Tác giả: Lê Anh Triều (18521536), Nguyễn Hoàng Thiên (18521427), Đặng Nguyễn Anh Tuấn (18521597)

Xây dựng hệ chuyên gia

Hệ thống tư vấn nhu cầu mua sắm máy tính xách tay

(Tháng 7 năm 2020)

***Tóm tắt* — Dựa vào lượng kiến thức đã học ở trường qua môn trí tuệ nhân tạo cùng với nhận thấy nhu cầu mua sắm máy tính xách tay của con người trong thời đại công nghệ hiện nay ngày càng cao. Chúng tôi quyết định tiến hành xây dựng một hệ thông tư vấn tự động, giúp cho người có nhu cầu mua sắm có thể mau chóng tìm ra những mẫu máy tính thích hợp cho công việc hằng ngày của mình. Để làm được điều này, xây dựng một hệ chuyên gia sẽ là phương pháp chúng tôi sử dụng để tạo nên chương trình. Những kiến thức sử dụng để xây dựng cơ sở luật trong đồ án này được chúng tôi thu thập chủ yếu từ những cửa hàng điện tử lớn như Phong Vũ, Thế giới di động, FPT Shop, Gearvn cùng với vốn hiểu biết, kinh nghiệm của bản thân. Bên cạnh những khó khăn đòi hỏi phải giải quyết được một kho tri thức đủ lớn thì chương trình cũng cần có một chương trình chuyên dụng để xây dựng nên, vì thế chúng tôi chọn ngôn ngữ Prolog, một ngôn ngữ đủ mạnh để xây dựng bất kì một chương trình thuộc lĩnh vực trí tuệ nhân tạo nào.**

# **CHƯƠNG I. GIỚI THIỆU BÀI TOÁN**

Trong thời đại ngày nay, khoa học công nghệ ngày càng phát triển nhanh chóng làm thay đổi và ảnh hưởng rất nhiều đến cuộc sống hằng ngày của chúng ta. Kể từ khi máy tính bàn được ra đời, những hoạt đồng thường nhật của con người như làm việc, học tập, giải trí hay các hoạt động chuyên môn khác đều được những chiếc máy tính này đáp ứng như cầu một cách dễ dàng, thuận tiện. Tuy nhiên máy tính để bàn vẫn là một thiết bị to với nhiều phần cứng khác nhau và rất thụ động. Từ những nhược điểm đó, những chiếc máy tính xách tay với thiết kế gọn nhẹ hơn, linh động và thậm chí giá thành rẻ hơn đã được ra đời, khắc phục hoàn toàn những nhược điểm mà máy tính để bàn để lại. Từ những ưu điểm đó, máy tính xách tay dần trở thành một thiết bị tiện dụng, thiết yếu bởi sự cơ động và linh hoạt đối với bất cứ ai đang trong quá trình học tập và làm việc.

Tuy nhiên, với thời buổi công nghệ ngày càng phát triển hiện nay thì càng ngày càng có nhiễu mẫu mã máy tính xách tay với nhiều hãng sản xuất khác nhau được ra đời. Nhiều mẫu mã mới thường được các hãng điện tử nâng cấp và phát triển dựa trên những khuyết điểm nhỏ của các mẫu mã được ra mắt trước đó, hay tạo ra những dòng máy cùng phân khúc nhưng tiết giảm một vài chi tiết nhằm giảm giá thành, cạnh tranh với các hãng đối thủ. Điiều này dẫn đến việc dường như có quá nhiều lựa chọn cho các dòng máy có cùng phân khúc mà những người tiêu dùng thông thường sẽ không có khả năng phân biệt, khiến cho họ khó đưa ra lựa chọn tuy đã có đủ kiến thức về thông số kỹ thuật của máy tính và dễ dẫn đến việc mua nhầm chiếc máy không phù hợp.

Nhằm để giải quyết những vấn đề cấp thiết đó, chúng tôi quyết định tiền hành xây dựng hệ chuyên gia, tạo ra một chương trình, hệ thống tư vấn người tiêu dùng giúp cho họ nhanh chóng có thể lựa chọn được những mẫu mã máy tính xách tay phù hợp với nhu cầu cũng như khả năng tài chính của họ. Tuy nhiên, kho tri thức sử dụng để tạo nên cơ sở luật của hệ chuyên gia này chúng tôi thu thập dựa trên những kiến thức, kinh nghiệm của bản thân và nhu cầu của đại đa số người dùng phổ thông hiện nay, nên hệ thống tư vấn này có thể vẫn đưa ra những kết quả không hoàn toàn chính xác đối với một số nhu cầu sử dụng của cá nhân khác.

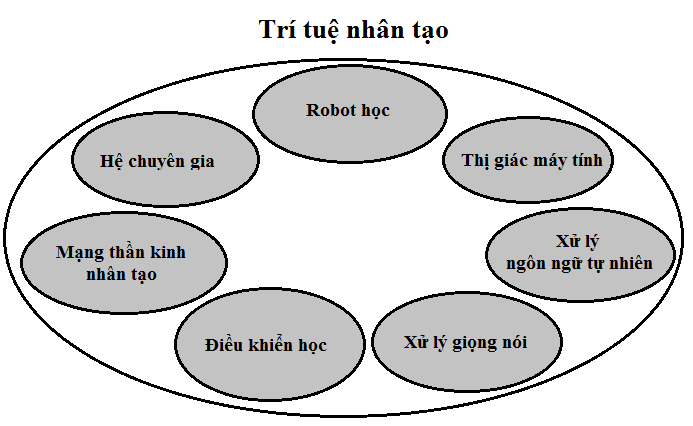
# **CHƯƠNG II. CƠ SỞ LÝ THUYẾT**

## **1. Hệ chuyên gia.**

### **1.1. Khái niệm hệ chuyên gia.**

Theo E.Feigenbaum: “Hệ chuyên gia (Expert System) là một chương trình máy tính thông minh sử dụng tri thức (knowledge) và các thủ tục suy luận (inference procedures) để giải những bài toán tương đối khó khăn đòi hỏi những chuyên giả mới giải được”.

Hệ chuyên gia là một hệ thống tin học có thể mô phỏng năng lực quyết đoán và hành động của một chuyên gia, con người. Hệ chuyên gia là một nhánh trong lĩnh vực ứng dụng của trí tuệ nhân tạo (Artificial Intelligence) và nó sử dụng tri thức của những chuyên gia để giải quyết các vấn đề, bài toán khác nhau trong mọi lĩnh vực.



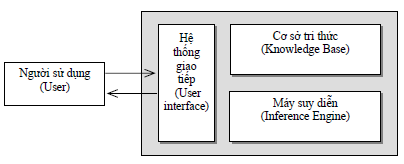
*Một số lĩnh vực ứng dụng của trí tuệ nhân tạo.*

Tri thức (knowledge) trong một hệ chuyên gia phản ánh sự tinh thông được tích tụ từ sách vở, tạp chí, từ các nhà chuyên gia hay các nhà bác học trong một lĩnh vực nhất định. Các thuật ngữ hệ chuyên gia, hệ thống dựa trên tri thức (knowledge – based sytem) hay hệ chuyên gia dựa trên tri thức (knowledge – based expert system) thường có cùng nghĩa.

Một hệ chuyên gia gồm ba thành phần cấu thành chính đó chính là cơ sở tri thức (knowledge base), máy suy diễn hay môtơ suy diễn (inference engine), và hệ thống giao tiếp với người sử dụng (user interface). Cơ sở tri thức chứa các tri thức để từ đó máy suy diễn có thể tạo ra những câu trả lời phù hợp, thoả đáng cho người sử dụng qua hệ thống giao tiếp.

Người sử dụng (user) cũng cấp những sự kiện là những gì đã biết, đã có thật hay những thông tin có ích cho hệ chuyên gia và nhận lại được những câu trả lời là những lời khuyên hay những lời gợi ý thích hợp, đúng đắn.

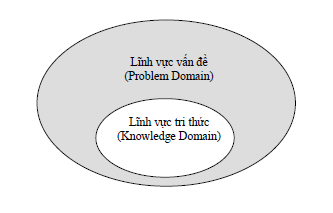
Hoạt động của một hệ chuyên gia dựa trên tri thức được minh hoạ như sau:



*Mô hình hoạt động tổng quát của hệ chuyên gia.*

Mỗi hệ chuyên gia chỉ đặc trưng cho một lĩnh vực nhất định hoặc một vấn đề nhất định nào đó như trong y học, luật pháp, tài chính, khoa học và công nghệ,… mà không phải cho bất cứ một lĩnh vực, vấn đề hỗn hợp nào.

Những tri thức chuyên gia dùng để giải quyết một vấn đề đặc trưng được gọi là lĩnh vực tri thức (knowledge domain). Lĩnh vực tri thức hoàn toàn nằm trong lĩnh vực vấn đề. Phần bên ngoài lĩnh vực tri thức nói lên rằng tri thức không dùng để giải quyết cho tất cả mọi vấn đề.

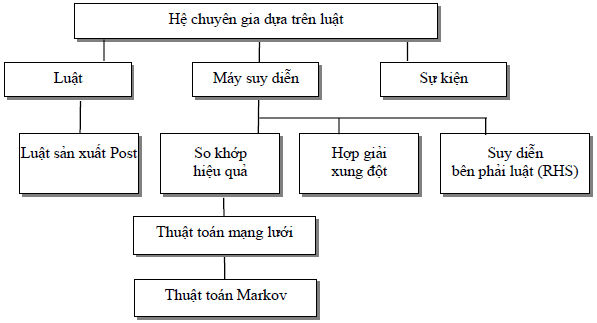


*Quan hệ giữa lĩnh vực vấn đề và lĩnh vực tri thức.*

### **1.2. Kỹ thuật suy luận trong các hệ chuyên gia**.

Có nhiều phương pháp tổng quát để suy luận trong các chiến lược giải quyết vấn đề của hệ chuyên gia. Những phương pháp hay gặp là suy diễn tiến (foward chaining), suy diễn lùi (backward chaining) và cả phối hợp hai phương pháp này. Ngoài ra còn có những phương pháp khác là phân tích phương tiện, rút gọn vấn đề, quay lui, kiểm tra lập kế hoạch, lập kế hoạch phân cấp...

Dưới đây là nền tảng của công nghệ hệ chuyên gia hiện đại (foundation of modern rele- based expert system).



*Nền tảng của công nghệ hệ chuyên gia dựa trên luật hiện đại.*

### **1.3. Phương pháp suy diễn tiến**.

Suy diễn tiến ( forward charning) là lập luận từ các sự kiện, sự việc để rút ra các kết luận.

Trong phương pháp này, người sử dụng cung cấp các sự kiện cho hệ chuyên gia để hệ thống (máy suy diễn) tìm cách rút ra các kết luận có thể. Kết luận được xem là những thuộc tính có thể được gán giá trị. Trong số những kết luận này, có thể có những kết luận làm ngườisử dụng quan tâm, một số khác không nói lên điều gì, một số khác có thể vắng mặt.Các sự kiện thường có dạng : Atthibute = value

Lần lượt các sự kiện trong cơ sở tri thức được chọn và hệ thống xem xét tất cả các luật mà các sự kiện này xuất hiện như là tiền đề. Theo nguyên tắc lập luận trên, hệ thống sẽ lấy ra những luật thoã mãn. Sau khi gán giá trị cho các thuộc tính thuộc kết luận tương ứng, ngườita nói rằng các sự kiện đã được thoã mãn. Các thuộc tính được gán giá trị sẽ là một phần của kết quả chuyên gia. Sau khi mọi sự kiện đã được xem xét, kết quả được xuất ra cho người sử dụng.

