Biểu diễn tri thức
(Knowledge Representation)

Chương 4

Nội dung

1. Giới thiệu về tri thức
2. Các phương pháp biểu diễn tri thức
3. Các vấn đề trong biểu diễn tri thức
4. Biểu diễn tri thức bằng logic
5. Biểu diễn tri thức bằng mạng ngữ nghĩa

1. Giới thiệu về tri thức

Để giải những bài toán phức tạp trong lãnh vực trí tuệ nhân tạo, chúng ta cần một lượng lớn *tri thức* và một số *cơ chế* (mechanism) để xử lý tri thức ấy và tạo ra những lời giải cho bài toán.

Có hai thực thể đáng quan tâm khi biểu diễn tri thức:

1. Các sự kiện (facts)
2. Các cách thức biểu diễn sự kiện (representations of facts).

Sự kiện và cách thức biểu diễn

Dạng thực	Dạng hình thức
- **Facts** (sự kiện): sự thật trong lĩnh vực	- **Representations** (sự biểu diễn): dạng biểu diễn của sự kiện theo lược đồ được chọn.
Cái cần biểu diễn	Cái có thể xử lý được

Cấu trúc hóa tri thức

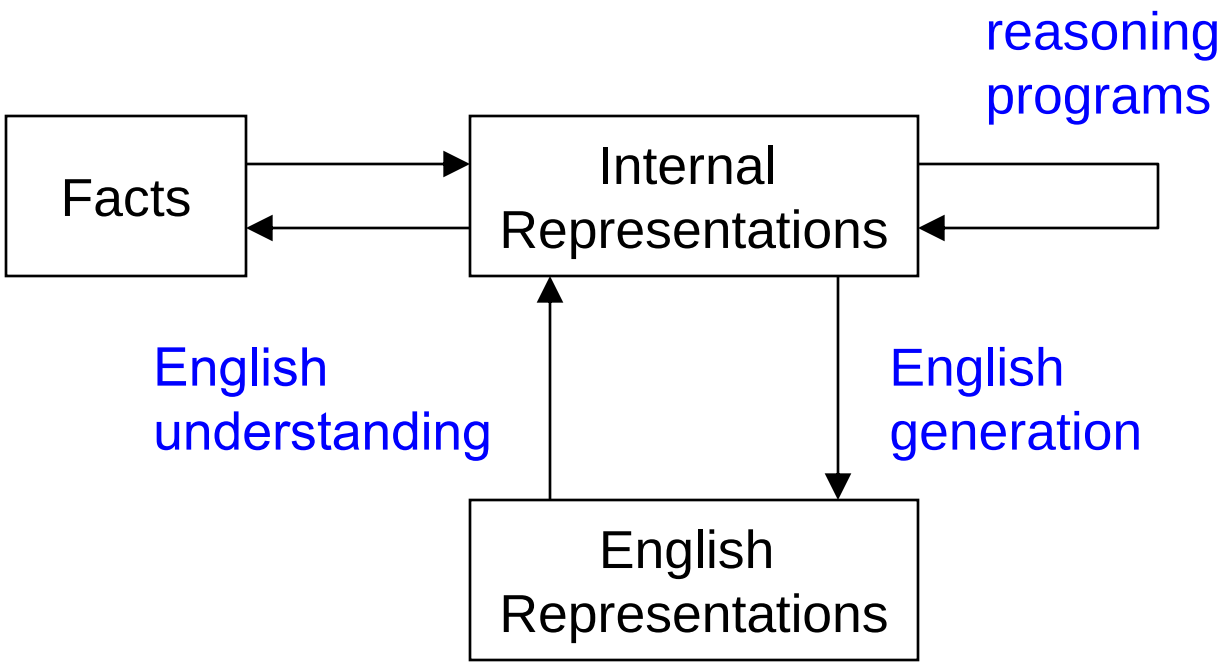
Chúng ta có thể cấu trúc hóa tri thức ở hai mức:

- **Mức tri thức** (knowledge level), mức mà các sự kiện được mô tả
- **Mức ký hiệu** (symbol level), Mức mà sự biểu diễn của các đối tượng đã được chọn trong mức tri thức được viết ra ở dạng ký hiệu để có thể xử lý được bằng chương trình.

Có tồn tại những ánh xạ hai chiều (two-way mapping) giữa hai mức này.

Biểu diễn và ánh xạ

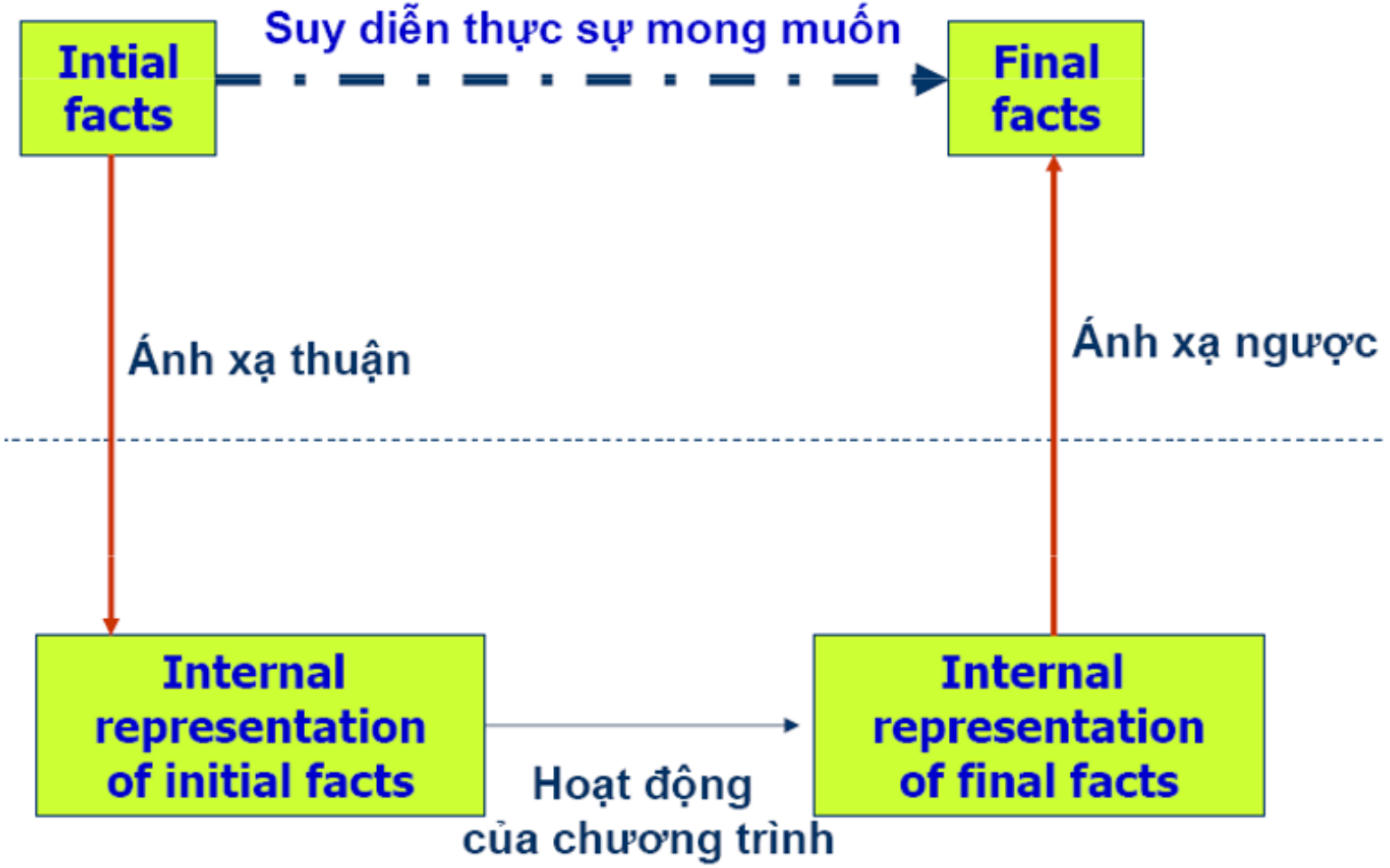
Ánh xạ thuận: từ sự kiện chuyển sang biểu diễn nội tại.
Ánh xạ ngược: từ biểu diễn nội tại chuyển thành sự kiện



Hình 4.1. Hai ánh xạ giữa những sự kiện và những biểu diễn

Các ***chương trình suy diễn*** (reasoning program) chỉ có thể làm việc với biểu diễn nội tại của các sự kiện

Hai ánh xạ giữa các sự kiện và các cách biểu diễn



Hình 4.2: Biểu diễn các sự kiện

Từ dữ liệu đến tri thức

- Đối với máy tính, *dữ liệu* (data) là các con số, ký hiệu mà máy tính có thể lưu trữ, biểu diễn, xử lý. Bản thân dữ liệu không có ý nghĩa.
- Chỉ khi con người cảm nhận, tư duy thì dữ liệu mới có một ý nghĩa nhất định, đó chính là *thông tin* (Information)
- *Tri thức* (knowledge) là kết tinh, cô đọng, chắt lọc của thông tin. Tri thức hình thành từ quá trình xử lý thông tin mang lại

Phân loại tri thức

- **Tri thức sự kiện** (factual knowledge) khẳng định về một sự kiện, hiện tượng hay một khái niệm nào đó trong một hoàn cảnh không gian hoặc thời gian nhất định: định lý toán học, định luật vật lý, …
- **Tri thức thủ tục** (procedural knowledge) mô tả cách giải quyết một vấn đề, quy trình xử lý các công việc, lịch trình tiến hành các thao tác: các luật, chiến lược, lịch trình như phương pháp điều chế hóa học, thuật toán, …
- **Tri thức mô tả** (declarative knowledge) các nhận định, kết luận về sự kiện, hiện tượng
- **Tri thức heuristic** (heuristic knowledge) các ước lượng, suy đoán hình thành qua kinh nghiệm. Không đảm bảo hòan tòan chính xác hoặc tối ưu theo một nghĩa nào đó về cách giải quyết vấn đề. Tri thức **heuristic** thường được coi là một mánh lới nhằm dẫn dắt tiến trình lập luận

Nhu cầu xử lý tri thức

- Trí tuệ, sự thông minh phải dựa trên nền tảng của tri thức. Tuy nhiên, nó còn phụ thuộc vào việc vận dụng, xử lý tri thức.
- Biểu diễn tri thức là việc đưa tri thức vào máy tính. Biểu diễn tri thức chỉ có ý nghĩa nếu việc "*xử lý tri thức*" được thực hiện.

