ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Trí Tuệ Nhân Tạo

(Artificial Intelligence)

Viện Công nghệ thông tin và Truyền thông
Trường Đại Học Bách Khoa Hà Nội

Nội dung môn học:

- Giới thiệu về Trí tuệ nhân tạo
- Tác tử
- Giải quyết vấn đề: Tìm kiếm, Thỏa mãn ràng buộc
- **Logic và suy diễn**
- Biểu diễn tri thức
- Biểu diễn tri thức không chắc chắn
- Học máy

Giới thiệu về logic

- **Logic** là ngôn ngữ hình thức cho phép (giúp) biểu diễn thông tin dưới dạng các kết luận có thể được đưa ra
  - Logic = Syntax + Semantics
- **Cú pháp (syntax)**: để xác định các mệnh đề (sentences) trong một ngôn ngữ.
- **Ngữ nghĩa (semantics)**: để xác định "ý nghĩa" của các mệnh đề trong một ngôn ngữ
  - Tức là, xác định sự đúng đắn của một mệnh đề
- Ví dụ: Trong ngôn ngữ của toán học
  - $(x+2 \geq y)$ là một mệnh đề; $(x+y > \{\})$ không phải là một mệnh đề
  - $(x+2 \geq y)$ là đúng nếu và chỉ nếu giá trị $(x+2)$ không nhỏ hơn giá trị y
  - $(x+2 \geq y)$ là đúng khi $x = 7, y = 1$
  - $(x+2 \geq y)$ là sai khi $x = 0, y = 6$

Cú pháp (syntax)

- Cú pháp = Ngôn ngữ + Lý thuyết chứng minh
- **Ngôn ngữ (Language)**
  - Các ký hiệu (symbols), biểu thức (expressions), thuật ngữ (terms), công thức (formulas) hợp lệ
  - Ví dụ: *one plus one equal two*
- **Lý thuyết chứng minh (Proof theory)**
  - Tập hợp các luật suy diễn cho phép chứng minh (suy luận ra) các biểu thức
  - Ví dụ: Luật suy diễn *any plus zero* ⊢ *any*
- Một **định lý (theorem)** là một mệnh đề logic cần chứng minh
- Việc chứng minh một định lý không cần phải xác định ngữ nghĩa (interpretation) của các ký hiệu!

Ngữ nghĩa (semantics)

- Ngữ nghĩa = Ý nghĩa (diễn giải) của các ký hiệu
- Ví dụ
  - $\mathcal{I}$(*one*) nghĩa là **1** ($\in$ N)
  - $\mathcal{I}$(*two*) nghĩa là **2** ($\in$ N)
  - $\mathcal{I}$(*plus*) nghĩa là phép cộng **+** : N x N $\rightarrow$ N
  - $\mathcal{I}$(*equal*) nghĩa là phép so sánh bằng **=** : N x N $\rightarrow$ {*true*, *false*}
  - $\mathcal{I}$(*one plus one equal two*) nghĩa là *true*
- Nếu diễn giải của một biểu thức là đúng (true), chúng ta nói rằng phép diễn giải này là một **mô hình (model)** của biểu thức
- Một biểu thức đúng đối với bất kỳ phép diễn giải nào thì được gọi là một biểu thức **đúng đắn (valid)**
  - Ví dụ: A OR NOT A

Tính bao hàm

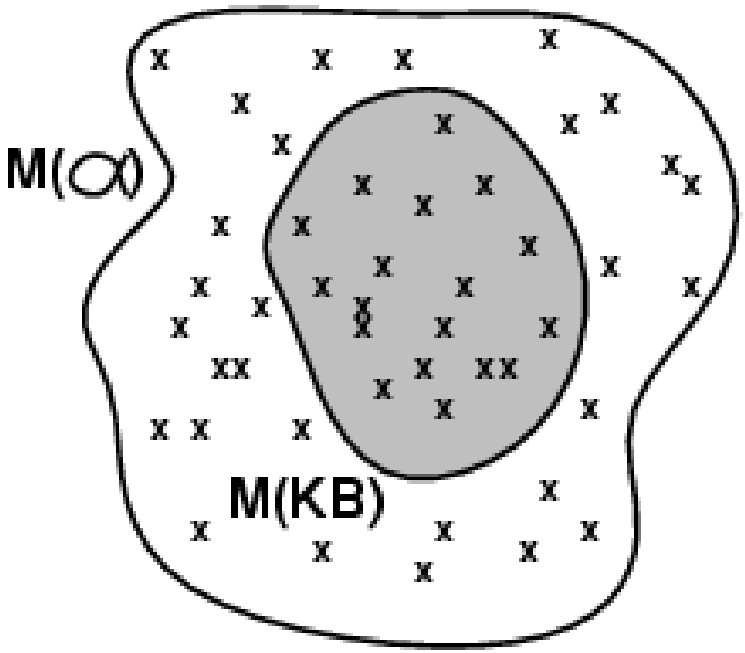
- Tính bao hàm có nghĩa là một cái gì đó tuân theo (bị hàm chứa ý nghĩa bởi, được suy ra từ) một cái gì khác:

$$KB \models \alpha$$

- Một cơ sở tri thức *KB* **bao hàm** (hàm chứa) mệnh đề α nếu và chỉ nếu α là đúng **trong mọi mô hình** (thế giới) mà trong đó *KB* là đúng. Tức là: nếu *KB* đúng, thì α cũng phải đúng
  - Ví dụ: Nếu một cơ sở tri thức *KB* chứa các mệnh đề "Đội bóng A đã thắng" và "Đội bóng B đã thắng", thì *KB* bao hàm mệnh đề "Đội bóng A hoặc đội bóng B đã thắng"
  - Ví dụ: Mệnh đề (x+y = 4) bao hàm mệnh đề (4 = x+y)

Các mô hình

- Các nhà logic học thường hay xem xét các sự việc theo các mô hình (models)
- Các mô hình là các không gian (thế giới) có cấu trúc, mà trong các không gian đó tính đúng đắn (của các sự việc) có thể đánh giá được
- **Định nghĩa:** *m* là một mô hình của mệnh đề α nếu α là đúng trong *m*
- *M(α)* là tập hợp tất cả các mô hình của α
- KB ⊨ α nếu và chỉ nếu $M(KB) \subseteq M(\alpha)$
  - Ví dụ: *KB* = "Đội bóng A đã thắng và đội bóng B đã thắng", α = "Đội bóng A đã thắng"



Suy diễn logic (1)

- $KB \vdash_i \alpha$
  - Mệnh đề α **được suy ra** từ *KB* bằng cách áp dụng thủ tục (suy diễn) *i*
  - (Nói cách khác) Thủ tục *i* **suy ra** mệnh đề α từ *KB*
- **Tính đúng đắn (soundness)**
  - Một thủ tục suy diễn *i* được gọi là **đúng đắn (sound)**, nếu thủ tục *i* suy ra **chỉ** các mệnh đề được bao hàm (entailed sentences)
  - Thủ tục *i* là đúng đắn, nếu bất cứ khi nào $KB \vdash_i \alpha$, thì cũng đúng đối với $KB \models \alpha$
  - Nếu thủ tục *i* suy ra mệnh đề α, mà α không được bao hàm trong *KB*, thì thủ tục *i* là không đúng đắn (unsound)

Suy diễn logic (2)

- **Tính hoàn chỉnh (completeness)**
  - Một thủ tục suy diễn *i* được gọi là **hoàn chỉnh (complete)**, nếu thủ tục *i* có thể suy ra **mọi** mệnh đề được bao hàm (entailed sentences)
  - Thủ tục *i* là hoàn chỉnh, nếu bất cứ khi nào $KB \models \alpha$, thì cũng đúng đối với $KB \vdash_i \alpha$
- (Trong phần tiếp theo của bài giảng) chúng ta sẽ xét đến logic vị từ bậc 1 (first-order logic)
  - Có khả năng biểu diễn (diễn đạt) hầu hết các phát biểu logic
  - Với logic vị từ bậc 1, tồn tại một thủ tục suy diễn *đúng đắn* và *hoàn chỉnh*

Suy diễn logic (3)

- Logic là một cách để biểu diễn hình thức và suy diễn tự động
- Việc suy diễn (reasoning) có thể được thực hiện ở mức cú pháp (bằng các chứng minh): **suy diễn diễn dịch (deductive reasoning)**
- Việc suy diễn có thể được thực hiện ở mức ngữ nghĩa (bằng các mô hình): **suy diễn dựa trên mô hình (model-based reasoning)**

Suy diễn logic (4)

- Suy diễn ngữ nghĩa ở mức của một phép diễn giải (mô hình):
  - Với một biểu thức, có tồn tại một mô hình không? **có thể thỏa mãn được (satisfiability)?**
  - Với một biểu thức và một phép diễn giải, kiểm tra xem phép diễn giải có phải là một mô hình của biểu thức không?: **kiểm tra mô hình (model checking)**
- Suy diễn ngữ nghĩa ở mức của tất cả các phép diễn giải có thể: **kiểm tra tính đúng đắn (validity checking)**

Logic định đề: Cú pháp (1)

- Logic định đề (propositional logic) là loại logic đơn giản nhất
- **Biểu thức định đề (propositional formula)**
  - Một ký hiệu định đề (S_1, S_2, …) là một biểu thức (định đề)
  - Các giá trị hằng logic **đúng (true)** và **sai (false)** là các biểu thức
  - Nếu S_1 là một biểu thức, thì ($\neg S_1$) cũng là một biểu thức (Phép **phủ định**)

Logic định đề: Cú pháp (2)

- **Biểu thức định đề (propositional formula)…**
  - Nếu S_1 và S_2 là các biểu thức, thì $(S_1 \wedge S_2)$ cũng là một biểu thức (Phép **kết hợp** / **và**)
  - Nếu S_1 và S_2 là các biểu thức, thì $(S_1 \vee S_2)$ cũng là một biểu thức (Phép **tuyển** / **hoặc**)
  - Nếu S_1 và S_2 là các biểu thức, thì $(S_1 \Rightarrow S_2)$ cũng là một biểu thức (Phép **suy ra** / **kéo theo**)
  - Nếu S_1 và S_2 là các biểu thức, thì $(S_1 \Leftrightarrow S_2)$ cũng là một biểu thức (Phép **tương đương**)
  - Không gì khác (các dạng trên) là một biểu thức

