



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

GIÁO TRÌNH

CÁC HỆ CƠ SỞ TRI THỨC

Biên soạn: GS. TSKH. Hoàng Kiếm
TS. Đỗ Phúc
TS. Đỗ Văn Nhơn



NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

LỜI NÓI ĐẦU

Giáo trình này là một trong các giáo trình chính yếu của chuyên ngành Công nghệ thông tin. Giáo trình được xây dựng theo phương châm vừa đáp ứng yêu cầu chuẩn mực của sách giáo khoa, vừa có giá trị thực tiễn, đồng thời tăng cường khả năng tự học, tự nghiên cứu của sinh viên. Trên cơ sở đó, chúng tôi đã tham khảo nhiều tài liệu có giá trị của các tác giả trong và ngoài nước và đã sử dụng nhiều ví dụ lấy từ các ứng dụng thực tiễn.

Giáo trình này được dùng kèm giáo trình điện tử trên đĩa CD trong đó có thêm phần trình bày của giảng viên, các bài tập và phần đọc thêm nhằm đáp ứng tốt nhất cho việc tự học của sinh viên.

Chúng tôi rất mong nhận được các ý kiến đóng góp để giáo trình ngày càng hoàn thiện.

Nhóm biên soạn

1.1. MỞ ĐẦU

Các 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, 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: khối tri thức hay còn được gọi là *cơ sở tri thức*, 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 mỏi của người sử dụng là

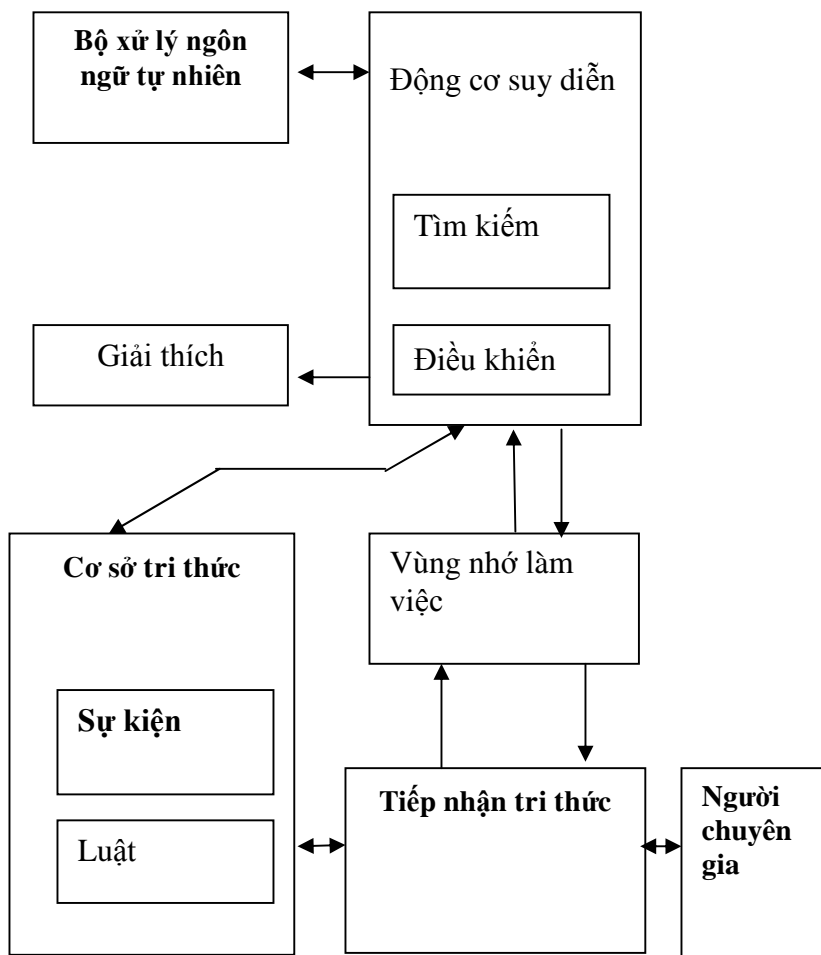
hệ CSTT sẽ cung cấp các sự kiện mới. Ngoài ra còn có cách điều khiển theo mục tiêu nhằm hướng đến các kết luận đã có và đi tìm các dẫn chứng để kiểm định tính đúng đắn của kết luận đó. Các kiểu suy diễn này sẽ được thảo luận chi tiết trong chương 3.

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 hồng học 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 hồng học 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

Bộ phận giải thích sẽ trả lời hai câu hỏi là WHY và HOW, câu hỏi WHY nhằm mục đích cung cấp các lý lẽ để thuyết phục người sử dụng đi theo con đường suy diễn của hệ chuyên gia. Câu hỏi HOW nhằm cung cấp các giải thích về con đường mà hệ chuyên gia sử dụng để mang lại kết quả.



Hình 1.1. Các thành phần của hệ chuyên gia

1.5. HỆ HỖ TRỢ RA QUYẾT ĐỊNH

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 tình 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... 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 quy 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 quy 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.