### **1.4. Phương pháp suy diễn lùi**.

Phương pháp suy diễn lùi tiến hành các lập luận theo chiều ngược lại (đối với phương pháp suy diễn tiến). Từ một giả thuyết (như là một kết luận), hệ thống đưa ra một tình huống trả lời gồm các sự kiện là cơ sở của giả thuyết đã cho này.

Suy diễn lùi là cho phép nhận được giá trị của một thuộc tính. Đó là câu trả lời cho câu hỏi « giá trị của thuộc tính A là bao nhiêu ? » với A là một đích (goal).

Để xác định giá trị của A, cần có các nguồn thông tin. Những nguồn này có thể là những câu hỏi hoặc có thể là những luật. Căn cứ vào các câu hỏi, hệ thống nhận được một cách trực tiếp từ người sử dụng những giá trị của thuộc tính liên quan. Căn cứ vào các luật, hệ thống suy diễn có thể tìm ra giá trị sẽ là kết luận của một trong số các kết luận có thể của thuộc tính liên quan, v.v...

Ý tưởng của thuật toán suy diễn lùi như sau. Với mỗi thuộc tính đã cho, người ta định nghĩa nguồn của nó:

- Nếu thuộc tính xuất hiện như là tiền đề của một luật (phần đầu của luật), thì nguồn sẽ thu gọn thành một câu hỏi.

- Nếu thuộc tính xuất hiện như là hậu quả của một luật (phần cuối của luật), thì nguồn sẽ là các luật mà trong đó, thuộc tính là kết luận.

- Nếu thuộc tính là trung gian, xuất hiện đồng thời như là tiền đề và như là kết luận, khi đó nguồn có thể là các luật, hoặc có thể là các câu hỏi mà chưa được nêu ra.

Nếu mỗi lần với câu hỏi đã cho, người sử dụng trả lời hợp lệ, giá trị trả lời này sẽ được gán cho thuộc tính và xem như thành công. Nếu nguồn là các luật, hệ thống sẽ lấy lần lượt các luật mà thuộc tính đích xuất hiện như kết luận, để có thể tìm giá trị các thuộc tính thuộc tiền đề. Nếu các luật thoã mãn, thuộc tính kết luận sẽ được ghi nhận.

### **1.5. Đặc trưng và ưu điểm của hệ chuyên gia**.

Có bốn đặc trưng cơ bản của một hệ chuyên gia:

* Hiệu quả cao (high performance): khả năng trả lời với mực độ tinh thông bằng hoặc là cao hơn cả so với chuyên gia trong cùng lĩnh vực.
* Thời gian trả lời thoả đáng (adequate respone time): thời gian đưa ra câu trả lời thích hợp, hợp lý bằng hoặc nhanh hơn so với chuyên gia để đi đến cùng một quyết định. Hệ chuyên gia là một hệ thống thời gian thực.
* Độ tin cậy cao (good reliability): không thể xảy ra sự cố hoặc giảm sút độ tin cây khi sử dụng.
* Dễ hiểu (understandable): hệ chuyên gia giải thích các bước suy luận một cách dễ hiểu và nhất quán, câu trả lời không mang tính chất hàn lâm.

Những ưu điểm nổi bật của hệ chuyên gia:

* Tính phổ cập (increased availability): là một sản phẩm chuyên gia, được phát triển không ngừng với hiệu quả sử dụng cao không thể phủ nhận.
* Giảm giá thành (reduced cost): có thể tái sử dụng nhiều lần thay vì tìm đến những chuyên gia trong lĩnh vực.
* Giảm thiểu rủi ro (reduced dangers): giúp con người tránh được các trường hợp trải nghiệm thực tế trong các môi trường rủi ro, nguy hiểm.
* Tính thường trực (permanance): bất kể lúc nào cũng có thể khai thác sử dụng ngay cả khi con người đang trong trạng thái mệt mỏi, đang nghỉ ngơi hay vắng mặt.
* Đa lĩnh vực (multiple expertise): chuyên gia về nhiều lĩnh vực khác nhau và được khai thác đồng thời bất kể thời gian sử dụng.
* Độ tin cậy (increased reliability): luôn đảm bảo được độ tin cậy khi khai thác.
* Khả năng giảng giải (explanation): câu trả lời với mức độ tinh thông được giải thích rõ ràng, chi tiết, dễ hiểu.
* Khả năng trả lời (fast response): trả lời theo thời gian thực, khách quan.
* Tính ổn định, suy luận có lý và đầy đủ mọi lúc mọi nơi
* Trợ giúp thông minh như một người hướng dẫn.
* Có thể truy cập như một cơ sở dữ liệu thông minh.

## **2. Các mô hình biểu diễn cơ sở tri thức.**

Biểu diễn tri thức là một lĩnh vực nghiên cứu quan trọng của trí tuệ nhân tạo. Chúng nhằm phát triển các phương pháp, cách thức biểu diễn tri thức và các công cụ dùng để hỗ trợ cho việc biểu diễn tri thức.

Có rất nhiều phương pháp biểu diễn tri thức và phải phụ thuộc vào tri thức cụ thể của từng bài toán khác nhau ta mới chọn được một phương pháp biểu diễn hợp lý nhất. Nhưng thông thường người ta sử dụng các cách sau đây:

* Biểu diễn tri thức bằng logic mệnh đề.
* Biễu diễn tri thức bằng luật sản xuất.
* Biểu diễn tri thức bằng mạng ngữ nghĩa.
* Biểu diễn tri thức bằng khung.

### **2.1. Biểu diễn tri thức bằng logic mệnh đề.**

Logic mệnh đề là một công cụ toán logic, trong đó các mệnh đề được mã hoá cho một biến, hoặc một hằng, còn các biểu thức là sự liên kết có nghĩa giữa các biếng hằng với một số toán tử nhất định.

Tri thức được mô tả dưới dạng các mệnh đề trong ngôn ngữ biểu diễn tri thức. Mỗi câu có thể xem như sự mã hoá một sự hiểu biết của ta về thế giới thực. Ngôn ngữ biểu diễn tri thức (cũng như mọi ngôn ngữ hình thức khác) gồm hai thành phần cơ bản là cú pháp và ngữ nghĩa.

#### **2.1.1. *Cú pháp.***

Cú pháp của logic mệnh đề rất đơn giản, nó cho phép xây dựng nên các công thức. Cú pháp của logic mệnh đề bao gồm tập các ký hiệu và tập các luật xây dựng công thức.

**\*Các ký hiệu:**

Hai hằng logic True và False.

Các ký hiệu mệnh đề (còn được gọi là các biến mệnh đề): P, Q, R,…

Các kết nối logic: ∧, ∨, ˥, ⇒, ⇔.

Các dấu mở ngoặc ( và đóng ngoặc ).

**\*Các quy tắc xây dựng công thức:**

Các biến mệnh đề là công thức.

Nếu A và B là công thức thì:

(A∧B) đọc “A hội B” hoặc “A và B”)

(A∨B) đọc “A tuyển B”hoặc “A hoặc B”.

(˥A) đọc “phủ định A”).

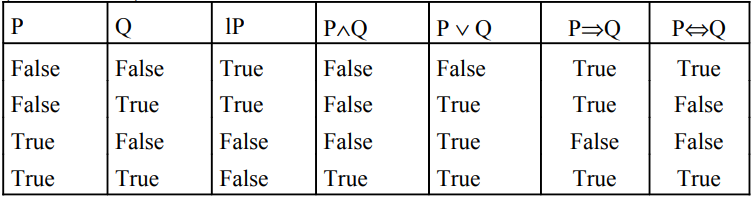
A⇒B) đọc “A kéo theo B”hoặc “nếu A thì B”.

(A⇔B) đọc “A và B kéo theo nhau”.

#### **2.1.2. *Ngữ nghĩa.***

Ngữ nghĩa của logic mệnh đề cho phép ta xác định ý nghĩa của các công thức trong thế giới hiện thực nào đó. Điều đó được thực hiện bằng cách kết hợp mỗi ký hiệu mệnh đề với sự kiện nào đó trong thế giới hiện thực. Chẳng hạn ký hiệu mệnh đề P có thể ứng với sự kiện “Paris là thủ đô nước Pháp” hoặc bất kỳ một sự kiện nào khác. Bất kỳ một sự kết hợp các kí hiệu mệnh đề với các sự kiện trong thế giới thực được gọi là một minh họa (interpretation). Chẳng hạn minh họa của kí hiệu mệnh đề P có thể là một sự kiện (mệnh đề) “Paris là thủ đô nước Pháp ”. Một sự kiện chỉ có thể đúng hoặc sai. Chẳng hạn, sự kiện “Paris là thủ đô nước Pháp “là đúng, còn sự kiện “Số Pi là số hữu tỉ “là sai.

Một cách chính xác hơn, ta hiểu một minh họa là một cách gán cho mỗi ký hiệu mệnh đề một giá trị chân lý True hoặc False. Trong một minh họa, nếu kí hiệu mệnh đề P được gán giá trị chân lý True/False (P ơTrue/ PơFalse) thì ta nói mệnh đề P đúng/sai trong minh họa đó. Trong một minh họa, ý nghĩa của các câu phức hợp được xác định bởi ý nghĩa của các kết nối logic. Chúng ta xác định ý nghĩa của các kết nối logic trong các bảng chân lý:



*Bảng chân trị của các kết nối logic*

Ý nghĩa của các kết nối logic ∧, ∨ và ˥ được xác định như ý nghĩa của các từ “và”,“hoặc là”và “phủ định”trong ngôn ngữ tự nhiên. Chúng ta cần phải giải thích thêm về ý nghĩa của phép kéo theo P ⇒ Q (P kéo theo Q), P là giả thiết, còn Q là kết luận. Trực quan cho phép ta xem rằng, khi P là đúng và Q là đúng thì câu “P kéo theo Q “là đúng, còn khi P là đúng Q là sai thì câu “P kéo theo Q”là sai. Nhưng nếu P sai và Q đúng, hoặc P sai Q sai thì “P kéo theo Q”là đúng hay sai ? Nếu chúng ta xuất phát từ giả thiết sai, thì chúng ta không thể khảng định gì về kết luận. Không có lý do gì để nói rằng, nếu P sai và Q đúng hoặc P sai và Q sai thì “P kéo theo Q”là sai. Do đó trong trường hợp P sai thì “P kéo theo Q “là đúng dù Q là đúng hay Q là sai.

Bảng chân lý cho phép ta xác định được ngữ nghĩa các câu phức hợp. Chẳng hạn ngữ nghĩa của các câu P ∧ Q trong minh họa {P ← True, Q ơ False} là False. Việc xác định ngữ nghĩa của một câu (P ∨ Q) ∧ ˥S trong một minh họa được tiến hành như sau: đầu tiên ta xác định giá trị chân lý của P ∨ Q và ˥S, sau đó ta sử dụng bảng chân lý của ∧ để xác định giá trị (P ∨ Q) ∧ ˥S.

#### **2.1.3. *Dạng chuẩn tắc.***

Chúng ta sẽ xét việc chuẩn hóa các công thức, đưa các công thức về dạng thuận lợi cho việc lập luận, suy diễn. Trước hết ta sẽ xét các phép biến đổi tương đương. Sử dụng các phép biển đổi này, ta có thể đưa một công thức bất kỳ về dạng chuẩn tắc.