Cú pháp của logic định đề: Ví dụ

- p
- q
- r
- true
- false
- $\neg p$
- $(\neg p) \wedge \text{true}$
- $\neg((\neg p) \vee \text{false})$
- $(\neg p) \Rightarrow (\neg((\neg p) \vee \text{false}))$
- $(p \wedge (q \vee r)) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$

Thứ tự ưu tiên của các toán tử logic

- Thứ tự ưu tiên của các toán tử logic (từ cao xuống thấp)
  - $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- Sử dụng cặp ký tự "()" để xác định mức độ ưu tiên
- Các ví dụ
  - $p \wedge q \vee r$ tương đương $(p \wedge q) \vee r$
    chứ không phải $p \wedge (q \vee r)$
  - $\neg p \wedge q$ tương đương $(\neg p) \wedge q$
    chứ không phải $\neg(p \wedge q)$
  - $p \wedge \neg q \Rightarrow r$ tương đương $(p \wedge (\neg q)) \Rightarrow r$
    chứ không phải $p \wedge (\neg(q \Rightarrow r))$ hoặc $p \wedge ((\neg q) \Rightarrow r)$

Logic định đề: Ngữ nghĩa (1)

- Với một mô hình (model) cụ thể, nó sẽ xác định giá trị đúng/sai cho mỗi ký hiệu định đề
  - Ví dụ: Với 3 ký hiệu S_1, S_2 và S_3, thì có thể lấy ví dụ một mô hình m_1 xác định như sau:

    $m_1 \equiv (S_1=\text{sai}, S_2=\text{đúng}, S_3=\text{sai})$

- Với 3 ký hiệu định đề như ví dụ trên, có thể chỉ ra 8 mô hình có thể

Logic định đề: Ngữ nghĩa (2)

- Ngữ nghĩa của một mô hình *m* = Các quy tắc để đánh giá giá trị chân lý (đúng/sai) của các mệnh đề trong mô hình *m* đó

  $\neg S_1$ là đúng, khi và chỉ khi S_1 là sai

  $S_1 \wedge S_2$ là đúng, khi và chỉ khi S_1 là đúng và S_2 là đúng

  $S_1 \vee S_2$ là đúng, khi và chỉ khi S_1 là đúng hoặc S_2 là đúng

  $S_1 \Rightarrow S_2$ là đúng, khi và chỉ khi S_1 là sai hoặc S_2 là đúng;

  là sai, khi và chỉ khi S_1 là đúng và S_2 là sai

  $S_1 \Leftrightarrow S_2$ là đúng, khi và chỉ khi $S_1 \Rightarrow S_2$ là đúng và $S_2 \Rightarrow S_1$ là đúng

- Ví dụ: Với mô hình m_1 như trong ví dụ trước, thì giá trị của biểu thức logic định đề sau sẽ là:

  $\neg S_1 \wedge (S_2 \vee S_3)$ = *đúng* $\wedge$ (*đúng* $\vee$ *sai*) = *đúng* $\wedge$ *đúng* = *đúng*

Ngữ nghĩa của logic định đề: Ví dụ (1)

- Xét mô hình $m_1 \equiv (p=đúng, q=sai)$, ta có ngữ nghĩa (giá trị logic) của các biểu thức sau
  - $\neg p$ là *sai*
  - $\neg q$ là *đúng*
  - $p \wedge q$ là sai
  - $p \vee q$ là đúng
  - $p \Rightarrow q$ là *sai*
  - $q \Rightarrow p$ là *đúng*
  - $p \Leftrightarrow q$ là sai
  - $\neg p \Leftrightarrow q$ là đúng

Ngữ nghĩa của logic định đề: Ví dụ (2)

- Xét mô hình $m_2 \equiv (p=sai, q=đúng)$, ta có ngữ nghĩa (giá trị logic) của các biểu thức sau
  - $\neg p$ là *đúng*
  - $\neg q$ là *sai*
  - $p \wedge q$ là sai
  - $p \vee q$ là đúng
  - $p \Rightarrow q$ là *đúng*
  - $q \Rightarrow p$ là *sai*
  - $p \Leftrightarrow q$ là sai
  - $\neg p \Leftrightarrow q$ là đúng

Bảng chân lý đối với các toán tử logic

S_1	S_2	$\neg S_1$	$S_1 \wedge S_2$	$S_1 \vee S_2$	$S_1 \Rightarrow S_2$	$S_1 \Leftrightarrow S_2$
sai	sai	đúng	sai	sai	đúng	đúng
sai	đúng	đúng	sai	đúng	đúng	sai
đúng	sai	sai	sai	đúng	sai	sai
đúng	đúng	sai	đúng	đúng	đúng	đúng

Tương đương logic

- Hai mệnh đề được gọi là tương đương logic khi và chỉ khi hai mệnh đề này luôn đúng trong cùng mô hình:

$$\alpha \equiv \beta \text{ khi và chỉ khi } \alpha \models \beta \text{ và } \beta \models \alpha$$

$$\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{de Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}$$

Biểu diễn bằng logic định đề: Ví dụ

- Giả sử chúng ta có các định đề sau
  - p ≡ "Chiều nay trời nắng"
  - q ≡ "Thời tiết lạnh hơn hôm qua"
  - r ≡ "Tôi sẽ đi bơi"
  - s ≡ "Tôi sẽ đi đá bóng"
  - t ≡ "Tôi sẽ về đến nhà vào buổi tối"
- Biểu diễn các phát biểu trong ngôn ngữ tự nhiên
  - "Chiều nay trời *không* nắng và thời tiết lạnh hơn hôm qua": $\neg p \wedge q$
  - "Tôi sẽ đi bơi *nếu như* chiều nay trời nắng": $p \rightarrow r$
  - "*Nếu* tôi (sẽ) *không* đi bơi *thì* tôi sẽ đi đá bóng": $\neg r \rightarrow s$
  - "Nếu tôi (sẽ) đi đá bóng thì tôi sẽ về nhà vào buổi tối": $s \rightarrow t$

Mâu thuẫn và Tautology

- Một biểu thức logic định đề luôn có giá trị sai (false) trong mọi phép diễn giải (mọi mô hình) thì được gọi là một **mâu thuẫn (contradiction)**
  - Ví dụ: $(p \wedge \neg p)$
- Một biểu thức logic định đề luôn có giá trị đúng (true) trong mọi phép diễn giải (mọi mô hình) thì được gọi là một **tautology**
  - Ví dụ: $(p \vee \neg p)$

    $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$

    $\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$

Tính thỏa mãn được và Tính đúng đắn

- Một biểu thức logic định đề là **thỏa mãn được (satisfiable)**, nếu biểu thức đó đúng trong *một mô mình* nào đó
  - Ví dụ: $A \vee B$, $A \wedge B$
- Một biểu thức là **không thể thỏa mãn được (unsatisfiable)**, nếu *không tồn tại bất kỳ mô hình nào* mà trong đó biểu thức là đúng
  - Ví dụ: $A \wedge \neg A$
- Một biểu thức là **đúng đắn (valid)**, nếu biểu thức đúng trong *mọi mô hình*
  - Ví dụ: *đúng*; $A \vee \neg A$; $A \Rightarrow A$; $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Bài toán chứng minh logic

- Với một cơ sở tri trức (một tập các mệnh đề) *KB* và một mệnh đề α cần chứng minh (gọi là một định lý)
- Cơ sở tri thức *KB* có bao hàm (về mặt ngữ nghĩa) α hay không: KB $\models \alpha$?
  - Nói cách khác, α có thể được suy ra (được chứng minh) từ cơ sở tri thức KB hay không?
- Câu hỏi đặt ra: Liệu có tồn tại một thủ tục (suy diễn) có thể giải quyết được bài toán chứng minh logic, trong một số hữu hạn các bước?
  - Đối với logic định đề, câu trả lời là có!

Giải quyết bài toán chứng minh logic

- Mục đích: để trả lời câu hỏi KB $\models \alpha$?
- Có 3 phương pháp (chứng minh) phổ biến:
  - Sử dụng bảng chân lý (Truth-table)
  - Áp dụng các luật suy diễn (Inference rules)
  - Chuyển về bài toán chứng minh thỏa mãn (SAT)
    - Phương pháp chứng minh bằng phản chứng (Refutation)

Chứng minh dựa trên bảng chân lý (1)

- Bài toán chứng minh: KB $\models \alpha$?
- Kiểm tra tất cả các phép diễn giải có thể (tất cả các mô hình có thể) mà trong đó KB là đúng, để xem α đúng hay sai
- Bảng chân lý: Liệt kê các giá trị chân lý (đúng/sai) của các mệnh đề, đối với tất cả các phép diễn giải có thể
  - Các phép gán giá trị đúng/sai đối với các ký hiệu định đề

		KB		α
p	q	$p \vee q$	$p \leftrightarrow q$	$(p \vee \neg q) \wedge q$
đúng	đúng	đúng	đúng	đúng
đúng	sai	đúng	sai	sai
sai	đúng	đúng	sai	sai
sai	sai	sai	đúng	sai

← chứng minh

Chứng minh dựa trên bảng chân lý (2)

- KB = $(p \vee r) \wedge (q \vee \neg r)$
- $\alpha = (p \vee q)$
- KB $\models \alpha$?

p	q	r	$p \vee r$	$q \vee \neg r$	KB	α
đúng	đúng	đúng	đúng	đúng	đúng	đúng
đúng	đúng	sai	đúng	đúng	đúng	đúng
đúng	sai	đúng	đúng	sai	sai	đúng
đúng	sai	sai	đúng	đúng	đúng	đúng
sai	đúng	đúng	đúng	đúng	đúng	đúng
sai	đúng	sai	sai	đúng	sai	đúng
sai	sai	đúng	đúng	sai	sai	sai
sai	sai	sai	sai	đúng	sai	sai