BIỂU DIỄN TRI THỨC

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

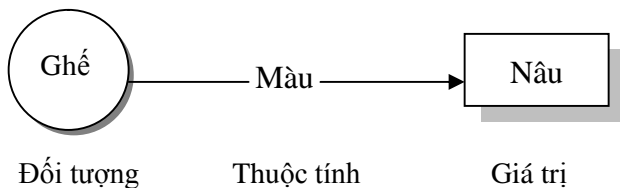
Phần này trình bày các kỹ thuật phổ biến nhất để biểu diễn tri thức, bao gồm:

- Bộ ba Đối tượng-Thuộc tính-Giá trị
- Các luật dẫn
- Mạng ngữ nghĩa
- Frame
- 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 2.1 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. Điển 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	Ho nhiều
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 có 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:

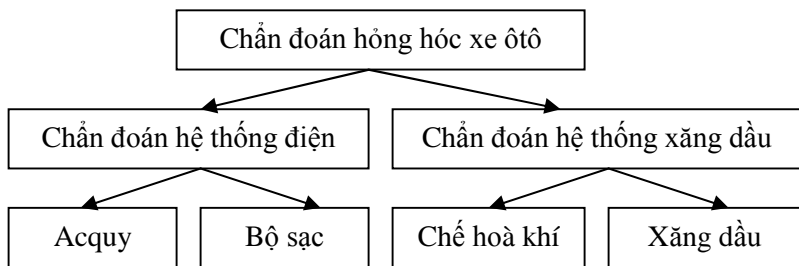
IF Lạm phát cao
THEN Lãi suất cao, CF = 0,8

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.

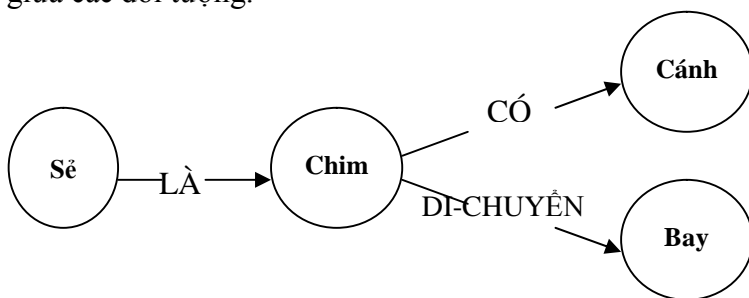


Hình 2.2. Tập các luật liên quan đến việc hỏng xe

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.

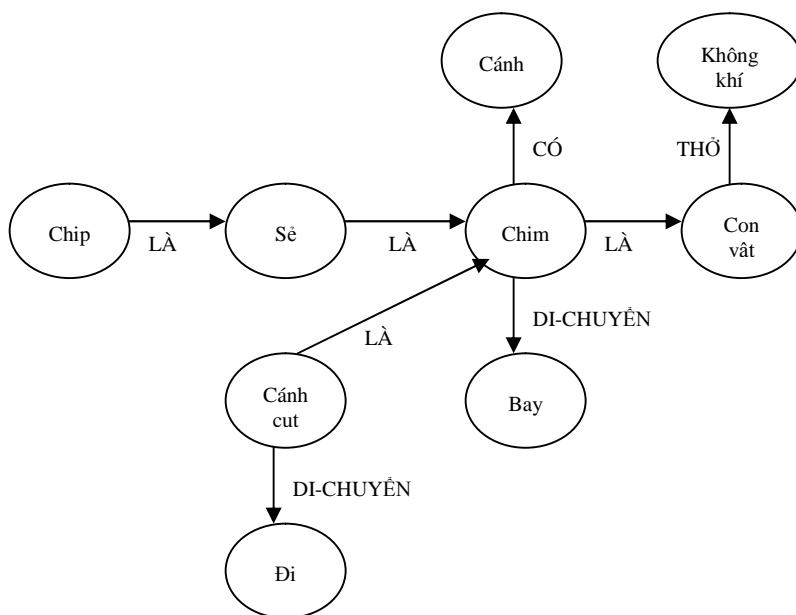


Hình 2.3. “Sẻ là Chim” thể hiện trên mạng ngữ nghĩa

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:

1. Thêm một đối tượng tương tự
2. Thêm một đối tượng đặc biệt hơn
3. Thêm một đối tượng tổng quát hơn

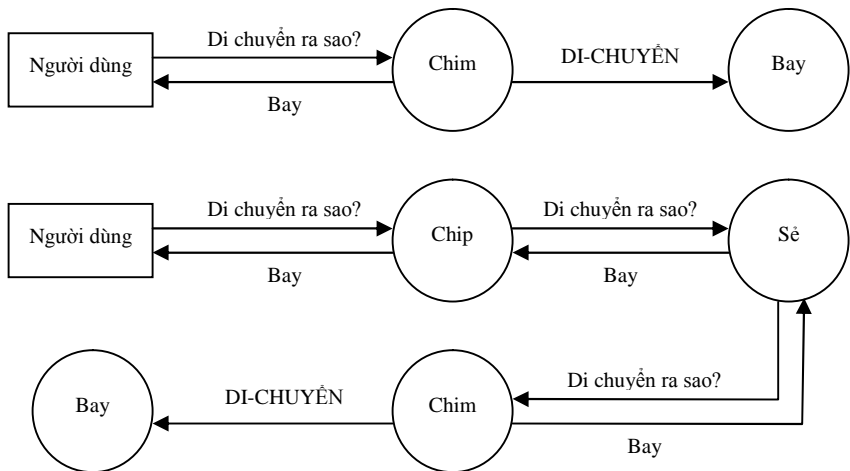
Thứ nhất, thêm “Cánh cụt” thể hiện một loại chim mới. *Thứ hai*, thêm “Chip” cũng có nghĩa nó là con “Sẻ” và đồng thời là “Chim”. *Thứ ba*, có thể đưa ra đối tượng tổng quát như “Con vật”. Lúc này, không những có thể biết được rằng “Chim là Con vật”, mà còn biết “Chip thở bằng không khí”.



