TOPIC TODAY

INFERENCE IN FIRST-ORDER LOGIC AND LOGICAL REASONING SYSTEMS

TEAM MEMBERS

1. Vũ Gia Huy
21021319

2. Nguyễn Tuấn Hưng
21021321

3. Nguyễn Tuấn Hưng
21020155

4. Trần Duy Hưng
21021322

5. Nguyễn Văn Hữu
21021325

TOPICS COVERED

Topic 1: LOGICAL REASONING SYSTEMS

Topic 2: INFERENCE IN FIRST-ORDER LOGIC

LOGICAL REASONING SYSTEM

CÚ PHÁP

KÝ HIỆU

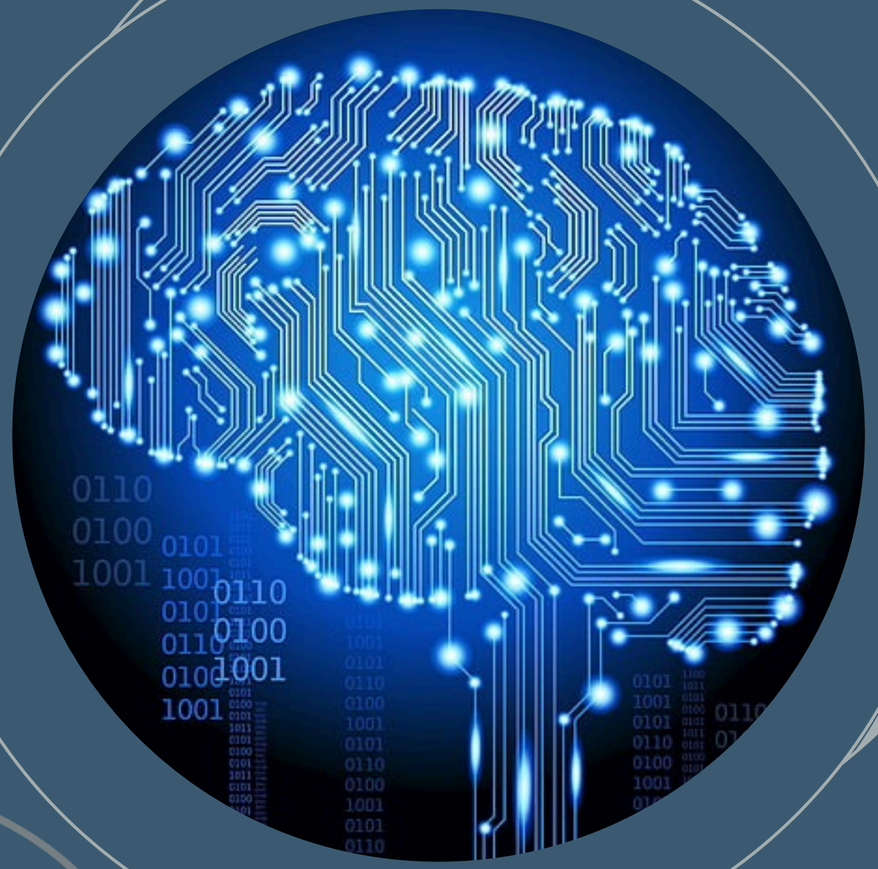
- Hằng: a, b, c, ...
- Biến: x, y, z, ...
- Vị từ: P, Q, R, ...
- Vị từ n biến $p(x_1, ..., x_n)$
- Vị từ không biến là mệnh đề
- Hàm: f, g, ... $f(x_1, ..., x_n)$ - hàm n biến
- Liên kết logic: $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$
- Lượng từ: $\forall$, $\exists$
- Dấu phẩy, đóng mở ngoặc

HẠNG THỨC

- Các ký hiệu hằng và biến
- Nếu $t_1, ..., t_n$ là các hạng thức, f là hàm n biến, thì $f(t_1, ..., t_n)$ là hạng thức

CÔNG THỨC PHÂN TỬ

- Các vị từ ko biến (mệnh đề)
- Nếu $t_1, ..., t_n$ là các hạng thức, P là vị từ n biến, thì $P(t_1, ..., t_n)$ là công thức phân tử