Hai công thức A và B được xem là tương đương nếu chúng có cùng một giá trị chân lý trong mọi minh họa. Để chỉ A tương đương với B ta viết A≡ B. Bằng phương pháp bảng chân lý, dễ dàng chứng minh được sự tương đương của các công thức sau đây:

A⇒B ≡ ˥A ∨ B

A ⇔ B ≡ (A⇒B) ∧ (B⇒A)

˥ (˥A) ≡ A

**• Luật De Morgan**

˥ (A ∨ B) ≡ ˥A ∧ ˥B

˥ (A ∧ B) ≡ ˥A ∨ ˥B

**• Luật giao hoán**

A ∨ B ≡ B ∨ A

A ∧ B ≡ B ∧ A

**• Luật kết hợp:**

(A ∨ B) ∨ C ≡ A ∨ (B ∨ C)

(A ∧ B) ∧ C ≡ A ∧ (B ∧ C)

**• Luật phân phối**

A ∧ (B ∨ C) ≡ (A ∧ B) ∨ (A ∧ C)

A ∨ (B ∧ C) ≡ (A ∨ B) ∧ (A ∨ C)

#### **2.1.4. *Các câu Horn.***

Ở trên ta đã chỉ ra, mọi công thức đều có thể đưa về dạng chuẩn hội, tức là hội của các tuyển, mỗi câu tuyển có dạng:

˥P1∨... ..∨ ˥Pm ∨ Q1 ∨.....∨ Qn

trong đó Pi, Qi là các ký hiệu mệnh đề (literal dương) câu này tương đương với câu:

P1^... .^ ˥Pm => Q1 ∨.....∨ Qn

Dạng câu này được gọi là **câu Kowalski** (do nhà logic Kowalski đưa ra năm 1971).

Khi n <= 1, tức là câu tuyển chỉ chứa nhiều nhất một literal dương, ta có một dạng câu đặc biệt quan trọng được gọi là **câu Horn** (mang tên nhà logic Alfred Horn, năm 1951).

Nếu m > 0, n = 1, câu Horn có dạng:

P1 ∧.....∧ Pm => Q

Trong đó Pi , Q là các literal dương. Các Pi được gọi là các điều kiện (hoặc giả thiết), còn Q được gọi là kết luận (hoặc hệ quả). Các câu Horn dạng này còn được gọi là các luật **if-then** và được biểu diễn như sau:

**If P1 and....and Pm then Q.**

Khi m=0, n=1 câu Horn trở thành câu đơn Q, hay sự kiện Q. Nếu m>0, n=0 câu Horn trở thành dạng ˥P1v......v lPm hay tương đương ˥(P1^...^ Pm).

Không phải mọi công thức đều có thể biểu diễn dưới dạng hội của các câu Horn. Tuy nhiên trong các ứng dụng, cơ sở tri thức thường là một tập nào đó các câu Horn (tức là một tập nào đó các luật if-then).

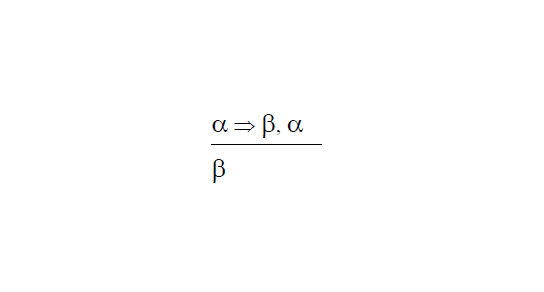
#### **2.1.5. *Luật suy diễn.***

Một công thức H mà nó được xem là một **hệ qủa logic** (logical consequence) của một tập công thức G ={G1,.....,Gm} nếu trong bất kỳ minh họa nào mà {G1,... ..,Gm} đúng thì H cũng đúng. Nói cách khác bất kỳ mô hình nào của G cũng là mô hình của H.

Khi có một cơ sở tri thức, ta muốn sử dụng các tri thức trong cơ sở này để suy ra tri thức mới mà nó là hệ quả logic của các công thức trong cơ sở tri thức. Điều đó được thực hiện bằng cách sử dụng **các luật suy diễn** (rule of inference). Luật suy diễn giống như một thủ tục mà chúng ta sử dụng để sinh ra một công thức mới từ các công thức đã có. Một luật suy diễn gồm hai phần: một tập các điều kiện và một kết luận. Chúng ta sẽ biểu diễn các luật suy diễn dưới dạng “phân số ”, trong đó tử số là danh sách các điều kiện, còn mẫu số là kết luận của luật, tức là mẫu số là công thức mới được suy ra từ các công thức ở tử số.

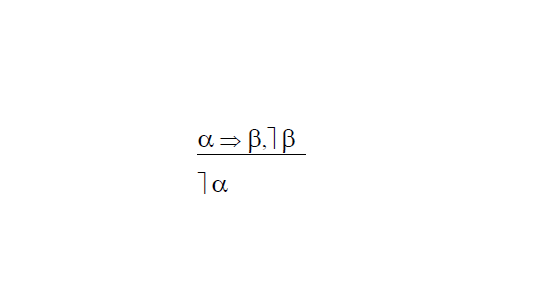
Sau đây là một số luật suy diễn quan trọng trong logic mệnh đề. Trong các luật này α, αi,β, γ là các công thức:

**• Luật Modus Ponens**



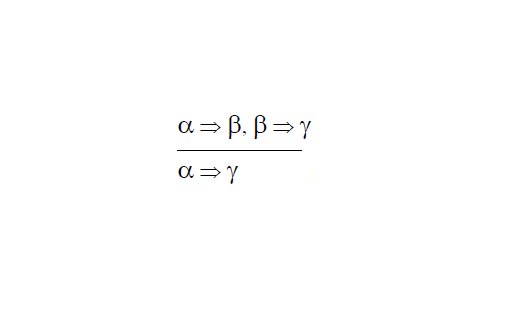
Từ một kéo theo và giả thiết của kéo theo, ta suy ra kết luận của nó.

**• Luật Modus Tollens**



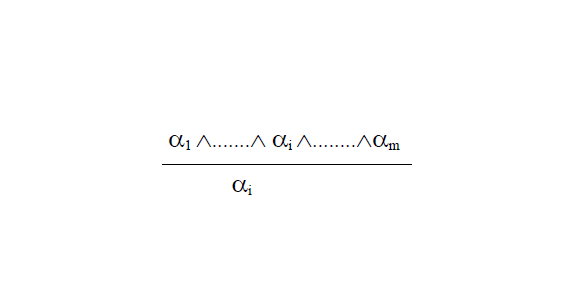
Từ một kéo theo và phủ định kết luận của nó, ta suy ra phủ định giả thiết của kéo theo.

**• Luật bắc cầu**

****

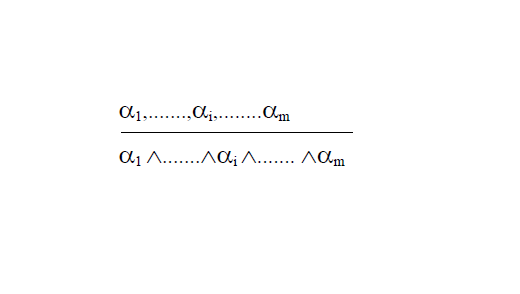
Từ hai kéo theo, mà kết luận của kéo theo thứ nhất trùng với giả thiết của kéo theo thứ hai, ta suy ra kéo theo mới mà giả thiết của nó là giả thiết của kéo theo thứ nhất, còn kết luận của nó là kết luận của kéo theo thứ hai.

**• Luật loại bỏ hội**

****

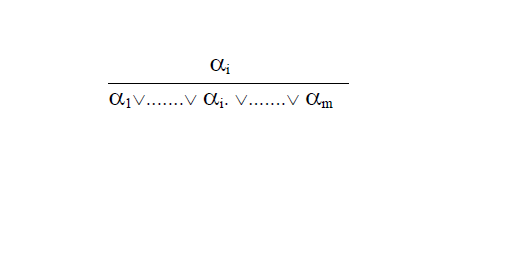
Từ một hội ta suy ra một nhân tử bất kỳ của hội.

**• Luật đưa vào hội**

****

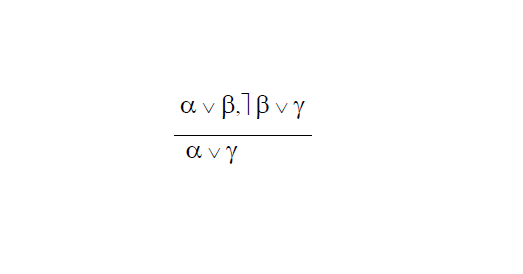
Từ một danh sách các công thức, ta suy ra hội của chúng.

**• Luật đưa vào tuyển**

****

Từ một công thức, ta suy ra một tuyển mà một trong các hạng tử của tuyển là công thức đó.

**• Luật phân giải**

****

Từ hai tuyển, một tuyển chứa một hạng tử đối lập với một hạng tử trong tuyển kia, ta suy ra tuyển của các hạng tử còn lại trong cả hai tuyển.

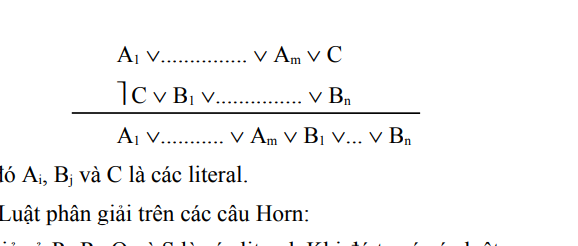
#### **2.1.6. *Chứng minh bác bỏ bằng luật phân giải.***

Phương pháp chứng minh bác bỏ (refutation proof hoặc proof by contradiction) là một phương pháp thường xuyên được sử dụng trong các chứng minh toán học. Tư tưởng của phương pháp này là như sau: Để chứng minh P đúng, ta giả sử P sai (thêm ˥P vào các giả thiết) và dẫn tới một mâu thuẫn. Sau đây ta sẽ trình bầy cơ sở của phương pháp chứng minh này.

Giả sử chúng ta có một tập các công thức G = {G1,.....,Gm} ta cần chứng minh công thức H là hệ quả logic của G. Điều đó tương đương với chứng minh công thức G1^....^Gm ⇒ H là vững chắc. Thay cho chứng minh G1^... ^Gm ⇒H là vững chắc, ta chứng minh G1^....^Gm ^˥H là không thỏa mãn được. Tức là ta chứng minh tập G’= (G1,... .,Gm,˥H) là không thỏa được. G sẽ không thoả được nếu từ G‘ta suy ra hai mệnh đề đối lập nhau. Việc chứng minh công thức H là hệ quả logic của tập các tiêu đề G bằng cách chứng minh tính không thỏa được của tập các tiêu đề được thêm vào phủ định của công thức cần chứng minh, được gọi là chứng minh bác bỏ.

Để thuận tiện cho việc sử dụng luật phân giải, chúng ta sẽ cụ thể hoá luật phân giải trên các dạng câu đặc biệt quan trọng.

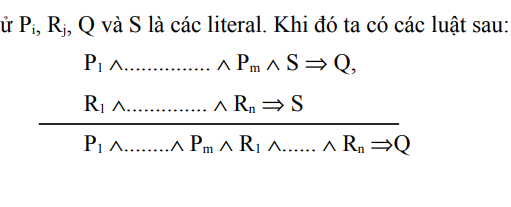
• **Luật phân giải trên các câu tuyển**



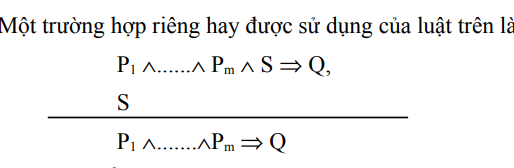
Trong đó Ai, Bj và C là các literal.