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 style="width: 100%;" type="text"/> Địa chỉ: <input style="width: 100%;" type="text"/> <table style="width: 100%; margin-top: 10px;"> <tr> <th style="width: 50%; text-align: center;">Môn</th> <th style="width: 50%; text-align: center;">Điểm</th> </tr> <tr> <td>Vật lý</td> <td><input style="width: 100%;" type="text"/></td> </tr> <tr> <td>Toán</td> <td><input style="width: 100%;" type="text"/></td> </tr> <tr> <td>...</td> <td><input style="width: 100%;" type="text"/></td> </tr> </table>	Môn	Điểm	Vật lý	<input style="width: 100%;" type="text"/>	Toán	<input style="width: 100%;" type="text"/>	...	<input style="width: 100%;" type="text"/>	Tên frame: <input style="width: 100%;" type="text"/> Lớp: <input style="width: 100%;" type="text"/> Thuộc tính: <table style="width: 100%; margin-top: 10px;"> <tr> <td style="width: 50%;">Thuộc tính 1</td> <td style="width: 50%;">Giá trị 1</td> </tr> <tr> <td>Thuộc tính 2</td> <td>Giá trị 2</td> </tr> <tr> <td>...</td> <td>...</td> </tr> <tr> <td>...</td> <td>...</td> </tr> </table>	Thuộc tính 1	Giá trị 1	Thuộc tính 2	Giá trị 2
Môn	Điểm																
Vật lý	<input style="width: 100%;" type="text"/>																
Toán	<input style="width: 100%;" type="text"/>																
...	<input style="width: 100%;" type="text"/>																
Thuộc tính 1	Giá trị 1																
Thuộc tính 2	Giá trị 2																
...	...																
...	...																

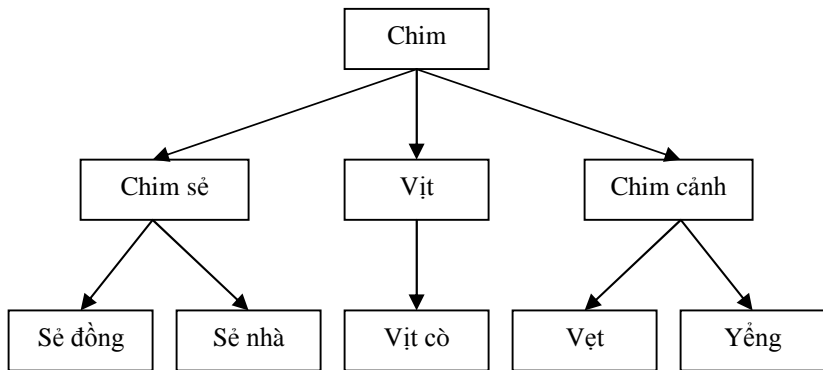
Hình 2.6. Cấu trúc frame

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**, **nhiệt 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 hai 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.

- c. **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 $X = \text{Nam}$, $Y = \text{Mai}$, $Y = \text{Cúc}$. Trong phép toán vị từ người ta dùng biến như đối số của biểu thức vị từ hay của hàm.

- d. **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:

$\text{cha}(\text{sơn}) = \text{dũng}$

$\text{mẹ}(\text{sơn}) = \text{an}$

- e. **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

3.1. 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.

Tiền 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 : “siết quạt lỏng thường gây ra tiếng ồn”. Kết luận này có được do kinh nghiệm sửa nhiều xe hơi. Người ta đoán ngay “siết 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 đang 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 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.

