn xộn của tập hợp

Đối với cơ sở dữ liệu thực, không phải bất kỳ thử nghiệm nào cũng có thể cho ra tập đồng nhất. Với cơ sở dữ liệu này người ta cần đo mức độ lộn xộn của dữ liệu, hay độ không đồng nhất trong các tập con được sinh ra. Công thức đo lý thuyết thông tin về độ lộn xộn trung bình:

$$\sum_i \frac{n_{bi}}{n} \sum_c -\frac{n_{bc}}{n_b} \log_2 \frac{n_{bc}}{n_b}$$

Trong đó nb là số mẫu trong nhánh b , nt là tổng số các mẫu, và nbc là số mẫu trong nhánh b của lớp c .

Thực chất người ta quan tâm đến số các mẫu tại cuối nhánh. Yêu cầu nb và nbc là cao khi thử nghiệm sinh ra các tập không đồng nhất, và là thấp khi thử nghiệm sinh ra các tập hoàn toàn thống nhất.

Độ lộn xộn tính bằng $\sum_c -\frac{nbc}{nb} \log_2 \frac{nbc}{nb}$

Dù công thức chưa cho thấy "sự lộn xộn", nhưng người ta dùng nó để đo thông tin. Để thấy được các khía cạnh quan tâm, giả sử có một tập gồm các phần tử của hai lớp A và B. Nếu số phần tử của hai lớp là cân bằng, thì độ lộn xộn là 1 và giá trị cực đại về độ lộn xộn được tính theo:

$$-1/2 \log_2 1/2 - 1/2 \log_2 1/2 = 1/2 + 1/2 = 1$$

Mặt khác nếu phần tử thuộc chỉ một trong A, B, độ lộn xộn là 0.

$$-1 \log_2 1 - 0 \log_2 0 = 0$$

Độ lộn xộn bằng 0 khi tập là hoàn toàn thống nhất, và bằng 1 khi tập hoàn toàn không đồng nhất. Độ đo lộn xộn có giá trị từ 0 đến 1.

Bằng công cụ này, người ta có thể tính được độ lộn xộn trung bình của các tập tại cuối các nhánh sau lần thử nghiệm.

Độ lộn xộn trung bình = $\sum_b \frac{nb}{n} \cdot$ độ lộn xộn trong tập nhánh b .

Trong thí dụ trước, thử nghiệm về màu tóc đã chia cơ sở dữ liệu thành ba phần. Tập tóc đen có hai người rám nắng và hai người không rám nắng. Tập tóc bạc chỉ có một người rám nắng. Trong tập tóc rậm, cả ba người không hề rám. Độ lộn xộn trung bình được tính cho kết quả 0.5.

$$4/8(-2/4 \log_2 2/4 - 2/4 \log_2 2/4) + 1/8 \cdot 0 + 3/8 \cdot 0 = 0.5$$

Thực hiện tính toán tương tự:

Thử nghiệm theo ...	Độ lộn xộn
---------------------	------------

Màu tóc	0.50
Chiều cao	0.69 0.94
Cân nặng	0.61
Dùng thuốc	

Bảng 7.2. Lựa chọn cách phân chia cơ sở dữ liệu theo độ đo về sự lộn xộn.

Do thử nghiệm theo màu tóc gây ra ít lộn xộn nhất, người ta dùng nó làm thử nghiệm đầu tiên. Tương tự, sau khi làm thử nghiệm này người ta làm thử nghiệm về việc dùng thuốc, do tính toán:

Thử nghiệm theo ...	Độ lộn xộn
Chiều cao	0.5 1.0
Cân	

nặng	0.0
Dùng thuốc	

Bảng 7.3. Lựa chọn tiếp theo.

Vậy để tạo ra cây định danh, người ta dùng thủ tục SINH được trình bày như sau:

✚Thủ tục SINH dùng cây định danh:

Cho đến khi mỗi nút lá được ghi tên các phần tử của tập mẫu đồng nhất, thực hiện:

- 1. Chọn nút lá ứng với tập mẫu không đồng nhất.*
- 2. Thay thế nút này bằng nút thử nghiệm cho phép chia tập không đồng nhất thành các tập không đồng nhất, dựa theo tính toán độ lộn xộn.*

♦7.3.4. Chuyển cây sang luật

Một khi đã dựng được cây định danh, nếu muốn chuyển các tri thức sang dạng luật thì cũng đơn giản. Người ta đi theo các nhánh của cây, từ gốc

đến các nút lá, lấy các thử nghiệm làm giả thiết và phân loại nút lá làm kết luận.

Thí dụ:

Các tri thức trong thí dụ về râm nắng khi nghỉ mát ở phần trên được viết thành luật:

.IF Tóc đen

Người đó dùng thuốc

THEN Không sao cả

.IF Người tóc đen

Không dùng thuốc

THEN Họ bị râm nắng

.IF Người tóc bạc

THEN Bị râm nắng

.IF Người tóc râm

THEN Không sao cả

7.3.4.1. Lược bỏ giả thiết không cần thiết trong luật

Sau khi thu được các luật chuyển từ cây định danh, có thể bỏ đi các luật không cần thiết để đơn giản tập các luật. Người ta kiểm tra giả thiết nào có thể bỏ đi mà không thay đổi tác dụng của luật đối với mẫu.

Thí dụ:

Xét luật đầu tiên trong các luật trên:

IF Tóc đen

Người đó dùng thuốc

THEN Không sao cả

Giả thiết có hai phần. Nếu bỏ đi phần đầu, còn điều kiện về "dùng thuốc". Theo các mẫu, người dùng thuốc chỉ có Lan, Xuân và Đào. Không ai trái với phần kết luận cả, tức không ai bị râm nắng. Do vậy người ta bỏ phần giả thiết đầu, thu được:

IF Người đó dùng thuốc

THEN Không sao cả

Để suy lý dễ dàng người ta thường đưa ra **bảng ngẫu nhiên**. Sở dĩ gọi vậy vì kết quả tùy thuộc vào thuộc tính.

Loại người	Không sao	Bị râm
Người có tóc đen	2 1	0 0
Người tóc không đen		

Bảng 7.4. Bảng ngẫu nhiên theo loại tóc đối với những người dùng thuốc.

Trong bảng người ta thấy số người dùng thuốc có tóc đen, không đen và số người bị râm nắng, không râm. Bảng cho thấy tri thức về màu của tóc không quyết định vì đến việc họ bị râm nắng.

Quay lại luật trên nếu bỏ đi phần giả thiết thứ hai "dùng thuốc", người ta thấy trong số bốn người tóc đen là Hoa, Lan, Hạ và Đào, có hai người dùng thuốc mà vẫn bị râm nắng. Còn bảng ngẫu nhiên cho biết:

Dùng thuốc ?	Không sao	Bị râm
Có dùng	2	0
Không dùng	0	2

Bảng 7.5. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Như vậy việc dùng thuốc có tác dụng đối với những người tóc đen. Các mẫu người tóc đen không râm nắng khi và chỉ khi họ dùng thuốc. Bởi vậy ý định bỏ giả thiết này không thực hiện.

Thí dụ:

Luật thứ hai trong tập các luật:

IF Người tóc đen

Không dùng thuốc

THEN Họ bị râm nắng

Tương tự như thí dụ trên, người ta dự tính bỏ đi giả thiết đầu trong hai giả thiết. Bảng ngẫu nhiên cho thấy:

Loại người	Không sao	Bị râm
Người có tóc đen	0 2	2 1
Người tóc không đen		

Bảng 7.6. Bảng ngẫu nhiên theo loại tóc đối với những người không dùng thuốc.

Như vậy giả thiết này là quan trọng. Thiếu nó người ta không thể đảm bảo việc khớp để kết

luận rằng không bị râm nắng. Nếu xét tiếp giả thiết còn lại, trong số bốn người tóc đen có hai bị râm và hai không. Bảng ngẫu nhiên cho thấy:

Dùng thuốc ?	Không sao	Bị râm
Có dùng	2	0
Không dùng	0	2

Bảng 7.7. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Nội dung bảng cho thấy không thể bỏ đi giả thiết này. Luật này không cần đơn giản hơn.

Thí dụ:

Xét hai luật còn lại. Chúng có một giả thiết. Việc bỏ giả thiết đi là không được. Các bảng ngẫu nhiên cũng cho các tri thức như vậy.

7.3.4.2. Lược bỏ luật thừa

Phần trên đã đơn giản hóa các luật riêng rẽ. Nhìn tổng thể, chúng cần được tính giản nữa. Các luật không cần thiết cần được bỏ đi. Quả thật có luật trong số bốn luật thu được sẽ bị bỏ đi.

Thí dụ:

Bốn luật thu gồm có:

• IF Người tóc đen

không dùng thuốc

THEN Họ bị râm nắng

• IF Người đỏ dùng thuốc

THEN Không sao cả

• IF Người tóc bạc

THEN Bị râm nắng

• IF Người tóc râm

THEN Không sao cả

Hai luật có kết luận "râm nắng" và hai luật khẳng định "không sao cả". Người ta có thể

thay thế hai luật khẳng định "rám nắng" bằng một luật. Gọi là **luật mặc định**. Luật mặc định là luật được dùng chỉ khi không có luật nào. Do có hai kết luận, có hai khả năng của luật mặc định:

• IF Không có luật nào

THEN Người đó bị rám nắng

• IF Không có luật nào

THEN Không sao cả

Chắc chắn chỉ dùng hai luật này là không thể được! Cả hai luật đều ứng với hai luật khác. Cần chọn luật mặc định là luật quét được kết luận chung trong tập mẫu, tức "không sao cả".
Thí dụ này chọn:

• IF Người tóc đen

THEN Không dùng thuốc

THEN Họ bị rám nắng

• IF Người tóc bạc

THEN Bị rám nắng

IF Không có luật nào

THEN Không sao cả

Một cách khác chọn luật mặc định không dựa vào số mẫu thu được, mà dựa vào số các giả thiết trong các luật, người ta có tập các luật sau:

IF Người đó dùng thuốc

THEN Không sao cả

IF Người tóc râu

THEN Không sao cả

IF Không có luật nào

THEN Người đó bị râm nắng

Tóm lại, để chuyển cây định danh về tập các luật, thực hiện thủ tục tên là CAT sau:

➤ **Dùng thủ tục CAT cho phép tạo nên các luật từ cây định danh:**

Tạo một luật từ mỗi nhánh gốc - lá của cây định danh.

Đơn giản hóa mỗi luật bằng cách khử các giả thiết không có tác dụng đối với kết luận của luật.

Thay thế các luật có chung kết luận bằng luật mặc định. Luật này được kích hoạt khi không có luật nào hoạt động. Khi có nhiều khả năng, dùng phép may rủi để chọn luật mặc định.

7.3. CÂY ĐỊNH DANH

Cây định danh là một công cụ khá phổ biến trong nhiều dạng ứng dụng, với cơ chế rút trích các luật nhân quả xác định các mẫu dữ liệu.

♦ 7.3.1. Thí dụ về thế giới thực thu gọn

Tưởng tượng có các dữ liệu về bài toán đánh giá độ rám nắng tại một nơi nghỉ mát. Có người vui vì ngăm đen, nhưng cũng có người rát vì rộp da. Dữ liệu quan sát trên tám người được ghi lại theo bảng sau đây.

TT	Tên người	Màu tóc	Chiều cao	Cân nặng	Dùng thuốc?	Kết quả
1	Hoa	Đen	Tầm	Nhẹ	Không	Bị rám

			thước			
2	Lan	Đen	Cao	Vừa phải	Có	Không
3	Xuân	Râm	Thấp	Vừa phải	Có	Không
4	Hạ	Đen	Thấp	Vừa phải	Không	Bị râm
5	Thu	Bạc	Tầm thước	Nặng	Không	Bị râm
6	Đông	Râm	Cao	Nặng	Không	Không
7	Mơ	Râm	Tầm thước	Nặng	Không	Không
8	Đào	Đen	Thấp	Nhẹ	Có	Không

Bảng 7.1. Số liệu quan sát về hiện tượng râm nắng.

Nếu có ba dữ liệu khác nhau về màu tóc, cân nặng, chiều cao và người ta có thể dùng thuốc hoặc không thì sẽ có $3 \times 3 \times 3 \times 2 = 54$ tổ hợp. Một người mới được chọn thì xác suất khớp được với mẫu là

$8/54 = 0.15$. Xác suất này cao hơn thực tế do còn nhiều thuộc tính và nhiều giá trị khác mà thuộc tính có thể nhận.

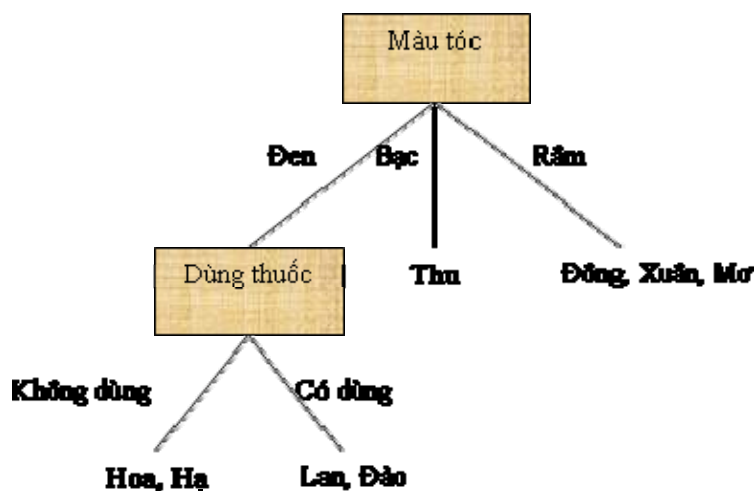
Do vậy nếu so sánh các thuộc tính của đối tượng chưa biết với thuộc tính của các đối tượng thống kê thì không chính xác. Một số nhận xét về việc giải bài toán này:

- Người ta có thể xử lý dữ liệu như không gian các đặc tính, nhưng nếu không rõ thuộc tính nào là quan trọng hơn thì cũng khó tìm thấy đối tượng khớp nhất.