• **Luật phân giải trên các câu Horn:**

Giả sử Pi, Rj, Q và S là các literal. Khi đó ta có các luật sau:



Một trường hợp riêng hay được sử dụng của luật trên là:



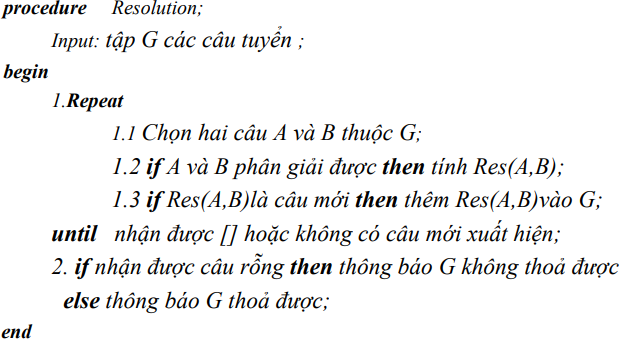
Khi ta có thể áp dụng luật phân giải cho hai câu, thì hai câu này được gọi là hai câu **phân giải được** và kết quả nhận được khi áp dụng luật phân giải cho hai câu đó được gọi là **phân giải thức** của chúng. Phân giải thức của hai câu A và B được kí hiệu là res(A,B). Chẳng hạn, hai câu tuyển phân giải được nếu một câu chứa một literal đối lập với một literal trong câu kia. Phân giải thức của hai literal đối lập nhau (P và ˥P) là câu rỗng, chúng ta sẽ ký hiệu câu rỗng là [], câu rỗng không thoả được.

Giả sử G là một tập các câu tuyển (bằng cách chuẩn hoá ta có thể đưa một tập các công thức về một tập các câu tuyển). Ta sẽ ký hiệu R(G) là tập câu bao gồm các câu thuộc G và tất cả các câu nhận được từ G bằng một dãy áp dụng luật phân giải.

Luật phân giải là luật đầy đủ để chứng minh một tập câu là không thỏa được. Điều này được suy từ định lý sau:

**Định lý phân giải:**

Một tập câu tuyển là không thỏa được nếu và chỉ nếu câu rỗng [9] ∈ R(G).



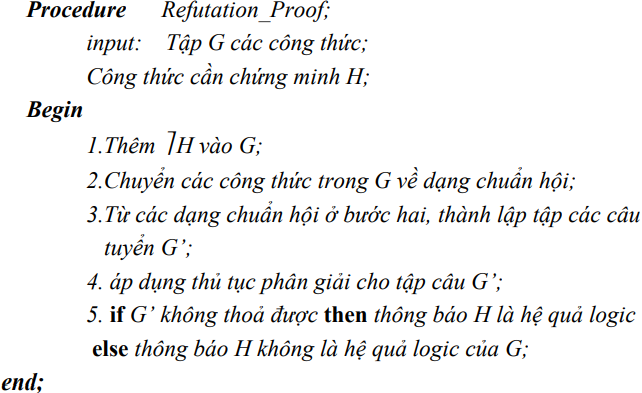
Định lý phân giải có nghĩa rằng, nếu từ các câu thuộc G, bằng cách áp dụng luật phân giải ta dẫn tới câu rỗng thì G là không thỏa được, còn nếu không thể sinh ra câu rỗng bằng luật phân giải thì G thỏa được. Lưu ý rằng, việc dẫn tới câu rỗng có nghĩa là ta đã dẫn tới hai literal đối lập nhau P và ˥ P (tức là dẫn tới mâu thuẫn).

Từ định lý phân giải, ta đưa ra thủ tục sau đây để xác định một tập câu tuyển G là thỏa được hay không. Thủ tục này được gọi là thủ tục phân giải.

Dễ thấy, nếu G là tập các câu hữu hạn thì các literal có mặt trong các câu của G là hữu hạn. Do đó, số các câu tuyển thành lập được từ các literal đó là hữu hạn. Vì vậy, chỉ có một số hữu hạn câu được sinh ra bằng luật phân giải. Thủ tục phân giải sẽ dừng lại sau một số hữu hạn bước.

Chỉ sử dụng luật phân giải ta không thể suy ra mọi công thức là hệ quả logic của một tập công thức đã cho. Tuy nhiên, sử dụng luật phân giải ta có thể chứng minh được một công thức bất kì có là hệ quả của một tập công thức đã cho hay không bằng phương pháp chứng minh bác bỏ. Vì vậy luật phân giải được xem là **luật đầy đủ cho bác bỏ***.*

Thủ tục chứng minh bác bỏ bằng luật phân giải:

**

**Ví dụ:** Giả giử G là tập hợp các câu tuyển sau:

˥ A ∨ ˥ B ∨ P (1)

˥ C ∨ ˥ D ∨ P (2)

E ∨ C (3)

A (4)

E (5)

D (6)

Giả sử ta cần chứng minh P. Thêm vào G câu sau:

˥P (7)

Ap dụng luật phân giải cho câu (2) và (7) ta được câu:

˥C ∨ ˥D (8)

Từ câu (6) và (8) ta nhận được câu:

˥C (9)

Từ câu (3) và (9) ta nhận được câu:

˥E (10)

Tới đây đã xuất hiện mâu thuẫn, vì câu (5) và (10) đối lập nhau. Từ câu (5) và (10) ta nhận được câu rỗng [9].

🡪 Vậy P là hệ quả logic của các câu (1) --(6).

Thông thường chúng ta có thể bảng chân lý để chứng minh tính đúng đắn của môt biểu thức. Nhưng phương pháp đó tỏ ra cồng kềnh và có tính “thủ công”. Thay vào đó, chúng ta có thể sử dụng hai thuật toán sau đây để chứng minh biêu thức là đúng hoặc sai.

**\*Thuật toán Havard (1970)**

**Bước 1**: Phát biểu lại giả thiết (GT) và kết luận của bài toán dưới dạng chuẩn sau: GT1,GT2,…, GTn → KL1,KL2,…, KLm

Trong đó các GTi, KLjđược xây dựng từ các biến mệnh đề và các phép nối ∧ ,∨,˥,

**Bước 2: Bước** bỏ phủ định (nếu cần). Khi cần bỏ các các phủ định: chuyển vế GTi sang vế kết luận KLj và ngược lại (giống như chuyển dấu âm trong đai số từ vế phai sang trái và ngược lại)

**Bước 3: Thay dấu** “∧” ở GTivà “∨” ở KLjbằng các dấu “,”

**Bước 4:** Nếu GTicòn dấu “∨” và KLjcòn dấu “∧” thì tách chúng thành hai dòng con

**Bước 5:** Một dòng được chứng minh nêu tồn tại chung một mệnh đề ở cả hai vế

**Bước 6:** Bài toán được chứng minh khi và chi khi tất cả các dòng được chứng minh. Ngược lại thì bài toán không được chứng minh.

**\*Thuật toán Robin son (1971)**

Robíson đã cai tiến thuật toán Havard. Cách thức chứng minh như sau:

**Bước 1**: Phát biểu lại giả thiết (GT) và kết luận của bài toán dưới dạng chuẩn sau: GT1,GT2,…, GTn → KL1,KL2,…, KLm

Trong đó các GTi, KLjđược xây dựng từ các biến mệnh đề và các phép nối ∧ ,∨,˥,

**Bước 2: Thay dấu** “∧” ở GTivà “∨” ở KLjbằng các dấu “,”

**Bước 3: Chuyển vế** KLj sang vế Gti với dấu phủ định để còn một vế, tức là : GT1,GT2,…, GTn , ˥, KL1,˥, KL2,…, ˥KLm

**Bước 4:** Xây dựng một mệnh đề mới bằng cách tuyển một cặp mệnh đề từ danh sách các mệnh đề. Nếu mệnh đề mới có các biến mệnh đề đối ngẫu thì mệnh đề đó được loại bỏ.

**Bước 5: Bổ sung mệnh đề mới này vào danh sách và lặp lại bước 4.**

**Bước 6:** Bài toán được chứng minh khi và chi khi chỉ còn hai mệnh đề đói ngẫu. Ngược lại thì bài toán không được chứng minh.

Thuật toán này thực chất là chứng minh bằng phản chứng

### **2.2. Biểu diễn tri thức bằng luật sản xuất.**

Biểu diễn tri thức bẳng các luật (nếu – thì), còn gọi là các luật sản xuất, là ngôn ngữ phổ biến nhất hiện nay dùng để biểu diễn tri thức. Trong đó một luật chứa đựng tri thức về việc giải quyết một vấn đề nào đó và các luật được tạo nên khá dễ dàng và dễ hiểu.

Một luật thường được biểu diễn dưới dạng:

**IF A1 AND A2 AND … AND An THEN B**

Trong đó:

Ai là các điều kiện (conditions, antecedents, premises).

B là kết luận (conclusion, consequence, action).

**Mệnh đề điều kiện của một luật:**

**•** Không cần sử dụng toán tử logic OR.

**•** Một luật với toán tử logic OR trong mệnh đề điều kiện, thì sẽ được chuyển thành một tập các luật tương ứng không chứa OR.

**•** Ví dụ: Luật (IF A1 ∨ A2 THEN B) được chuyển thành 2 luật (IF A1 THEN B) và (IF A2 THEN B).

**Mệnh đề kết luận của một luật:**

**•** Không cần sử dụng toán tử logic AND.

**•** Một luật với toán tử logic AND trong mệnh đề kết luận, thì sẽ được chuyển thành một tập các luật tương ứng không chứa AND.

**•** Ví dụ: Luật (IF … THEN B1 ∧ B2 ) được chuyển thành 2 luật(IF … THEN B1) and (IF … THEN B2).

**•**  Không cho phép sử dụng toán tử OR!

Các hệ tri thức mà cơ sở tri thức bao gồm các luật sẽ được gọi là các hệ dựa trên luật (rule - based system).

#### **2.2.1. *Đồ thị AND/OR.***

**Ví dụ**: cho tập luật sau:

IF (Shape=long) AND (Color=(green OR yellow))

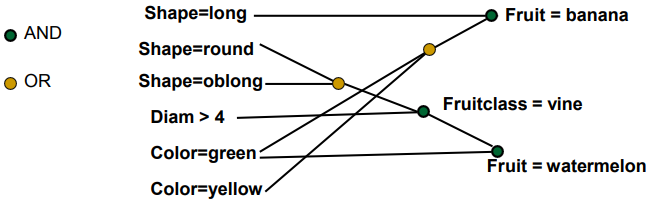
THEN (Fruit=banana)

IF (Shape=(round OR oblong)) AND (Diam > 4)

THEN (Fruitclass=vine)

IF (Fruitclass=vine) AND (Color=green)

THEN (Fruit=watermelon)



*Đồ thị AND/OR của tập luật đã cho.*

**Trong đó:**

- Luật IF (Shape=long) **AND** (Color=(green **OR** yellow)) THEN (Fruit=banana) được tạo nên bởi các luật:

**•** IF (Shape=long) **AND** (Color=green) THEN (Fruit=banana)

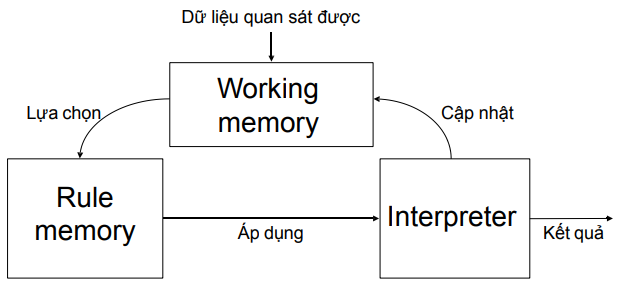
**•** IF (Shape=long) **AND** (Color=yellow) THEN (Fruit=banana)

- Luật IF (Shape=(round **OR** oblong)) **AND** (Diam > 4) THEN (Fruitclass=vine) được tạo nên bởi các luật:

**•** IF (Shape=round) **AND** (Diam > 4) THEN (Fruitclass=vine)

**•** IF (Shape=oblong) **AND** (Diam > 4) THEN (Fruitclass=vine)

#### **2.2.2. *Hệ thống suy diễn dựa trên luật.***



*Kiến trúc điển hình của một hệ thống suy diễn dựa trên luật (Rulebased system – RBS).*

**Bộ nhớ làm việc (Working memory)**

Lưu giữ các sự kiện (các giả thiết đúng, đã được chứng minh).