Chứng minh dựa trên bảng chân lý (3)

- Đối với logic định đề, phương pháp chứng minh dựa trên bảng chân lý có tính *đúng đắn (sound)* và *hoàn chỉnh (complete)*
- Độ phức tạp tính toán của phương pháp chứng minh dựa trên bảng chân lý
  - Hàm mũ đối với số lượng (n) các ký hiệu định đề: 2^n
  - Nhưng chỉ có một tập con (nhỏ) của tập các khả năng gán giá trị chân lý, mà trong đó KB và α là đúng

Chứng minh bằng các luật suy diễn (1)

- Luật suy diễn **Modus ponens**

$$\frac{p \rightarrow q,\ \ p}{q}$$

- Luật suy diễn loại bỏ liên kết VÀ (**And-Elimination**)

$$\frac{p_1 \wedge p_2 \wedge \ldots \wedge p_n}{p_i} \qquad (i=1..n)$$

- Luật suy diễn đưa vào liên kết VÀ (**And-Introduction**)

$$\frac{p_1, p_2, \ldots, p_n}{p_1 \wedge p_2 \wedge \ldots \wedge p_n}$$

- Luật suy diễn đưa vào liên kết HOẶC (**Or-Introduction**)

$$\frac{p_i}{p_1 \vee p_2 \vee \ldots \vee p_i \vee \ldots \vee p_n}$$

Chứng minh bằng các luật suy diễn (2)

- Luật suy diễn loại bỏ phủ định hai lần (**Elimination of Double Negation**)

$$\frac{\neg\neg p}{p}$$

- Luật suy diễn hợp giải (**Resolution**)

$$\frac{p \vee q, \quad \neg q \vee r}{p \vee r}$$

- Luật suy diễn hợp giải đơn (**Unit Resolution**)

$$\frac{p \vee q, \quad \neg q}{p}$$

- Tất cả các luật suy diễn trên đều có tính *đúng đắn (sound)*!

Chứng minh bằng luật suy diễn: Ví dụ (1)

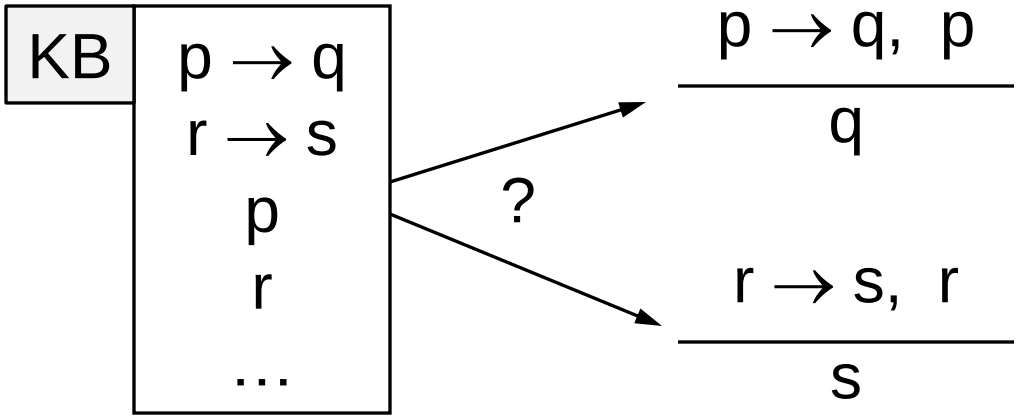
- Giả sử có tập giả thiết KB
  1) $p \wedge q$
  2) $p \rightarrow r$
  3) $(q \wedge r) \rightarrow s$
- Cần chứng minh định lý *s*
- Từ 1) và sử dụng luật And-Elimination, ta có:
  4) p
- Từ 2), 4), và sử dụng luật Modus Ponens, ta có:
  5) r

Chứng minh bằng luật suy diễn: Ví dụ (2)

- …
- Từ 1), và sử dụng luật And-Elimination, ta có:
  6) q
- Từ 5), 6), và sử dụng luật And-Introduction, ta có:
  7) $(q \wedge r)$
- Từ 7), 3), và sử dụng luật Modus-Ponens, ta có:
  8) s
- Vậy định lý (biểu thức logic) *s* được chứng minh là đúng!

Suy diễn logic và Tìm kiếm

- Để chứng minh định lý α là đúng đối với tập giả thiết *KB*, cần áp dụng một chuỗi các luật suy diễn đúng đắn
- Vấn đề: Ở mỗi bước suy diễn, có nhiều luật có thể áp dụng được
  - Chọn luật nào để áp dụng tiếp theo?
- Đây là vấn đề của bài toán tìm kiếm (search)



Chuyển đổi các biểu thức logic

- Trong logic định đề
  - Một biểu thức có thể bao gồm nhiều liên kết: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
  - Một biểu thức có thể bao gồm nhiều biểu thức con (lồng) khác
- Chúng ta có cần sử dụng tất cả các liên kết logic để biểu diễn một biểu thức phức tạp?
  - Không.
  - Chúng ta có thể viết lại (chuyển đổi) một biểu thức logic định đề thành một biểu thức tương đương *chỉ chứa các liên kết* $\neg, \wedge, \vee$

Các dạng chuẩn

- Các biểu thức trong logic định đề có thể được chuyển đổi về một trong các dạng chuẩn (Normal forms)
  - Giúp đơn giản hóa quá trình suy diễn
- **Dạng chuẩn kết hợp** (Conjunctive normal form – CNF)
  - Là kết hợp (liên kết VÀ) của các mệnh đề (clauses)
  - Mỗi mệnh đề (clause) là một liên kết HOẶC của các ký hiệu định đề đơn
  - Ví dụ: $(p \vee q) \wedge (\neg q \vee \neg r \vee s)$
- **Dạng chuẩn tuyển** (Disjunctive normal form – DNF)
  - Là liên kết HOẶC của các mệnh đề (clauses)
  - Mỗi mệnh đề (clause) là một liên kết VÀ của các ký hiệu định đề đơn
  - Ví dụ: $(p \wedge \neg q) \vee (\neg p \wedge r) \vee (r \wedge \neg s)$

Chuyển đổi về dạng chuẩn CNF: Ví dụ

Chuyển đổi về dạng chuẩn CNF: $\neg(p \rightarrow q) \vee (r \rightarrow p)$

1. Loại bỏ các liên kết $\rightarrow, \leftrightarrow$

   $\neg(\neg p \vee q) \vee (\neg r \vee p)$

2. Sử dụng các phép biến đổi tương đương (vd: luật De Morgan và phép phủ định 2 lần)

   $(p \wedge \neg q) \vee (\neg r \vee p)$

3. Sử dụng các luật kết hợp (associative rules) và phân bố (distributive rules)

   $(p \vee \neg r \vee p) \wedge (\neg q \vee \neg r \vee p)$

   $(p \vee \neg r) \wedge (\neg q \vee \neg r \vee p)$

Bài toán chứng minh thỏa mãn (SAT)

- Mục đích của bài toán chứng minh thỏa mãn (Satisfiability – SAT) là xác định một biểu thức ở dạng chuẩn kết hợp (CNF) có thể thỏa mãn được hay không
  - Tức là chứng minh biểu thức đó là đúng hay không
  - Ví dụ: $(p \vee q \vee \neg r) \wedge (\neg p \vee \neg r \vee s) \wedge (\neg p \vee q \vee \neg t)$
- Đây là một trường hợp của bài toán thỏa mãn ràng buộc (CSP)
  - Tập các biến
    - Các ký hiệu định đề (ví dụ: p, q, r, s, t)
    - Các giá trị (hằng) logic *đúng*, *sai*
  - Tập các ràng buộc
    - Tất cả các mệnh đề (được liên kết bởi phép VÀ) trong biểu thức phải đúng
    - Với mỗi mệnh đề, ít nhất một trong các định đề đơn phải đúng

Giải quyết bài toán SAT

- Phương pháp **Backtracking**
  - Áp dụng chiến lược tìm kiếm theo chiều sâu (Depth-first search)
  - Xét một biến (một định đề đơn), xét các khả năng gán giá trị (đúng/sai) cho biến đó
  - Lặp lại, cho đến khi tất cả các biến được gán giá trị, hoặc việc gán giá trị cho tập con của tập tất cả các biến, làm cho **biểu thức là sai**
- Các phương pháp **tối ưu hóa lặp (Iterative optimization methods)**
  - Bắt đầu với một phép gán ngẫu nhiên các giá trị đúng/sai cho các ký hiệu định đề
  - Đổi giá trị (*đúng* thành *sai* / *sai* thành *đúng*) đối với một biến
  - Heuristic: ưu tiên các phép gán giá trị làm cho nhiều mệnh đề (hơn) đúng
  - Sử dụng các phương pháp tìm kiếm cục bộ: Simulated Annealing, Walk-SAT

Bài toán suy diễn vs. Bài toán thỏa mãn được

- **Bài toán suy diễn logic**
  - Cần chứng minh: biểu thức logic (định lý) α được bao hàm bởi tập các mệnh đề *KB*
  - Nói cách khác: với mọi phép diễn giải mà trong đó *KB* đúng, thì α có đúng?
- **Bài toán thỏa mãn được (SAT)**
  - Có tồn tại một phép gán giá trị đúng/sai cho các ký hiệu định đề (một phép diễn giải) sao cho biểu thức α là đúng?
- Mối quan hệ:

$KB \models \alpha$	nếu và chỉ nếu:
$(KB \wedge \neg\alpha)$	là **không thể thỏa mãn được (unsatisfiable)**

Luật suy diễn hợp giải (1)

- Luật suy diễn **hợp giải (Resolution)**

$$\frac{p \vee q, \quad \neg q \vee r}{p \vee r}$$

- Luật suy diễn hợp giải áp dụng được đối với các biểu thức logic ở dạng chuẩn CNF
- Luật suy diễn hợp giải có tính *đúng đắn (sound)*, nhưng *không có tính hoàn chỉnh (incomplete)*
  - Tập giả thiết (cơ sở tri thức) *KB* chứa biểu thức $(p \wedge q)$
  - Cần chứng minh: $(p \vee q)$?
  - Luật suy diễn hợp giải không thể suy ra được biểu thức cần chứng minh!