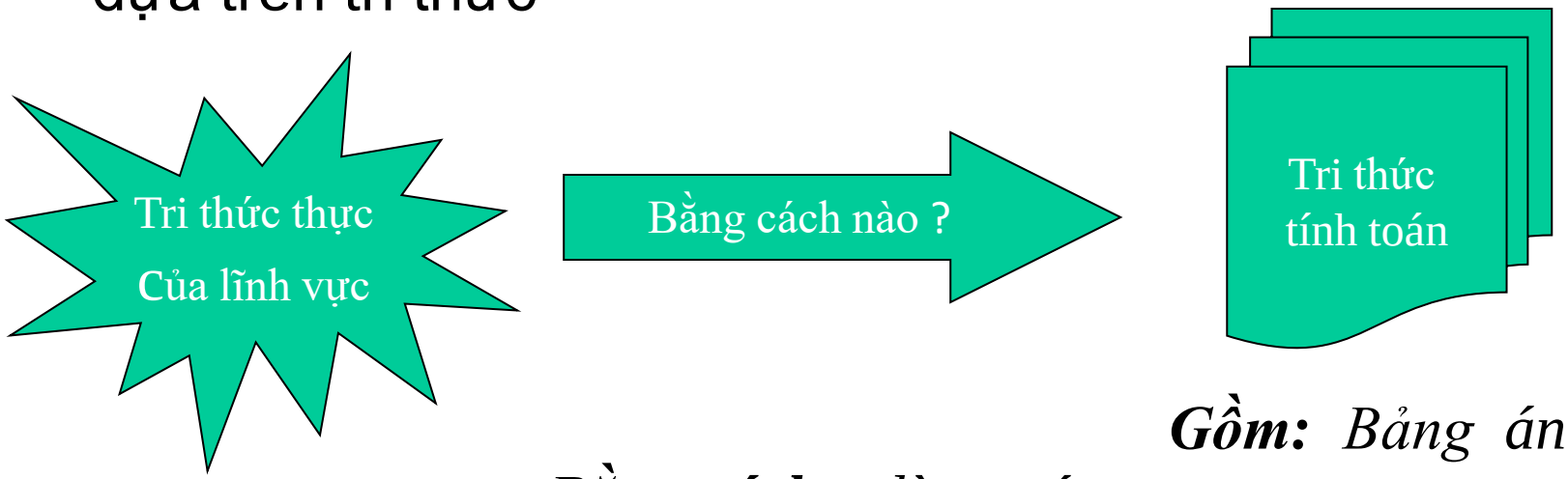
Ngôn ngữ biểu diễn tri thức

Ngôn ngữ biểu diễn tri thức = Cú pháp + Ngữ nghĩa + Cơ chế suy diễn

- **Cú pháp** bao gồm các ký hiệu và các *quy tắc* liên kết các ký hiệu (các luật cú pháp) để tạo thành các câu (công thức) trong ngôn ngữ
- **Ngữ nghĩa** xác định ý nghĩa của các câu trong một miền nào đó của thế giới hiện thực
- **Cơ chế suy diễn** để từ các tri thức trong cơ sở tri thức (knowledge base) và các sự kiện ta suy ra được các tri thức mới

2. Các phương pháp biểu diễn tri thức

- Biểu diễn tri thức là phương pháp mã hoá tri thức, nhằm thành lập ***cơ sở tri thức*** cho các hệ thống dựa trên tri thức



Gồm: *đối tượng và các quan hệ giữa chúng trong lĩnh vực*

Bằng cách: *dùng các lược đồ biểu diễn (scheme)* → *Chọn lược đồ cho loại tri thức là vấn đề quan trọng*

Gồm: *Bảng ánh xạ giữa*
Đối tượng thực → *đối tượng tính toán*
Quan hệ thực → *quan hệ tính toán*

Danh mục các lược đồ biểu diễn tri thức

- Lược đồ logic (logical scheme)
  - Logic mệnh đề
  - Logic vị từ
- Lược đồ thủ tục (procedural scheme)
- Lược đồ mạng (network scheme)
- Lược đồ cấu trúc (structural scheme)
  - Khung (Frame)
  - Kịch bản (Script)

Lược đồ logic

- Dùng các biểu thức trong logic hình thức, như phép toán vị từ, để biểu diễn tri thức
- Các luật suy diễn áp dụng cho loại lược đồ này
- Ngôn ngữ lập trình hiện thực tốt nhất cho loại lược đồ này là: PROLOG

Logic mệnh đề

IF *Xe không khởi động được* (A) AND *Khoảng cách từ nhà đến chỗ làm là xa* (B) THEN *Sẽ trễ giờ làm* (C)

- Luật trên có thể biểu diễn lại như sau:

$$A \wedge B \Rightarrow C$$

Logic vị từ

- Tương tự logic mệnh đề
- Dùng các ký hiệu để thể hiện tri thức.
- Những ký hiệu này gồm hằng số, vị từ, biến và hàm

Ví dụ: Câu tiếng Anh:

"Spot is a dog"

"Every dog has a tail"

- Dạng logic vị từ:

1. *dog(Spot).*
2. $\forall X$ *dog(X)*$\rightarrow$*hastail(X)*

Ta muốn suy ra một sự kiện mới *hastail(Spot)*

Lược đồ thủ tục

- Khác với lược đồ khai báo, lược đồ này biểu diễn tri thức như một tập các câu lệnh để giải quyết vấn đề
- Các câu lệnh trong lược đồ thủ tục chỉ ra bằng cách nào giải quyết vấn đề
- Ví dụ: **hệ luật sinh** (production rule system) điển hình cho loại lược đồ này

Hệ luật sinh

- ***Luật*** (rule) là cấu trúc tri thức dùng để liên kết thông tin đã biết với các thông tin khác giúp đưa ra các suy luận, kết luận từ những thông tin đã biết
- Thu thập các tri thức của lĩnh vực thành một tập và lưu chúng trong ***cơ sở tri thức*** (knowledge-base) của hệ thống. Hệ thống dùng các luật này cùng với các thông tin để giải bài toán
- Việc xử lý các luật trong hệ thống dựa trên các luật được quản lý bằng một module gọi là ***bộ máy suy diễn*** (inference engine)

Vài thí dụ về luật sinh

1. IF Bình điện hỏng THEN Xe sẽ không khởi động được
2. IF Xe không khởi động được THEN Đi bộ
3. IF Xe không khởi động được AND Hệ thống nhiên liệu tốt THEN Kiểm tra hệ thống điện
4. IF Xe không khởi động được THEN Đầu tiên hãy kiểm tra hệ thống nhiên liệu, sau đó kiểm tra hệ thống điện
5. IF Xe nổ AND tiếng giòn THEN Động cơ hoạt động bình thường
6. IF Sốt cao AND hay ho AND Họng đỏ THEN Viêm họng
7. IF Là nữ AND Da sáng THEN Nên chọn Xe Spacy AND Chọn màu sáng

Phương pháp suy diễn dùng trong hệ luật sinh: *suy diễn tiến (forward chaining)* và *suy diễn lùi (backward chaining)*