Các sự kiện này sẽ quyết định những luật nào được áp dụng (bởi thành phần Interpreter).

**Bộ nhớ các luật (Rule memory)**

Chính là cơ sở tri thức của hệ thống.

Lưu giữ các luật có thể áp dụng.

**Bộ diễn dịch (Interpreter)**

Hệ thống bắt đầu bằng việc đưa một sự kiện (dữ liệu) phù hợp vào bộ nhớ làm việc.

Khi sự kiện (dữ liệu) trong bộ nhớ làm việc phù hợp với các điều kiện của một luật trong bộ nhớ các luật, thì luật đó sẽ được áp dụng.

#### **2.2.3. *Ưu điểm của hệ thống suy diễn dựa trên luật.***

Cách biểu diễn phù hợp, rất gần với cách diễn đạt trong ngôn ngữ tự nhiên.

Rất dễ dàng để diễn đạt các tri thức bởi các luật.

Các luật dạng IF-THEN rất dễ hiểu (có lẽ là dễ hiểu nhất) đối với người sử dụng.

Trong một lĩnh vực (bài toán) cụ thể, cách biểu diễn bằng luật giúp các chuyên gia trong lĩnh vực này có thể đánh giá và cải tiến các luật.

Là một cách biểu diễn tri thức theo kiểu khai báo, kỹ sư tri thức chỉ cần thu thập các tri thức (ở dạng các luật IF-THEN) về một lĩnh vực cụ thể, và đưa chúng vào trong một cơ sở các luật.

Kỹ sư tri thức (có thể) không cần phải quan tâm đến khi nào, làm thế nào, và theo trật tự nào mà các luật được sử dụng – Hệ thống sẽ tự động đảm nhận các nhiệm vụ này.

Dễ dàng mở rộng cơ sở tri thức bằng cách bổ sung thêm các luật mới (các tri thức mới) vào cuối của cơ sở các luật.

#### **2.2.4. *Nhược điểm của hệ thống suy diễn dựa trên luật.***

Khả năng biểu diễn bị giới hạn trong nhiều lĩnh vực bài toán thực tế, tri thức của lĩnh vực bài toán đó không phù hợp với cách biểu diễn dạng (IF-THEN)

Sự tương tác giữa các luật và trật tự của các luật trong cơ sở luật có thể gây ra các hiệu ứng không mong muốn.

Trong quá trình thiết kế và bảo trì một cơ sở luật, mỗi luật mới được đưa vào cần phải được cân nhắc (kiểm tra) với các luật đã có từ trước.

Rất khó khăn và chi phí tốn kém để xem xét tất cả các tương tác có thể giữa các luật.

### **2.3. Biểu diễn tri thức bằng mạng ngữ nghĩa.**

Mạng ngữ nghĩa (Semantic Network) được đề cử bởi Quillian vào năm 1966 như là một mô hình biểu diễn bộ nhớ của con người. Cũng như trong logic mệnh đề, nói đến ngữ nghĩa là chúng ta nói đến ý nghĩa của các công thức trong một thế giới hiện thực nào đó mà nó được biểu diễn dựa trên đồ thị.

#### **2.3.1. *Cú pháp.***

Một mạng ngữ nghĩa bao gồm một tập các nút và các cung liên kết để biểu diễn định nghĩa của một (tập) khái niệm. Các nút (nodes) biểu diễn các khái niệm, hành động hoặc đối tượng trong lĩnh vực bài toán đang xét. Các cung liên kết biểu diễn các mối quan hệ giữa các khái niệm với nhau, được gán nhãn và có chiều giữa các nút.

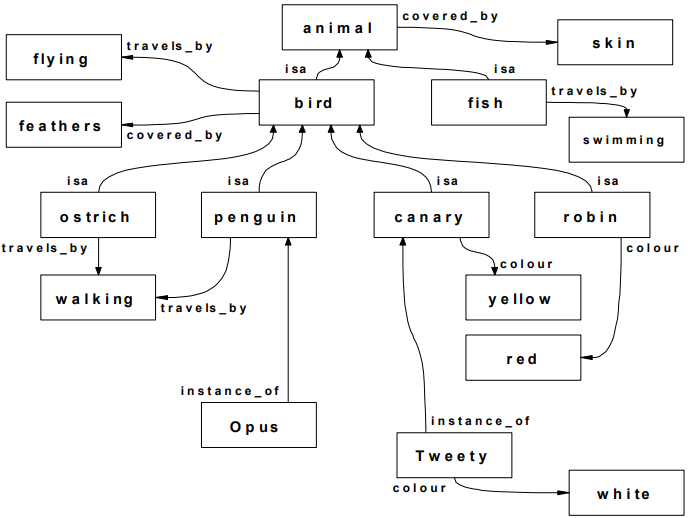
Quá trình suy diễn (reasoning/inference) trong mạng ngữ nghĩa được thực hiện thông qua cơ chế lan truyền tác động (Activation) và kế thừa (Inheritance).

**Liên kết kế thừa (Inheritance-oriented link)** biểu diễn nút A là một lớp con của nút B hoặc A là một ví dụ của nút B (Ví dụ: liên kết INSTANCE – OF).

**Liên kết cụ thể (Domain-specific link)** biểu diễn: Nút A liên quan tới (có quan hệ với) nút B (Ví dụ: HAS, CAN, HAS-PART, CAUSES, HAS-COLOR, …).

Các nút biểu diễn các đối tượng hoặc các lớp đối tượng (các khái niệm), hoặc các giá trị của các thuộc tính. Chúng ta quy ước các đỉnh hình chữ nhật biểu diễn các đối tượng hoặc các lớp, còn các đỉnh elip biểu diễn các giá trị.

Các cung liên kết biểu diễn các quan hệ. Cung gắn nhãn isa đi từ đỉnh biểu diễn các đối tượng tới đỉnh biểu diễn lớp mà đối tượng là thành viên của lớp. Cung gắn nhãn ako đi từ đỉnh biểu diễn một lớp này tới đỉnh biểu diễn một lớp khác nếu lớp ứng với đỉnh nguồn của cung là lớp con của lớp ứng với đỉnh đích của cung. Các cung khác được gắn nhãn bởi tên của các thuộc tính, các cung này đi từ các đỉnh biểu diễn các đối tượng (hoặc các lớp), tới các đỉnh biểu diễn giá trị của thuộc tính.



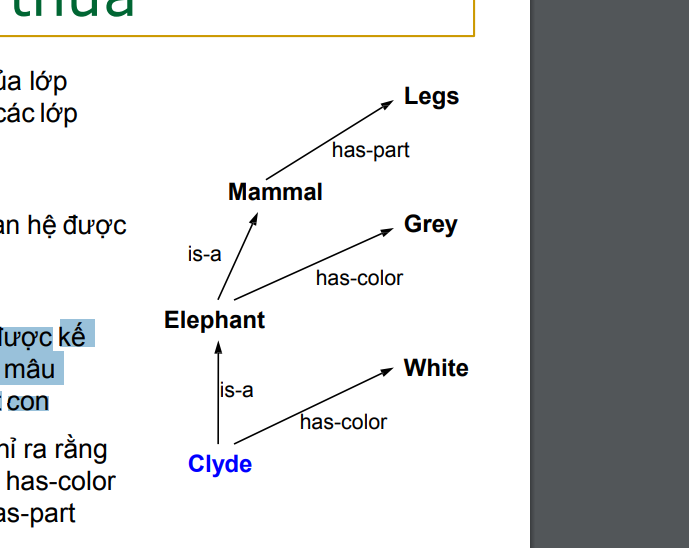
*Ví dụ về mô hình ngữ nghĩa.*

#### **2.3.2. *Tính kế thừa.***

Các thuộc tính (properties) của lớp (loại) cha được kế thừa cho các lớp (loại) con.

**Kế thừa toàn bộ (Universal inheritance):** Tất cả các quan hệ được kế thừa.

**Kế thừa mặc định (Default inheritance):** Các quan hệ được kế thừa, trừ khi có các thông tin mâu thuẫn (với nút cha) ở một nút con.



*Ví dụ về mô hình có tính chất kế thừa.*

#### **2.3.3. *Ưu điểm của* biểu diễn tri thức bằng mạng ngữ nghĩa*.***

Rõ ràng (trực quan) trong hiển thị, dễ hiểu đối với người dùng, thường được sử dụng như là một công cụ trao đổi (làm việc) giữa các kỹ sư tri thức (knowledge engineers) và các chuyên gia (experts) trong giai đoạn thu thập tri thức.

Mạng ngữ nghĩa là rất phù hợp đối với các bài toán biểu diễn tri thức ở dạng phân cấp các khái niệm, tri thức được phân loại (phân lớp) thành một cấu trúc phân cấp.

Cơ chế biểu diễn phân cấp hỗ trợ quá trình suy diễn nhanh chóng.

Hỗ trợ cơ chế suy diễn mặc định (default reasoning).

Tập trung vào các thành phần chính của tri thức và liên kết giữa chúng.

#### **2.3.4. *Nhược điểm của* biểu diễn tri thức bằng mạng ngữ nghĩa*.***

Không tồn tại cách diễn dịch (interpretation) chung (chuẩn), ngữ nghĩa của các mạng ngữ nghĩa không được định nghĩa một cách chuẩn tắc.

Gặp vấn đề trong việc biểu diễn các thông tin tin phủ định và tuyển (Ví dụ: “John does not go fishing”, “John eats pizza or fish and chips”).

Khó khăn trong việc chọn các thành phần cơ bản (primitives) phù hợp.

Khả năng suy diễn hạn chế.

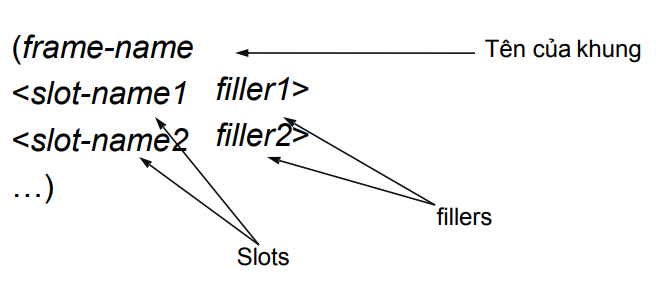
### **2.4. Biểu diễn tri thức bằng khung.**

Khung là một phương pháp biểu diễn tri thức có cấu trúc dữ liệu chứa tất cả tri thức liên quan đến một đối tượng cụ thể nào đó. Khung (Frame) có liên hệ chặt chẽ đến khái niệm hương đối tượng. Khung thương được dùng đẻ biểu diễn những tri thức “chuẩn” hoặc những tri thức dựa trên kinh nghiệm hoặc các điểm đã được hiểu biết cặn kẽ. Có 2 kiểu khung: khung cụ thể và khung tổng quát.