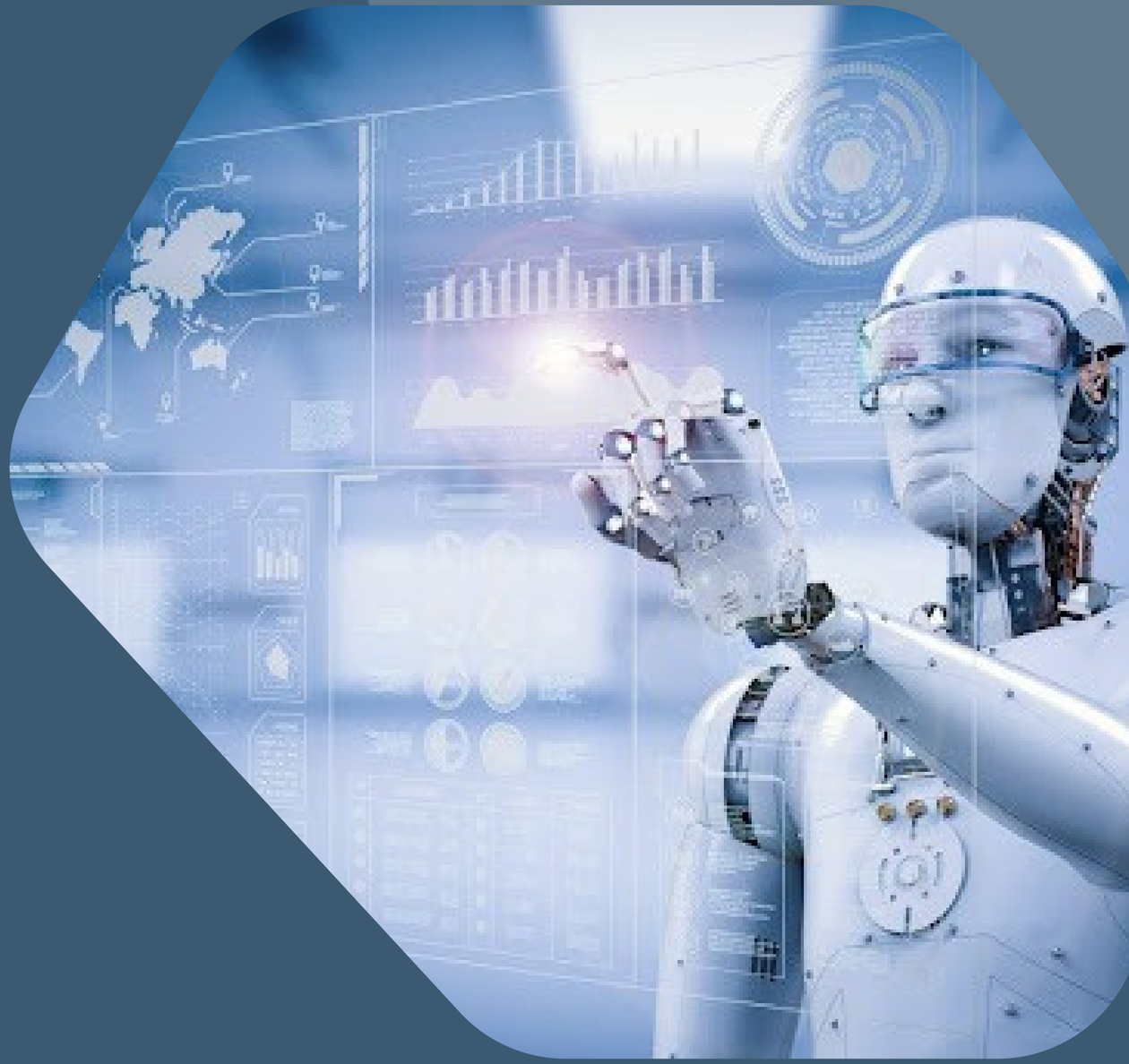
CÚ PHÁP

- CÔNG THỨC:

- Các công thức phân tử là các công thức
- Nếu P, Q là các công thức thì $P \wedge Q$, $P \vee Q$, $\neg P$, $P \Rightarrow Q$, $P \Leftrightarrow Q$ là các công thức
- Nếu P là công thức, x là biến thì $\forall xP$, $\exists xP$ là các công thức
- Literal: công thức phân tử hoặc phủ định của công thức phân tử
- Công thức đóng: công thức mà tất cả các biến đều là biến bị buộc
- Biến bị buộc x nếu trong công thức có dạng $\forall xP$ hoặc $\exists xP$, còn lại là biến tự do
- Ví dụ: $\forall xP(x, f(x,y)) \wedge \exists xQ(x)$