Lược đồ mạng

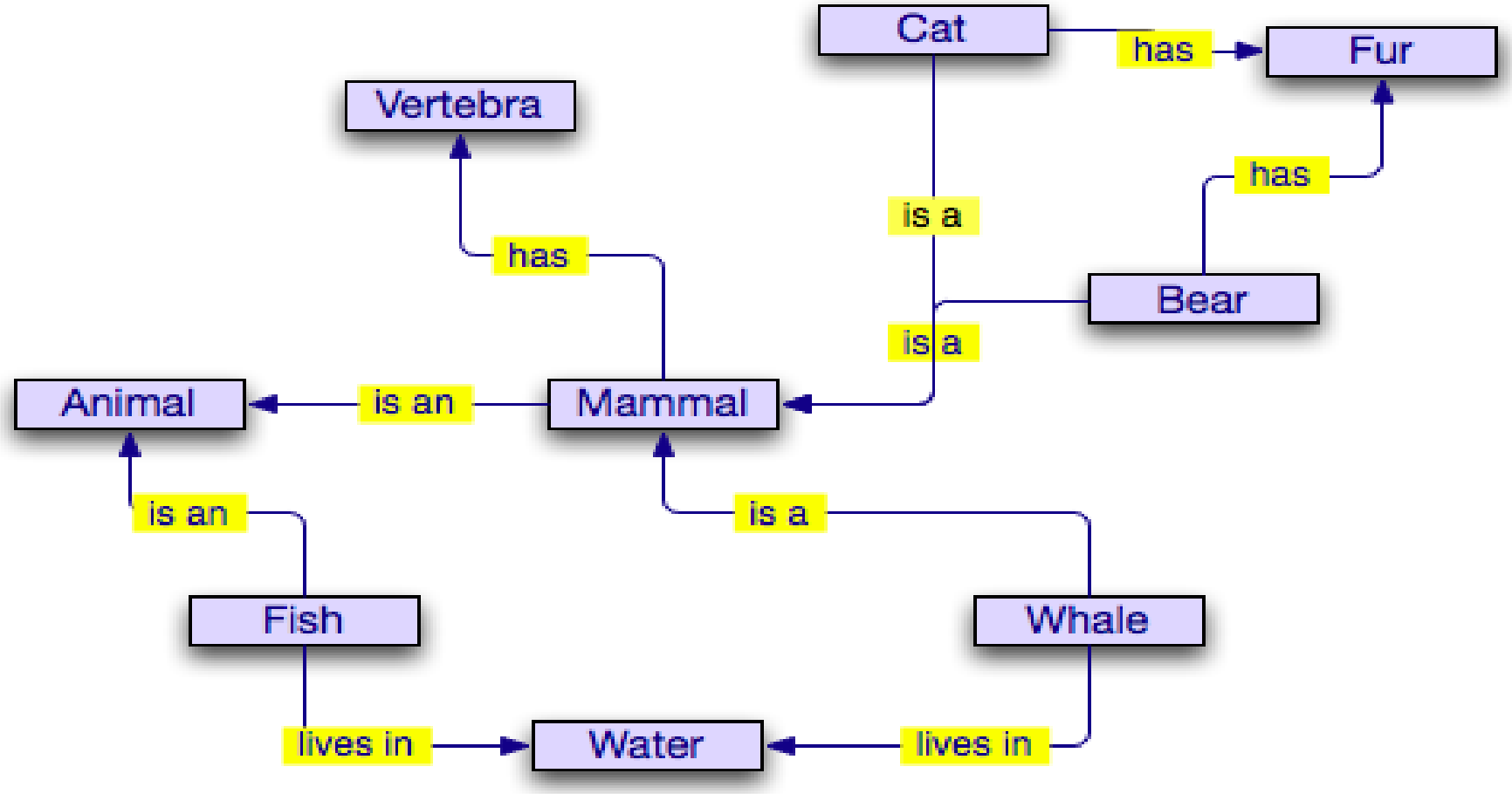
- Biểu diễn tri thức như là đồ thị; các đỉnh như là các đối tượng hoặc khái niệm, các cung như là quan hệ giữa chúng
- Ví dụ: mạng ngữ nghĩa (semantic network), lược đồ quan hệ phụ thuộc khái niệm (conceptual dependency), đồ thị khái niệm (conceptual graph).

Mạng ngữ nghĩa

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

Thí dụ về mạng ngữ nghĩa

Hình 4.3: Mạng ngữ nghĩa



Lược đồ cấu trúc

- Là một mở rộng của lược đồ mạng; bằng cách cho phép các nút có thể là một Cấu Trúc Dữ Liệu phức tạp gồm các khe (slot) có tên và trị hay một thủ tục
- Ví dụ: kịch bản (script), khung (frame)

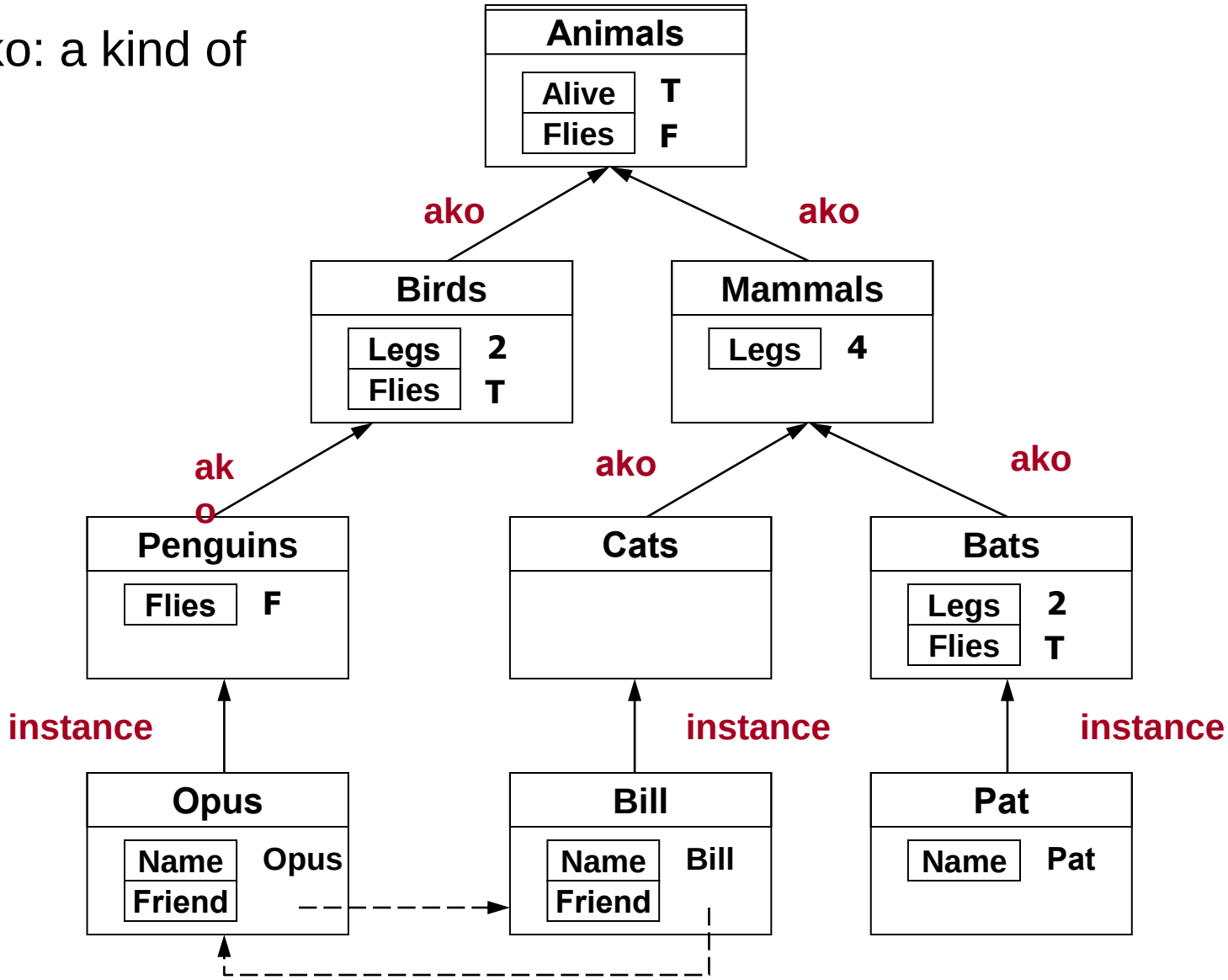
Khung (frame)

- Mỗi frame mô tả một *đối tượng* (object)
- Một frame bao gồm 2 thành phần cơ bản là **slot** và **facet**
- **slot** là một thuộc tính đặc tả đối tượng được biểu diễn bởi frame
- Mỗi slot có thể chứa một hoặc nhiều **facet**
- Các facet (đôi lúc được gọi là slot "con") đặc tả một số thông tin hoặc *thủ tục* liên quan đến *thuộc tính* được mô tả bởi slot. Facet có nhiều loại khác nhau, sau đây là một số facet thường gặp: *value, default value, range, ...*

Frame

Hình 4.4. Một hệ thống frame

ako: a kind of



3. Các vấn đề trong biểu diễn tri thức

Có những câu hỏi cần trả lời trước khi chọn một phương pháp biểu diễn tri thức.

- Có những thuộc tính cơ bản nào của đối tượng mà chúng xuất hiện trong mọi lĩnh vực không? Nếu có: những thuộc tính nào?
- Có chắc chắn là chúng sẽ được xử lý thích hợp trong từng cơ chế được đề nghị không?
- Có quan hệ quan trọng nào tồn tại cùng với thuộc tính không?
- Các đối tượng và các quan hệ có thể biểu diễn cho cái gì trong lĩnh vực?

Ví dụ: để biểu diễn cho ý "Nam cao 1 mét 70", có thể dùng: ***chieucao(Nam, 170).*** Để diễn tả "An cao hơn Nam"?

Các vấn đề trong biểu diễn tri thức (tt.)

- Tri thức được biểu diễn đến mức chi tiết nào?
- Bằng cách nào thể hiện tính phân cấp của tri thức: các hình thức: kế thừa, trị mặc định, ngoại lệ, đa thừa kế phải đặc tả như thế nào?
- Khi mô tả đối tượng, bằng cách nào có thể tích hợp một tri thức thủ tục vào bản mô tả, và khi nào thủ tục được thực hiện?
- Với số lượng lớn tri thức được chứa trong CSDL, Bằng cách nào truy xuất những thành phần cần thiết?