**Khung cụ thể (Individual frames):** biểu diễn một đối tượng cụ thể, chẳng hạn như một người cụ thể, một chuyến đi nghỉ mát cụ thể, …

**Khung tổng quát (Generic frames):** biểu diễn một lớp (loại) các đối tượng, chẳng hạn như các sinh viên, các chuyến đi nghỉ mát, …

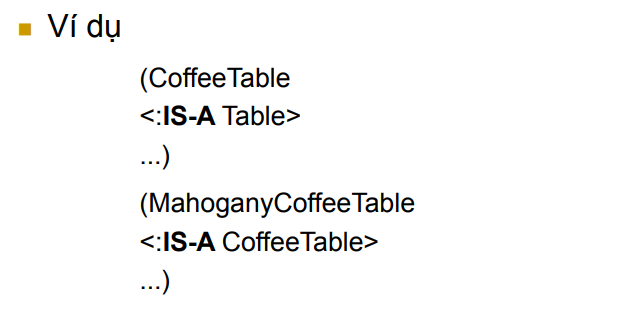
Cấu trúc của khung gồm hai thành phần cơ bản “Slot” và “Filler”. Slot là một thuộc tính đặc tả đối tượng ví dụ khung mô tả *xe hơi* có hai Slot là *Trọng lượng* và *Loại động cơ*, còn giá trị gán cho một thuộc tính được gọi là filler of the slot Một Slot có thể chứa nhiều Filler. Các Filler đôi khi còn được gọi là Slot con.



*Các thành phần biểu diễn một khung.*

Các thủ tục và các giá trị thuộc tính của một khung tổng quát hơn sẽ được áp dụng (kế thừa) bởi một khung cụ thể hơn, thông qua cơ chế kế thừa.

**Ví dụ:**

****

*Khung có tính kế thừa.*

#### **2.4.1. *Ưu điểm của* biểu diễn tri thức bằng khung*.***

Kết hợp được cả tri thức khai báo (declarative knowledge) và tri thức thủ tục (procedural knowledge) trong cùng một phương pháp biểu diễn.

Các khung được tổ chức có cấu trúc phân cấp, cho phép dễdàng phân loại tri thức.

Cấu trúc phân cấp các khung cho phép giảm bớt sự phức tạp và chi phí trong quá trình xây dựng cơ sở tri thức.

Cho phép thiết lập các ràng buộc đối với các giá trị được gán cho các thuộc tính (ví dụ: ràng buộc giá trị nhập vào phải nằm trong một khoảng giá trị cụ thể).

Cho phép lưu giữ các giá trị mặc định (sử dụng thuộc tính đặc biệt IS-A, các giá trị của các thuộc tính của một khung tổng quát hơn được sử dụng để gán cho các thuộc tính của một khung cụ thể hơn).

#### **2.4.2. *Nhược điểm của* biểu diễn tri thức bằng khung*.***

Trong quá trình thiết kế cấu trúc phân cấp của các khung, cần rất để ý đến sự hợp lý của việc phân loại (các khung).

Có thể gặp vấn đề chi phí cao cho việc thiết kế các thủ tục (IF-ADDED và IF-NEEDED) – Quá nhiều công sức dành cho việc thiết kế các thủ tục phù hợp, thay vì tập trung vào việc kiểm tra cấu trúc và nội dung của các khung.

Quá trình khai thác các khung có thể không hiệu quả, vì không có phương pháp hiệu quả để lưu trữ dữ liệu của các khung trong máy tính.

## **3. Tổng quan ngôn ngữ lập trình logic Prolog.**

Prolog là một ngôn ngữ cấp cao, có đặc điểm gần với ngôn ngữ tự nhiên, từ những người mới học đến những lập trình viên chuyên nghiệp đều có thể tiếp cận một cách nhanh chóng, viết ra một chương trình ứng dụng hữu ích.

Prolog ra đời vào năm 1973 (do [Alain Colmerauer](https://vi.wikipedia.org/w/index.php?title=Alain_Colmerauer&action=edit&redlink=1) và [Robert Kowalski](https://vi.wikipedia.org/w/index.php?title=Robert_Kowalski&action=edit&redlink=1) thiết kế), mục tiêu của Prolog là giúp người dùng mô tả lại bài toán trên ngôn ngữ của logic, dựa trên đó, máy tính sẽ tiến hành suy diễn tự động dựa vào những cơ chế suy diễn có sẵn ([hợp nhất](https://vi.wikipedia.org/wiki/H%E1%BB%A3p_nh%E1%BA%A5t_(ph%C3%A9p_to%C3%A1n)" \o "Hợp nhất (phép toán)), [quay lui](https://vi.wikipedia.org/wiki/Quay_lui_(khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh)) và [tìm kiếm theo chiều sâu](https://vi.wikipedia.org/wiki/T%C3%ACm_ki%E1%BA%BFm_theo_chi%E1%BB%81u_s%C3%A2u" \o "Tìm kiếm theo chiều sâu)) để tìm câu trả lời cho người dùng. Từ đó đến nay, qua nhiều lần cải tiến, đặc biệt hãng Borland cho ra đời phần mềm TURBO PROLOG với nhiều ưu điểm, thuận tiện cho người sử dụng. Để giải quyết một số vấn đề, ta nhận thấy sử dụng ngôn ngữ Prolog cho ta chương trình gọn nhẹ hơn nhiều so với các ngôn ngữ khác.

Khác với những ngôn ngữ cấu trúc như Pascal, hay C mà ta đã làm quen, Prolog là một ngôn ngữ mô tả, với một số sự kiện và quy luật suy diễn đã mô tả, Prolog sẽ suy luận cho ta các kết quả.

### **3.1. Cấu trúc của một chương trình Prolog.**

Một chương trình Prolog thường gồm có 3 hoặc 4 đoạn cơ bản: clauses, predicates, domains và goal. Phần goal có thể bỏ đi, nếu ta không thiết kế goal trong chương trình, thì khi thực hiện, hệ thống sẽ yêu cầu ta nhập goal vào.

#### **3.1.1. Phần Domains.**

Đây là phần định nghĩa kiểu mới dựa vào các kiểu đã biết. Các kiểu được định nghĩa ở đây sẽ được sử dụng cho các đối số trong các vị từ. Nếu các vị từ sử dụng đối số có kiểu cơ bản thì có thể không cần phải định nghĩa lại các kiểu đó. Tuy nhiên để cho chương trình sáng sủa, người ta sẽ định nghĩa lại cả các kiểu cơ bản.

Cú pháp:

<**danh sách kiểu mới**>**=**<**kiểu đã biết**>

hoặc <**danh sách kiểu mới**>**=**<**danh sách kiểu đã biết**>

Trong đó các kiểu mới phân cách nhau bởi dấu phẩy, còn các kiểu đã biết phân cách nhau bởi dấu chấm phẩy.

**Ví dụ:**

Domains

ten, tac\_gia, nha\_xb, dia\_chi = string

nam, thang, so\_luong = integer

dien\_tich = real

nam\_xb = nxb(thang, nam)

do\_vat = sach(tac\_gia, ten, nha\_xb, nam\_xb); xe(ten, so\_luong); nha(dia\_chi, dien\_tich)

Trong ví dụ trên, ta đã định nghĩa các kiểu mới, trong đó các kiểu mới ten, tac\_gia, nha\_xb, dia\_chi dựa vào cùng một kiểu đã biết là string; các kiểu mới nam, thang, so\_luong dựa vào cùng một kiểu đã biết là integer; kiểu mới dien\_tich dựa vào kiểu đã biết là real; kiểu mới năm\_xb dựa vào kiểu nxb được xây dựng từ các kiểu đã biết là thang, nam; còn kiểu do\_vat lại dựa vào các kiểu sach, xe, nha mà các kiểu này lại dựa vào các kiểu đã biết.

#### **3.1.2. Phần Predicates.**

Đây là phần bắt buộc phải có. Trong phần này chúng ta cần phải khai báo đầy đủ các vị từ sử dụng trong phần Clauses, ngoại trừ các vị từ mà Turbo Prolog đã xây dựng sẵn.

Cú pháp:

<**Tên vị từ**> **(**<**danh sách các kiểu**>**)**

Các kiểu là các kiểu cơ bản hoặc là các kiểu đã được định nghĩa trong phần domains và được viết phân cách nhau bơi dấu phẩy.

**Ví dụ:**

Predicates

so\_huu (ten, do\_vat)

so\_nguyen\_to(integer)

Trong ví dụ trên ta khai báo hai vị từ. Trong đó vị từ so\_huu (ten, do\_vat) để chỉ một người có tên là ten sẽ sở hữu môt do\_vat nào đó. Còn vị từ so\_nguyen\_to(integer) để xét xem một số integer nào đó có phải là số nguyên tố hay không.

#### **3.1.3. Phần Clause.**

Đây là phần bắt buộc phải có dùng để mô tả các sự kiện và các luật, sử dụng các vị từ đã khai báo trong phần predicates.

Cú pháp:

<**Tên vị từ**>**(**<**danh sách các tham số**>**)** <**kí hiệu**>

<**Tên vị từ 1**>**(**<**danh sách các tham số 1**>**)** <**kí hiệu**>

… … …

<**Tên vị từ N**>**(**<**danh sách các tham số N**>**)** <**kí hiệu**>

Trong đó: Tên vị từ phải là các tên vị từ đã được khai báo trong phần predicates. Các tham số có thể là các hằng hoặc biến có kiểu tương thích với các kiểu tương ứng đã được khai báo trong các vị từ ở trong phần predicates; các tham số được viết cách nhau bởi dấu phẩy. Các kí hiệu bao gồm:

**:-** (điều kiện nếu).

**,** (điều kiện và).

**;** (điều kiện hoặc).

**.** (kết thúc vị từ)

**Ví dụ:**

Clauses

so\_nguyen\_to(2):- !.

so\_nguyen\_to(N):- N>0,

so\_nguyen\_to(M),

M<N,

N MOD M <>0.

so\_huu(“Nguyen Van A”, sach(“Do Xuan Loi”, “Cau truc DL”, “Khoa hoc Ky thuat”, nxb(8,1985))).

**Chú ý**: Nếu trong các tham số của một vị từ có biến thì biến này phải xuất hiện ít nhất 2 lần trong vị từ đó hoặc trong các vị từ dùng để suy diễn ra vị từ đó. Nếu chỉ xuất hiện một lần thì bắt buộc phải dùng biến tự do.

**Ví dụ:** Để diễn tả sự kiện: Tổ hợp chập 0 của N (N bất kỳ) bằng 1, ta không thể viết Tohop(N,0,1) vì biến N chỉ xuất hiện đúng một lần trong vị từ này, do đó ta phải viết Tohop(\_,0,1) .

#### **3.1.4. Phần Goal.**

Bao gồm các mục tiêu mà ta yêu cầu Turbo Prolog xác định và tìm kết quả. Đây là phần không bắt buộc phải có. Nếu ta viết sẵn trong chương trình thì đó gọi là goal nội; Nếu không, khi chạy chương trình Turbo Prolog sẽ yêu cầu ta nhập goal vào, lúc này gọi là goal ngoại.

Cú pháp phần goal giống như cú pháp phần clauses. Tức là ta đưa vào một hoặc một số các vị từ.