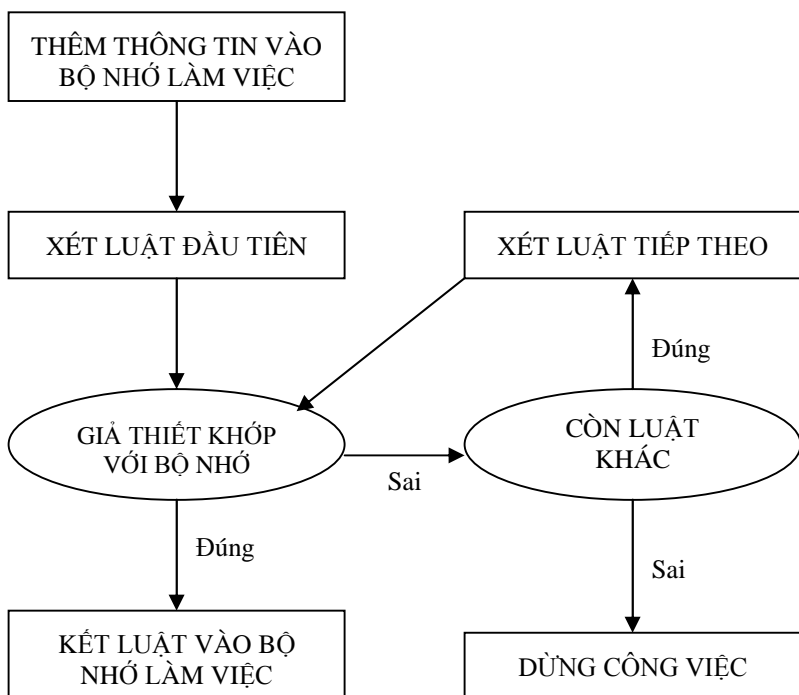
Suy diễn có mặt trái là khi các dữ liệu thừa cứ sinh ra có thể càng tiếp tục suy diễn càng không đi đến trạng thái

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 như 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	Thân nhiệt bệnh nhân quá 37 độ
THEN	Bệnh nhân bị sốt

Luật 3.

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

Thông tin từ bệnh nhân là:

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

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.

Vòng 1

Bộ nhớ làm việc

Luật 1, giả thuyết 1,

Rát họng ←

Đúng

**Thân nhiệt 39 độ,
ốm hai tuần, rát**

Luật 2, giả thuyết 2,

Viêm nhiễm ←

Không rõ

**Thân nhiệt 39 độ,
ốm hai tuần, rát**

Luật 2, giả thuyết 1,

Thân nhiệt quá 37 ————— Đúng ————— Thân nhiệt 39 độ,
Thực hiện luật 2 ồm hai tuần, rất
Bệnh nhân sốt ————— Kết luận ————— họng,bệnh nhân sốt

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

Luật 3, giả thuyết 1,

Ôm trên một tuần ←———— Đúng ————— Thân nhiệt 39 độ,
ôm hai tuần, rất
họng,bệnh nhân sốt

Luật 3, giả thuyết 2,

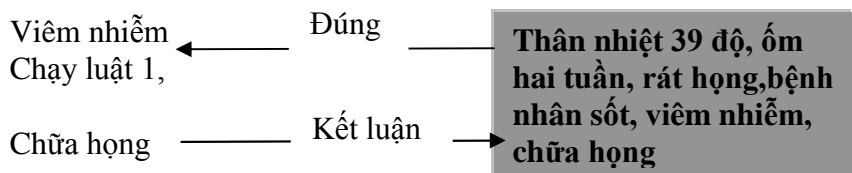
Bệnh nhân sốt ←———— Đúng ————— Thân nhiệt 39 độ,
Chạy luật 3 ồm hai tuần, rất
Viêm nhiễm ————— Đúng ————— họng,bệnh nhân sốt,
viêm nhiễm

Vòng 3

Luật 1, Giả thuyết ,

Rất họng ←———— Đúng ————— Thân nhiệt 39 độ, ồm
hai tuần, rất
họng,bệnh nhân sốt,
viêm nhiễm

Luật 1, giả thuyết 2,



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

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

1. Bệnh nhân sốt
2. Nghi viêm nhiễm
3. 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 ra 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 luật 4 và luật 5 chạy. 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, nhờ các luật khác.

Quá trình này tiếp tục đệ quy 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 thích 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.

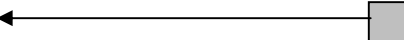
IF	Họng bệnh nhân đỏ
THEN	Có dấu hiệu viêm nhiễm.

Luật 3

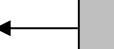
IF	Cơ quan bị thương tổn
-----------	-----------------------

AND	Có khuẩn cầu
AND	Có hạt
THEN	Chắc chắn cơ quan nội tạng bị viêm.

Bước 1. Đích : Bệnh nhân bị viêm hong

Bước 2. Đã thấy đích : không ← 

Bước 3. Tìm đích trong phần **THEN** của luật 1 ← 

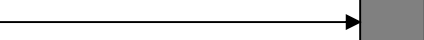
Bước 4. Xem luật 1. Phần giả thiết ”dấu hiệu viêm”? ← 

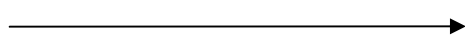
Bước 5. Tìm thấy các luật có giả thiết như phần **THEN** của luật 2?