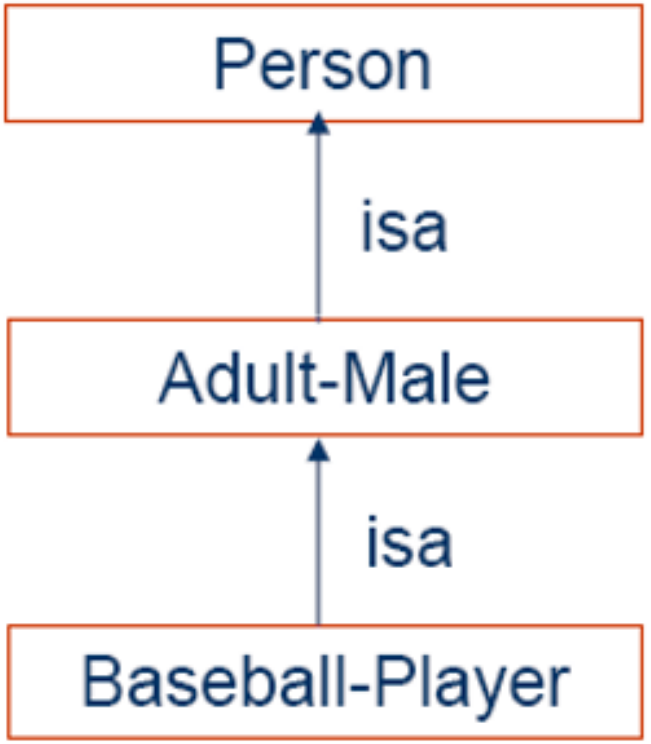
Đặc điểm của hệ thống biểu diễn tri thức

- Phải có khả năng **biểu diễn** tất cả các tri thức cần thiết cho lĩnh vực đó
- Phải có khả năng **xử lý** các cấu trúc sẵn có để sinh ra các cấu trúc mới tương ứng với tri thức mới được sinh ra từ tri thức cũ.
- Phải có khả năng **thêm vào** cấu trúc những tri thức, thông tin bổ sung mà nó có thể được dùng để hướng dẫn cơ chế suy luận theo hướng có nhiều triển vọng nhất.
- Phải có khả năng thu được thông tin mới dễ dàng.
  - Trường hợp đơn giản nhất là **chèn** trực tiếp tri thức mới (do con người) vào cơ sở tri thức. Lý tưởng nhất là chương trình có thể kiểm soát việc thu gom được tri thức.

Tri thức có khả năng thừa kế

- Một dạng bổ sung cơ chế suy diễn vào cơ sở tri thức quan hệ nói trên, đó là: **thừa kế thuộc tính**.
- Thừa kế thuộc tính:
  - Tổ chức các đối tượng thành các lớp (class).
  - Các lớp được sắp xếp vào hệ thống phân cấp (hierachy) – có lớp cha (tổng quát) và lớp con (cụ thể).

$\Rightarrow$ Các lớp con ***thừa kế*** các thuộc tính từ lớp cha



Hình 4.5. Một thí dụ về tính thừa kế.

Tri thức suy diễn

- Thừa kế thuộc tính nêu trên cũng là 1 dạng suy diễn
- Logic truyền thống: cung cấp dạng suy diễn mạnh hơn
- Tri thức suy diễn: cần ***thủ tục suy diễn*** (reasoning procedure)
- Thủ tục suy diễn: nhiều dạng
  - Forward (tiến): sự kiện → kết luận
  - Backward (lùi): kết luận → sự kiện đã cho
  - Thủ tục thường dùng để suy diễn: hợp giải (resolution)

Tri thức thủ tục

- Tri thức khai báo (declarative knowledge) có tính chất tĩnh.
- Một dạng tri thức khác: chỉ ra hành động được thi hành khi điều kiện nào đó thỏa, đó là *tri thức thủ tục* (procedural knowledge)
- Cách biểu diễn trong chương trình
  - **Viết bằng các Ngôn Ngữ Lập Trình (Ví Dụ LISP hay Prolog)**

    ⇒ ***Máy tính sẽ thực thi mã để thực hiện công việc***
- Tri thức thủ tục hay dùng *luật sinh* (production rule)

Các mối quan hệ quan trọng

1. Instance

- Cho biết quan hệ thành viên giữa đối tượng và lớp nó thuộc vào

2. Isa

- Cho biết một lớp là lớp con của lớp khác

Cặp quan hệ trên cho phép khả năng thừa kế thuộc tính

Chúng có thể được gọi và biểu diễn khác nhau trong nhiều hệ thống tri thức

4.Biểu diễn tri thức bằng Logic

Biểu diễn những sự kiện đơn giản bằng **logic mệnh đề** (propositional logic).

Vài thí dụ:

It is raining
RAINING

It is sunny
SUNNY

It is windy
WINDY

If it is raining, then it is not sunny
RAINING $\rightarrow \neg$SUNNY

Sử dụng logic mệnh đề

- Logic mệnh đề khá tiện dụng để biểu diễn tri thức: ***thủ tục quyết định*** (decision procedure) của loại logic này đã tồn tại. Thủ tục này thực hiện một công việc giống như ***chứng minh định lý*** (theorem proving).

- Logic mệnh đề không thể biểu diễn các đối tượng như là các biến và không thể đưa vào các lượng từ ($\forall$,$\exists$).

Logic mệnh đề

- Hai hằng số logic (TRUE hoặc FALSE)
- Ký hiệu mệnh đề (biến mệnh đề - propotional variale): thí dụ P, Q
- Các phép kết nối logic: $\vee$ (OR), $\wedge$(AND), $\neg$ (NOT), $\Rightarrow$ (hàm ý), $\Leftarrow$ (IF), $\Leftrightarrow$ (*tương đương*)

- Công thức (formula), thí dụ: $P \wedge Q$, $P \Rightarrow Q$,

$$(P \vee Q) \wedge R$$

Ví dụ về mệnh đề

Mệnh đề thực tế

- "Nếu trời mưa thì bầu trời có mây"
- Trời đang mưa

Vậy ➔ Bầu trời có mây

Mệnh đề logic

- P="Trời mưa"
- Q= "Bầu trời có mây"

Ta có hai phát biểu sau đúng:

- P➔ Q
- P

Theo luật suy diễn ➔ Q là đúng.

Nghĩa là: "Bầu **trời có mây"**

Mệnh đề :

Mệnh đề là một phát biểu khai báo.

Mệnh đề chỉ nhận một trong hai giá trị:

ĐÚNG (***True***) hoặc SAI (***False***)

Qui tắc xây dựng công thức

- Các ***công thức*** (formula) mà chỉ là các ký hiệu mệnh đề sẽ được gọi là các ***mệnh đề đơn*** .
- Các công thức không phải là mệnh đề đơn sẽ được gọi là ***mệnh đề phức hợp.***
- Nếu P là ký hiệu mệnh đề thì P và $\neg$P được gọi là ***literal***, P là ***literal dương***, còn $\neg$ P là ***literal âm***

Ngữ nghĩa

- ***Xác định*** ý nghĩa của các công thức trong thế giới thực bằng cách kết hợp mỗi *ký hiệu mệnh đề* với *sự kiện* nào đó trong thế giới hiện thực
- Ký hiệu mệnh đề có thể ứng với sự kiện nào đó
- Sự kết hợp các kí hiệu mệnh đề với các sự kiện trong thế giới thực được gọi là một ***diễn giải*** (interpretation)
- Một sự kiện chỉ có thể đúng hoặc sai. Ví dụ: sự kiện "Paris là thủ đô nước Pháp" là đúng

Dạng chuẩn của công thức logic mệnh đề

Chuẩn hóa các công thức

- Đưa các công thức về dạng thuận lợi cho việc lập luận, suy diễn.
- Sử dụng các phép biến đổi tương đương ta có thể đưa một công thức bất kỳ về dạng chuẩn.
- **Dạng chuẩn hội**: conjunctive normal form

$(L11 \vee L12 \vee \ldots \vee L1n_1) \wedge (L21 \vee L22 \vee \ldots \vee L2n_2) \wedge \ldots \wedge (Lk1 \vee Lk2 \vee \ldots \vee Lkn_k)$

Sự tương đương giữa các công thức

- $A \Rightarrow B \equiv \neg A \vee B$
- $A \Leftrightarrow B \equiv (A \Rightarrow B) \wedge (B \Rightarrow A)$
- $\neg(\neg A) \equiv A$

Luật De Morgan

- $\neg(A \vee B) \equiv \neg A \wedge \neg B$
- $\neg(A \wedge B) \equiv \neg A \vee \neg B$

Luật giao hoán-luật kết hợp-luật phân phối

Luật giao hoán

- $A \vee B \equiv B \vee A$
- $A \wedge B \equiv B \wedge A$

Luật kết hợp

- $(A \vee B) \vee C \equiv A \vee (B \vee C)$
- $(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$

Luật phân phối

- $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
- $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$

Chuyển thành dạng chuẩn

- Để dễ dàng viết các chương trình máy tính, ta đưa chúng về dạng biểu diễn ***dạng chuẩn hội***
- Một công thức ở dạng chuẩn hội nếu nó là hội (AND) của các **mệnh đề tuyển** (disjunct)
- Cách thực hiện
  1. Bỏ các dấu kéo theo (⇒) bằng cách thay (A⇒B) bởi (¬A∨B)
  2. Chuyển các dấu phủ định (¬) vào sát các ký hiệu mệnh đề bằng cách áp dụng luật De Morgan và thay ¬ (¬A) bởi A
  3. Áp dụng luật phân phối, thay các công thức có dạng A∨(B∧C) bởi (A ∨ B) ∧ (A ∨ C)