Luật suy diễn hợp giải (2)

- Chuyển bài toán chứng minh logic về bài toán SAT
  - Phương pháp chứng minh bằng phản chứng
  - Việc chứng minh sự mâu thuẫn của: $(KB \wedge \neg\alpha)$
  - Tương đương việc chứng minh sự bao hàm: $KB \models \alpha$
- Luật suy diễn hợp giải (Resolution rule)
  - Nếu các biểu thức trong tập *KB* và biểu thức $\neg\alpha$ đều ở dạng CNF, thì áp dụng luật suy diễn hợp giải sẽ xác định tính (không) thỏa mãn được của $(KB \wedge \neg\alpha)$

Giải thuật hợp giải

- Chuyển đổi tất cả các biểu thức trong KB về dạng chuẩn CNF
- Áp dụng liên tiếp luật suy diễn hợp giải (Resolution rule) bắt đầu từ: $(KB \wedge \neg\alpha)$
  - KB là kết hợp của các biểu thức ở dạng chuẩn CNF
  - Do đó, $(KB \wedge \neg\alpha)$ cũng là một biểu thức ở dạng chuẩn CNF!
- Quá trình áp dụng luật suy diễn hợp giải dừng lại khi:
  - Có mâu thuẫn xảy ra
    - Sau khi hợp giải, thu được (suy ra) biểu thức rỗng (mâu thuẫn)

$$\frac{p, \quad \neg p}{\{\}}$$

  - Không có biểu thức mới nào được sinh ra nữa

Kết luận sao?

Chứng minh bằng hợp giải: Ví dụ (1)

- Giả sử có tập giả thiết KB
  - $p \wedge q$
  - $p \rightarrow r$
  - $(q \wedge r) \rightarrow s$
- Cần chứng minh định lý *s*
- Bước 1. Chuyển đổi KB về dạng chuẩn CNF
  - $(p \rightarrow r)$ được chuyển thành $(\neg p \vee r)$
  - $((q \wedge r) \rightarrow s)$ được chuyển thành $(\neg q \vee \neg r \vee s)$
- Bước 2. Phủ định biểu thức cần chứng minh
  - $\neg s$
- Bước 3. Áp dụng liên tiếp luật hợp giải đối với $(KB \wedge \neg\alpha)$:

$\{p,\ q,\ \neg p \vee r,\ \neg q \vee \neg r \vee s,\ \neg s\}$

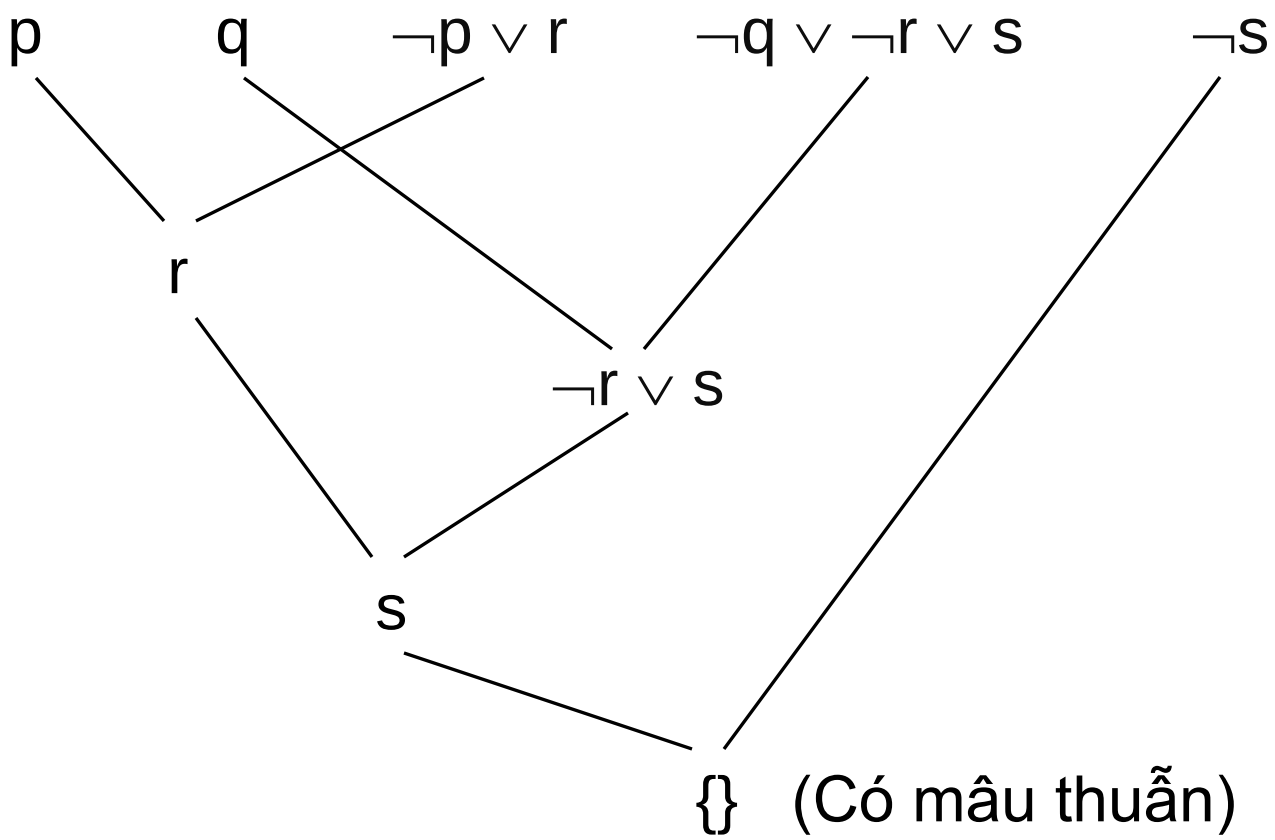
Chứng minh bằng hợp giải: Ví dụ (2)

- Bắt đầu quá trình hợp giải, ta có tập các mệnh đề:

1) p
2) q
3) $\neg p \vee r$
4) $\neg q \vee \neg r \vee s$
5) $\neg s$

- Hợp giải 1) và 3), ta thu được
  6) r
- Hợp giải 2) và 4), ta thu được
  7) $\neg r \vee s$
- Hợp giải 6) và 7), ta thu được
  8) s
- Hợp giải 8) và 5), ta thu được mâu thuẫn ({})
- Tức là biểu thức ban đầu (*s*) được chứng minh là đúng

Chứng minh bằng hợp giải: Ví dụ (3)



Dạng chuẩn Horn

- Một biểu thức logic ở dạng chuẩn Horn nếu:
  - Biểu thức đó là một liên kết VÀ của các mệnh đề
  - Mỗi mệnh đề là một liên kết HOẶC các ký hiệu (literals), và có tối đa là 1 ký hiệu khẳng định (positive literal)
  - Ví dụ: $(p \vee \neg q) \wedge (\neg p \vee \neg r \vee s)$
- Không phải mọi biểu thức logic định đề đều có thể được chuyển về dạng chuẩn Horn!
- Biểu diễn tập giả thiết KB ở dạng chuẩn Horn
  - **Các luật (Rules)**
    - $(\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n \vee q)$
    - Tương đương với luật: $(p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow q)$
  - **Các sự kiện (Facts)**
    - p, q
  - **Các ràng buộc toàn vẹn (Integrity constraints)**
    - $(\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n)$
    - Tương đương với luật: $(p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow \textit{sai})$

Luật suy diễn Modus Ponens tổng quát

$$\frac{(p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow q),\ p_1,\ p_2,\ \ldots,\ p_n}{q}$$

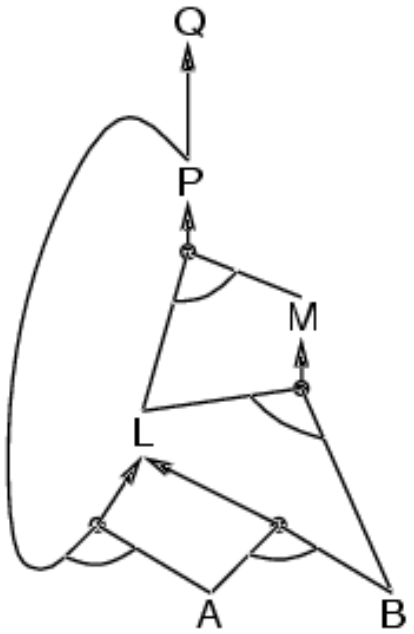
- Luật suy diễn Modus Ponens có tính *đúng đắn (sound)* và *hoàn chỉnh (complete)*, đối với các ký hiệu định đề và đối với tập các biểu thức KB ở dạng chuẩn Horn

- Luật suy diễn Modus Ponens có thể được sử dụng với cả 2 chiến lược suy diễn (suy diễn tiến và suy diễn lùi)

Suy diễn tiến (forward chaining)

- Với một tập các mệnh đề giả thiết (cơ sở tri thức) *KB*, cần suy ra mệnh đề kết luận *Q*
- Ý tưởng: Lặp lại 2 bước sau cho đến khi suy ra được kết luận
  - Áp dụng các luật có mệnh đề giả thiết được thỏa mãn trong *KB*
  - Bổ sung kết luận của các luật đó vào *KB*

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Suy diễn tiến: Ví dụ (1)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