- Có thể dùng không gian các thể hệ để cô lập các thuộc tính liên quan và thuộc tính không liên quan. Tuy nhiên thường không tìm thấy lý do thuyết phục để tin được mô hình phân loại có thể được diễn tả như tổ hợp các giá trị của tập các thuộc tính. Mặt khác cũng có thể các mẫu bị nhiễu, không thực chính xác như thế giới thực.

- Người ta dùng thủ tục phân loại chính xác các mẫu. Khi thủ tục đã học mẫu với số lượng mẫu

"khá đủ", thủ tục có thể nhận ra đối tượng chưa biết.



Hình 7.1. Cây định danh (*Người có tên ghi đậm là người bị râm nắng*).

Một cách phù hợp cho phép thực hiện các thủ tục thử nghiệm các thuộc tính là sắp xếp các thử nghiệm trên **cây định danh**. Do cây định danh thuộc loại cây quyết định, đặc tả của nó như đặc tả cây quyết định.

➤ **Định nghĩa 7.1 : Cây định danh (Identification tree)**

Cây định danh có thể hiện như cây quyết định, trong đó mỗi tập các kết luận được thiết lập ngầm định bởi một danh sách đã biết.

Cây định danh dùng để xác định người bị râm nắng với kiểm tra đầu tiên là màu tóc. Nếu thấy tóc đen người ta kiểm tra có dùng thuốc không? Ngược lại, nếu tóc bạc hay râu, người ta không cần kiểm tra. Nói chung việc chọn tiến hành loại kiểm tra nào là phụ thuộc vào kết quả của lần kiểm tra trước. Xem ví dụ thể hiện trong hình 7.1.

Mỗi đối tượng đưa vào định danh đi xuống theo một nhánh cây, tùy theo các giá trị thuộc tính của nó. Cây định danh như hình vẽ có thể phân loại người trong cơ sở dữ liệu "râm nắng" vì mỗi người ứng với một nút lá trên cây định danh. Nút này dùng cho một hay nhiều người. Trong hình cho thấy người bị râm nắng được đánh dấu bằng tên in đậm hơn.

Người ta có thể đưa ra một cây, có các nút lá ứng với mỗi người, không liên quan gì đến thuộc tính râm nắng. Liệu nó được dùng phân loại không?

So sánh cây định danh và cây tổng quát, người ta thấy cây thứ nhất là cây định danh, có vẻ liên quan đến sự râm nắng nhiều hơn cây thứ hai. Cây định

danh tỏ ra đã biết rằng màu tóc và phần lộ ra ánh nắng liên quan trực tiếp đến tính rám nắng.

Làm sao có thể chương trình hóa để đến được cùng một kết luận mà không cần biết trước màu tóc và việc dùng thuốc liên quan đến đặc tính của da? Một trong các giải đáp là phát biểu của Occam.

• **Phát biểu Occam dùng cho các cây định danh:**

Thế giới vốn đơn giản. Do vậy cây định danh gồm các mẫu là cái thích hợp nhất để định danh các đối tượng chưa biết một cách chính xác.

Theo phát biểu này, cây định danh đầu tiên nhỏ hơn cây sau nên nó phù hợp hơn.

♦ **7.3.2. Phân loại đối tượng theo các thuộc tính**

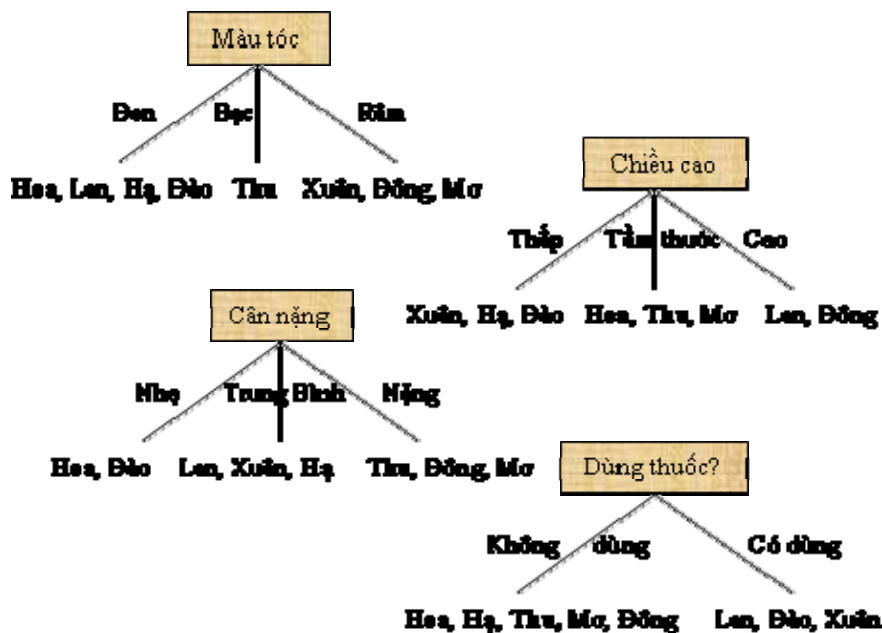
Nếu như ta tìm kiếm cây định danh nhỏ nhất khi cần có rất nhiều thử nghiệm thì thực là không thực tế. Chính vì vậy mà cũng nên dừng lại ở thử tục xây dựng những cây định danh nhỏ, dù rằng nó không phải là nhỏ nhất. Người ta chọn thử nghiệm cho phép chia cơ sở dữ liệu các mẫu thành các tập con.

Trong đó nhiều mẫu cùng chung một loại. Đối với mỗi tập có nhiều loại mẫu, dùng thử nghiệm khác để chia các đối tượng không đồng nhất thành các tập chỉ gồm đối tượng đồng nhất.

Xét ví dụ thể hiện ở hình 7.2. Cơ sở dữ liệu "rám nắng" có thể được chia nhỏ theo bốn thử nghiệm ứng với bốn thuộc tính:

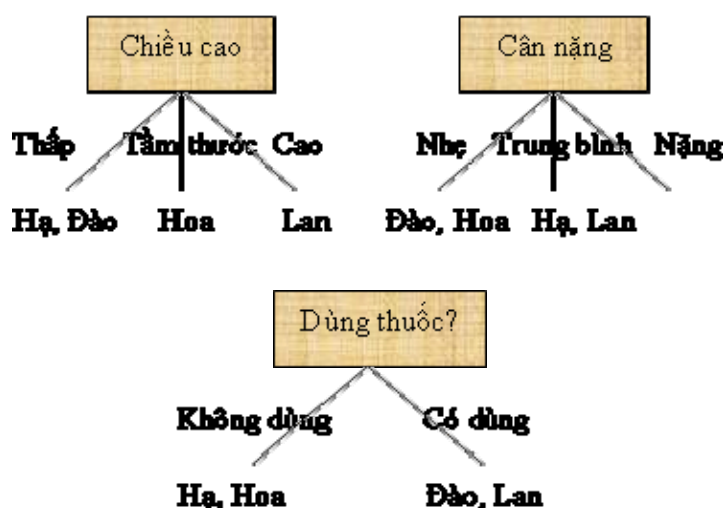
- Thử nghiệm theo cân nặng là tồi nhất nếu người ta đánh giá thử nghiệm theo các tập đồng nhất, có cùng tính chất rám nắng. Sau khi dùng thử nghiệm này, những mẫu rám nắng nằm đều ở các tập.
- Thử nghiệm theo chiều cao có vẻ tốt hơn. Có hai người trong một tập đồng nhất. Hai tập kia có lần cả người rám và không rám nắng.
- Thử nghiệm về việc dùng thuốc thu được ba đối tượng trong một tập đồng nhất gồm những người không rám nắng.
- Thử nghiệm theo màu tóc là tốt nhất. Trong tập đồng nhất rám nắng có một người là Thu, và tập

đồng nhất không râm nắng có ba người là Đông, Mơ và Xuân.



Hình 7.2. Bốn cách phân chia cơ sở dữ liệu theo bốn thuộc tính khác nhau

Theo các thử nghiệm này người ta sử dụng trước tiên thử nghiệm về màu tóc. Thử nghiệm này có một tập không đồng nhất ứng với màu tóc, lẫn lộn người râm nắng và không râm nắng. Bốn người Hoa, Lan, Hạ và Đào được chia nhỏ ra.



Hình 7.3. Ba cách phân chia tiếp theo đối với bốn người thuộc tập không đồng nhất

Sau lần chia này người ta nhận thấy trong ba cách chia, cách chia theo việc dùng thuốc cho phép tách bốn đối tượng thành hai tập đồng nhất.

♦ 7.3.3. Độ lộn xộn của tập hợp

Đối với cơ sở dữ liệu thực, không phải bất kỳ thử nghiệm nào cũng có thể cho ra tập đồng nhất. Với cơ sở dữ liệu này người ta cần đo mức độ lộn xộn của dữ liệu, hay độ không đồng nhất trong các tập con được sinh ra. Công thức đo lý thuyết thông tin về độ lộn xộn trung bình:

$$\sum_i \frac{n_{bi}}{n} \sum_c - \frac{n_{bc}}{n_b} \log_2 \frac{n_{bc}}{n_b}$$

Trong đó nb là số mẫu trong nhánh b , nt là tổng số các mẫu, và nbc là số mẫu trong nhánh b của lớp c .

Thực chất người ta quan tâm đến số các mẫu tại cuối nhánh. Yêu cầu nb và nbc là cao khi thử nghiệm sinh ra các tập không đồng nhất, và là thấp khi thử nghiệm sinh ra các tập hoàn toàn thống nhất.

Độ lộn xộn tính bằng $\sum_c -\frac{nbc}{nb} \log_2 \frac{nbc}{nb}$

Dù công thức chưa cho thấy "sự lộn xộn", nhưng người ta dùng nó để đo thông tin. Để thấy được các khía cạnh quan tâm, giả sử có một tập gồm các phần tử của hai lớp A và B. Nếu số phần tử của hai lớp là cân bằng, thì độ lộn xộn là 1 và giá trị cực đại về độ lộn xộn được tính theo:

$$-1/2 \log_2 1/2 - 1/2 \log_2 1/2 = 1/2 + 1/2 = 1$$

Mặt khác nếu phần tử thuộc chỉ một trong A, B, độ lộn xộn là 0.

$$-1 \log_2 1 - 0 \log_2 0 = 0$$

Độ lộn xộn bằng 0 khi tập là hoàn toàn thống nhất, và bằng 1 khi tập hoàn toàn không đồng nhất. Độ đo lộn xộn có giá trị từ 0 đến 1.

Bằng công cụ này, người ta có thể tính được độ lộn xộn trung bình của các tập tại cuối các nhánh sau lần thử nghiệm.

Độ lộn xộn trung bình = $\sum_b \frac{nb}{n} \cdot$ độ lộn xộn trong tập nhánh b .

Trong thí dụ trước, thử nghiệm về màu tóc đã chia cơ sở dữ liệu thành ba phần. Tập tóc đen có hai người rám nắng và hai người không rám nắng. Tập tóc bạc chỉ có một người rám nắng. Trong tập tóc nâu, cả ba người không hề rám. Độ lộn xộn trung bình được tính cho kết quả 0.5.

$$4/8(-2/4 \log_2 2/4 - 2/4 \log_2 2/4) + 1/8 \cdot 0 + 3/8 \cdot 0 = 0.5$$

Thực hiện tính toán tương tự:

Thử nghiệm theo ...	Độ lộn xộn
---------------------	------------

Màu tóc	0.50
Chiều cao	0.69 0.94
Cân nặng	0.61
Dùng thuốc	

Bảng 7.2. Lựa chọn cách phân chia cơ sở dữ liệu theo độ đo về sự lộn xộn.

Do thử nghiệm theo màu tóc gây ra ít lộn xộn nhất, người ta dùng nó làm thử nghiệm đầu tiên. Tương tự, sau khi làm thử nghiệm này người ta làm thử nghiệm về việc dùng thuốc, do tính toán:

Thử nghiệm theo ...	Độ lộn xộn
Chiều cao	0.5 1.0
Cân	

nặng	0.0
Dùng thuốc	

Bảng 7.3. Lựa chọn tiếp theo.