NGỮ NGHĨA

·Trong 1 diễn giải:
-Hằng → đối tượng cụ thể
-Hàm → hàm cụ thể
· Ngữ nghĩa của các câu đơn
-Ví dụ: Sinhviên(Lan)
· Ngữ nghĩa các câu phức
-Ví dụ: Sinhviên(Lan) ∧ Thích(Lan, Bóngdá)
· Ngữ nghĩa các câu chứa lượng từ
-∀xP : hội của tất cả các công thức nhận dược từ P bằng cách thay x bởi 1 đối tượng trong miền
-∃xP : tuyển của tất cả các công thức nhận dược từ P bằng cách thay x bởi 1 đối tượng trong miền



CÁC CÔNG THỨC TƯƠNG ĐƯƠNG

$\forall x\, P(x) \equiv \forall y\, P(y)$

$\exists x\, P(x) \equiv \exists y\, P(y)$

$\neg(\forall x\, P(x)) \equiv \exists x(\neg P(x))$

$\neg(\exists x\, P(x)) \equiv \forall x(\neg P(x))$

$\forall x\, (P(x) \wedge Q(x)) \equiv \forall x\, P(x) \wedge \forall x\, Q(x)$

$\exists x\, (P(x) \vee Q(x)) \equiv \exists x\, P(x) \vee \exists x\, Q(x)$

Ví dụ: $\forall x$ Thích(x, Chồng(x)) $\equiv \forall y$ Thích(y, Chồng(y))

CÁC LUẬT SUY DIỄN

Có 2 luật suy diễn cơ bản là: Suy diễn tiến và suy diễn lùi

SUY DIỄN TIẾN

Khái niệm:
Bắt đầu từ các câu có trong KB. Sử dụng GMP và các quy tắc suy diễn để sinh ra các câu mới cho đến khi không thể sinh ra câu mới nào nữa.

2 dạng GMP thường gặp

Dạng 1: p1 và p1’ có thể hợp nhất tại phép thế q
VD:
"x Cat(x) => Like(x,Fish) (1)
Cat(Tom) (2)
GMP (1), (2) => Like(Tom,Fish) {x /Tom}

Dạng 2: p1 và p1’, p2 và p2’ có thể hợp nhất tại phép thế q
VD:
"x,y Cat(x) ^ Like(x,y) => Eat(x,y) (1)
Cat(Tom)(2)
Like(Tom,Fish)(3)
GMP(1),(2),(3) => Eat(Tom,Fish) {x/Tom,y/Fish}

THUẬT TOÁN SUY DIỄN TIẾN

Forward Chaining Algorithm

```
function FOL-FC-ASK(KB, α) returns a substitution or false
  repeat until new is empty
    new ← ∅
    for each sentence r in KB do
      (p1 ∧ ... ∧ pn ⟹ q) ← STANDARDIZE-APART(r)
      for each θ such that (p1 ∧ ... ∧ pn)θ = (p'1 ∧ ... ∧ p'n)θ
                  for some p'1, ..., p'n in KB
        q' ← SUBST(θ, q)
        if q' is not a renaming of a sentence already in KB or new then do
          add q' to new
          φ ← UNIFY(q', α)
          if φ is not fail then return φ
    add new to KB
  return false
```

SUY DIỄN TIẾN

Khái niệm:
Ngược lại so với suy diễn tiến, suy diễn lùi bắt đầu từ các câu truy vấn sau đó tìm được các sự kiện và quy tắc trong KB cho phép chứng minh câu truy vấn là đúng.

Thủ tục suy diễn lùi:

Với mỗi câu q, nếu tồn tại q' hợp nhất được với q thì trả về hợp tử q được chứng minh.
VD:
KB: Cat(Tom)
Q:Cat(Tom)?
Câu truy vấn hợp nhất được với Cat(Tom) trong KB -> Q được chứng minh
Với mỗi quy tắc có vế phải q' hợp nhất được với q cố gắng chứng minh các phần tử vế trái bằng suy diễn lùi

Luật Modus Ponens:
VD:
KB:"x (Hoc(x)^Gioi(x) => VanHoa(x))
Q: VanHoa(Nam)
Câu truy vấn Q hợp nhất được vế phải của KB
=> Cần chứng minh vế trái Hoc(Nam)^Gioi(Nam) bằng suy diễn lùi.
Chú ý : Nếu câu truy vấn Q có dạng cấc câu đơn hội với nhau : A ^ B ^ C ...