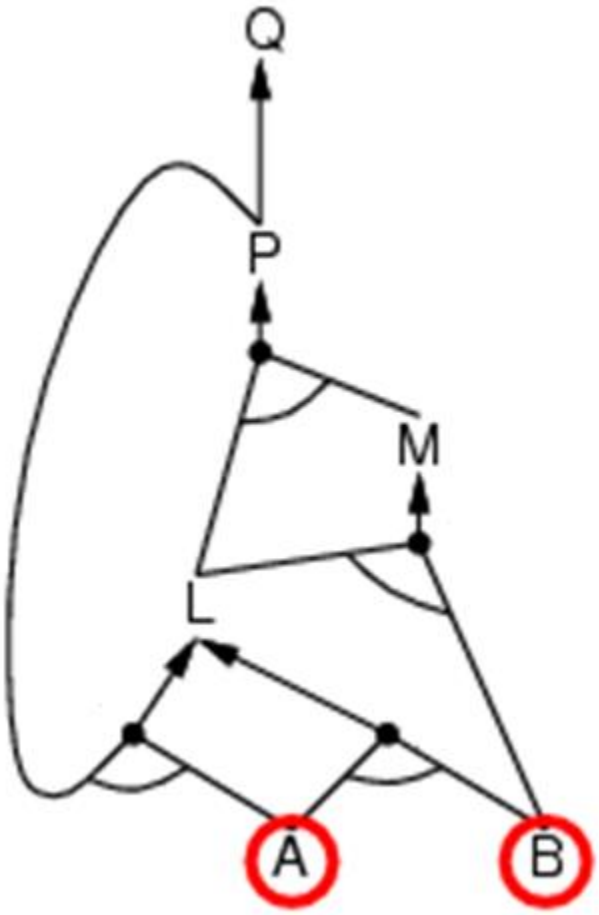
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (2)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

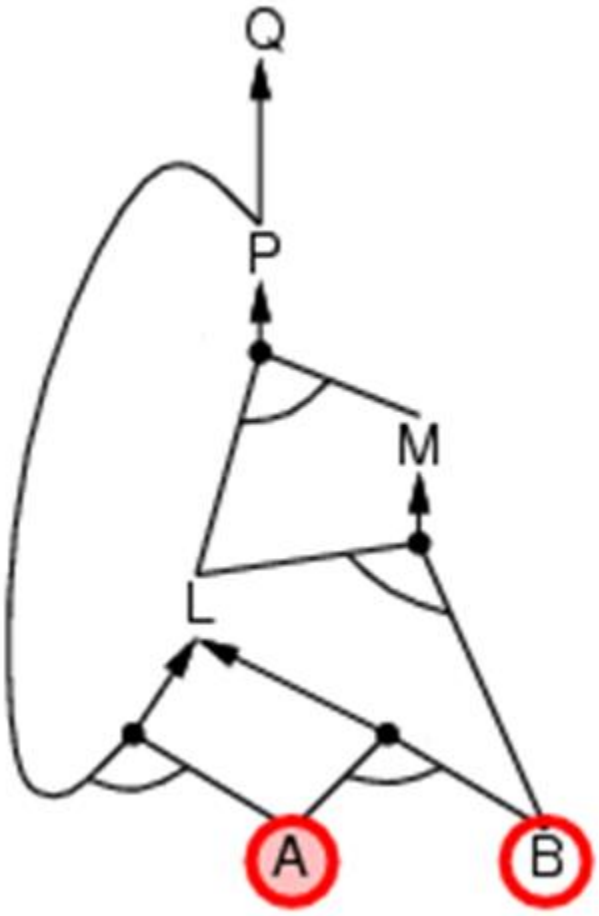
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (3)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

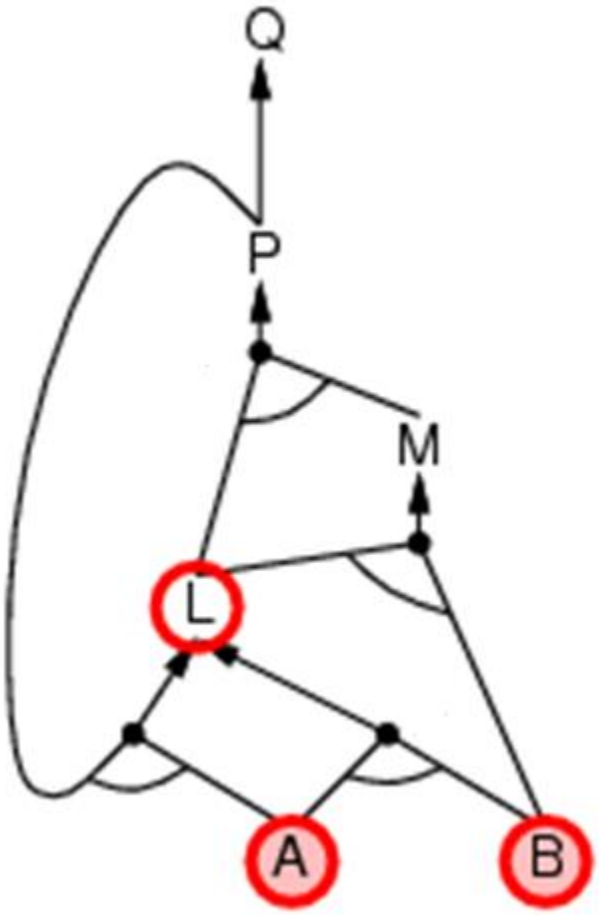
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (4)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

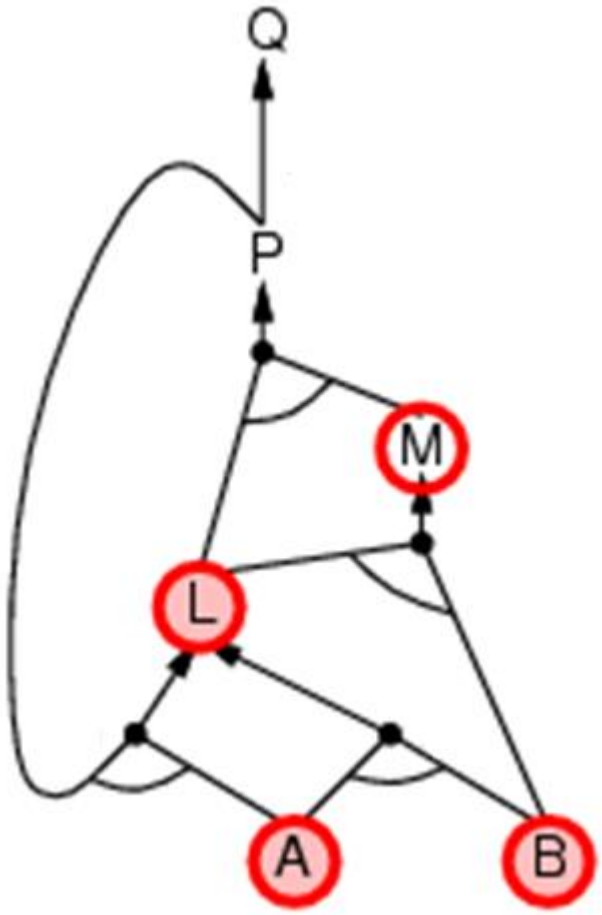
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (5)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

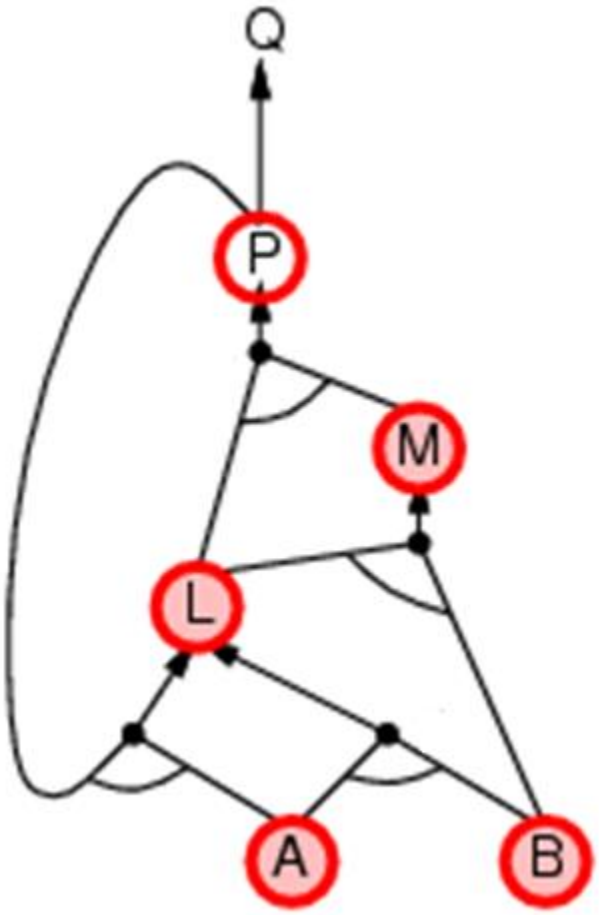
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (6)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

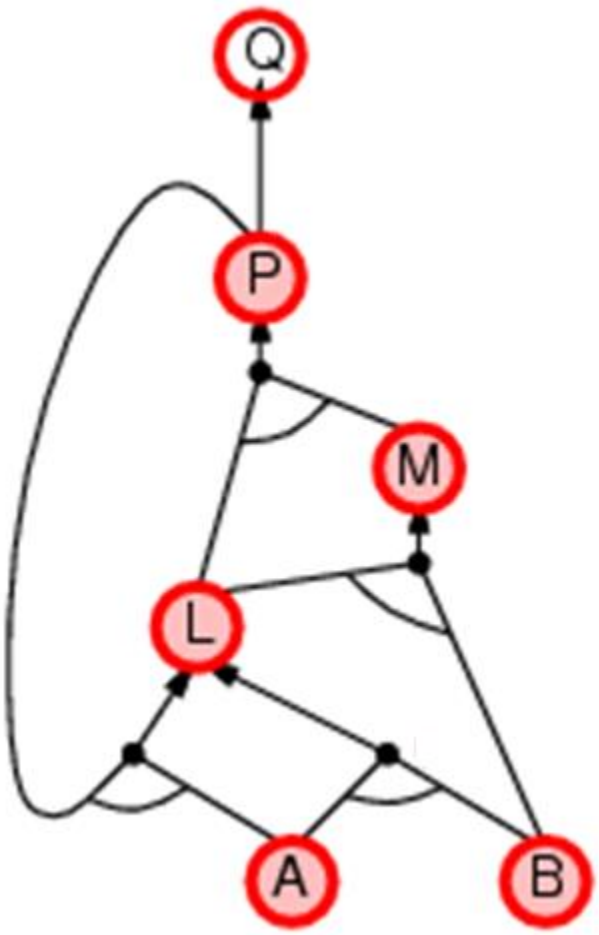
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn tiến: Ví dụ (7)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