Vậy để tạo ra cây định danh, người ta dùng thủ tục SINH được trình bày như sau:

✚Thủ tục SINH dùng cây định danh:

Cho đến khi mỗi nút lá được ghi tên các phần tử của tập mẫu đồng nhất, thực hiện:

- 1. Chọn nút lá ứng với tập mẫu không đồng nhất.*
- 2. Thay thế nút này bằng nút thử nghiệm cho phép chia tập không đồng nhất thành các tập không đồng nhất, dựa theo tính toán độ lộn xộn.*

♦7.3.4. Chuyển cây sang luật

Một khi đã dựng được cây định danh, nếu muốn chuyển các tri thức sang dạng luật thì cũng đơn giản. Người ta đi theo các nhánh của cây, từ gốc

đến các nút lá, lấy các thử nghiệm làm giả thiết và phân loại nút lá làm kết luận.

Thí dụ:

Các tri thức trong thí dụ về râm nắng khi nghỉ mát ở phần trên được viết thành luật:

.IF Tóc đen

Người đó dùng thuốc

THEN Không sao cả

.IF Người tóc đen

Không dùng thuốc

THEN Họ bị râm nắng

.IF Người tóc bạc

THEN Bị râm nắng

.IF Người tóc râm

THEN Không sao cả

7.3.4.1. Lược bỏ giả thiết không cần thiết trong luật

Sau khi thu được các luật chuyển từ cây định danh, có thể bỏ đi các luật không cần thiết để đơn giản tập các luật. Người ta kiểm tra giả thiết nào có thể bỏ đi mà không thay đổi tác dụng của luật đối với mẫu.

Thí dụ:

Xét luật đầu tiên trong các luật trên:

IF Tóc đen

Người đó dùng thuốc

THEN Không sao cả

Giả thiết có hai phần. Nếu bỏ đi phần đầu, còn điều kiện về "dùng thuốc". Theo các mẫu, người dùng thuốc chỉ có Lan, Xuân và Đào. Không ai trái với phần kết luận cả, tức không ai bị râm nắng. Do vậy người ta bỏ phần giả thiết đầu, thu được:

IF Người đó dùng thuốc

THEN Không sao cả

Để suy lý dễ dàng người ta thường đưa ra **bảng ngẫu nhiên**. Sở dĩ gọi vậy vì kết quả tùy thuộc vào thuộc tính.

Loại người	Không sao	Bị râm
Người có tóc đen	2 1	0 0
Người tóc không đen		

Bảng 7.4. Bảng ngẫu nhiên theo loại tóc đối với những người dùng thuốc.

Trong bảng người ta thấy số người dùng thuốc có tóc đen, không đen và số người bị râm nắng, không râm. Bảng cho thấy tri thức về màu của tóc không quyết định vì đến việc họ bị râm nắng.

Quay lại luật trên nếu bỏ đi phần giả thiết thứ hai "dùng thuốc", người ta thấy trong số bốn người tóc đen là Hoa, Lan, Hạ và Đào, có hai người dùng thuốc mà vẫn bị râm nắng. Còn bảng ngẫu nhiên cho biết:

Dùng thuốc ?	Không sao	Bị râm
Có dùng	2	0
Không dùng	0	2

Bảng 7.5. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Như vậy việc dùng thuốc có tác dụng đối với những người tóc đen. Các mẫu người tóc đen không râm nắng khi và chỉ khi họ dùng thuốc. Bởi vậy ý định bỏ giả thiết này không thực hiện.

Thí dụ:

Luật thứ hai trong tập các luật:

IF Người tóc đen

Không dùng thuốc

THEN Họ bị râm nắng

Tương tự như thí dụ trên, người ta dự tính bỏ đi giả thiết đầu trong hai giả thiết. Bảng ngẫu nhiên cho thấy:

Loại người	Không sao	Bị râm
Người có tóc đen	0 2	2 1
Người tóc không đen		

Bảng 7.6. Bảng ngẫu nhiên theo loại tóc đối với những người không dùng thuốc.

Như vậy giả thiết này là quan trọng. Thiếu nó người ta không thể đảm bảo việc khớp để kết

luận rằng không bị râm nắng. Nếu xét tiếp giả thiết còn lại, trong số bốn người tóc đen có hai bị râm và hai không. Bảng ngẫu nhiên cho thấy:

Dùng thuốc ?	Không sao	Bị râm
Có dùng	2	0
Không dùng	0	2

Bảng 7.7. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Nội dung bảng cho thấy không thể bỏ đi giả thiết này. Luật này không cần đơn giản hơn.

Thí dụ:

Xét hai luật còn lại. Chúng có một giả thiết. Việc bỏ giả thiết đi là không được. Các bảng ngẫu nhiên cũng cho các tri thức như vậy.

7.3.4.2. Lược bỏ luật thừa

Phần trên đã đơn giản hóa các luật riêng rẽ. Nhìn tổng thể, chúng cần được tính giản nữa. Các luật không cần thiết cần được bỏ đi. Quả thật có luật trong số bốn luật thu được sẽ bị bỏ đi.

Thí dụ:

Bốn luật thu gồm có:

• IF Người tóc đen

không dùng thuốc

THEN Họ bị râm nắng

• IF Người đỏ dùng thuốc

THEN Không sao cả

• IF Người tóc bạc

THEN Bị râm nắng

• IF Người tóc râu

THEN Không sao cả

Hai luật có kết luận "râm nắng" và hai luật khẳng định "không sao cả". Người ta có thể

thay thế hai luật khẳng định "rám nắng" bằng một luật. Gọi là **luật mặc định**. Luật mặc định là luật được dùng chỉ khi không có luật nào. Do có hai kết luận, có hai khả năng của luật mặc định:

- IF Không có luật nào

THEN Người đó bị rám nắng

- IF Không có luật nào

THEN Không sao cả

Chắc chắn chỉ dùng hai luật này là không thể được! Cả hai luật đều ứng với hai luật khác. Cần chọn luật mặc định là luật quét được kết luận chung trong tập mẫu, tức "không sao cả".
Thí dụ này chọn:

- IF Người tóc đen

Không dùng thuốc

THEN Họ bị rám nắng

- IF Người tóc bạc

THEN Bị rám nắng

IF Không có luật nào

THEN Không sao cả

Một cách khác chọn luật mặc định không dựa vào số mẫu thu được, mà dựa vào số các giả thiết trong các luật, người ta có tập các luật sau:

IF Người đó dùng thuốc

THEN Không sao cả

IF Người tóc râu

THEN Không sao cả

IF Không có luật nào

THEN Người đó bị râm nắng

Tóm lại, để chuyển cây định danh về tập các luật, thực hiện thủ tục tên là CAT sau:

➤ **Dùng thủ tục CAT cho phép tạo nên các luật từ cây định danh:**

Tạo một luật từ mỗi nhánh gốc - lá của cây định danh.

Đơn giản hóa mỗi luật bằng cách khử các giả thiết không có tác dụng đối với kết luận của luật.

Thay thế các luật có chung kết luận bằng luật mặc định. Luật này được kích hoạt khi không có luật nào hoạt động. Khi có nhiều khả năng, dùng phép may rủi để chọn luật mặc định.

7.4. THUẬT GIẢI ILA

Thuật giải ILA (Inductive Learning Algorithm) được dùng để xác định các luật phân loại cho tập hợp các mẫu học. Thuật giải này thực hiện theo cơ chế lặp, để tìm luật riêng đại diện cho tập mẫu của từng lớp. Sau khi xác định được luật, ILA loại bỏ các mẫu liên quan khỏi tập mẫu, đồng thời thêm luật mới này vào tập luật. Kết quả có được là một danh sách có thứ tự các luật chứ không là một cây quyết định. Các ưu điểm của thuật giải này có thể được trình bày như sau:

Dạng các luật sẽ phù hợp cho việc khảo sát dữ liệu, mô tả mỗi lớp một cách đơn giản để dễ phân biệt với các lớp khác.

Tập luật được sắp thứ tự, riêng biệt – cho phép quan tâm đến một luật tại thời điểm bất kỳ. Khác với việc xử lý luật theo phương pháp cây quyết định, vốn rất phức tạp trong trường hợp các nút cây trở nên khá lớn.

♦ 7.4.1. Xác định dữ liệu

1. Tập mẫu được liệt kê trong một bảng, với mỗi dòng tương ứng một mẫu, và mỗi cột thể hiện một thuộc tính trong mẫu.
2. Tập mẫu có m mẫu, mỗi mẫu gồm k thuộc tính, trong đó có một thuộc tính quyết định. Tổng số n các giá trị của thuộc tính này chính là số lớp của tập mẫu.
3. Tập luật R có giá trị khởi tạo là ϕ .
4. Tất cả các cột trong bảng ban đầu chưa được đánh dấu (kiểm tra).

♦ 7.4.2. Thuật giải ILA

Bước 1: Chia bảng m mẫu ban đầu thành n bảng con. Mỗi bảng con ứng với một giá trị của thuộc tính phân lớp của tập tập mẫu.

(* thực hiện các bước 2 đến 8 cho mỗi bảng con*)

Bước 2: Khởi tạo bộ đếm kết hợp thuộc tính j , $j=1$.

Bước 3: Với mỗi bảng con đang khảo sát, phân chia danh sách các thuộc tính theo các tổ hợp phân biệt, mỗi tổ hợp ứng với j thuộc tính phân biệt.

Bước 4: Với mỗi tổ hợp các thuộc tính, tính số lượng các giá trị thuộc tính xuất hiện theo cùng tổ hợp thuộc tính trong các dòng chưa được đánh dấu của bảng con đang xét (mà đồng thời không xuất hiện với tổ hợp thuộc tính này trên các bảng còn lại). Gọi tổ hợp đầu tiên (trong bảng con) có số lần xuất hiện nhiều nhất là *tổ hợp lớn nhất*.

Bước 5: Nếu tổ hợp lớn nhất bằng ϕ , tăng j lên 1 và quay lại bước 3.

Bước 6: Đánh dấu các dòng thoả tổ hợp lớn nhất của bảng con đang xử lý theo lớp.

Bước 7: Thêm luật mới vào tập luật R , với vế trái là tập các thuộc tính của tổ hợp lớn nhất (kết hợp các thuộc tính bằng toán tử AND) và vế phải là giá trị thuộc tính quyết định tương ứng.

Bước 8: Nếu tất cả các dòng đều đã được đánh dấu phân lớp, tiếp tục thực hiện từ bước 2 cho các bảng con còn lại. Ngược lại (nếu chưa đánh dấu hết các dòng) thì quay lại bước 4. Nếu tất cả các bảng con đã được xét thì kết thúc, kết quả thu được là tập luật cần tìm.

◆ 7.4.3. Mô tả thuật giải ILA

ILA là một thuật giải khá đơn giản rút trích các luật dẫn từ một tập mẫu. Mỗi mẫu được mô tả dưới dạng một tập xác định các thuộc tính, mỗi thuộc tính ứng với một vài giá trị nào đó.

Để minh họa thuật giải ILA, chúng ta sử dụng tập mẫu cho trong bảng 7.8, gồm có 7 mẫu ($m=7$), 3 thuộc tính ($k=3$), và thuộc tính quyết định (phân lớp) có hai giá trị là {yes, no} ($n=2$). Trong ví dụ này, "Size", "Color" và "Shape" là các thuộc tính

với các nhóm giá trị {small, medium, large}, {red, blue, green}, và {brick, wedge, sphere, pillar}.

Mẫu số	Size	Color	Shape	Decision
1	medium	blue	brick	yes
2	small	red	wedge	no
3	small	red	sphere	yes
4	large	red	wedge	no
5	large	green	pillar	yes
6	large	red	pillar	no
7	large	green	sphere	yes

Bảng 7.8. Tập mẫu học cho bài toán phân lớp đối tượng

Do $n=2$, bước đầu tiên ta chia tập mẫu thành hai bảng con như trong bảng 7.9.

<i>Bảng con 1</i>				
Mẫu	Size	Color	Shape	Decision

số cũ, mới				
1 1	medium	blue	brick	yes
3 2	small	red	sphere	yes
5 3	large	green	pillar	yes
7 4	large	green	sphere	yes

Bảng con 2

Mẫu số cũ mới	Size	Color	Shape	Decision
2 1	small	red	wedge	no
4 2	large	red	wedge	no

6	large	red	pillar	no
3				

Bảng 7.9. Chia thành hai bảng con theo thuộc tính Decision

Áp dụng bước 2 của thuật giải vào bảng con thứ nhất trong bảng 7.9. Với $j=1$, danh sách các tổ hợp thuộc tính gồm có $\{Size\}$, $\{Color\}$, và $\{Shape\}$.