Ví dụ chuẩn hóa (P ⇒ Q) ∨ ¬(R ∨ ¬S)

- (P ⇒ Q) ∨ ¬ (R ∨ ¬S)
- (¬P ∨ Q) ∨ ¬ (R ∨ ¬S) : Loại bỏ dấu ⇒
- (¬P ∨ Q) ∨ (¬R ∧ S) : Luật De Morgan
- ((¬P ∨ Q) ∨ ¬R) ∧ ((¬ P ∨ Q) ∨ S) : Luật phân phối
- (¬ P ∨ Q ∨ ¬ R) ∧ (¬ P ∨ Q ∨ S)

Ta đi đến một tập hợp các *mệnh đề tuyển* (*disjunct*)

Mệnh đề Horn

- Mọi công thức đều có thể đưa về dạng chuẩn hội, có dạng:
  - $\neg P_1 \vee ... .. \vee \neg P_m \vee Q_1 \vee ..... \vee Qn$
  - tương đương với: $P_1 \wedge ... \wedge P_m => Q_1 \vee ... \vee Qn$ gọi là ***mệnh đề Kowalski*** (do nhà logic Kowalski đưa ra năm 1971)
- Khi n <=1, tức là câu tuyển chỉ chứa nhiều nhất một literal dương → gọi là ***mệnh đề Horn*** (Horn clause- mang tên nhà logic Alfred Horn, năm 1951)
- Nếu m > 0, n=1, mệnh đề ***Horn*** có dạng: $P_1 \wedge ... \wedge P_m => Q$
- Có thể biểu diễn thành các luật ***if-then***:

  If *P_1 and ... and P_m* ***then*** *Q*

Luật suy diễn

- H được xem là ***hệ qủa logic*** (logical consequence) của một tập công thức G ={G1,.....,Gm} nếu trong bất kỳ sự diễn giải (interpretation) nào mà {G1,... ..,Gm} đúng thì H cũng đúng
- Dùng các tri thức trong cơ sở tri thứcđể suy ra tri thức mới mà nó là hệ quả logic của các công thức trong cơ sở tri thức: sử dụng ***các luật suy diễn*** (rule of inference)
- Một luật suy diễn gồm hai phần: một tập các điều kiện và một kết luận

Một số luật suy diễn quan trọng

Luật	Điều kiện	Kết luận
Modus Ponens	$\alpha \Rightarrow \beta, \alpha$	β
Modus Tollens	$\alpha \Rightarrow \beta, \neg\beta$	$\neg\alpha$
Bắc cầu	$\alpha \Rightarrow \beta, \beta \Rightarrow \gamma$	$\alpha \Rightarrow \gamma$
Loại bỏ hội	$\alpha_1 \wedge \ldots \wedge \alpha_i \wedge \ldots \wedge \alpha_m$	α_i
Đưa vào hội	$\alpha_1, \ldots, \alpha_i, \ldots, \alpha_m$	$\alpha_1 \wedge \ldots \wedge \alpha_i \wedge \ldots \wedge \alpha_m$
Đưa vào tuyển	α_i	$\alpha_1 \vee \ldots \vee \alpha_i \vee \ldots \vee \alpha_m$
Hợp giải	$\alpha \vee \beta, \neg\beta \vee \gamma$	$\alpha \vee \gamma$

Tiên đề, định lý, chứng minh

- Các luật suy diễn cho phép suy ra các **công thức mới** từ **các công thức đã có** bằng một chuỗi áp dụng các luật suy diễn
- Các công thức đã cho: các ***tiên đề***
- Các công thức được suy ra: các ***định lý***
- Chuỗi các luật suy diễn được áp dụng để dẫn tới định lý: ***chứng minh*** của định lý
- Nếu các luật suy diễn là tin cậy, thì các định lý là **hệ quả logic** của các tiên đề

Vấn đề chứng minh phép suy diễn

- Tính đúng đắn của phép suy diễn: a → b?
- Hai phép suy diễn cơ bản của logic mệnh đề (Modus Ponens, Modus Tollens) cộng với các phép biến đổi hình thức có thể giúp ta chứng minh được phép suy diễn (reasoning)
- Tuy nhiên, thao tác biến đối hình thức là rất khó cài đặt trên máy tính

Giải thuật Robinson

- Giải thuật này hoạt động dựa trên phương pháp chứng minh phản chứng (proof by contradiction) và phép **hợp giải** Robinson (Robinson resolution).
- Chứng minh phản chứng:
  - Chứng minh phép suy luận (a → b) là đúng (với a là giả thiết, b là kết luận).
  - Phản chứng : giả sử ¬ b sai suy ra b là đúng

Giải thuật suy diễn của Robinson

Cho một tập tiên đề F, để chứng minh mệnh đề P là đúng, giải thuật suy diễn gồm các bước sau đây:

- B1 : Chuyển các mệnh đề trong tập F về dạng chuẩn hội.
- B2 : Phủ định mệnh đề P và chuyển kết quả này về dạng chuẩn. Đưa nó vào tập hợp mệnh đề đạt được ở bước B1.
- B3 : Lặp các thao tác sau đây cho đến khi tìm thấy sự mâu thuẫn xảy ra hoặc không có bước tiến triển nào:

  a) Chọn hai mệnh đề có thể hợp giải. Hợp giải hai mệnh đề này thành một mệnh đề mới.

  b) Nếu mệnh đề mới rỗng, thì sự mâu thuẫn đã xảy ra. Nếu không thì đưa mệnh đề mới vào tập mệnh đề đang có.

Giải thuật suy diễn

- Hai mệnh đề có thể hợp giải là hai mệnh đề có tính chất như sau. Có tồn tại một literal có dạng dương ở mệnh đề này và có dạng âm ở mệnh đề.
- Thí dụ: nếu ta có 2 mệnh đề:

  winter $\vee$ summer

  $\neg$winter $\vee$ cold

Từ hai mệnh đề này ta có thể diễn dịch ra mệnh đề mới:

summer $\vee$ cold.

Mệnh đề mới được gọi là ***mệnh đề hợp giải*** (resolvent)

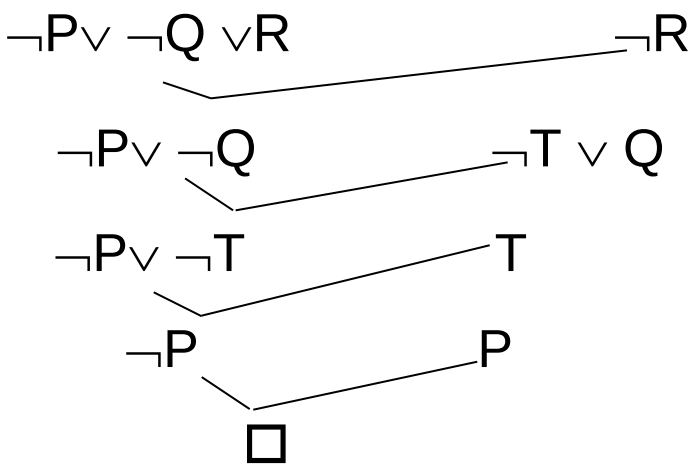
(Ghi chú: Luật hợp giải: $\alpha \vee \beta$, $\neg\beta \vee \gamma$ dẫn đến $\alpha \vee \gamma$)

Thí dụ về hợp giải

Cho tập mệnh đề = {P, (P $\wedge$ Q) $\rightarrow$ R, S $\rightarrow$ Q, T $\rightarrow$ Q, T}

cần chứng minh R đúng.

Quá trình suy diễn:



Tập mệnh đề sau biến đổi:
P
¬(P ∧Q) ∨R
¬S ∨ Q
¬T ∨ Q
T
¬R (thêm vào)

Hình 4.6. Quá trình hợp giải để chứng minh R đúng.

Logic vị từ

- Ký hiệu **hằng**:
  - Hằng của ngôn ngữ: true, false
  - Hằng do người sử dụng đặt cho tên đối tượng cụ thể: An, Binh, ..., a, b, c, … (đối tượng là các chủ ngữ hoặc tân ngữ trong mệnh đề)
- Ký hiệu **biến** (thường là biến đối tượng, đại diện cho chủ ngữ hoặc túc từ): x, y, z, t, u, …
- **Ký hiệu vị từ** (predicate symbol): P, Q, … hoặc Sinhvien, Yeu, Father, …(mỗi ký hiệu tương ứng với một vị từ (predicate) trong mệnh đề). Mỗi ký hiệu vị từ là mệnh đề đơn trong logic vị từ và có ngữ nghĩa true hay false.