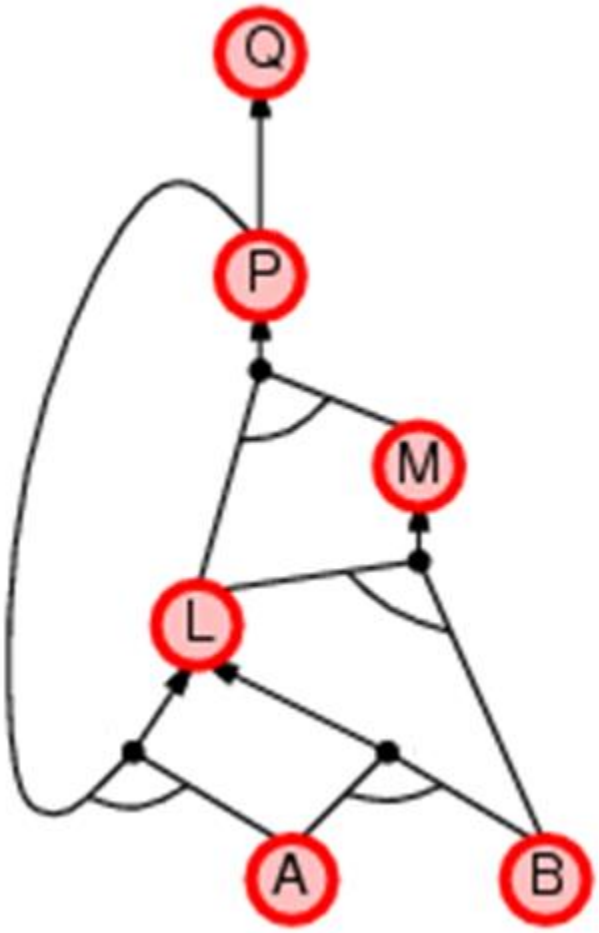
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Thuật toán suy diễn tiến cho logic mệnh đề

function PL-FC-ENTAILS?(KB, q) **returns** *true* or *false*

inputs: KB, the knowledge base, a set of propositional definite clauses
q, the query, a proposition symbol

$count \leftarrow$ a table, where $count[c]$ is the number of symbols in c's premise

$inferred \leftarrow$ a table, where $inferred[s]$ is initially *false* for all symbols

$agenda \leftarrow$ a queue of symbols, initially symbols known to be true in KB

while $agenda$ is not empty **do**
  $p \leftarrow$ POP($agenda$)
  if $p = q$ **then return** *true*
  if $inferred[p] = false$ **then**
    $inferred[p] \leftarrow true$
    for each clause c in KB where p is in c.PREMISE **do**
      decrement $count[c]$
      if $count[c] = 0$ **then** add c.CONCLUSION to $agenda$

return *false*

Suy diễn lùi (backward chaining)

- Ý tưởng: Quá trình suy diễn bắt đầu từ mệnh đề kết luận *Q*
- Để chứng minh *Q* bằng tập mệnh đề (cơ sở tri thức) *KB*
  - Kiểm tra xem Q đã được chứng minh (trong *KB*) chưa,
  - Nếu chưa, tiếp tục chứng minh tất cả các mệnh đề giả thiết của một luật nào đó (trong *KB*) có mệnh đề kết luận là *Q*
- Tránh các vòng lặp
  - Kiểm tra xem các mệnh đề mới đã có trong danh sách các mệnh đề cần chứng minh chưa? – Nếu rồi, thi không bổ sung (lại) nữa!
- Tránh việc chứng minh lặp lại đối với 1 mệnh đề
  - Đã được chứng minh (trước đó) là đúng
  - Đã được chứng minh (trước đó) là không thể thỏa mãn được (sai) trong *KB*

Suy diễn lùi: Ví dụ (1)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

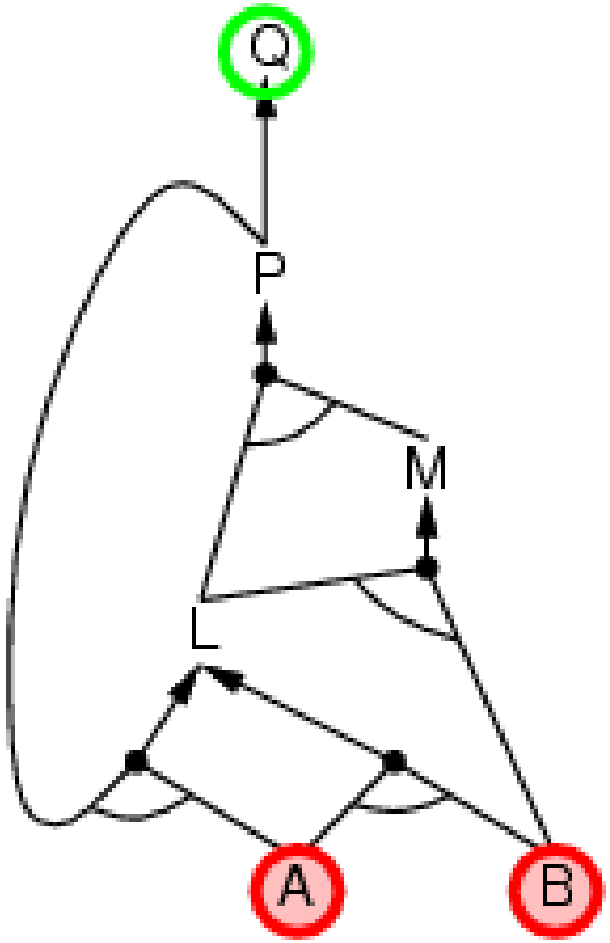
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn lùi: Ví dụ (2)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

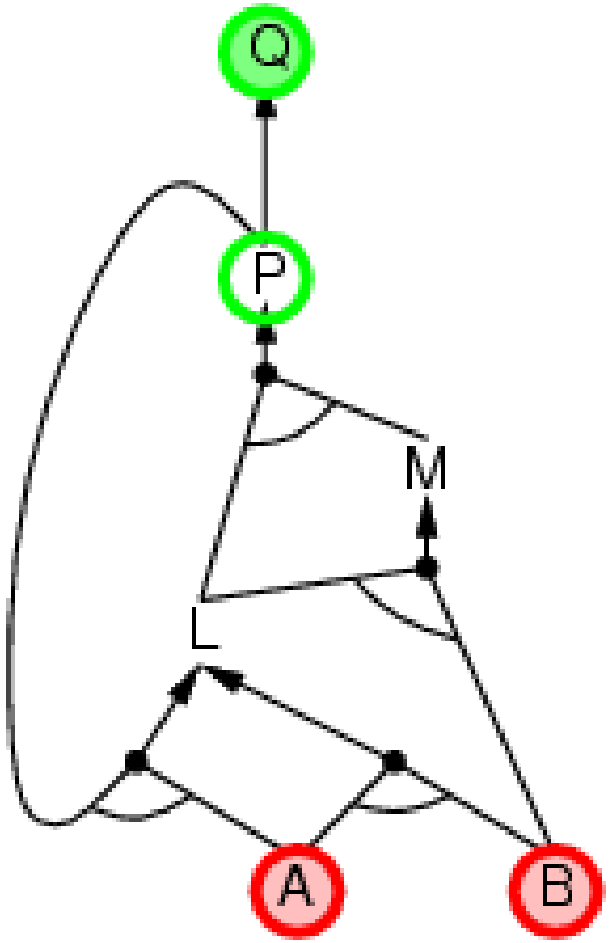
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn lùi: Ví dụ (3)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

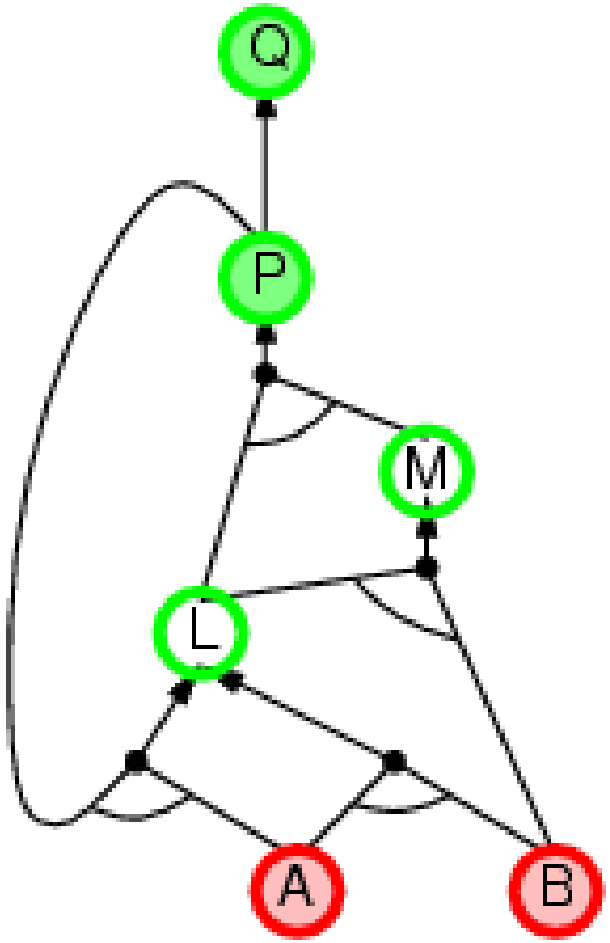
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn lùi: Ví dụ (4)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