Với tổ hợp $\{Size\}$, giá trị thuộc tính "medium" xuất hiện trong bảng con thứ nhất nhưng không có trong bảng con thứ hai, do đó giá trị tổ hợp lớn nhất là "medium". Bởi vì các giá trị thuộc tính "small" và "large" xuất hiện trong cả hai bảng con, nên không được xét trong bước này. Với tổ hợp $\{Size\}$, giá trị thuộc tính "medium" chỉ bằng 1, ta xét tiếp cho tổ hợp $\{Color\}$ thì giá trị tổ hợp lớn nhất là bằng 2, ứng với thuộc tính "green", còn thuộc tính "blue" là bằng 1. Tương tự như vậy, với tổ hợp $\{Shape\}$, ta có "brick" xuất hiện một lần, và "sphere" hai lần. Đến cuối bước 4, ta có tổ hợp $\{Color\}$ với thuộc tính "green" và $\{Shape\}$ với thuộc tính "sphere" đều có số lần xuất hiện lớn nhất là 2. Thuật toán mặc định chọn trường hợp thứ nhất để xác định luật

tổ hợp lớn nhất. Dòng 3 và 4 được đánh dấu đã phân lớp, ta có luật dẫn như sau:

➤ Rule 1: **IF color là green THEN decision là yes**

Ta tiếp tục thực hiện từ bước 4 đến 8 cho các mẫu còn lại (chưa đánh dấu) trong bảng con này (tức dòng 1 và 2). Áp dụng tương tự như trên, ta thấy giá trị thuộc tính "medium" của {Size}, "blue" của "Color", "brick" và "sphere" của {Shape} đều xuất hiện một lần. Bởi vì số lần xuất hiện này giống nhau, thuật giải áp dụng luật mặc định chọn trường hợp đầu tiên. Ta có thêm luật dẫn sau:

➤ Rule 2: **IF size là medium THEN decision là yes**

Đánh dấu cho dòng 1 trong bảng con thứ nhất. Tiếp tục áp dụng bước 4 đến 8 trên dòng còn lại (tức dòng 2). Giá trị thuộc tính "sphere" của {Shape} xuất hiện một lần, ta có luật dẫn thứ ba:

➡ **Rule 3: IF shape là sphere THEN decision là yes**

Dòng 2 được đánh dấu. Như vậy, tất cả các dòng trong bảng con 1 đã được đánh dấu, ta chuyển qua xử lý tiếp bảng con 2. Thuộc tính "wedge" của {Shape} xuất hiện hai lần trong dòng 1 và 2 của bảng con này. Đánh dấu các dòng này với luật dẫn thứ tư như sau:

➡ **Rule 4: IF shape là wedge THEN decision là no**

Với dòng còn lại (tức dòng 3) của bảng con 2, ta có thuộc tính {Size} với giá trị "large" có xuất hiện trong bảng con 1. Do đó, theo thuật giải, ta loại bỏ trường hợp này. Tương tự như vậy cho giá trị "red" của {Color} và "pillar" của {Shape}. Khi đó, ILA tăng j lên 1, và khởi tạo các tổ hợp 2 thuộc tính là {Size và Color}, {Size và Shape}, và {Color và Shape}. Các tổ hợp thứ nhất và thứ ba thoả mãn điều kiện không xuất hiện trong bảng con 1 với các cặp thuộc tính hiện có của dòng này. Theo luật mặc

định, ta chọn luật theo trường hợp thứ nhất. Đánh dấu dòng này, ta có thêm luật dẫn thứ 5:

➡ **Rule 5: IF size là large AND color là red THEN decision là no**

Bởi vì lúc này tất cả các dòng trong bảng con hai cũng đều đã được đánh dấu phân lớp, đồng thời không còn bảng con nào chưa xét, thuật giải kết thúc.

♦ 7.4.4. Đánh giá thuật giải

Số lượng các luật thu được xác định mức độ thành công của thuật giải. Đây chính là mục đích chính của các bài toán phân lớp thông qua một tập mẫu học. Một vấn đề nữa để đánh giá các hệ học quy nạp là khả năng hệ thống có thể phân lớp các mẫu được đưa vào sau này.

Thuật giải ILA được đánh giá mạnh hơn hai thuật giải khá nổi tiếng về phương pháp học quy nạp trước đây là ID3 và AQ, đã thử nghiệm trên một số tập mẫu như Balloons, Balance, và Tic-tac-toe (lấy từ Kho Dữ liệu Máy học và Giả thuyết - Đại học California Irvine).

CHƯƠNG 8. KẾT HỢP CƠ SỞ TRI THỨC VÀ CƠ SỞ DỮ LIỆU

8.1. DẪN NHẬP

8.2. KẾT HỢP CSDL VÀ CSTT

8.2.1. Dạng luật trong CSTT

8.2.2. Ý nghĩa của các phép toán logic trong các luật suy diễn

8.2.2.1. Phép toán AND

cuu duong than cong. com

8.2.2.2. Phép toán OR

8.2.2.3. Phép toán NOT (\sim)

8.3. MÔ HÌNH SUY DIỄN

8.3.1. So khớp và đồng nhất biến

cuu duong than cong. com

8.3.2. Phân giải luật suy diễn không đệ qui

8.4. VÍ DỤ MINH HỌA

8.4.1. Phần Cơ sở dữ liệu

8.4.2. Phần Cơ sở tri thức

8.5. XÂY DỰNG ĐỘNG CƠ SUY DIỄN THÔNG TIN TỪ CSDL DỰA TRÊN CÁC LUẬT TRONG CSTT

8.5.1. Tổ chức dữ liệu

8.5.2. Quản trị CSTT

8.5.2.1. Các chức năng
quản trị CSTT

8.5.2.2. Các chức năng
thêm, xóa, sửa các luật
trong CSTT

8.5.3. Một số thuật toán để
trong động cơ suy diễn cho
các luật suy diễn dữ liệu

8.1. MỞ ĐẦU

Trong phần này chúng tôi trình bày cách kết hợp cơ sở tri thức (CSTT) với cơ sở dữ liệu (CSDL) nhằm suy diễn thông tin từ CSDL bằng các luật suy diễn

trong CSTT. Trước hết, chúng tôi nêu ra dạng của các luật suy diễn, cách suy diễn thông tin từ các luật có trong CSTT và dữ liệu trong CSDL, cách quản trị CSTT. Trong các phần sau, chúng tôi sẽ trình bày cách kết hợp CSTT với CSDL để khai thác dữ liệu và suy diễn thông tin dựa trên tri thức và hiểu biết của chuyên gia về dữ liệu.

8.2. KẾT HỢP CSDL VÀ CSTT

♦ 8.2.1. Dạng luật trong CSTT

Chúng tôi gọi các luật của CSTT kết hợp với CSDL là các luật suy diễn dữ liệu, các luật này có dạng tổng quát như sau:

$$H: - G_1 \& G_2 \dots \& G_k$$

Trong đó: **H** được gọi là phần đầu hay kết luận của luật. $G_1 \& G_2 \dots \& G_k$ là phần thân của luật. Các G_k là đích con hay tiền đề của luật.

Ví dụ 1: $\text{parent}(X,Y): - \text{father}(X,Y) \mid \text{mother}(X,Y)$

Với **father**, **mother**, **parent** là các vị từ **X,Y** là các biến. Mỗi vị từ $p(X,Y,Z)$ ứng với một quan hệ

$P(X,Y,Z)$. Chúng tôi phân biệt hai loại vị từ, một là vị từ được suy và hai là vị từ nền.

• **Vị từ được suy IDB** (intensional predicate): là vị từ xuất hiện trong phần kết luận của luật. Vị từ được suy cũng có thể xuất hiện trong phần thân của luật.

• **Vị từ nền EDB** (extensional predicate): ứng với một quan hệ được lưu trữ trong CSDL. Mỗi vị từ ứng với một quan hệ. Một vị từ có thể nhận được giá trị đúng hay sai. Nếu p là vị từ nền và P là quan hệ nền thì $p(a,b,c)$ với a, b, c là đối sẽ có giá trị đúng nếu bộ (a,b,c) sẽ tạo được trong tiến trình suy diễn.

Trong ví dụ trên, father và mother là các vị từ nền(EDB) còn parent là vị từ được suy(IDB). Ta cũng phân ra hai dạng luật suy diễn dữ liệu là:

• **Luật không đệ qui**: Vị từ ở phần đầu không xuất hiện trong phần thân của luật.

Ví dụ 2: $sibling(X,Y): - parent(Z,X) \ \& \ mother(Z,Y)$.

Luật đệ qui: Vị từ ở phần đầu xuất hiện trong phần thân của luật.

Ví dụ 3: $\text{ancestor}(X,Y) :- \text{parent}(X,Y).$

$\text{ancestor}(X,Y) :- \text{parent}(X,Z) \ \& \ \text{ancestor}(Z,Y)$

Trong hệ thống của mình, chúng tôi chỉ tập trung khảo sát các luật suy diễn không đệ qui cho CSTT.

♦ 8.2.2. Ý nghĩa của các phép toán logic trong các luật suy diễn

Trong các luật suy diễn dữ liệu có các phép toán logic *and*(&), *or*(|), *not*(~) để kết nối các vị từ trong phần thân của luật. Các phép toán này được định nghĩa dựa trên các phép toán của đại số quan hệ và các câu lệnh SQL tương đương như sau:

8.2.2.1. Phép toán AND

Phép **AND(&)** được xây dựng trên cơ sở phép kết và phép chiếu của đại số quan hệ.

Với luật $t(a,b,d,e) :- r(a,b,c) \ \& \ s(c,d,e)$, quan hệ trong $T(a,b,d,e)$ ứng với vị từ $t(a,b,d,e)$ được tính theo cách sau:

$$T(a,b,d,e) = \Pi_{a,b,d,e} (R(a,b,c) \bowtie S(c,d,e)).$$

Nếu dùng câu SQL, ta có câu lệnh tương ứng:

SELECT r.a, r.b, s.d, s.e

FROM table r, table s

WHERE r.c = s.c.

Nếu một vị từ trong phần tiền đề của luật có dạng một biểu thức so sánh với luật $s(a,b,c)$: - $r(a,b,c) \ \& \ (b \geq 2)$. Quan hệ $S(a,b,c)$ ứng với vị từ $s(a,b,c)$ \emptyset được suy $s(a,b,c)$ được tính theo công thức sau:

$$s(a,b,c) = \delta_{b \geq 2} (r(a,b,c))$$

Trong đó δ là phép chọn theo điều kiện $b \geq 2$, nếu dùng câu SQL,ta có câu lệnh tương ứng:

SELECT *

FROM table r

WHERE r.b \geq 2

8.2.2.2. Phép toán OR

Phép toán **OR** (\cup) được xây dựng trên cơ sở phép hợp sau đây:

$$t(a,b,c): - r(a,b,c) \cup s(a,b,c)$$

Quan hệ **T(a,b,c)** trong **t(a,b,c)** được tính theo cách sau:

$$T(a,b,c) = R(a,b,c) \cup S(a,b,c)$$

Nếu dùng SQL, ta có câu lệnh tương ứng :

SELECT *

FROM table1 **r**

UNION SELECT * FROM table2 **r INTO**
table t

8.2.2.3. Phép toán NOT (\sim)

Phép **not**(\sim) được xây dựng trên cơ sở phép hiệu, ví dụ:

$$t(a,b,c): - r(a,b,c) \& (\sim s(a,b,c))$$

Quan hệ được suy **T(a,b,c)** của vị từ **t(a,b,c)** được tính theo cách sau:

$$t(a,b,c) = r(a,b,c) \setminus s(a,b,c)$$

Nếu dùng SQL, ta có thể cài đặt như sau:

SELECT a, b, e

FROM table r

WHERE a NOT IN (SELECT a FROM s)

8.3. MÔ HÌNH SUY DIỄN

♦ 8.3.1. So khớp và đồng nhất biến

Mục đích là để so sánh các thành phần để tìm vị từ và sự kiện trong tiến trình suy diễn. Sau khi so khớp sẽ xảy ra tiến trình đồng nhất biến theo nghĩa thay thế biến bằng một giá trị cụ thể. Xét hai luật sau:

r1: grandfather(X,Y): -father(X,Z) & parent(Z,Y)

r2: parent(X,Y): - father(X,Y) | mother(X,Y)

Quan hệ **Father(A,B)**, **Mother(A,B)** được mô tả bằng vị từ **father(A,B)**, **mother(A,B)** là các quan hệ nền (hay **EDB**). Từ các vị từ ngoài, nhờ các luật suy diễn chúng ta có thể tạo sinh các quan hệ cho

các vị từ được suy(**IDB**) là **Parent(A,B)** hay **Grandfather(A,B)** như trong ví dụ trên.

Có thể mô tả các luật suy diễn bằng đồ thị suy diễn. Ví dụ với hai luật trên ta có thể tạo đồ thị dạng cây suy diễn ở hình 8.1.

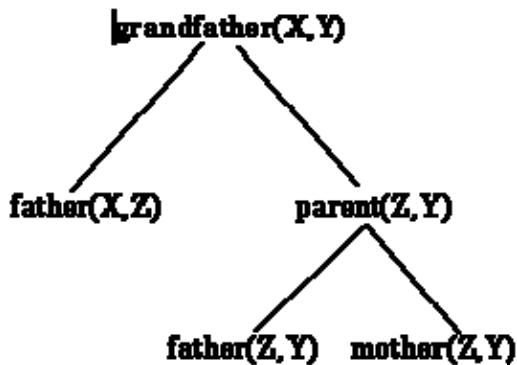
Trong tiến trình suy diễn để tạo quan hệ cho vị từ được suy **grandfather(X,Y)** chúng ta cần tạo các quan hệ cho các vị từ **father(X,Z)** và **parent(Z,Y)**. Do **father(X,Z)** là quan hệ nền nên chỉ cần thẩm định **parent(Z,Y)** bằng cách so khớp và đồng nhất biến để có dạng sau:

parent(Z,Y):- father(Z,Y) | mother(Z,Y)

Với vị từ nền **father(Z,Y)** chúng ta sử dụng so khớp và đồng nhất biến theo thứ tự xuất hiện của đối trong vị từ và thứ tự xuất hiện trường trong quan hệ nền. Từ quan hệ **father(F,C)**, chúng ta có các đồng nhất biến sau cho **father(X,Z)** và **father(Z,Y)**.