Nếu tất cả các tham số của vị từ là hằng thì kết quả nhận được là Yes (đúng) hoặc No (sai). Nếu trong các tham số của vị từ có biến thì kết quả trả về sẽ là các giá trị của biến.

Ngoài các phần chủ yếu nói trên, ta có thể đưa vào các phần liên quan đến khai báo hằng, các tập tin liên quan hoặc chỉ thị dịch.

**Ví dụ:**

Constants

Pi = 3.141592653

### **3.2. Kết nối Prolog với ngôn ngữ lập trình C#.**

Prolog tuy đã là một ngôn ngữ mạnh mẽ chuyên phục vụ cho trí tuệ nhân tạo và xử lý ngôn ngữ tự nhiên nhưng đôi lúc, chúng ta cũng phải kết hợp Prolog với các ngôn ngữ khác để tăng sự trực quan cho ứng dụng. Và ngôn ngữ chúng tôi chọn để trực quan hoá hệ chuyên gia này đó chính là ngôn ngữ C#, ngôn ngữ hỗ trợ sử dụng WinForm rất thuận tiện, dễ sử dụng, hỗ trợ nhiều thao tác, thân thiện cho người dùng.

Những tập tin yêu cầu để hỗ trợ kết nối:

**•** Môi trường lập trình Prolog (version 6.6.5 x86).

**•** Visual Studio phiên bản 2008 trở lên.

**•** Thư viện Prolog để kết nối với C# – SwiPlCs version 1.1.60605.0.

# **CHƯƠNG III. GIẢI QUYẾT BÀI TOÁN**

## **1. Thu thập tri thức.**

Thu thập tri thức là bước quan trọng mở đầu cho mỗi bài toán. Đối với bài toàn tư vấn nhu cầu mua sắm laptop cho những người đã có những kiến thức, am hiểu căn bản về thông số kỹ thuật của máy thì các tri thức cụ thể được thu thập dựa trên các thông số kỹ thuật phần cứng quan trọng chính của máy tính cùng với thông tin về giá tiền của sản phẩm.

### **1.1. CPU – Central Processing Unit.**

CPU là một đơn vị xử lý trung tâm. Đây là bộ phận rất bé của laptop nhưng có sức mạnh lớn nhất, nó được xem là bộ não để điều khiển các phần còn lại của máy giúp phân tích mọi dữ liệu và xử lí các yêu cầu tính toán mà người dùng thao tác trên máy.

Tốc độ xử lý của CPU là tần số tính toán và làm việc của nó được đo bằng đơn vị GHz hoặc MHz. Nếu cùng một dòng chip ví dụ như [Core i3](https://www.dienmayxanh.com/laptop?g=core-i3) thì xung nhịp cao hơn đồng nghĩa với tốc độ xử lý nhanh hơn, khả năng làm việc tốt hơn. Tuy nhiên, nếu giữa 2 dòng chip khác nhau như Core i3 hai nhân xung nhịp 2.2GHz và Intel [Pentium](https://www.dienmayxanh.com/laptop?g=pentium) Dual core 2.3GHz thì không thể so sánh ngay được bởi vì tốc độ xử lý của Laptop còn phụ thuộc rất nhiều vào bộ nhớ đệm và các bộ phận khác như RAM, chip đồ họa, ổ cứng…

Hiện nay có hai nhà sản xuất CPU cho Laptop lớn nhất là Intel và AMD. Các hãng xuất Laptop khá phổ biến hiện nay như [Acer](https://www.dienmayxanh.com/laptop-acer), [Asus](https://www.dienmayxanh.com/laptop-asus), [Lenovo](https://www.dienmayxanh.com/laptop-lenovo), [Dell](https://www.dienmayxanh.com/laptop-dell), [HP](https://www.dienmayxanh.com/laptop-hp-compaq) và [Apple](https://www.dienmayxanh.com/laptop-apple-macbook) đều đã đưa ra rất nhiều mẫu Laptop sử dụng các loại CPU khác nhau có giá thành từ bình dân đến cao cấp.

### **1.2. RAM – Random Access Memory.**

RAM (Random Access Memory) là bộ nhớ truy xuất ngẫu nhiên được sử dụng làm**nơi lưu trữ tạm dữ liệu** và lệnh thực thi của hệ điều hành và các ứng dụng trước khi ghi chúng lên ổ cứng khi kết thúc phiên làm việc. Khi người dùng mở một phần mềm trên laptop thì dữ liệu sẽ được truyền tải từ ổ đĩa cứng lên RAM và truyền tải vào CPU để xử lý, sau đó lưu ngược lại vào ổ cứng vì RAM có tốc độ rất nhanh hơn rất nhiều lần so với ổ cứng.

Dữ liệu trên RAM được lưu trên từng ô nhớ và mỗi ô nhớ đều có địa chỉ khác nhau, bên cạnh đó, thời gian để đọc và ghi dữ liệu trên cùng một ô nhớ là bằng nhau.

Nếu không có RAM, [máy tính](https://www.thegioididong.com/laptop) sẽ không thể thực hiện những tác vụ cơ bản bởi việc truy cập các tệp dữ liệu sẽ cực kỳ chậm.

### **1.3. Hardware – Ổ cứng.**

Ổ cứng máy tính hay còn gọi là ổ cứng (Hard Disk Drive, viết tắt: HDD) là thiết bị dùng để**lưu trữ dữ liệu**trong máy tính. Với sự phát triển nhanh chóng của công nghệ, ổ đĩa cứng ngày nay có kích thước mỏng, gọn, nhẹ nhưng dung lượng thì ngày càng tăng lên.

Ổ cứng ngoài việc phụ trách lưu trữ dữ liệu còn **liên quan trực tiếp** đến những vấn đề quan trọng khi sử dụng máy tính như: tốc độ khởi động máy, tốc độ chép xuất dữ liệu của máy, độ an toàn của dữ liệu cá nhân để trên máy.

Bất cứ các thao tác phần mềm trên máy tính của bạn như sao chép, cắt dán, khởi động phần mềm,… nhanh hay chậm đều **phụ thuộc vào cấu tạo phần cứng của ổ cứng** tốt hay không.

* Hiện nay có 2 loại ổ cứng phổ biến là HDD và SDD:
* **• HDD** là viết tắt của Hard Disk Drive hay ổ đĩa cứng truyền thống, dữ liệu được lưu trên các bề mặt phiến đĩa tròn làm bằng nhôm, thủy tinh hoặc gốm được phủ vật liệu từ tính. Tâm của đĩa có gắn một động cơ, khi hoạt động các tấm đĩa sẽ được quay bởi động cơ này để đọc ghi dữ liệu.
* **• SSD** là viết tắt của Solid State Drive, tức ổ cứng thể rắn, ra đời như một giải pháp thay thế cho tốc độ chậm chạp của HDD truyền thống. Bên cạnh tốc độ đọc ghi nhanh, SSD còn sở hữu vô số ưu điểm khác.

### **1.4. Card đồ hoạ.**

Nếu chỉ có bộ xử lý (CPU) và một màn hình hiển thị, thì chưa đủ để ta có thể nhìn thấy được sự sống động và trung thực của các khung hình hiển thị trên màn hình đó. Lúc này ta bắt đầu cần đến sự trợ giúp của card đồ họa. Như vậy, card đồ hoạ là một bộ phận chịu trách nhiệm xử lý các tác vụ liên quan đến đồ họa bao gồm hình ảnh, video trên một chiếc laptop, thông qua đó, mọi thứ sẽ trở nên sống động và mượt mà hơn.

Trên Laptop hiện nay có 2 loại card đồ họa chính đó là **card onboard (tích hợp)**và **card rời gắn ngoài.** Card onboard trước đây được nhà sản xuất mainboard (như Intel, AMD…) tích hợp sẵn trên bo mạch (mainboard) sau này là nằm trên vi xử lý (CPU).

Card đồ họa tích hợp thường đáp ứng nhu cầu đồ họa ở mức vừa phải. Nếu người dùng cần một chiếc máy mạnh để đáp ứng nhu cầu xử lý độ hoạ nặng thì cần chọn sản phẩm có card đồ họa rời. Card rời hỗ trợ nhiều tập lệnh cao cấp hơn, hiệu năng mạnh hơn có thể lên đến hàng chục lần.

### **1.5. Kích thước màn hình.**

Kích thước màn hình trên laptop rất đa dạng, phù hợp với mọi nhu cầu sử dụng của khách hàng, các hãng sản xuất lớn thường sẽ đưa ra các loại màn hình với kích thước tiêu chuẩn như: **12.5 inch, 13.3 inch, 14 inch, 15.6 inch...**

**• 11 – 12 inch:** Các dòng laptop mỏng nhất và nhẹ nhất đều có màn hình kích thước 11 – 12 inch, thường nặng 1,1 – 1,6 kg và có tính linh hoạt cao.

**• 13 – 14 inch:** Đạt được sự cân bằng giữa sự linh động và hiệu quả sử dụng, đặc biệt đúng với laptop có cân nặng dưới 1,8 kg.

**• 15 inch:** Kích thước phổ biến nhất, máy tính xách tay 15 inch thường nặng 2 – 3 kg.

**• 17 – 18 inch:** Dòng máy có kích thước 17 – 18 inch sẽ mang lại hiệu năng vượt trội, công suất xử lý mạnh mẽ để xử lý đồ hoạ nặng hoặc khả năng làm việc ở mức độ máy trạm (workstation).

## **2. Biểu diễn thi trức với Prolog.**

Đối với bài toán tư vấn mua sắm máy tính xách tay (dạng bài toán đưa ra quyết định) thì sử dụng phương pháp suy diễn tiến để biểu diễn tri thức là phương pháp phù hợp.

Tập các tri thức tiền đề:

**• Tình trạng máy:**

|  |  |
| --- | --- |
| **label** | new |
|  | old |

**• Phân loại máy:**

|  |  |
| --- | --- |
| **class** | common |
|  | luxury |

**• Hãng sản xuất:**

|  |  |
| --- | --- |
| **brand** | Acer |
|  | Asus |
| Dell |
| HP |
| Lenovo |
| Macbook |
| MSI |

**• Giá sản phẩm:**

|  |  |
| --- | --- |
| **price** | less\_than\_10m |
|  | 10\_15m |
|  | 15\_20m |
|  | less\_than\_20m |
|  | 20\_30m |
|  | greater\_than\_30m |

**• Ram:**

|  |  |
| --- | --- |
| **ram** | 4gb |
|  | 8gb |
|  | 16gb |

**• CPU:**

|  |  |
| --- | --- |
| **cpu** | core\_i3 |
|  | core\_i5 |
|  | core\_i7 |
|  | core\_i9 |
|  | ryzen\_3 |
|  | ryzen\_5 |
|  | ryzen\_7 |

**• Ổ cứng:**

|  |  |
| --- | --- |
| **hardware** | hdd |
|  | ssd |

**• Dung lượng ổ cứng:**

|  |  |
| --- | --- |
| **diskspace** | 128gb |
|  | 256gb |
|  | 500gb |
|  | 512gb |
|  | 1tb |

**• Kích thước màn hình:**

|  |  |
| --- | --- |
| **inch** | common |
|  | luxury |

**• Card đồ hoạ:**

|  |  |
| --- | --- |
| **graphic** | intel\_hd |
|  | intel\_uhd |
|  | intel\_iris\_plus |
|  | amd |
|  | nvidia |

Khi thực hiện suy diễn tiến với đầy đủ các tập luật trên chương trình sẽ đưa ra kết quả tương ứng với 240 giá trị khác nhau là tên của dòng máy tính xách tay trong tập luật kết quả ***laptop***.