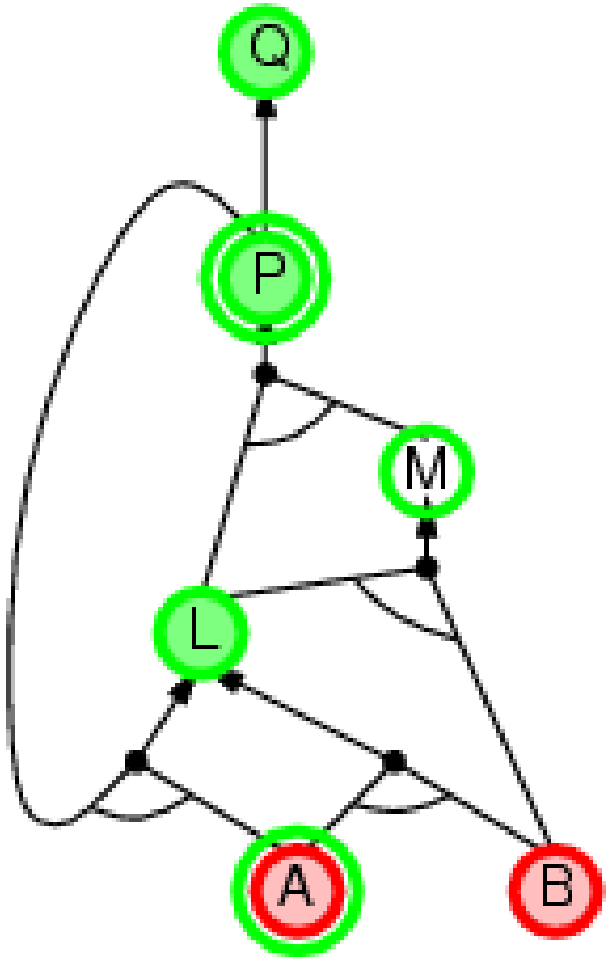
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Suy diễn lùi: Ví dụ (5)

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

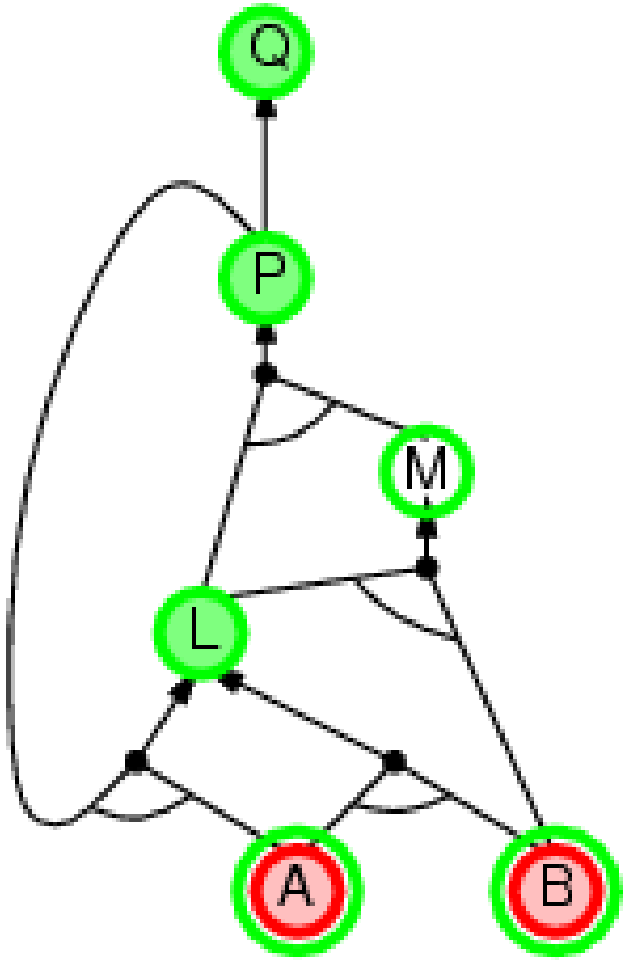
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Thuật toán suy diễn lùi cho logic mệnh đề

function DPLL-SATISFIABLE?(s) **returns** *true* or *false*
  inputs: s, a sentence in propositional logic

  $clauses \leftarrow$ the set of clauses in the CNF representation of s
  $symbols \leftarrow$ a list of the proposition symbols in s
  return DPLL($clauses, symbols, \{\ \}$)

---

function DPLL($clauses, symbols, model$) **returns** *true* or *false*

  if every clause in $clauses$ is true in $model$ **then return** *true*
  if some clause in $clauses$ is false in $model$ **then return** *false*
  $P, value \leftarrow$ FIND-PURE-SYMBOL($symbols, clauses, model$)
  if P is non-null **then return** DPLL($clauses, symbols - P, model \cup \{P{=}value\}$)
  $P, value \leftarrow$ FIND-UNIT-CLAUSE($clauses, model$)
  if P is non-null **then return** DPLL($clauses, symbols - P, model \cup \{P{=}value\}$)
  $P \leftarrow$ FIRST($symbols$); $rest \leftarrow$ REST($symbols$)
  return DPLL($clauses, rest, model \cup \{P{=}true\}$) **or**
        DPLL($clauses, rest, model \cup \{P{=}false\}$))

Suy diễn tiến hay Suy diễn lùi?

- Suy diễn tiến là quá trình dựa trên dữ liệu (data-driven)
  - Ví dụ: việc nhận dạng đối tượng, việc đưa ra quyết định
- Suy diễn tiến có thể thực hiện nhiều bước suy diễn dư thừa – chẳng liên quan tới (cần thiết cho) mục tiêu cần chứng minh
- Suy diễn lùi là quá trình hướng tới mục tiêu (goal-driven), phù hợp cho việc giải quyết vấn đề

Logic định đề: Ưu và nhược điểm

- (+) Logic định đề cho phép dễ dàng phát biểu (biểu diễn) cơ sở tri thức bằng tập các mệnh đề
- (+) Logic định đề cho phép làm việc với các thông tin ở dạng phủ định, dạng tuyển (disjunctive)
- (+) Logic định đề có tính cấu tạo (kết cấu)
  - Ngữ nghĩa của mệnh đề ($S_1 \wedge S_2$) được suy ra từ ngữ nghĩa của S_1 và ngữ nghĩa của S_2
- (+) Ngữ nghĩa trong logic định đề không phụ thuộc ngữ cảnh (context-independent)
  - Không như trong ngôn ngữ tự nhiên (ngữ nghĩa phụ thuộc vào ngữ cảnh của các câu nói)
- (-) Khả năng diễn đạt (biểu diễn) của logic định đề là rất hạn chế
  - Logic định đề không thể diễn đạt được (như trong ngôn ngữ tự nhiên): "Nếu X là cha của Y, thì Y là con của X"
  - Logic định đề phải liệt kê (xét) mọi khả năng gán giá trị chân lý (đúng/sai) cho X và Y

Giới hạn của Logic định đề

- Hãy xét ví dụ sau đây:
  - Tuấn là một sinh viên của HUST
  - Mọi sinh viên của HUST đều học môn Đại số
  - Vì Tuấn là một sinh viên của HUST, nên Tuấn học môn Đại số
- Trong logic định đề:
  - Định đề *p*: "Tuấn là một sinh viên của HUST"
  - Định đề *q*: "Mọi sinh viên của HUST đều học môn Đại số"
  - Định đề *r*: "Tuấn học môn Đại số"
  - Nhưng: (trong logic định đề) *r* không thể suy ra được từ *p* và *q*!

Logic vị từ (FOL): Ví dụ

- Ví dụ nêu trên có thể được biểu diễn trong logic vị từ bởi các biểu thức (logic vị từ) sau
  - $HUT\_Student(Tuan)$: "Tuấn là một sinh viên của HUT"
  - $\forall x: HUT\_Student(x) \rightarrow Studies\_Algebra(x)$: "Mọi sinh viên của HUT đều học môn Đại số"
  - $Studies\_Algebra(Tuan)$: "Tuấn học môn Đại số"
- Trong logic vị từ, chúng ta có thể chứng minh được:

$$\{HUT\_Student(Tuan), \forall x: HUT\_Student(x) \rightarrow Studies\_Algebra(x)\} \vdash Studies\_Algebra(Tuan)$$

- Với ví dụ trên, trong logic vị từ:
  - Các ký hiệu *Tuan*, *x* được gọi là các **phần tử** (*Tuan* là hằng, *x* là biến)
  - Các ký hiệu $HUT\_Student$ và $Studies\_Algebra$ là các **vị từ**
  - Ký hiệu $\forall$ là **lượng từ với mọi**
  - Các phần tử, các vị từ và các lượng từ cho phép biểu diễn các biểu thức

FOL: Ngôn ngữ (1)

- 4 kiểu **ký hiệu (symbols)**
  - **Hằng (Constants)**: Các tên của các đối tượng trong một lĩnh vực bài toán cụ thể (ví dụ: *Tuan*)
  - **Biến (Variables)**: Các ký hiệu mà giá trị thay đổi đối với các đối tượng khác nhau (ví dụ: *x*)
  - **Ký hiệu hàm (Function symbols)**: Các ký hiệu biểu diễn ánh xạ (quan hệ hàm) từ các đối tương của miền (domain) này sang các đối tượng của miền khác (ví dụ: *plus*)
  - **Các vị từ (Predicates)**: Các quan hệ mà giá trị logic là đúng hoặc sai (ví dụ: *HUT_Student* and *Studies_Algebra*)
- Mỗi ký hiệu hàm hoặc vị từ đều có một tập các tham số
  - Ví dụ: *HUT_Student* và *Studies_Algebra* là các vị từ có 1 tham số
  - Ví dụ: *plus* là một ký hiệu hàm có 2 tham số

FOL: Ngôn ngữ (2)

- Một **phần tử (term)** được định nghĩa (truy hồi) như sau
  - Một hằng số là một phần tử
  - Một biến là một phần tử
  - Nếu $t_1, t_2, \ldots, t_n$ là các thành phần và f là một ký hiệu hàm có n tham số, thì $f(t_1, t_2, \ldots, t_n)$ là một phần tử
  - Không còn gì khác là một phần tử
- Các ví dụ của phần tử (term)
  - `Tuan`
  - `2`
  - `friend(Tuan)`
  - `friend(x)`
  - `plus(x,2)`

FOL: Language (3)