FATHER (<u>F</u> <u>C</u>)	father(<u>X</u> <u>Z</u>)
father(<u>Z</u> <u>Y</u>)	

nam	son	nam
son	nam	son
loc	vinh	loc
vinh	loc	vinh



Hình 8.1. Đồ thị suy diễn

♦ 8.3.2. Phân giải luật suy diễn không đệ qui

Nhìn chung có hai bước chính là:

- Tạo cây suy diễn theo các luật.
- Duyệt cây để thăm định và tạo sinh dữ liệu cho vị từ được suy.

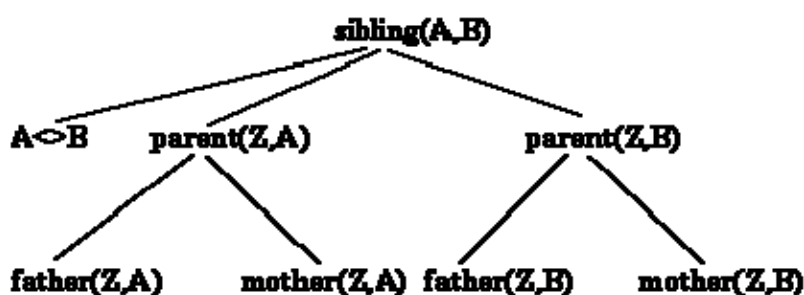
Với hai luật:

r1: sibling(X,Y):- parent(X,Z) & parent(Z,Y) & (X<>Y)

r2: parent(X,Y):- father(X,Y) | mother(X,Y)

Vị từ được suy **sibling(X,Y)** với các đích **sibling(A,B)** sẽ được thăm định như sau:

- Tìm luật so khớp được với đích và thực hiện đồng nhất biến.
- Tạo cây con có gốc chính là đích của bài toán. Xử lý đệ qui với các đích con là các lá của cây con vừa mới tạo được. Nếu vị từ trong lá là vị từ nền thì không thể mở rộng được cây con.
- Kết quả ta đồ thị suy diễn ở hình 8.2:



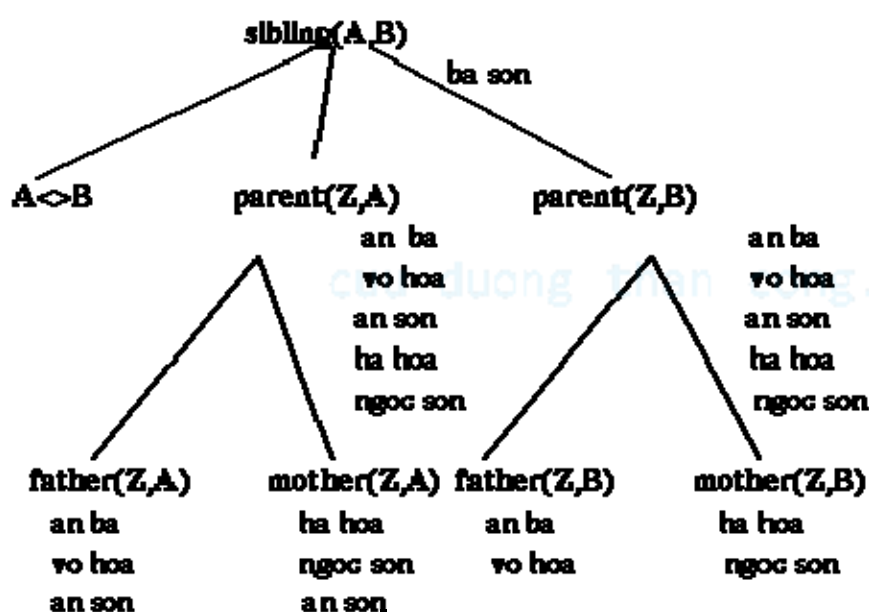
Hình 8.2. Đồ thị suy diễn

Sau khi đã tạo xong cây suy diễn, chúng ta bước sang suy diễn bằng cách duyệt cây để tạo sinh dữ liệu cho các vị từ được suy. Ý tưởng của thuật toán như sau:

- Tìm luật có phần đầu so khớp được với đích.
- Tạo dữ liệu cho các vị từ trong phần thân của luật.
- Thực hiện các phép toán logic của cơ sở dữ liệu suy diễn để tạo dữ liệu cho vị từ được suy.

Trong phần 4, chúng tôi sẽ trình bày chi tiết các thuật toán trên.

Ví dụ: với các vị từ nền **father(X,Y)** và **mother(X,Y)** sau đây, chúng ta có thể tạo dữ liệu cho quan hệ **sibling(X,Y)** thông qua đồ thị suy diễn ở hình 8.3.



Hình 8.3. Tạo ra các quan hệ IDB trên đồ thị suy diễn

8.4. VÍ DỤ MINH HỌA

Trong phần này, chúng tôi trình bày một ví dụ kết hợp CSTT và CSDL để suy diễn thông tin từ CSDL.

♦ 8.4.1. Phần Cơ sở dữ liệu

Xét một CSDL theo dõi kết quả học tập của sinh viên với các quan hệ nền sau.

• **Nganhhoc**(manganh,tennganh): chứa các ngành học của trường.

• **Sinhvien**(masv,manganh,holot,ten): chứa hồ sơ sinh viên.

• **Moncoso**(manganh,mamon,tenmon,sotc): các môn cơ sở của từng ngành học và số tín chỉ của môn.

• **Chuan**(manganh,mamon,dchuan): điểm chuẩn của môn cơ sở chính của ngành dùng để suy diễn.

• **Diemcoso**(masv,mamon,dmon): điểm kết quả môn của sinh viên.

• **Chuyennganh**(manganh,machnganh): các chuyên ngành của từng ngành.

• **Chuyende**(machnganh,machde,tenchde,sotc): các chuyên đề của ngành.

• **Diemchde**(masv,machde,dchde): điểm kết quả chuyên đề của sinh viên.

♦ 8.4.2. Phần Cơ sở tri thức

Phần CSTT bao gồm các luật suy diễn dữ liệu sau:

• **Các luật suy diễn 1: Sinh viên giỏi ở giai đoạn cơ sở**

r1:

IF diemcoso(masv,mamon,dmon)

AND chuan(manganh, mamon, dmon,dchuan)

THEN

diemchinh(masv,manganh,mamon,dmon,dchuan)

Tạo quan hệ **diemchinh**(masv,manganh,mamon,dmon,dc huan) chứa điểm môn quan trọng của ngành ở giai đoạn của sinh viên và điểm chuẩn để suy diễn của môn đó.

r2:

IF **dchinh**(masv,manganh, mamon, dmon,dchuan)

AND (dmon > dchuan)

THEN **svgioicoso** (masv,manganh)

Tạo quan hệ **svgioicoso**(masv,manganh) chứa các sinh viên được đánh giá là giỏi ở giai đoạn cơ sở theo quan điểm nếu điểm môn quan trọng của sinh viên lớn hơn điểm chuẩn.

➡ **Các luật suy diễn 2: Sinh viên giỏi ở giai đoạn chuyên ngành**

r3:

IF

chuyende(machnganh,machde,tenchde,sotchi)

AND (sotchi>5)

THEN

chdeqtrong(machnganh,machde)

Tạo quan hệ **chdeqtrong**(machnganh,machde) chứa các chuyên đề quan trọng theo nghĩa có số tín chỉ lớn hơn 5.

r4: **IF diemde**(masv,machde,dchde)
AND (dchde>8)

THEN svgioicde(masv, machde)

Tạo quan hệ **svgioicde**(masv,machde) chứa các chuyên đề mà sinh viên đạt kết quả tốt.

r5: **IFsvgioicde**(masv,machde)

AND

chdeqtrong(machnganh,machde)

THEN svgioichng(masv, machnganh)

Tạo quan hệ **svgioichng**(masv, machnganh) chứa các sinh viên được đánh giá là giỏi chuyên ngành được xác định bởi machnganh. Ở đây có thể dùng độ đo niềm tin vào để đánh giá.

➤ **Các luật suy diễn 3: Sinh viên sẽ làm sẽ làm nghiên cứu sinh theo chuyên ngành nào sau khi tốt nghiệp**

r6: IF svgioicoso(masv,manganh)

AND svgioichng(masv, machnganh)

THEN

svtheochnganh(masv,machnganh)

Tạo quan hệ **svtheochnganh**(masv,machnganh) suy luận về chuyên ngành sinh viên sẽ làm nghiên cứu sinh. Luật này có độ đo chính xác của luật.

8.5. XÂY DỰNG ĐỘNG CƠ SUY DIỄN THÔNG TIN TỪ CSDL DỰA TRÊN CÁC LUẬT TRONG CSTT

♦ 8.5.1. Tổ chức dữ liệu

Chúng tôi dùng một đồ thị suy diễn $G = (E, V)$ để chứa các luật suy diễn dữ liệu, trong đó E là các cạnh do các luật suy diễn dữ liệu tạo ra và V là tập các đỉnh ứng với các quan hệ **EDB** hay **IDB**. Đồ thị này được lưu trong các bảng sau:

a) Bảng 1: EvidenceTable(RuleNo, Term)

Thuộc tính Term của bảng này chứa dạng Postfix của phần giả thuyết của luật suy diễn dữ liệu.

Với dạng lưu trữ của luật

r1: IF (NOT a) AND (NOT b)
THEN sẽ là a ~ b ~ &

b) Bảng 2: RuleTable(Ruleno, Evidence, Conclusion, Cfrule)

Chứa luật suy diễn dữ liệu. Cfrule là độ tin cậy của luật

c) Bảng 3: NodeTable(NodeNo, NodeType, NodeDescription, Cfnode)

Chứa thông tin của các node trên đồ thị suy diễn. Thuộc tính Nodetype có ba giá trị là 1 nếu là node lá, 2 nếu là nối mục tiêu trung gian và 3 nếu là mục tiêu cuối.

◆8.5.2. Quản trị CSTT

Chúng tôi xây dựng bộ quản trị CSTT bao gồm các chức năng thêm, xoá, sửa luật. Với bộ quản trị CSTT này, chúng ta có thể tiến hành soạn thảo các luật suy diễn dữ liệu.

8.5.2.1. Các chức năng quản trị CSTT

a) Hàm tạo dữ liệu cho các bảng EvidenceTable

```
Function CreaPostFix( Ruleno,
Cfrule:string): table
Begin
    ExpArray =
    ChangeToPostFixArray(Cfrule
    );
    For ( each Term of ExpArray
    ) do
```

```
WriteDataToTableEvidence(Ruleno, Term);  
  
Return (EvidenceTable);  
  
End;
```

b) Hàm tạo dữ liệu cho các bảng RuleTable

```
Function CreaRule( Ruleno,  
RuleEvidence, RuleConclusion,  
Cfrule:string): table  
  
Begin  
WriteDataToTableRule(Ruleno  
,RuleEvidence,  
RuleConclusion, Cfrule);  
  
Return (RuleTable);  
  
End;
```

c) Hàm tạo dữ liệu cho các bảng NodeTable

```
Function CreaNode( Ruleno,  
Cfrule:string): table  
  
Begin
```

```

ExpArray=
CreateAllNodesOfNet
(EvidenceTable,
RuleTable);

For ( each Nodeno of
ExpArray) do

Begin

NodeTypeInfo =
AssignNodeType (Rule, Nodeno);
WriteDataToTableNode (NodeInfo, NodeTypeInfo);

End;

Return (NodeTable);

End;

```

8.5.2.2. Các chức năng thêm, xóa, sửa các luật trong CSTT

a) Thêm luật vào CSTT

```
Function AddRule (RuleNo,
```



```

RuleEvidence,      RConclusion:
string;           Cfrule:      real;
EvidenceTable,RuleTable,NodeT
able: table) : BOOL

```

Begin

```

EvidenceTable      =
CreaPostFix(      Ruleno,
Cfrule);

```

```

RuleTable      =      CreaRule(
Ruleno,      RuleEvidence,
RuleConclusion,
Cfrule:string);

```

```

NodeTable      =      CreaNode(
EvidenceTable,
RuleTable);

```

```

If      (      NOT
Validate(Ruleno)) then

```

Begin

```

DeleteRule(Ruleno,
EvidenceTable,
RuleTable,

```

```

        NodeTable) ;

        return FALSE;

    End;

    return TRUE;

End;

```

b) Hàm xử lý xóa luật khỏi CSTT

```

Function      DeleteRule (Ruleno:
string;
EvidenceTable, RuleTable, NodeT
able: table) : BOOL

Begin

    DeleteRuleInEvidenceTable (
Ruleno) ;

    DeleteRuleInRuleTable (Rulen
o) ;

    NodeTable =
    CreaNode (EvidenceTable,
RuleTable) ;

    If ( NOT Validate (Ruleno)

```

Then

Begin

UnDeleteRule (Ruleno,
EvidenceTable,
RuleTable, NodeTable) ;

return FALSE;

End;

return TRUE;

End;

c) Hàm sửa luật suy diễn trong CSTT

Function **EditRule** (Ruleno:
string;
EvidenceTable, RuleTable, NodeT
able: table): BOOL