Logic vị từ

- Ký hiệu **hàm**: sin, cos, log, f, …
- Ký hiệu **kết nối logic**: ¬, ∧, ∨, ⇒, ⇔
- Ký hiệu **lượng từ**:

  ∀, ∃ (∀: lượng từ phổ quát, ∃: lượng từ tồn tại
- Các ký hiệu “(“, ”)” và ”,”

Logic vị từ - Thí dụ

- “An là sinh viên”

  Sinhvien(An)

  % Sinhvien là vị từ một ngôi.
- “Nam là cha của Hoàn”

  Cha(Nam, Hoàn)

  % Cha là vị từ hai ngôi (2-place predicate)
- “Mọi sinh viên đều học giỏi”

  $\forall x$ Sinhvien(x) $\Rightarrow$ Hocgioi(x)
- “Trong sinh viên có bạn học giỏi”

  $\exists x$ Sinhvien(x) $\wedge$ Hocgioi(x)

Logic vị từ - Thí dụ

1. Marcus was a man.

   man(Marcus)

2. Marcus was a Pompeian.

   Pompeian(Marcus)

3. All Pompeians were Romans.

   $\forall x$: Pompeian(x) $\rightarrow$ Roman(x)

4. Caesar was a ruler.

   ruler(Caesar)

5. All Romans were either loyal to Caesar or hated him.

   inclusive-or

   $\forall x$: Roman(x) $\rightarrow$ loyalto(x, Caesar) $\vee$ hate(x, Caesar)

   exclusive-or

   $\forall x$: Roman(x) $\rightarrow$ (loyalto(x, Caesar) $\wedge \neg$hate(x, Caesar)) $\vee$ ($\neg$loyalto(x, Caesar) $\wedge$ hate(x, Caesar))

Logic vị từ - Thí dụ

6. Every one is loyal to someone.

   $\forall x: \exists y:$ loyalto(x, y) $\exists y: \forall x:$ loyalto(x, y)

7. People only try to assassinate rulers they are not loyal to.

   $\forall x: \forall y:$ person(x) $\wedge$ ruler(y) $\wedge$ tryassassinate(x, y)

   $\rightarrow \neg$loyalto(x, y)

8. Marcus tried to assassinate Caesar.

   tryassassinate(Marcus, Caesar)

Logic vị từ- thí dụ (tt.)

Giả sử chúng ta muốn dựa vào những phát biểu vừa nêu để trả lời câu hỏi sau đây:

Was Marcus loyal to Caesar?

man(Marcus)

ruler(Caesar)

tryassassinate(Marcus, Caesar)

$\Downarrow$ $\forall x$: man(x) $\rightarrow$ person(x) // thêm một luật mới

$\neg$loyalto(Marcus, Caesar) // kết hợp (7) và (8)

Phân giải trong logic vị từ

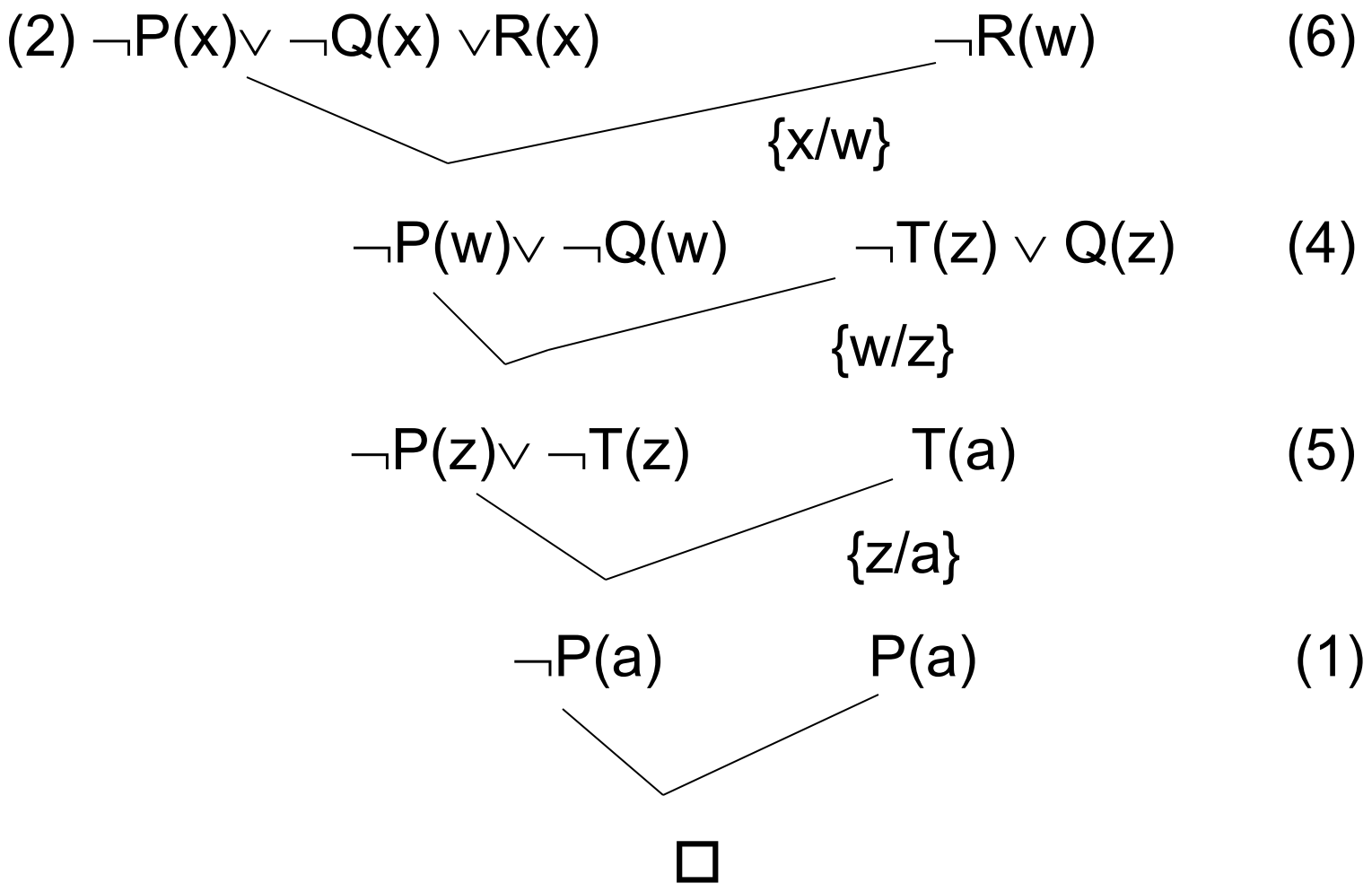
Thí dụ:

Tập mệnh đề = {P(a), $\forall$x: (P(x) $\wedge$ Q(x)) $\rightarrow$ R(x), $\forall$y: (S(y) $\vee$ T(y)) $\rightarrow$ Q(y), T(a)}

Chứng minh $\exists$a R(a) là đúng.

Tập mệnh đề được biến đổi về dạng chuẩn:

P(a)	(1)
$\neg$P(x)$\vee$ $\neg$Q(x) $\vee$R(x)	(2)
$\neg$ S(y) $\vee$ Q(y)	(3)
$\neg$T(y) $\vee$Q(y)	(4)
T(a)	(5)
$\neg$R(b)	(6)



Hình 4.8. Cây phân giải cho logic vị từ

Phép hợp nhất

Phép hợp nhất (unification) hai literal p và q:

UNIFY(p, q) = unifier θ where SUBST(θ, p) = SUBST(θ, q)

Cho các mệnh đề:

$\forall$x: knows(John, x) $\rightarrow$ hates(John, x)
knows(John, Jane)
$\forall$y: knows(y, Leonid)
$\forall$y: knows(y, mother(y))
$\forall$x: knows(x, Elizabeth)

Vài thí dụ về phép hợp nhất
UNIFY(knows(John, x), knows(John, Jane)) = {Jane/x}
UNIFY(knows(John, x), knows(y, Leonid)) = {Leonid/x, John/y}
UNIFY(knows(John, x), knows(y, mother(y))) = {John/y, mother(John)/x}

Vài thídụ về phép hợp nhất

UNIFY(knows(John, x), knows(y, Elizabeth)) = {John/y, Elizabeth/x}
UNIFY(knows(John, x), knows(x, Elizabeth)) = FAIL

Phép hợp nhất tổng quát nhất

Có thể tìm thấy nhiều phép hợp nhất có thể có giữa hai literal:

UNIFY(knows(John, x), knows(y, z)) = {John/y, John/x, John/z}
= {John/y, Jane/x, Jane/z}
= {John/y, v/x, v/z}
= {John/y, z/x, Jane/z}
= {John/y, z/x}

{John/y, z/x} là phép hợp nhất ***tổng quát nhất*** (most general unifier).

Kiểm tra occur check

Kiểm tra Occur check khi hợp nhất.

Thí dụ:

UNIFY(knows(x, x), knows(y, mother(y))) = FAIL

//x can not unify with f(x)

Chuyển đổi logic vị từ về dạng mệnh đề

Giải thuật: chuyển đổi công thức logic vị từ về dạng mệnh đề.

1. Loại trừ phép toán (implication) $\rightarrow$.

   $P \rightarrow Q \equiv \neg P \vee Q$

2. Thu giảm tầm vực của mỗi phép toán $\neg$ về một toán hạng (term) đơn.

   $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
   $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
   $\neg \forall x: P \equiv \exists x: \neg P$
   $\neg \exists x: P \equiv \forall x: \neg P$
   $\neg\neg P \equiv P$

3. Chuẩn hóa các biến sao cho một lượng từ liên kết với một biến đơn nhất.

   $(\forall x: P(x)) \vee (\exists x: Q(x)) \equiv (\forall x: P(x)) \vee (\exists y: Q(y))$

Chuyển đổi logic vị từ về dạng mệnh đề

4. Di chuyển tất cả các lượng từ sang bên trái mà không thay đổi thứ tự tương đối của chúng.

   $(\forall x: P(x)) \vee (\exists y: Q(y)) \equiv \forall x: \exists y: (P(x) \vee (Q(y))$

5. Loại trừ lượng từ tồn tại $\exists$ (Skolemization).

   $\exists x: P(x) \equiv P(c)$ Skolem constant

   $\forall x: \exists y\ P(x, y) \equiv \forall x: P(x, f(x))$ Skolem function

6. Bỏ các lượng từ $\forall$.

   $\forall x: P(x) \equiv P(x)$

7. Chuyển công thức sang dạng hội (conjunct) của các disjunct.

   $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$

8. Tạo một mệnh đề riêng lẻ cho mỗi mệnh đề tuyển (disjunct)

9. Làm cho các tên biến ***khác biệt nhau*** giữa các mệnh đề trong tập hợp các mệnh đề đã chuyển đổi.

Thí dụ: Suy diễn trong thế giới thực

1. Marcus was a man.
2. Marcus was a Pompeian.
3. All Pompeians were Romans.
4. Caesar was a ruler.
5. All Romans were either loyal to Caesar or hated him.
6. Every one is loyal to someone.
7. People only try to assassinate rulers they are not loyal to.
8. Marcus tried to assassinate Caesar.