- **Các nguyên tử (Atoms)**
  - Nếu $t_1,t_2,\ldots,t_n$ là các thành phần (terms) và p là một vị từ có n tham số, thì $p(t_1,t_2,\ldots,t_n)$ là một nguyên tử (atom)
  - Ví dụ: `HUT_Studies(Tuan)`, `HUT_Studies(x)`, `Studies_Algebra(Tuan)`, `Studies(x)`

- **Các biểu thức (Formulas)** được định nghĩa như sau
  - Một nguyên tử (atom) là một biểu thức
  - Nếu ϕ và ψ là các biểu thức, thì $\neg\phi$ và $\phi\wedge\psi$ là các biểu thức
  - Nếu ϕ là một biểu thức và x là một biến, thì $\forall x:\phi(x)$ là một biểu thức
  - Không còn gì khác là một biểu thức

- Lưu ý: $\exists x:\phi(x)$ được định nghĩa bằng $\neg\forall x:\neg\phi(x)$

FOL: Ngữ nghĩa (1)

- Một **phép diễn giải (interpretation)** của một biểu thức ϕ được biểu diễn bằng cặp $<\mathcal{D},\mathcal{I}>$
- **Miền giá trị (Domain)** $\mathcal{D}$ là một tập khác rỗng
- **Hàm diễn giải (Interpretation function)** $\mathcal{I}$ là một phép gán giá trị đối với mỗi hằng, ký hiệu hàm, và ký hiệu vị từ – sao cho:
  - Đối với hằng `c`: $\mathcal{I}(\texttt{c}) \in \mathcal{D}$
  - Đối với ký hiệu hàm (có *n* tham số) `f`: $\mathcal{I}(\texttt{f}): \mathcal{D}^n \rightarrow \mathcal{D}$
  - Đối với ký hiệu vị từ (có *n* tham số) `P`: $\mathcal{I}(\texttt{P}): \mathcal{D}^n \rightarrow \{\texttt{true}, \texttt{false}\}$

FOL: Ngữ nghĩa (2)

- **Diễn giải đối với một biểu thức logic vị từ**. Giả sử ϕ, ψ và λ là các biểu thức vị từ
  - Nếu ϕ là $\neg\psi$, thì $\mathcal{I}(\phi)$=`sai` nếu $\mathcal{I}(\psi)$=`đúng`, và $\mathcal{I}(\phi)$=`đúng` nếu $\mathcal{I}(\psi)$=`sai`
  - Nếu ϕ là $(\psi \wedge \lambda)$, thì $\mathcal{I}(\phi)$=`sai` nếu $\mathcal{I}(\psi)$ hoặc $\mathcal{I}(\lambda)$ là `sai`, và $\mathcal{I}(\phi)$=`true` nếu cả $\mathcal{I}(\psi)$ và $\mathcal{I}(\lambda)$ là `đúng`
  - Giả sử $\forall x : \phi(x)$ là một biểu thức, thì $\mathcal{I}(\forall x : \phi(x))$=`đúng` nếu $\mathcal{I}(\phi)(d)$=`đúng` với mọi giá trị $d \in \mathcal{D}$

FOL: Ngữ nghĩa (3)

- Một biểu thức ϕ là **thỏa mãn được (satisfiable)** nếu và chỉ nếu tồn tại một phép diễn giải $<\mathcal{D}, \mathcal{I}>$ sao cho $\mathcal{I}(\phi)$ – Chúng ta ký hiệu là: $\models_{\mathcal{I}} \phi$
- Nếu $\models_{\mathcal{I}} \phi$, thì chúng ta nói rằng $\mathcal{I}$ là một **mô hình (model)** của ϕ. Nói cách khác, $\mathcal{I}$ **thỏa mãn (satisfies)** ϕ
- Một biểu thức là **không thể thỏa mãn được (unsatisfiable)** nếu và chỉ nếu không tồn tại bất kỳ phép diễn giải nào
- Một biểu thức ϕ là **đúng (valid)** nếu và chỉ nếu mọi phép diễn giải $\mathcal{I}$ đều thỏa mãn ϕ – Chúng ta ký hiệu là: $\models \phi$

Lượng tử logic Với mọi

- Cú pháp của lượng tử logic **Với mọi** (universal quantifier): $\forall$<*Biến*$_1$,…,*Biến*$_n$>: <*Mệnh đề*>
- Ví dụ: Tất cả (mọi) sinh viên đang ngồi học trong lớp K4 đều chăm chỉ

$\forall$x: *Ngoi_trong_lop*(x,K4) $\Rightarrow$ *Cham_chi*(x)

- Mệnh đề ($\forall$x: *P*) là đúng trong một mô hình *m*, khi và chỉ khi *P* đúng với *x* là **mỗi** (mọi) đối tượng trong mô hình đó
- Tức là, mệnh đề ($\forall$x: *P*) tương đương với sự **kết hợp (và)** của tất cả các trường hợp của *P*

Ngoi_trong_lop(Hue,K4) $\Rightarrow$ *Cham_chi*(Hue)
$\wedge$ *Ngoi_trong_lop*(Cuong,K4) $\Rightarrow$ *Cham_chi*(Cuong)
$\wedge$ *Ngoi_trong_lop*(Tuan,K4) $\Rightarrow$ *Cham_chi*(Tuan)
$\wedge$ …

Lượng tử logic Tồn tại

- Cú pháp của lượng tử logic Tồn tại (existential quantifier): $\exists$<*Biến*$_1$,…,*Biến*$_n$>: <*Mệnh đề*>
- Ví dụ: Tồn tại (có) sinh viên đang ngồi học trong lớp K4, và là sinh viên chăm chỉ:

$\exists$x: *Ngoi_trong_lop*(x,K4) $\wedge$ *Cham_chi*(x)

- Mệnh đề ($\exists x$: *P*) là đúng trong một mô hình *m*, khi và chỉ khi *P* là đúng với *x* là **một** đối tượng trong mô hình đó
- Tức là, mệnh đề ($\exists x$: *P*) tương đương với phép **tuyển (hoặc)** của các trường hợp của *P*

Ngoi_trong_lop(Hue,K4) $\wedge$ *Cham_chi*(Hue)
$\vee$ *Ngoi_trong_lop*(Cuong,K4) $\wedge$ *Cham_chi*(Cuong)
$\vee$ *Ngoi_trong_lop*(Tuan,K4) $\wedge$ Cham_chi(Tuan)
$\vee$ …

Các đặc điểm của các lượng từ logic

- Tính hoán vị:
  - $(\forall x\ \forall y)$ là tương đương với $(\forall y\ \forall x)$
  - $(\exists x\ \exists y)$ là tương đương với $(\exists y\ \exists x)$
- Tuy nhiên, $(\exists x\ \forall y)$ **không** tương đương với $(\forall y\ \exists x)$
  - $\exists x\ \forall y$: Yeu(x,y) - "Trên thế giới này, tồn tại (có) một người mà người đó yêu quý tất cả mọi người khác"
  - $\forall y\ \exists x$: Yeu(x,y) - "Trên thế giới này, mọi người đều được ít nhất một người khác yêu thích"
- Mỗi lượng từ logic ($\exists$ hoặc $\forall$) đều có thể được biểu diễn bằng lượng từ kia
  - $(\forall x$: Thich(x,Kem)) là tương đương với $(\neg\exists x$: $\neg$Thich(x,Kem))
  - $(\exists x$: Thich(x,BongDa)) là tương đương với $(\neg\forall x$: $\neg$Thich(x,BongDa))

Sử dụng logic vị từ

Biểu diễn các phát biểu trong ngôn ngữ tự nhiên

- "*x* là anh/chị/em của *y*" tương đương với "*x* và *y* là anh em ruột"

  $\forall x,y: Anh\_chi\_em(x,y) \Leftrightarrow Anh\_em\_ruot(x,y)$

- "Mẹ của *c* là *m*" tương đương với "*m* là phụ nữ và *m* là bậc cha mẹ của *c*"

  $\forall m,c: Me(c) = m \Leftrightarrow (Phu\_nu(m) \wedge Cha\_me(m,c))$

- Quan hệ "anh em ruột" có tính chất đối xứng

  $\forall x,y: Anh\_em\_ruot(x,y) \Leftrightarrow Anh\_em\_ruot(y,x)$

Thuật toán suy diễn tiến cho logic vị từ

function FOL-FC-ASK(KB, α) **returns** a substitution or *false*
  inputs: KB, the knowledge base, a set of first-order definite clauses
    α, the query, an atomic sentence
  local variables: *new*, the new sentences inferred on each iteration

  repeat until *new* is empty
    new $\leftarrow \{\}$
    for each *rule* **in** KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-VARIABLES(*rule*)
      for each θ such that SUBST($\theta, p_1 \wedge \ldots \wedge p_n$) = SUBST($\theta, p'_1 \wedge \ldots \wedge p'_n$)
          for some $p'_1, \ldots, p'_n$ in KB
        $q' \leftarrow$ SUBST(θ, q)
        if q' does not unify with some sentence already in KB or *new* **then**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q', α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

Thuật toán suy diễn lùi cho logic vị từ

function FOL-BC-ASK(KB, $query$) **returns** a generator of substitutions
  return FOL-BC-OR(KB, $query$, $\{\}$)

---

generator FOL-BC-OR(KB, $goal$, θ) **yields** a substitution
  for each rule ($lhs \Rightarrow rhs$) in FETCH-RULES-FOR-GOAL(KB, $goal$) **do**
    (lhs, rhs) $\leftarrow$ STANDARDIZE-VARIABLES((lhs, rhs))
    for each θ' **in** FOL-BC-AND(KB, lhs, UNIFY(rhs, $goal$, θ)) **do**
      yield θ'

---

generator FOL-BC-AND(KB, $goals$, θ) **yields** a substitution
  if $\theta = failure$ **then return**
  else if LENGTH($goals$) = 0 **then yield** θ
  else do
    $first$, $rest$ $\leftarrow$ FIRST($goals$), REST($goals$)
    for each θ' **in** FOL-BC-OR(KB, SUBST(θ, $first$), θ) **do**
      for each θ'' **in** FOL-BC-AND(KB, $rest$, θ') **do**
        yield θ''