Begin

RuleInfo=
ReadInfoFromTable (Ruleno,
RuleTable) ;

```

Edit (RuleInfo) ;

If (DeleteRule (Ruleno, Eviden
ceTable, RuleTable,
NodeTable)

Then

Begin

    AddRule (Ruleno, RuleInfo
    .RuleEvidence,
    RuleInfo.RConclusion, Ru
    leInfo.Cfrule, EvidenceT
    able,          RuleTable,
    NodeTable) ;

    return TRUE;

End;

return TRUE;

End;

```

d) Hàm kiểm tra tính hợp lệ của CSTT

```

Function ValidateRule (Ruleno:
string;          EvidenceTable,

```

```

RuleTable,
NodeTable:table) :BOOL

Begin

    If (
        HasLoopInNet (EvidenceTable,
            RuleTable, NodeTable))

        return FALSE;

    return TRUE;

End;

```

♦ 8.5.3. Một số thuật toán để trong động cơ suy diễn cho các luật suy diễn dữ liệu

a) Hàm tính trị của một nút

```

Function
EvaluateNode (NodeName:string)
: real;

Begin

    If ( NodeName is a
        Terminator) Then

```

Begin

Do case

Case SimpleNode: Ask for
the Cfnode;

Case CompoundNode:
Evaluate the Node

Expression;

Create CfNode;

Endcase

Return (CfNode) ;

End;

If (NodeName had Cfnode)
Then

return (Cfnode) ;

MatchingRuleArray =
FindMatchRules (NodeName) ;

Ruleno =
GetNextRuleFromArray (

```

MatchingRuleArray) ;

Val1 = EvaluateRule(Ruleno) ;

For      (      each      rule      of
MatchingRuleArray ) do

Begin

        Ruleno                                =
        GetNextRuleFromArray (Matc
hingRuleArray) ;

        Val2                                =
        EvaluateRule (Ruleno) ;

        Val1      =      CombinesRules
        (Val1, Val2) ;

End;

return (Val1) ;

End;

```

b) Hàm tính trị của một luật

```

Function
EvaluateRule (Ruleno:string) :

```

```
string;
```

```
Begin
```

```
PostFixArray =  
CreateTermFromEvidenceTable (R  
uleno) ;
```

```
InitStack() ;
```

```
For ( each element of  
PostfixArray ) do
```

```
Begin
```

```
cuu duong than cong. com  
X =  
GetNextTermFromArray (PostFi  
xArray) ;
```

```
If ( IsOperand(X) ) Then  
PushStack (X) ;
```

```
If ( IsOperator(X) ) Then
```

```
cuu duong than cong. com  
Begin
```

```
y = PopStack() ;
```

```
Val1 =  
(IsNode (Y) ) ?EvaluateNode (
```



```
Y) : Y ;
```

```
Do case
```

```
Case IsAnd(X) :
```

```
Y = PopStack() ;
```

```
Val2 = EvaluateNode(Y) ;
```

```
Val = ANDCombine(Val1,  
Val2) ;
```

```
PushStack(Val) ;
```

```
Case IsOr(X) :
```

```
Y = PopStack() ;
```

```
Val2 = EvaluateNode(Y) ;
```

```
Val = ORCombine(Val1,  
Val2) ;
```

```
PushStack(Val) ;
```

```
Case IsNot(X) :
```

```
Val =  
NotEvaluate(Val1) ;
```

```
        PushStack (Val) ;  
  
    EndCase ;  
  
End ;  
  
Return ( ( PopStack() * Cfrule  
)  
  
End ;
```

c) Các hàm phân giải các phép toán AND, OR, NOT

Các hàm **ANDCombine()**; **ORCombine()**; **NOTEvaluate()** nhằm thực hiện các phép toán **AND**, **OR**, **NOT** nhằm tạo sinh ra quan hệ ứng với vị từ kết quả như đã nêu trong mục 8.2.2.1, 8.2.2.2, và 8.2.2.3.

CHƯƠNG 9. HỆ THỐNG MỜ CHO CÁC BIẾN LIÊN TỤC

9.1. DẪN NHẬP

9.2. CÁC KHÁI NIỆM CƠ BẢN

9.2.1. Tập rõ và hàm thành viên

9.2.2. Tập mờ và hàm thành viên

9.2.3. Các dạng của hàm thành viên

9.2.4. Các phép toán trên tập mờ

9.3. CÁC HỆ THỐNG MỜ

9.4. NGUYÊN LÝ XỬ LÝ CÁC BÀI TOÁN MỜ

9.4.1. Bài toán 1

9.4.2. Bài toán 2

9.1. DẪN NHẬP

Các chuyên gia sử dụng các lập luận một cách tự nhiên để giải quyết các bài toán. Các tri thức này thường là các tri thức không rõ ràng và rất khó diễn tả bằng các hệ thống logic truyền thống.

Từ những năm 1920, Lukasiewicz đã nghiên cứu cách diễn đạt toán học khái niệm mờ. Năm 1965, Lofti Zadeh đã phát triển lý thuyết khả năng và đề xuất hệ thống logic mờ (fuzzy logic). Hiện nay, logic mờ đang được ứng dụng rộng rãi trong rất nhiều lĩnh vực, đặc biệt là các hệ thống điều khiển mờ. Chương này trình bày các khái niệm cơ bản về logic mờ, lập luận mờ và hệ thống điều khiển mờ tiêu biểu.

9.2. CÁC KHÁI NIỆM CƠ BẢN

♦ 9.2.1. Tập rõ và hàm thành viên

Tập rõ (crisp set) là tập hợp truyền thống theo quan điểm của Cantor (crisp set). Gọi A là một tập hợp rõ, một phần tử x có thể có $x \in A$ hoặc $x \notin A$, Có thể sử dụng hàm χ để mô tả khái niệm thuộc về. Nếu $x \in A$, $\chi(x) = 1$, ngược lại nếu $x \notin A$, $\chi(x) = 0$. Hàm χ được gọi là hàm đặc trưng của tập hợp A .

♦ 9.2.2. Tập mờ và hàm thành viên

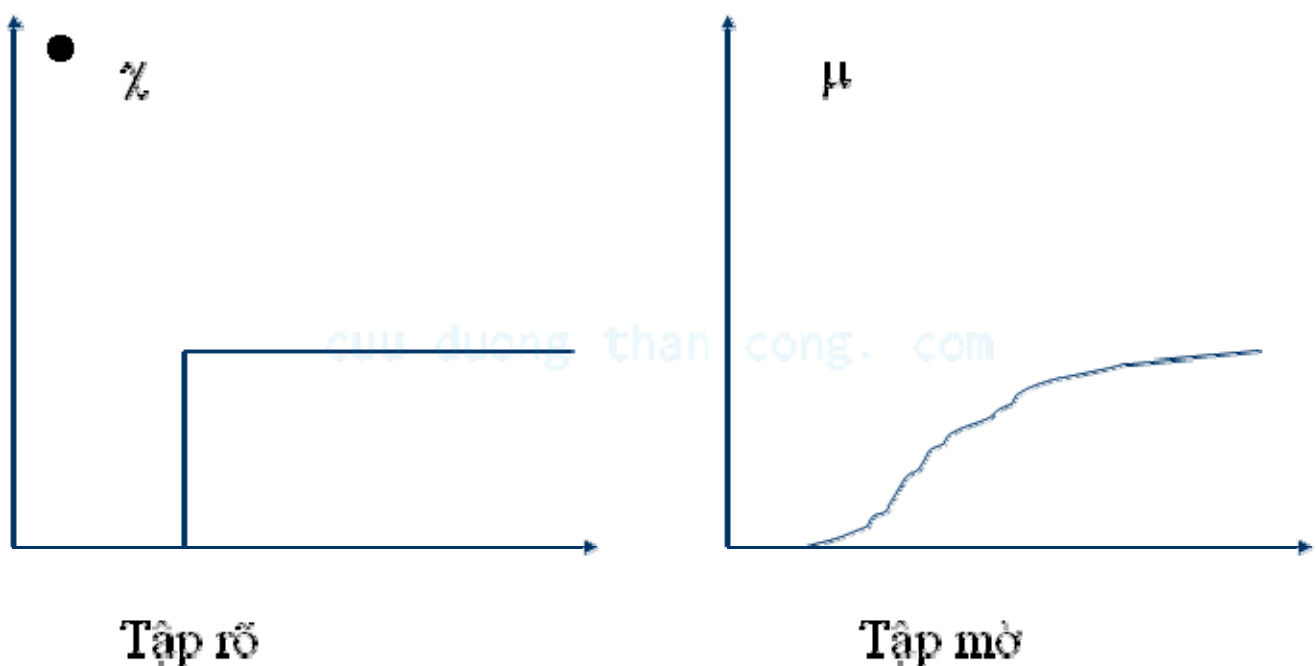
Khác với tập rõ, khái niệm thuộc về được mở rộng nhằm phản ánh mức độ x là phần tử của tập mờ A . Một tập mờ (fuzzy set): A được đặc trưng bằng hàm

thành viên μ và cho x là một phần tử $\mu(x)$ phản ánh mức độ x thuộc về A .

Ví dụ: Cho tập mờ High

Lan cao 1.5m, $\mu(\text{Lan})=0.3$

Hùng cao 2.0 m, $\mu(\text{Hùng})=0.9$



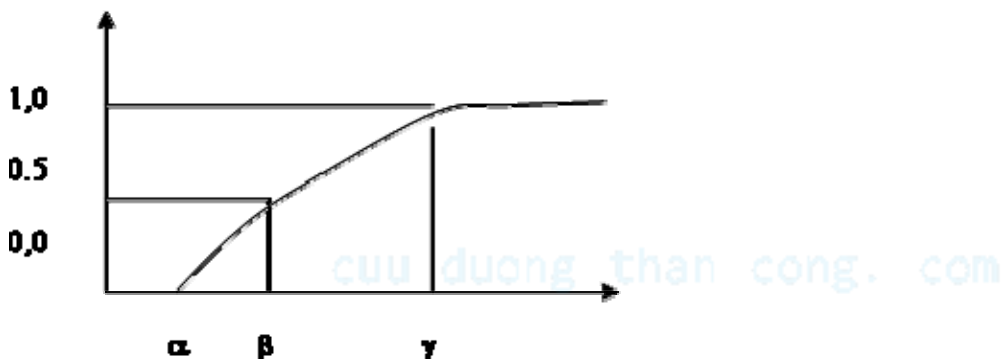
Hình 9.1. Đường biểu diễn của hàm đặc trưng và hàm thành viên

♦ 9.2.3. Các dạng của hàm thành viên

Các hàm thành viên của tập mờ có 3 dạng cơ bản là: dạng tăng, dạng giảm và dạng chuông.

a) Dạng S tăng

$$\mu(x) = S(x, \alpha, \beta, \gamma) = \begin{cases} 0 & \text{nếu } x \leq \alpha \\ \frac{2(x - \alpha)}{(\gamma - \alpha)} & \text{nếu } \alpha < x \leq \beta \\ 1 - \frac{2(x - \beta)}{(\gamma - \beta)} & \text{nếu } \beta < x < \gamma \\ 1 & \text{nếu } x \geq \gamma \end{cases}$$



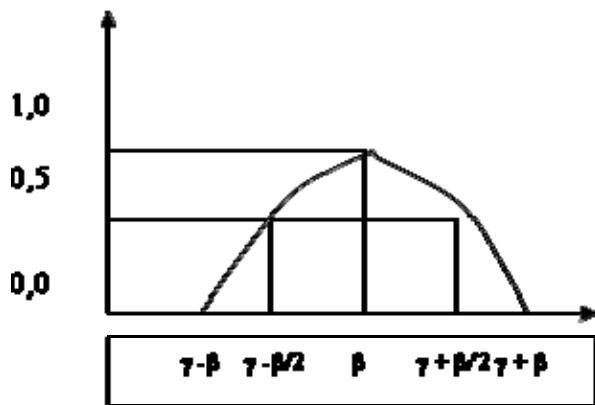
Hình 9.2. Hàm S tăng

b) Dạng S giảm

$$\mu(x) = 1 - S(x, \alpha, \beta, \gamma)$$

c) Dạng hình chuông

$$\Pi(x; \gamma, \beta) = \begin{cases} S(x; \gamma - \beta, \gamma - \beta/2; \gamma) & \text{if } x \leq \gamma \\ S(x; \gamma, \gamma + \beta/2; \gamma + \beta) & \text{if } x > \gamma \end{cases}$$



Hình 9.3. Hàm dạng chuông

♦ 9.2.4. Các phép toán trên tập mờ

Cho ba tập mờ A, B, C với $\mu A(x), \mu B(x), \mu C(x)$

$$C = A \cap B: \mu C(x) = \min(\mu A(x), \mu B(x))$$

$$C = A \cup B: \mu C(x) = \max(\mu A(x), \mu B(x))$$

$$C = \neg A: \mu C(x) = 1 - \mu A(x)$$

9.3. CÁC HỆ THỐNG MỜ

♦ Hàm thành viên cho các biến rời rạc

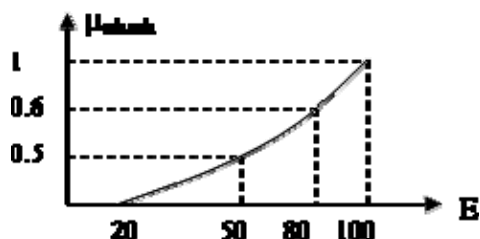
Cho tập vũ trụ $E = \text{Tốc độ} = \{ 20, 50, 80, 100 \}$ đơn vị là Km/g.