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:

Hỏi: Họng có đỏ không? → 

Trả lời: Đúng ! → 

Luật 2: được chạy, do vậy “có dấu hiệu nhiễm khuẩn” ← 

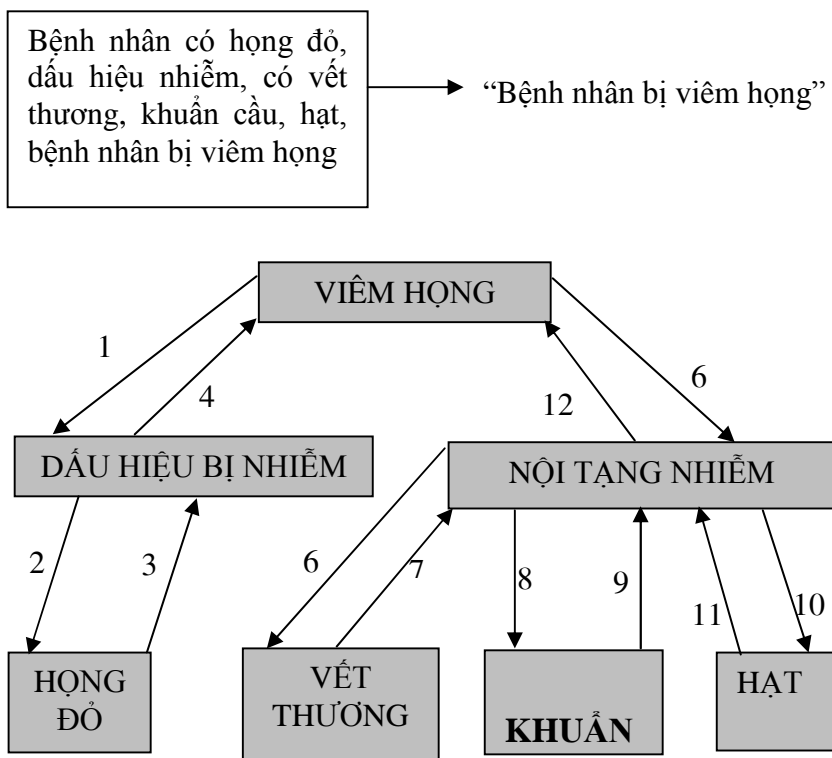
Bước 9. Xem luật 1, giả thiết 2 “có nội tạng viêm nhiễm” không?

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 ba 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 chạy.

Bước 12. Luật 1 chạy, 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 chúng 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 các 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 liên quan 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: Hỏng 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 quy 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 Hồng phần in.

R4: NẾU KHÔNG Khởi động được VÀ KHÔNG Thông báo VÀ Có âm thanh THÌ Hồng RAM.

R5: NẾU KHÔNG Khởi động được VÀ Thông báo đĩa THÌ Hồng 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 khóa bắt đầu của phần giả thiết.
- THÌ: Từ khóa 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 (edge). Một luật được biểu diễn dưới dạng:

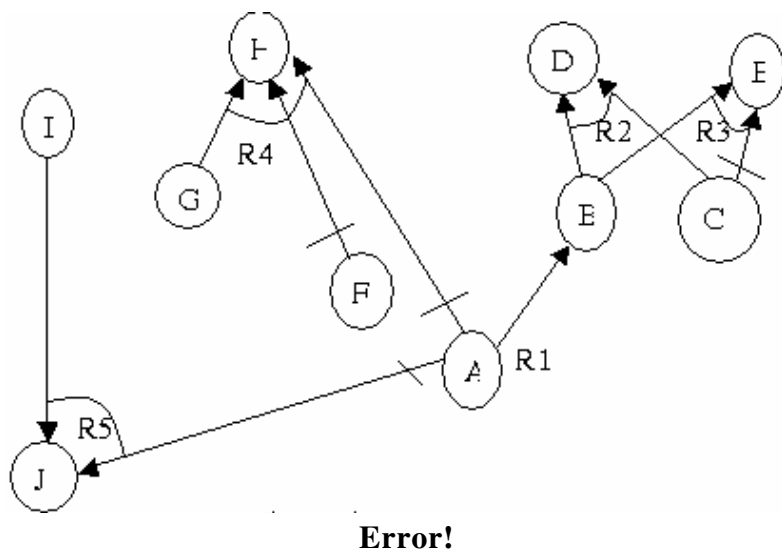
VỀ TRÁI (Giả thiết)	→	VỀ PHẢI (Kết luận)
R1: A	→	B
R2: B^C	→	D
R3: B^ (~ C)	→	E
R4: (~A)^ (~F)^G	→	H
R5: (~A)^I	→	J

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    Giatri;
}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: $(\sim A) \wedge (\sim F) \rightarrow H$.

STT nút	Tên nút	Vế trái (vt)	Vế 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	~A	1	0	16
17	~B	0	0	17
18	~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

❖ 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.*

Đề 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 vế TRÁI, hoặc chỉ ở vế PHẢI, đồng thời có những nút xuất hiện trong cả hai vế. 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 ba tập nút khác nhau (Loại 1,2,3):

Tập Nút kết luận = VẾ PHẢI \ VẾ TRÁI = { D,E,H,J } = { Các nút chỉ có trong VẾ PHẢI }

Tập Nút trung gian = VẾ TRÁI \cap 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 sự kiện	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} = {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 một trong bốn sự cố nằm trong tập Nút kết luận trên:

D: Không hồng

E: Hồng phản in

H: Hồng RAM

J: Hồng 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 hồng gì. Hệ chuyên gia sẽ phỏng vấn chúng ta và cố gắng đưa sự cố hư hỏng của máy chúng ta vào một trong bốn trường hợp trên. Khi đã có một trong bốn 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 bốn 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) $\ni \{ D, E, H, J, \text{“Không giải đáp được”} \}$