**Ví dụ:**

laptop('dell\_inspiron\_n3481',label('new'),

class('common'),

brand('dell'),

price('10\_15m'),

ram('4gb'),

cpu('core\_i3'),

hardware('hdd'),

diskspace('1tb'),

inch('14'),

graphic('intel\_uhd'))

Tuy nhiên, trong tập luật kết quả ***laptop*** vẫn mang những giá trị “null” để khi đến một bước suy diễn nào đó mà các giá trị là tên của dòng máy đang có hiện hành không đáp ứng được nhu cầu của người sử dụng thì kết quả vẫn trả ra giác trị không có máy phù hợp, đề phòng chương trình bị lỗi.

**Ví dụ:**

laptop('null', label('new'),

class('common'),

brand('asus'),

price('10\_15m'),

ram('4gb'),

cpu('core\_i3'),

hardware('hdd'),

diskspace('1tb'),

inch('14'),

graphic('intel\_iris\_plus')).

# **CHƯƠNG IV. CÀI ĐẶT VÀ THỬ NGHIỆM CHƯƠNG TRÌNH**

## **1. Cài đặt.**

**•** Chương trình được trực quan hoá giao diện bởi ngôn ngữ lập trình C#.

**•** Dữ liệu tri thức được lưu trữ và tổng hợp bởi thông qua ngữ lập trình logic Prolog.

**•** Chương trình được chạy bằng môi trường Visual Studio.

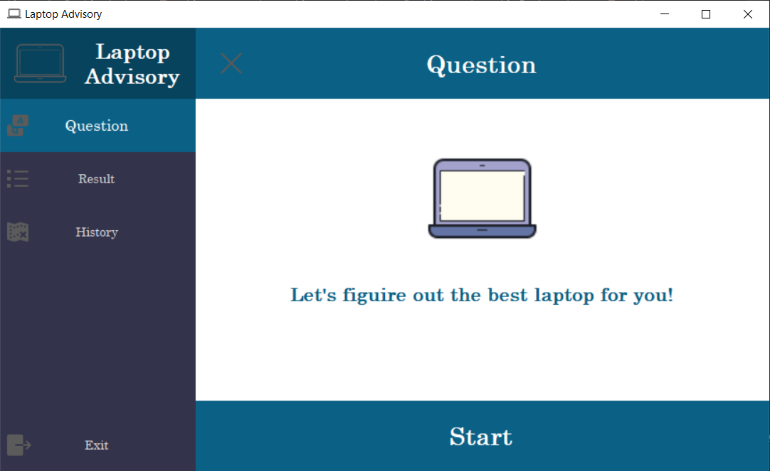
## **2. Thử nghiệm chương trình.**

Form giao diện khi khởi chạy chương trình.

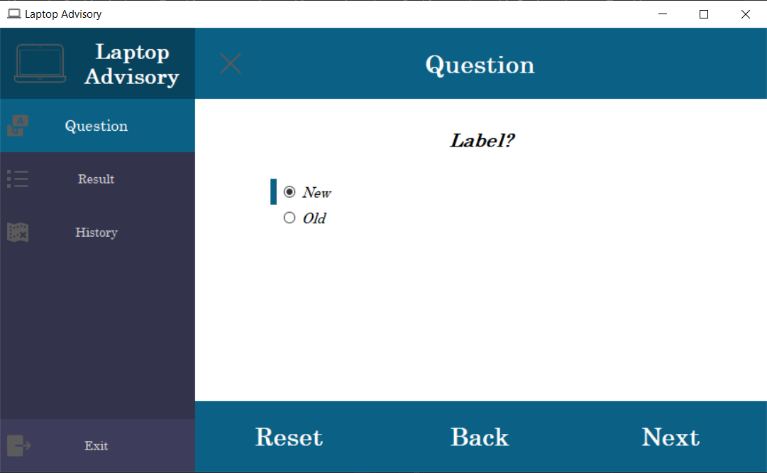
**• Tab Question:** form này cho phép người dụng lựa chọn các giá trị phù hợp về tình trạng máy mong muốn, giá thành, hãng sản xuất và thông số kị thuật theo nhu cầu cá nhân.

**• Tab Result:** form có chức năng trả về tất cả thông số cơ bản của máy tính phù hợp với nhu cầu của người sử dụng kèm theo hình ảnh và đường link dẫn thẳng đến trang web hiện hành đang mở bán sản phẩm tương ứng.

**• Tab History:** lưu lại tất cả những giá trị mà người dùng đã lựa chọn tương ứng với mỗi câu hỏi.

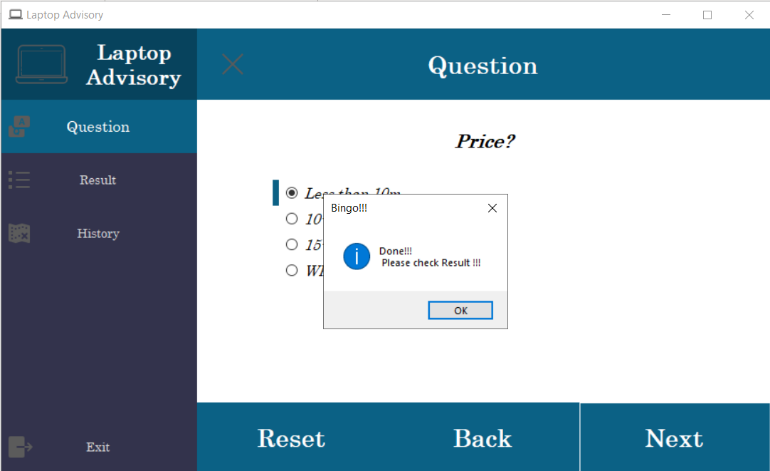


Khi người dung nhấn “Start” chương trình sẽ mở đầu với câu hỏi lựa chọn về dòng sản máy cũ hay máy mới.

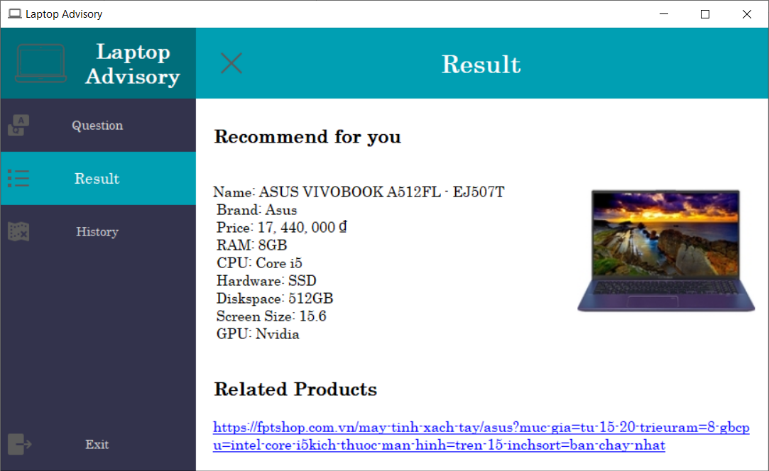


Sau khi chọn giá trì phù hợp với nhu cầu hiện tại, người dùng di chuyển chuột nhấn vào nút “Next” đế tiến đến những câu hỏi tiếp theo cho đến khi đạt được kết quả cuối cùng.

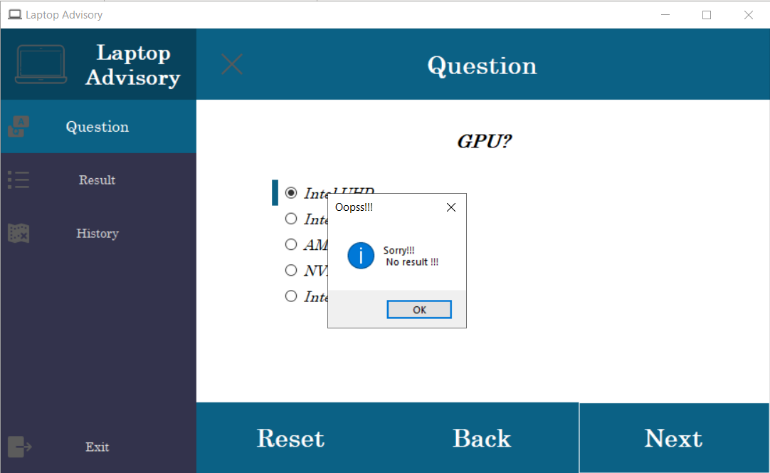
Khi có sản phẩm phù hợp bảng thông báo “Bingo!!!” sẽ hiện ra.



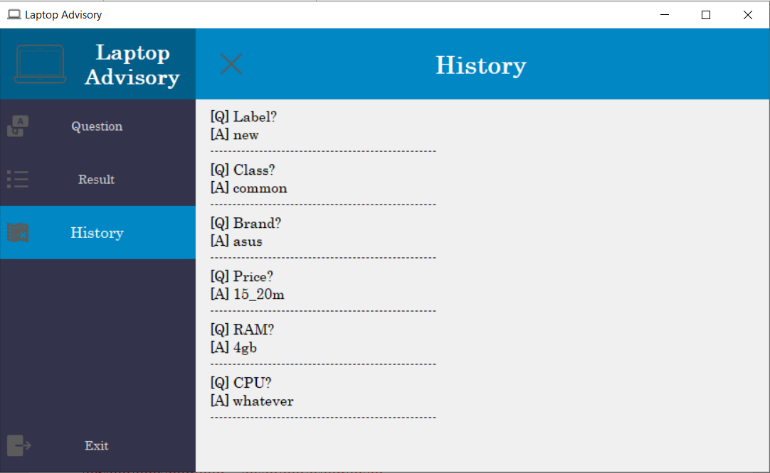
Người dùng nhấn vào tab Result sẽ nhận được kết quả thông số cơ bản của máy kèm hình ảnh máy và đường dẫn đến trang bán hàng.



Đối với những kết quả mà hệ thống chưa có khả năng đưa ra kết quả phù hợp để đáp ứng chon hu cầu của người dùng thì bảng thông báo “Oopss!!!” sẽ hiện ra kèm lời xin lỗi.



Khi người dùng có nhu cầu muốn xem lại những kết quả mà mình đã chọn thì có thể nhấn vào tab History để tra kết quả.



# **CHƯƠNG V. KẾT LUẬN VÀ TỰ ĐÁNH GIÁ**

## **1. Vấn đề mắc phải.**

Trong quá trình xây dựng chương trình, do số lượng tri thức thu thập chưa nhiều, lượng thông tin câu hỏi chưa đa dạng nên kết quả đưa ra vẫn chưa hoàn toàn phù hợp và chính xác.

Tri thức thu thập còn chưa dựa trên thị yếu người dùng nam hay nữ, phân loại rõ nhu cầu người sử dụng đang cần tím máy tính cho mục đích học tập, làm việc, đồ hoạ hay chơi game,…

## **2. Hướng phát triển của đề tài.**

Xây dựng mở rộng thêm tri thức, cho phép học và bổ sung thêm các luật vào cơ sở tri thức, cải tiến các tri thức giúp nâng cao độ chính xác.

Thử nghiệm phương thức biểu diễn tri thức bằng thông tin không chắc chắn dựa trên lý thuyết xác suất.

Mở rộng tư vấn mua sắm các sản phẩm công nghệ khác: điện thoại, máy tính bảng, tai nghe,…

## **3. Tự đánh giá.**

Chương trình còn nhiều hạn chế về kết quả tư vấn vì số lượng tập luật, số lượng dữ liệu còn bị hạn chế.

Giao diện còn đơn giản, chưa thật sự bắt mắt.