Chứng minh:

hate(Marcus, Caesar)

Thí dụ

Chuyển từ những phát biểu bằng ngôn ngữ tự nhiên thành các mệnh đề logic vị từ

1. man(Marcus).
2. Pompeian(Marcus).
3. $\forall x$: Pompeian(x) $\rightarrow$ Roman(x).
4. ruler(Caesar).
5. $\forall x$: Roman(x) $\rightarrow$ loyalto(x, Caesar) $\vee$ hate(x, Caesar).
6. $\forall x$: $\exists y$: loyalto(x, y).
7. $\forall x$: $\forall y$: man(x) $\wedge$ ruler(y) $\wedge$ tryassassinate(x, y)
   $\rightarrow \neg$loyalto(x, y).
8. tryassassinate(Marcus, Caesar).

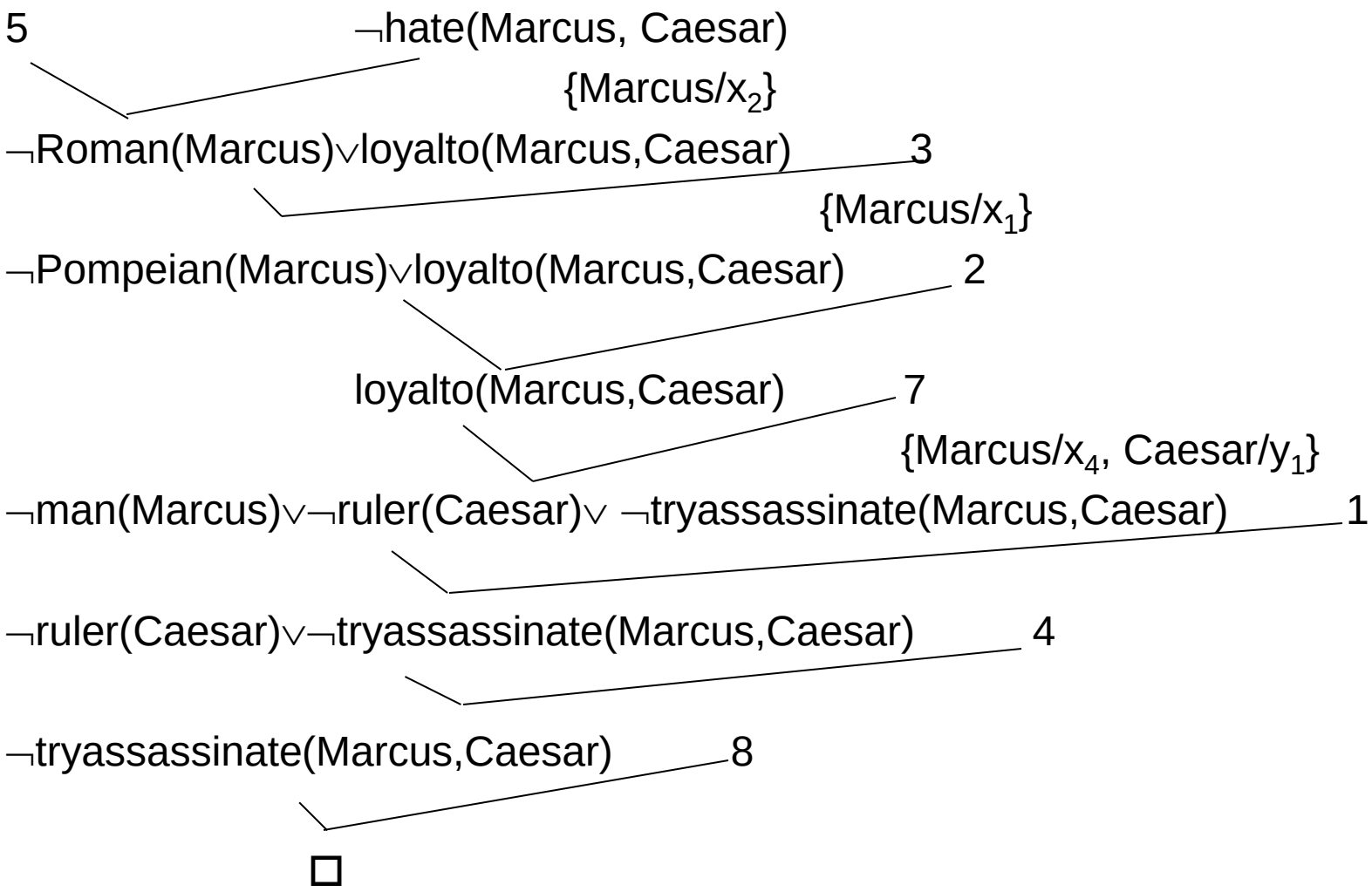
Chuyển tập phát biểu của thí dụ về dạng chuẩn

1. man(Marcus).
2. Pompeian(Marcus).
3. $\neg$ Pompeian(x1) $\vee$ Roman(x1).
4. ruler(Caesar).
5. $\neg$ Roman(x2) $\vee$ (loyalto(x2, Caesar) $\vee$ hate(x2, Caesar)).
6. loyalto(x3, f(x3)).
7. $\neg$ man(x4) $\wedge$ $\neg$ ruler(y1) $\wedge$ $\neg$ tryassassinate(x4, y1) $\vee$ $\neg$loyalto(x4, y1).
8. tryassassinate(Marcus, Caesar).

Chứng minh:

hate(Marcus, Caesar)

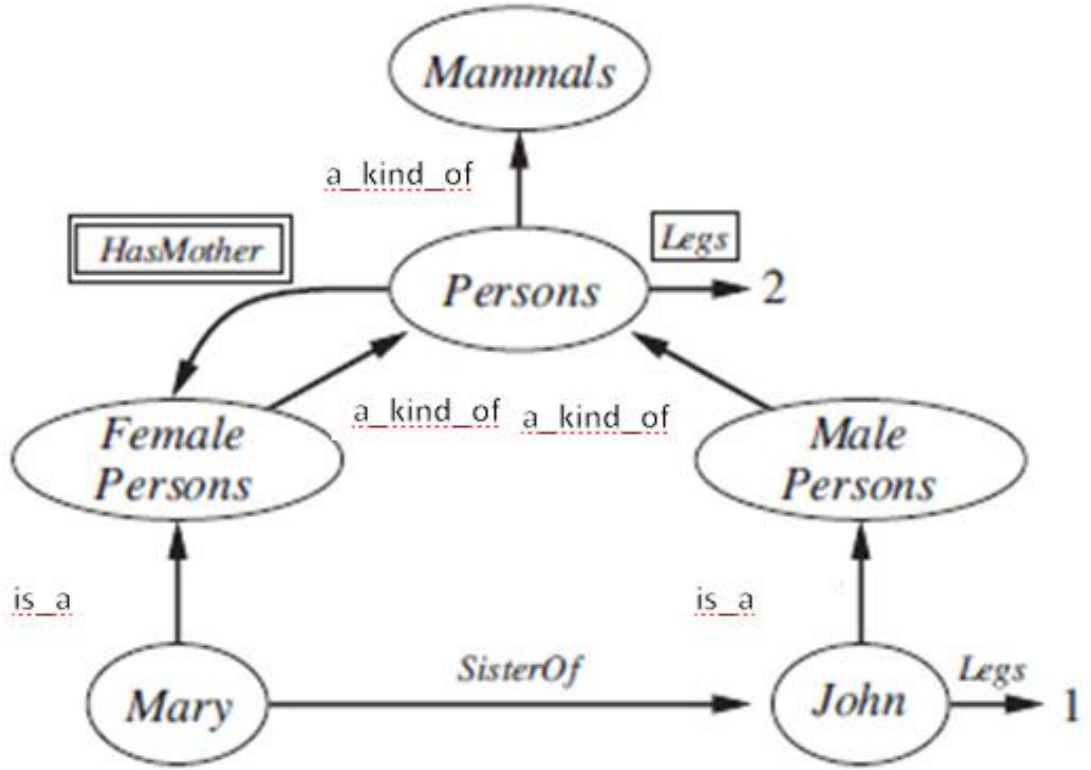
Hình 4.9. Quá trình phân giải để chứng minh hate(Marcus, Caesar)



5. Mạng ngữ nghĩa – Semantic Network

- *Mạng ngữ nghĩa* là một đồ thị có hướng được gán nhãn, trong đó mỗi nút thể hiện một đối tượng (một mệnh đề), và mỗi liên kết thể hiện mối quan hệ giữa 2 đối tượng

Hình 4.10. Một thí dụ về mạng ngữ nghĩa.