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 các nút trong vế trái của Luật

+ Nếu một nút đã có giá trị thì

$giá trị luật = giá trị luật \text{ AND } giá trị nút$

+ 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

$giá trị luật = giá trị luật \text{ AND } giá trị nút$

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)	(1)
--------------------	------------	------------	------------

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 hai 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, “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 vé 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) $\ni \{D, E, H, J, \text{“Không giải đáp được”}\}$

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
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
giátriluật = giátriluật AND giátrínút
- + 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
giátriluật = giátriluật AND giátrínút
Câu hỏi Why? Xuất hiện
Gọi thủ tục Giải thích Why 0;

Hết các nút trong về trái của Luật

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

+ Nếu giá trị của nút đang xét = 1

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

Hết tập Luật

Chương 4

HỆ HỖ TRỢ 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 bao gồm:

- 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

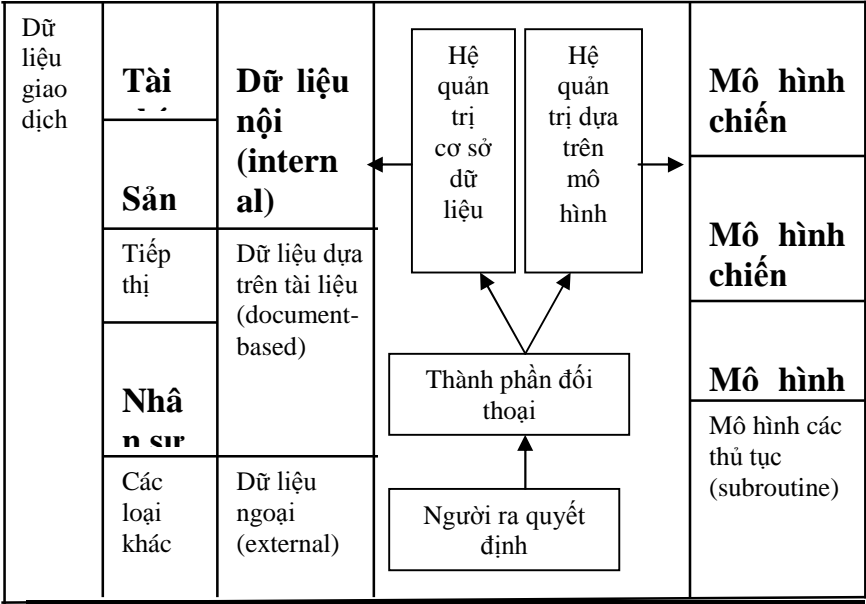
Tên	Lĩnh vực ứng dụng
GADS Geodata Analysis Display System	Phân tích và cung cấp tài nguyên địa lý.
PMS Portfolio Management System	Tư vấn và quản trị đầu tư
IRIS Industrial Relations Information	Phân tích chất lượng và bố trí nhân lực trong sản xuất
PROJECTOR	Hoạch định kế hoạch tài chính
IFPS Interactive 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 trở 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 tất định (deterministic). Các mô hình tiền 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

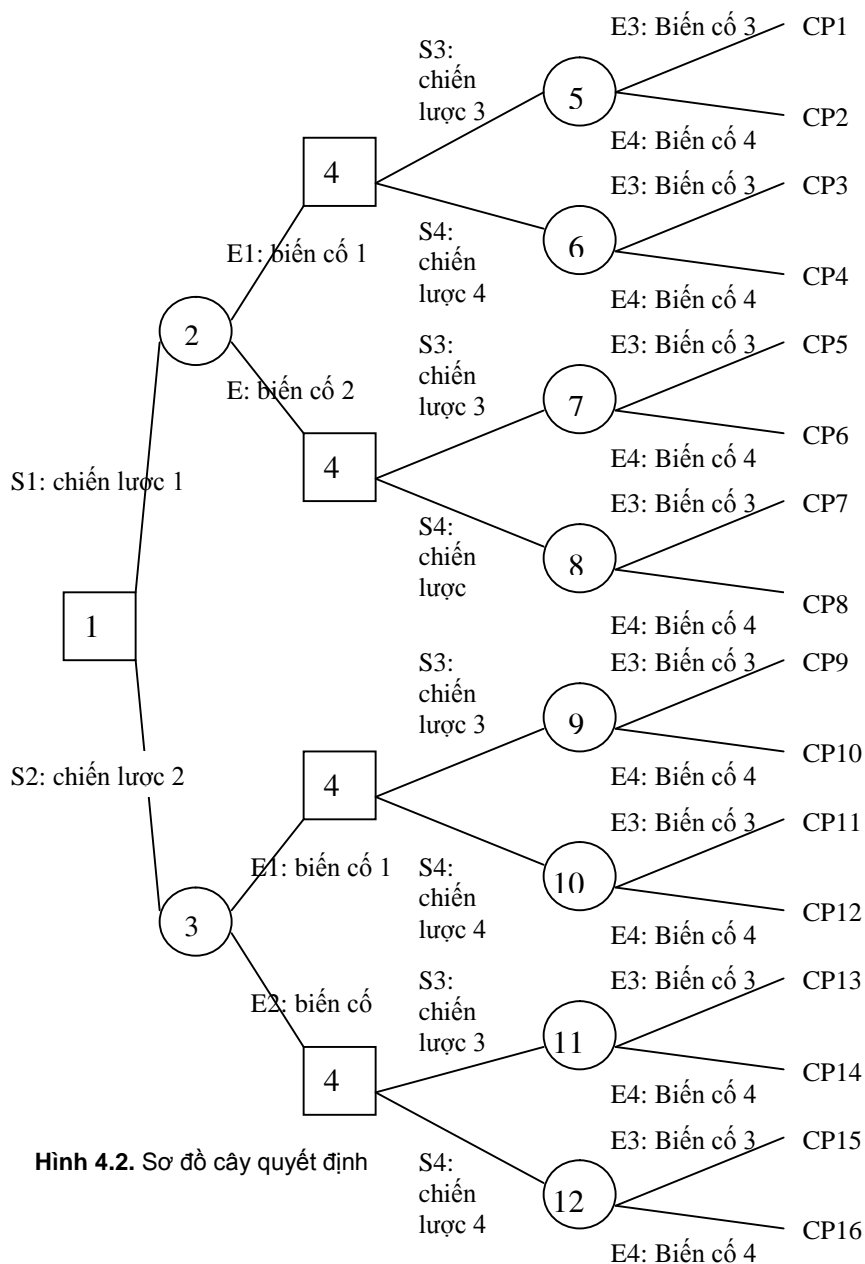
trong cơ sở dữ liệu nên có thể làm dữ liệu nhập cho các mô hình khác, do đó có thể tích hợp nhiều mô hình lại với nhau.