a. Xét tập mờ $F = \text{Nhanh}$ xác định bởi hàm membership

$$\mu_{\text{nhanh}}: E \rightarrow [0, 1]$$

$x_1 \text{ ----} > \mu_{\text{nhanh}}(X)$

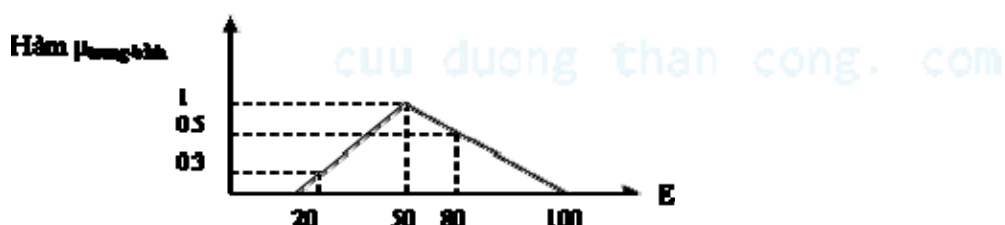
Khi ta gán $\mu_{\text{nhanh}}(20) = 0$ nghĩa là tốc độ 20 Km/g được xem như là không nhanh.



theo nguyên tắc đó tập mờ nhanh = $\{ (20,0), (50,0.5), (80,0.6), 100, 1) \}$ hay vắn tắt hơn Nhanh = $\{ 0,0.5,0.6,1 \}$

Vậy hàm thành viên đánh giá mức độ đúng của các tốc độ trong tập vũ trụ E với khái niệm nhanh. Hàm này có tính chủ quan và do kinh nghiệm hay do thực nghiệm.

b. Xét tập mờ trung_bình với hàm thành viên xác định như sau:



thì tập Trung Bình = $\{ 0.3,1,0.5,0 \}$

• Hàm thành viên trong không gian các biến liên tục

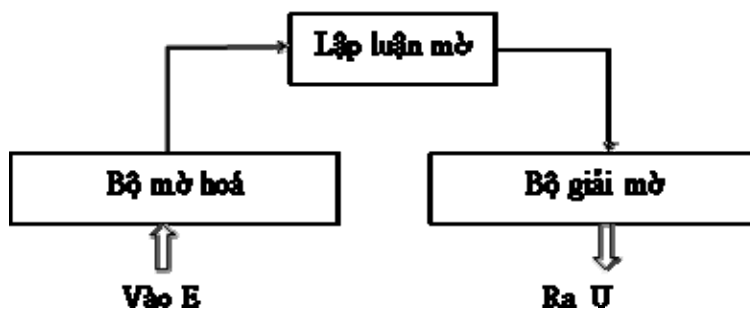
Chẳng hạn như các tập mờ Nhanh và Trung bình ở trên có thể định nghĩa như là các hàm

$$\mu_{\text{nhanh}}(x) = (x/100)^2$$

$$\mu_{\text{trung-bình}}(x) = \begin{cases} 0 & \text{if } x \leq 20 \\ (x-20)/30 & \text{if } 20 \leq x \leq 50 \\ (100-x)/50 & \text{if } 50 \leq x \leq 100 \end{cases}$$

Trong phần sau chỉ xét các hàm thành viên

9.4. NGUYÊN LÝ XỬ LÝ CÁC BÀI TOÁN MỜ



Hình 9.4. Hệ thống mờ

Các dữ liệu nhập qua bộ mờ hoá để biến thành các trị mờ. Sau đó các giá trị mờ được đưa vào bộ lập luận mờ. Các kết quả là các giá trị mờ ứng với phần kế luật. Bộ giải mờ sẽ biến đổi trị mờ trở lại trị rõ.

♦9.4.1. Bài toán 1

Dữ liệu Input là các giá trị rõ.

Ví dụ: Xét bài toán mờ xác định bởi các luật sau:

Luật 1: if x is A_1 and y is B_1 Then z is C_1

Luật 2: if x is A_2 or y is B_2 Then z is C_2

Vào: trị x_0, y_0

--

Ra : trị z_0 tương ứng

Bài toán được giải quyết như sau:

Ứng với tập mờ A_1 ta có hàm thành viên $\mu_{A_1}(x)$

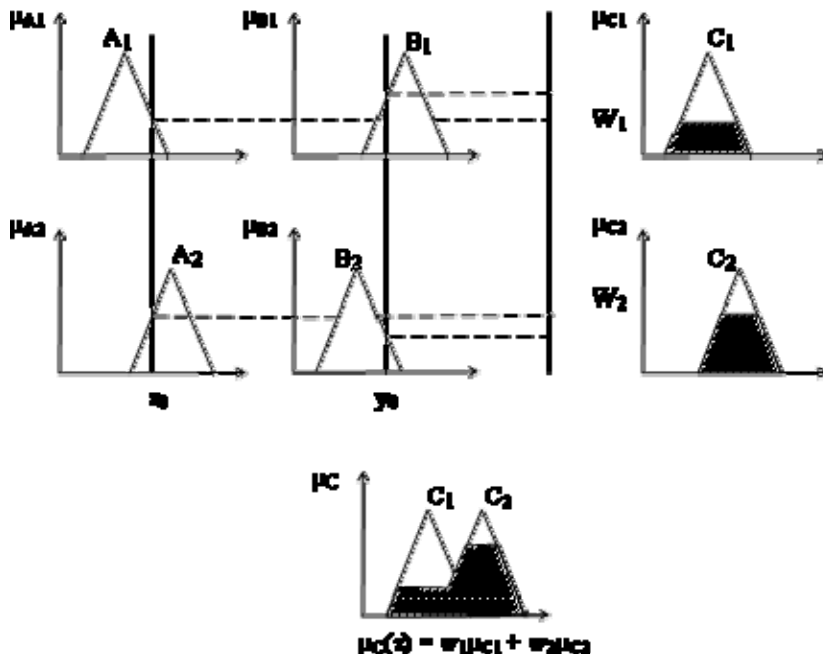
Ứng với tập mờ A_2 ta có hàm thành viên $\mu_{A_2}(x)$

Ứng với tập mờ B_1 ta có hàm thành viên $\mu_{B_1}(y)$

Ứng với tập mờ B_2 ta có hàm thành viên $\mu_{B_2}(x)$

Ứng với tập mờ C_1 ta có hàm thành viên $\mu_{C_1}(x)$

Ứng với tập mờ C_2 ta có hàm thành viên $\mu_{C_2}(x)$

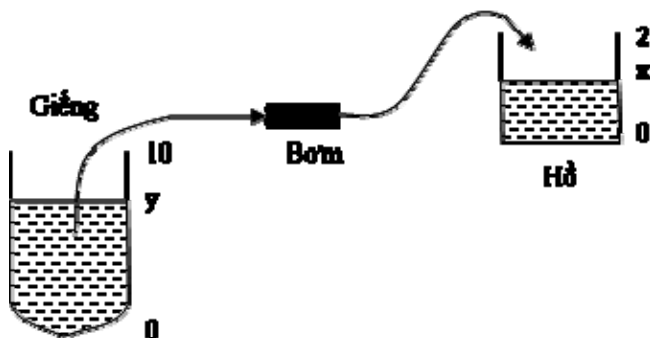


Vấn đề là khi cho các giá trị Input $x = x_0$ và $y = y_0$ hãy tìm hàm thành viên của các luật theo hình vẽ W_1 là min của hai giao điểm, W_2 là max của hai giao điểm: chúng gọi là trọng các luật. Khi đó hàm thành viên của kết luận là:

$$\mu_C(z) = \sum_{i=1}^N W_i \mu_{K1i}(z)$$

với $\mu_{K1i}(z)$ là hàm thành viên của kết luận cho luật thứ i . Từ đó suy ra trị Output z_0 là hệ thống mờ trên.

Ví dụ: Giải bài toán điều khiển tự động mờ cho hệ thống bơm nước lấy nước từ giếng. Trong khi hồ hết nước và trong giếng có nước thì máy bơm tự động bơm.



	H.Đầy	H.Lưng	H.Cạn
N.Cao	0	B.Vừa	B.Lâu
N.Vừa	0	B.Vừa	B.HơiLâu
N.Ít	0	0	0

Với biến ngôn ngữ Hồ có các tập mờ hồ đầy (H.Đầy), hồ lưng (H.Lưng) và hồ cạn (H.Cạn).

Với biến ngôn ngữ Giếng có các tập mờ nước cao (N.Cao), nước vừa (N.Vừa), nước ít (N.Ít).

Với biến ngôn ngữ kết luận xác định thời gian bơm sẽ có các tập mờ bơm vừa (B.Vừa), bơm lâu (B.Lâu), bơm hơi lâu (B.HơiLâu).

Các tập mờ trên được xác định bởi các hàm thành viên sau:

• Hàm thành viên của Hồ nước:

$$H.Đầy(x) = x/2 \quad 0 \leq x \leq 2$$

$$H.Lưng(x) = \{ x \text{ if } 0 \leq x \leq 1 \\ (2-x) \text{ if } 1 \leq x \leq 2 \}$$

$$H.Cạn(x) = 1-x/2 \quad 0 \leq x \leq 2$$

• Hàm thành viên cho giếng:

$$N.Cao(y) = y/10 \quad 0 \leq y \leq 10$$

$$N.Vừa(y) = \{ y/5 \text{ if } 0 \leq y \leq 5 \\ (10-y)/5 \text{ if } 5 \leq y \leq 10 \}$$

$$N.Ít(y) = 1-y/10 \quad 0 \leq y \leq 10$$

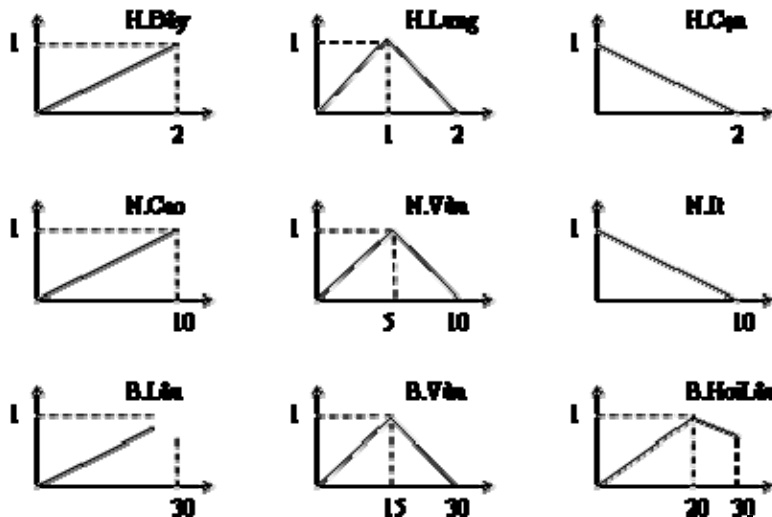
• Hàm thành viên của Kết luận cho từng luật:

$$B.Vừa(z) = \{ z/15 \text{ if } 0 \leq z \leq 15 \\ (30-z)/15 \text{ if } 15 \leq z \leq 30 \}$$

$$B.lâu(z) = z \quad 0 \leq z \leq 30$$

$$B.Hơi\ lâu(z) = \{ z/20 \text{ if } 0 \leq z \leq 20 \\ 1+0.05(z-20) \text{ if } 20 \leq z \leq 30 \}$$

Trong đó x chỉ độ sâu của Hồ ($0 \leq x \leq 2$), y chỉ độ sâu của Giếng ($0 \leq y \leq 10$) và z chỉ thời gian bơm ($0 \leq z \leq 30$).