Mạng ngữ nghĩa

- **“IS-A”** (là một thể hiện của). Đề cập đến một thành viên của một **class**, trong đó một class là một nhóm đối tượng với một hay nhiều thuộc tính (properties). Ví dụ, “Tom IS-A bird”.
- **“A-KIND-OF”**. Đề cập một class với một class khác. Ví dụ, “Birds are A-KIND-OF animals”.
- **“HAS-A”**. Thuộc tính liên quan đến đối tượng. Ví dụ, “Mary HAS-A cat”.
- **“CAUSE”**. Thể hiện quan hệ nhân quả. Ví dụ, “Fire CAUSES smoke”.

Đặc điểm của mạng ngữ nghĩa

- Ưu điểm:
  - Mô hình thực thi rất đơn giản
  - Mô ta rất dễ đọc giúp nó dễ dàng thể hiện các bước suy diễn.
- Mở rộng mạng ngữ nghĩa:

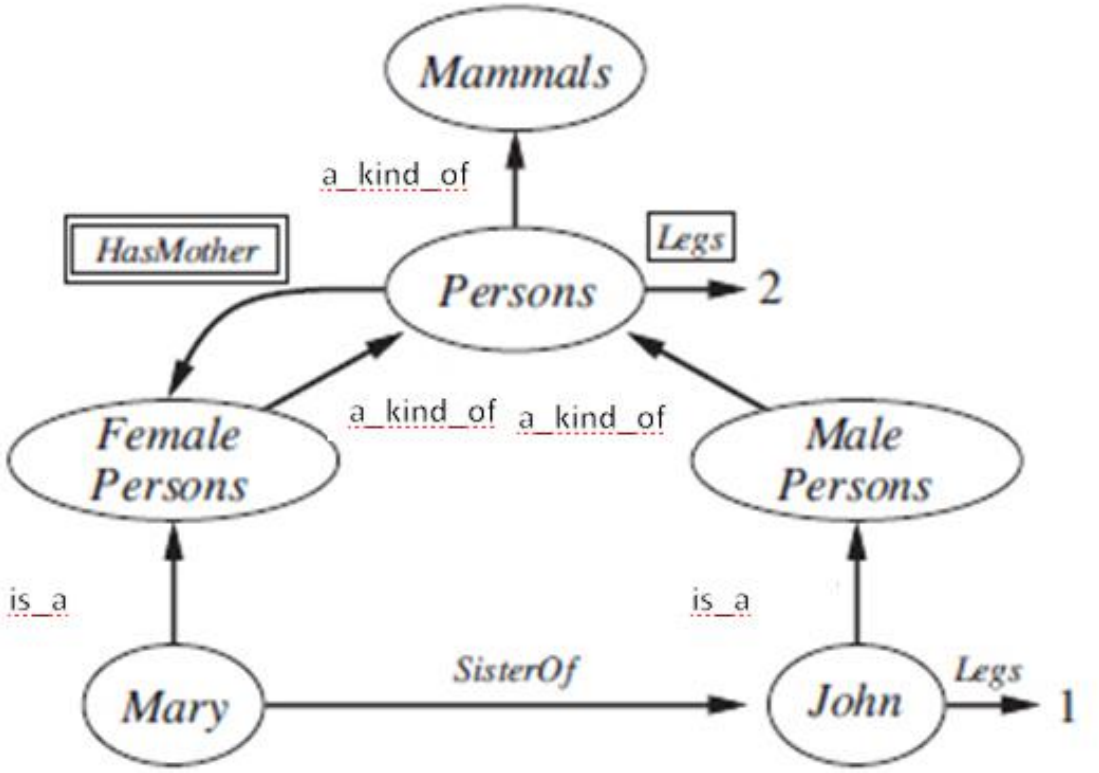
  Chỉ cần thêm các đỉnh và các cung quan hệ với các đỉnh có sẵn. Các đỉnh được thêm vào mạng hoặc là biểu diễn đối tượng hoặc là biểu diễn thuộc tính
- Tính thừa kế:

  Quan hệ đặc biệt “A-KIND-OF” để chỉ ra sự thừa kế
- Tính ngoại lệ:

  Định nghĩa một cung quan hệ mới đến một đỉnh có trị khác

Suy diễn trong mạng ngữ nghĩa

- Thủ tục suy diễn trong mạng ngữ nghĩa được gọi là thừa kế, và nó cho phép đặc điểm của một nút thành bản sao của nút con.



Ví dụ

"Persons are A-KIND-OF mammals"

"Male persons are A-KIND-OF persons"

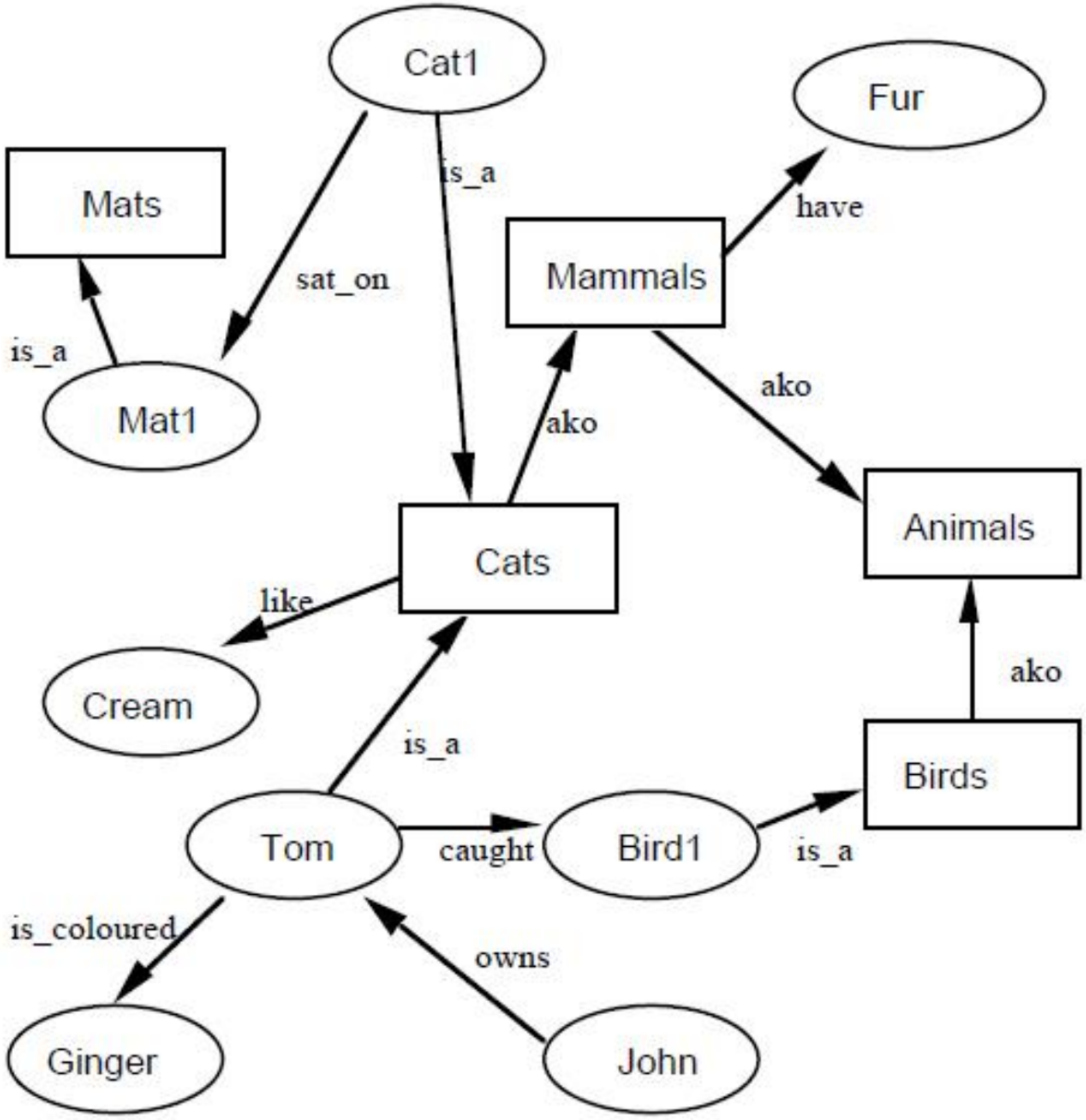
Giả sử thuộc tính của Persons gồm:

- "Persons HAS-A 2 legs"

Tất cả các thuộc tính của lớp cha (superclass): "Persons" sẽ được kế thừa bởi các lớp con

Ví dụ - Xây dựng mạng ngữ nghĩa

- Tom is a cat
- Tom caught a bird
- Tom is ginger in color
- Cats like cream
- The cat sat on the mat
- A cat is a kind of mammal
- A bird is a kind of animal.
- All mammals are animals
- Mammals have fur.



Hình 4.12 Xây dựng mạng ngữ nghĩa.

Kết luận

- Biểu diễn tri thức là một công việc quan trọng khi xây dựng một hệ thống thông minh.
- Có bốn lược đồ biểu diễn tri thức: lược đồ logic, lược đồ thủ tục, lược đồ mạng và lược đồ cấu trúc.
- Chương này tập trung vào hai phương pháp biểu diễn tri thức tiêu biểu: logic và mạng ngữ nghĩa.