- 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 biên 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 biên 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.



Hình 4.2. Sơ đồ cây quyết định

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.

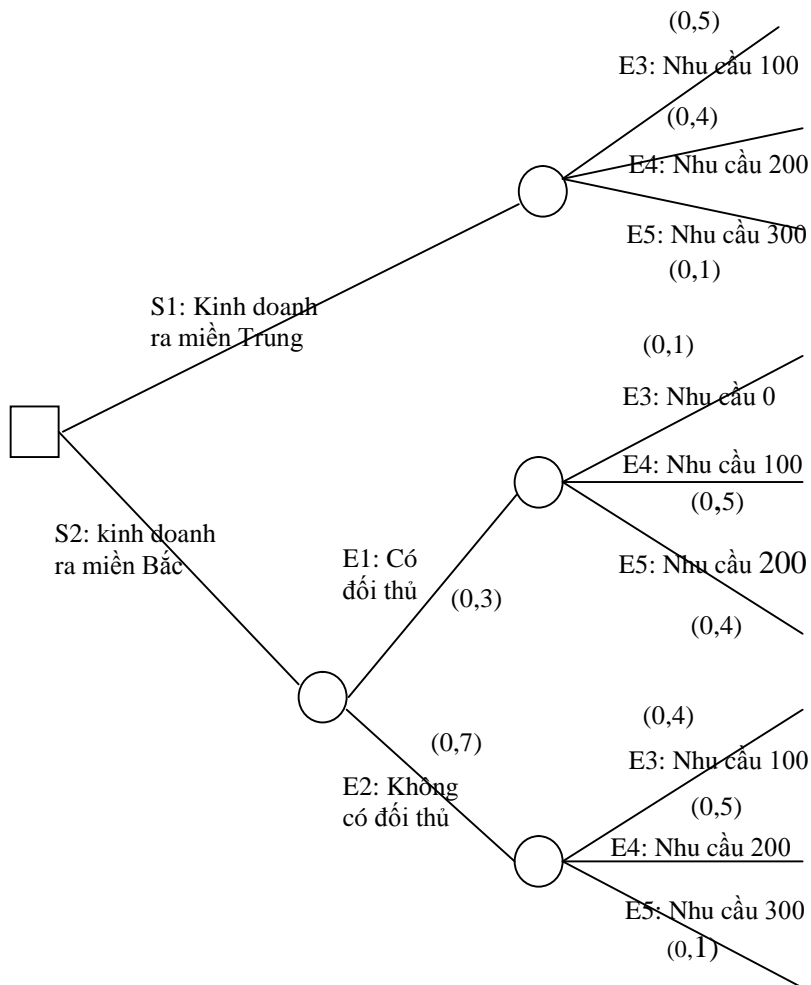
4.4.2. Suy diễn trên cây quyết định

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.

$200 \times 3.000.000 = 600.000.000$ đ. Giá bán dự kiến cho mỗi bộ là 5.000.000 đ, chúng ta có kết cuộc CP1 và CP2 tương ứng:

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

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



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:

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

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

Do vậy:

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

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.