Từ bảng trên ta có các luật:

- **Luật 1:** if x is H.Lưng and y is N.Cao
Then z is B.Vừa
- **Luật 2:** if x is H.Cạn and y is N.Cao
Then z is B.Lâu
- **Luật 3:** if x is H.Lưng and y is N.Vừa
Then z is B.Vừa
- **Luật 4:** if x is H.Cạn and y is N.Vừa
Then z is B.Hơi lâu

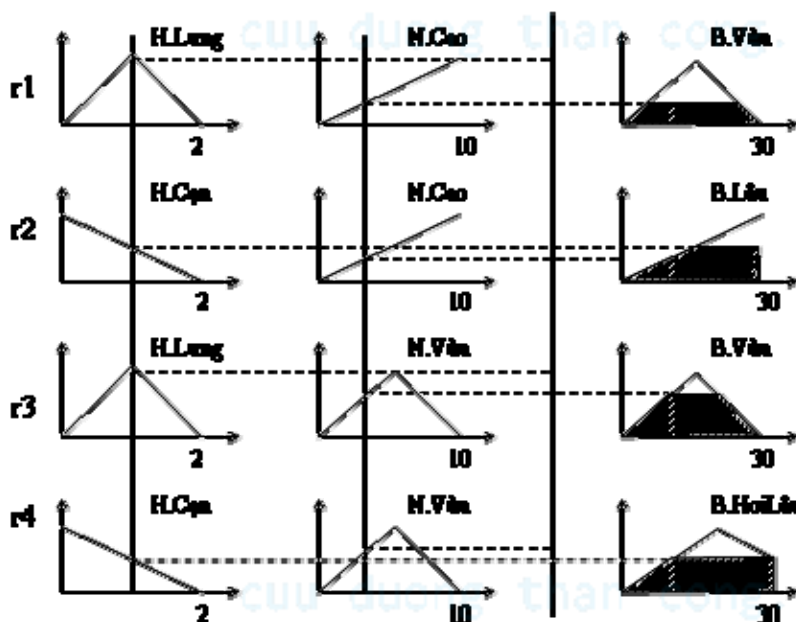
Bây giờ nếu ta nhập trị Input $x_0 = 1$ (Độ cao của nước trong hồ), $y_0 = 3$ (Độ cao của nước trong giếng)

$$\left. \begin{array}{l} \mu_{H.Lung}(x_0) = 1 \\ \mu_{H.Cao}(y_0) = 3/10 \end{array} \right\} \Rightarrow W_1 = \min \{1, 3/10\}$$

$$\left. \begin{array}{l} \mu_{H.Cao}(x_0) = 0.5 \\ \mu_{H.Vua}(y_0) = 3/5 \end{array} \right\} \Rightarrow W_2 = \min \{0.5, 3/5\} = 0.5$$

$$\left. \begin{array}{l} \mu_{H.Lung}(x_0) = 1 \\ \mu_{H.Vua}(y_0) = 3/5 \end{array} \right\} \Rightarrow W_3 = \min \{1, 3/5\} = 3/5$$

$$\left. \begin{array}{l} \mu_{H.Cao}(x_0) = 0.5 \\ \mu_{H.Vua}(y_0) = 3/5 \end{array} \right\} \Rightarrow W_4 = \min \{0.5, 3/5\} = 0.5$$



Các W_i gọi là các trọng số của luật thứ i

Theo lý thuyết hàm thành viên của kết luận cho bởi công thức:

$$\mu_c(z) = \sum_{i=1}^N W_i \mu_{K1i}(Z)$$

$$\mu_c(z) = W_1.B.Vừa(z) + W_2.B.Lâu(z) + W_3.B.Vừa(z) + W_4.B.Hơi Lâu(z)$$

$$\mu_c(z) = 3/10.B.Vừa(z) + 0.5.B.Lâu(z) + 3/5.B.Vừa(z) + 0.5.B.HơiLâu(z)$$

Bước tiếp theo là ta phải giải mờ từ hàm thành viên của kết luận bằng cách tính trọng tâm của hàm $\mu_c(z)$

Moment $\mu_c(z)$ là $\int_{-\infty}^{\infty} z \cdot \mu_c(z) dz = 17.12$

và $\int_{-\infty}^{\infty} \mu_c(z) dz = 2.3$

Vậy Defuzzy(z) = $17.12/2.3=8.15$

Do đó nếu mực nước trong hồ và giếng là 1m và 3m thì thời gian cần bơm là 8 phút và 15 giây.

♦ 9.4.2. Bài toán 2

Khi các trị Input là các tập mờ thì bài toán trên được giải quyết như thế nào?

Xét ví dụ sau:

Luật : **If** trời nắng **Then** mở khẩu độ nhỏ

Sự kiện : Trời rất nắng

Kết luận : Mở khẩu độ bao nhiêu?

Trong trường hợp này trị Input là một tập hợp mờ Rất Nắng trong trường hợp biến liên tục nó được xác định qua hàm thành viên của nó.

👉 **Các gia tử trên tập mờ**

Cho F là tập mờ trong tập vũ trụ E

Ta có các tập mờ phát sinh từ F như sau:

Very F = F^2

More or less F = $F^{1/2}$

Plus F = $F^{1.25}$

Ví dụ: nếu $F = \{0, 0.1, 0.5, 1\}$ thì $\text{very } F = F^2 = \{0, 0.01, 0.25, 1\}$

Để giải quyết bài toán 2 ta xét mô hình sau:

Cho bài toán mờ xác định bởi các quy luật

***Luật 1* : if x is A_1 and y is B_1 Then z is C_1**

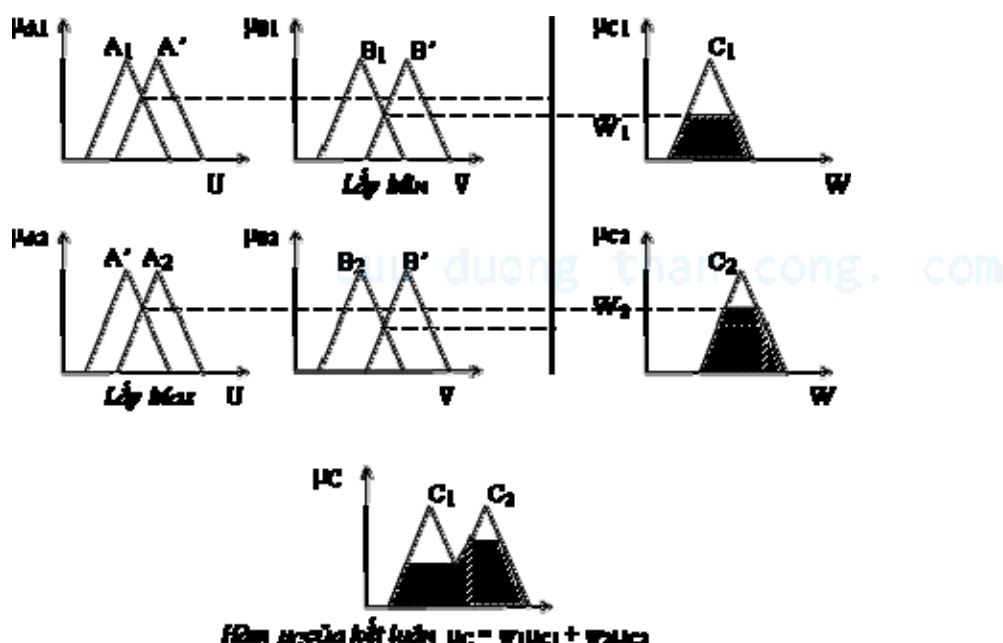
Luật 2 : if x is A_2 and y is B_2 Then z is C_2

Input : x = A' và y = B'

(A' có thể là very A , more or less A , plus A
... Cũng vậy cho B)

Kết luận : Trị rõ của Output là bao nhiêu?

Giả sử hàm thành viên của các luật trên có dạng:



Trong luật 1 ta tìm trị min của giao điểm của đồ thị A_1 và A' với giao điểm của đồ thị B_1 và B' trị min này làm trọng W_1 cho luật 1.

Tương tự cho luật 2 nhưng lần này ta lấy max (vì toán tử or) ta tìm được trọng W_2

Khi ấy hàm thành viên của Kết luận sẽ là:

$$\mu_c(z) = \sum W_i \mu_{K1i}(z) \quad i = 1 \dots N$$

Cuối cùng dùng công thức mờ ta được trị rõ

Ví dụ: Trong bài toán 1 nếu ta cho dữ liệu Input là các tập mờ như:

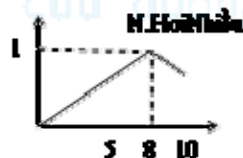
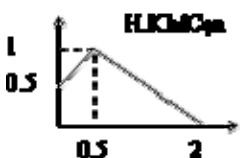
x is H.Khá Cạn (Hồ khá cạn)

y is N.Hơi nhiều (Nước trong giếng hơi nhiều)

Giả sử các tập mờ này được xác định bởi các hàm thành viên là:

$$\mu_{H.KhaCạn}(x) = \begin{cases} x+0.5 & \text{if } 0 \leq x \leq 0.5 \\ (2-x)/1.5 & \text{if } 0.5 \leq x \leq 2 \end{cases}$$

$$\mu_{N.HoiNhiều}(y) = \begin{cases} y & \text{if } 0 \leq y \leq 8 \\ 1+0.25(8-y) & \text{if } 8 \leq y \leq 10 \end{cases}$$

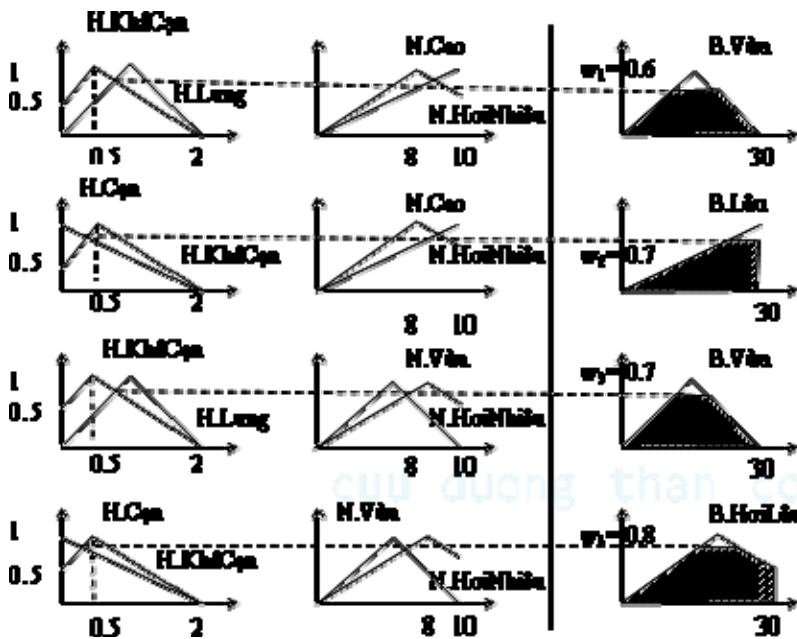


Tìm các trọng W_i cho từng luật (lấy min của các giao điểm)

Xây dựng hàm thành viên của kết luận

$$\mu_c(z) = W_1.B.Vừa(z) + W_2.B.Lâu(z) + W_3.B.Vừa(z) + W_4.B.HơiLâu(z)$$

Cuối cùng là giải mờ để tìm trị rõ z_0



Tóm lại : Muốn giải các bài toán mờ ta có các bước:

B1: Xác định các luật mờ của bài toán

B2: Xác định các hàm thành viên của các tập mờ có trong luật

B3: Tìm các trọng W_i của từng luật

B4: Nhập trị Input và tìm hàm thành viên cho kết luận

$$\mu_c(z) = \sum_i W_i \mu_{K1i}(z)$$

B5: Giải mờ để được giá trị rõ

❗Chú thích:

1. Hàm thành viên cho kết uận có thể tính bằng công thức:

$$a) \mu_c(z) = \sum_i W_i \mu_{K1i}(z) \quad \forall x \in E$$

$$b) \mu_c(z) = \sum_i \text{Min}(W_i, \mu_{K1i}(z)) \quad \forall x \in E$$

$$c) \mu_c(z) = \sum_i \text{Max}(\text{Min}(W_i, \mu_{K1i}(z))) \quad \forall x \in E$$

2. Giải mờ ta có thể áp dụng 1 trong 2 phương pháp sau:

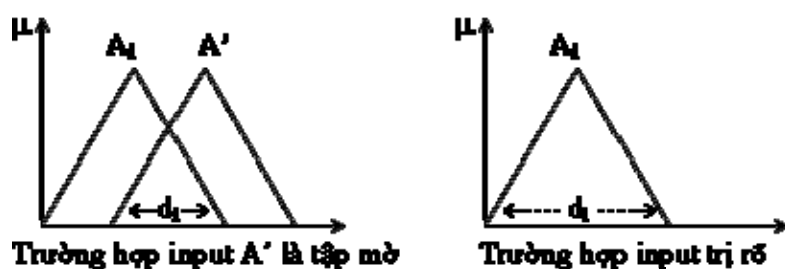
a) Tìm trọng tâm

b) Tìm trị trung bình

$$c) \text{Defuzzy}(z) = (\sum_i \alpha_i \cdot W_i) / \sum_i \alpha_i$$

Trong đó α_i là khoảng tin cậy của đồ thị của luật thứ I (fere strength)

W_i là trọng số của luật thứ i



TÀI LIỆU THAM KHẢO

[1].Bạch Hưng Khang, Hoàng Kiêm. *Trí tuệ nhân tạo, các phương pháp và ứng dụng*. Nhà xuất bản Khoa học và Kỹ thuật, 1989.

[2].Đỗ Trung Tuấn. *Trí tuệ nhân tạo*. Nhà xuất bản Giáo dục, 1998

[3].Đỗ Trung Tuấn. *Hệ chuyên gia*. Nhà xuất bản Giáo dục. 1999

[4].Đỗ Phúc. *Các đề án môn học Cơ sở Trí thức*. Khoa công nghệ thông tin - Đại học Khoa học tự nhiên, 1998.

[5].Rich Elaine. *Artificial Intelligence*. Addison Wesley, 1983.

[6].John Durkin. *Expert Systems-design and development*. Prentice Hall International, Inc, 1994.

[7].Adrian A. Hopgood, *Knowledge-based systems for engineers and scientists*. CRC Press, 1993.

[8].Stuart Russell & Peter Norvig. *Artificial Intelligence – a modern approach*. Prentice Hall, 1995

[9].Kurt Sundermeyer. *Knowledge based systems*. Wissenschafts Verlag, 1991.

[10].Mehmet R. Tolun & Saleh M. Abu-Soud. *An Inductive Learning Algorithm for Production Rule Discovery*. IEEE.

[11].Patrick Henry Winston. *Artificial Intelligence*. Addison Wesley, 1992.

cuu duong than cong. com