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 luận 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 phần 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:

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

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

Độ đo chắc chắn **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 **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)

$$\text{Thì } \mathbf{CF(e1 \text{ OR } e2) = MAX(CF(e1), CF(e2))}$$

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

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

$$\text{Thì } \mathbf{CF((e1 \text{ AND } e2) \text{ OR } e3) = MAX(MIN(CF(e1), CF(e2)), CF(e3))}$$

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

$$\mathbf{CF(NOT e) = - CF(e)}$$

Sau khi tính được độ đo chắc chắn của chứng cứ 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 hai 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:

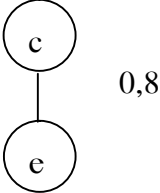
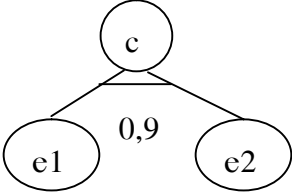
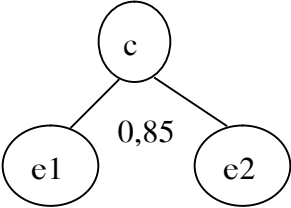
$$\text{Ctổng} = (\text{CF}(t1) + \text{CF}(t2)) / (1 - \text{MIN}(\text{ABS}(\text{CF}(t1)), \text{ABS}(\text{CF}(t2))))$$

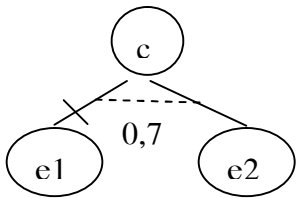
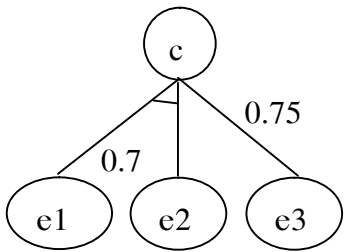
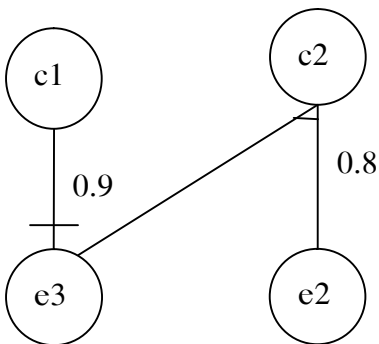
5.3. CHUỖI LẬP LUẬN

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) CF(r) = 0,8</p>
<p>b. Suy diễn có AND</p> 	<p>If (e1 AND e2) Then (c) CF(r) = 0,9</p>
<p>c. Suy diễn OR</p> 	<p>If (e1 OR e2) Then (c) CF(R) = 0.85</p>

<p>d. Suy diễn có NOT</p> 	<p>If ((NOT e1) OR Then (c2) $CF(r) = 0.7$</p>
<p>e. Nhiều luật cho cùng kết luận</p> 	<p>If (e1 AND e2) Then (c) $CF(r1) = 0.7$ If (e3) Then (c) $CF(r2) = 0.75$</p>
<p>f. Một chứng cứ được dùng trong hai luật</p> 	<p>If (NOT (e1) Then (c1) $CF(r1) = 0.9$ If (e1 AND e2) Then (c2) $CF(r2) = 0.8$</p>

Ví dụ: chúng ta có bảy 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

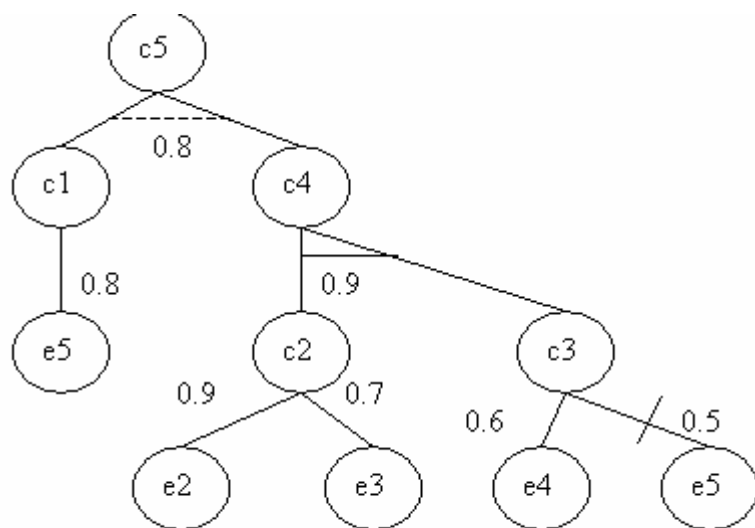
r6: If (c2 AND c3) Then (c4)

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:

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

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

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

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

$$CF(e5) = -0,3 \quad \text{Hãy tính } CF(c5)$$

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)**:

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

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

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

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

Do **CF(c2)** của **r2** trái dấu với **CF(c2)** của **r3**, nên:

$$\mathbf{CF(c2)}_{\text{tổng}} = (0,81 + (-0,21)) / (1 - \mathbf{MIN}(0,81, 0,21)) = 0,74$$

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

Với luật **r4**:

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

Với luật **r5**:

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

Do **CF(c3)** của **r4** và **CF(c3)** của **r5** cùng dương nên $\mathbf{CF(c3)}_{\text{tổng}} = 0,24 + 0,15 = 0,39$

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

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

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

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

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