THUẬT TOÁN SUY DIỄN LÙI

Backward Chaining Algorithm

function FOL-BC-ASK(*KB*, *goals*, θ) **returns** a set of substitutions
  inputs: *KB*, a knowledge base
    goals, a list of conjuncts forming a query (θ already applied)
    θ, the current substitution, initially the empty substitution $\varnothing$
  local variables: *answers*, a set of substitutions, initially empty

  if *goals* is empty **then return** $\{\theta\}$
  $q' \leftarrow$ SUBST(θ, FIRST(*goals*))
  for each sentence *r* **in** *KB*
    where STANDARDIZE-APART(*r*) = $(p_1 \wedge \ldots \wedge p_n \Rightarrow q)$
    and $\theta' \leftarrow$ UNIFY(q, q') succeeds
    new_goals $\leftarrow [p_1, \ldots, p_n | \text{REST}(goals)]$
    answers $\leftarrow$ FOL-BC-ASK(*KB*, *new_goals*, COMPOSE(θ', θ)) $\cup$ *answers*
  return *answers*

CHỨNG MINH BẰNG LUẬT PHÂN GIẢI

Chứng minh công thức H là hoặc không là hệ quả logic của tập công thức G bằng luật phân giải

Procedure Proof_by_Resolution

Input

G (các tiền đề)

H – công thức cần chứng minh;

Begin

1. Biến đổi G_I,¬H về dạng chuẩn hội;
2. Thành lập các câu tuyển C từ bước 1;
3. Repeat
   - Chọn 2 câu A, B từ C;
   - If A, B phân giải được then tính Res(A, B);
   - If Res(A, B) là câu mới then thêm Res(A, B) vào C;

Until nhận được câu rỗng hoặc không có câu mới nào được sinh ra;

4. If nhận được câu rỗng then thông báo H đúng else H sai;

End;

INFERENCE IN FIRST ORDER LOGIC

KHÁI NIỆM

- Là một khung sườn quan trọng trong lĩnh vực trí tuệ nhân tạo và khoa học máy tính, giúp máy móc thực hiện suy nghĩ và ra quyết định theo cách tương tự như con người.

GỒM HAI THÀNH PHẦN CHÍNH: CƠ SỞ TRI THỨC VÀ CƠ CHẾ SUY LUẬN

Cơ sở tri thức	Cơ chế suy luận
- Là một tập hợp dữ liệu và quy tắc được cấu trúc hóa, nơi lưu trữ các thông tin đã được xác minh hoặc được cho là đúng. - Các dữ liệu này có thể bao gồm: • Sự kiện (Facts) • Quy tắc (Rules) • Giả định (Assumptions)	- Là bộ phận trung tâm của hệ thống, sử dụng để phân tích và kết hợp thông tin từ cơ sở tri thức để suy ra kết luận mới. - Cơ chế suy luận có thể sử dụng các phương pháp sau: • Suy luận tiến (Forward Chaining) • Suy luận lùi (Backward Chaining)

VÍ DỤ ỨNG DỤNG

Hệ thống suy luận logic được ứng dụng trong nhiều lĩnh vực như:

- Hệ thống chẩn đoán y tế: Giúp xác định bệnh dựa trên triệu chứng và tiền sử y tế của bệnh nhân.
- Hệ thống hỗ trợ quyết định: Cung cấp các khuyến nghị và dự báo trong kinh doanh và quản lý.
- Hệ thống pháp lý tự động: Phân tích và áp dụng luật pháp để đưa ra phán quyết hoặc khuyến nghị về pháp lý.
- Robot thông minh: Sử dụng suy luận logic để đưa ra quyết định trong môi trường động và không xác định.

Hệ thống suy luận logic không chỉ hỗ trợ giải quyết các vấn đề phức tạp mà còn tạo cơ hội cho các hệ thống máy tính hiểu và tương tác với thế giới theo cách giống con người hơn.
Có nhiều loại suy luận khác nhau mà mỗi loại đều có cách tiếp cận riêng trong việc phân tích thông tin và đưa ra kết luận. Dưới đây là một số loại suy luận phổ biến:

- Suy Luận Quy Nạp (Inductive Reasoning)
- Suy Luận Diễn Dịch (Deductive Reasoning)
- Suy Luận Bổ Túc (Abductive Reasoning)
- Suy Luận Tương Quan (Analogical Reasoning)
- Suy Luận Khả Năng (Probabilistic Reasoning)
- Suy Luận Non-Monotonic

CÁC LOẠI SUY LUẬN

SUY LUẬN QUY NẠP

Là quá trình tạo ra các kết luận chung từ một tập hợp các quan sát cụ thể. Quá trình này thường không đảm bảo tính chính xác tuyệt đối của kết luận, nhưng nó có thể cung cấp một dự đoán hợp lý dựa trên các dữ liệu có sẵn.

SUY LUẬN DIỄN DỊCH

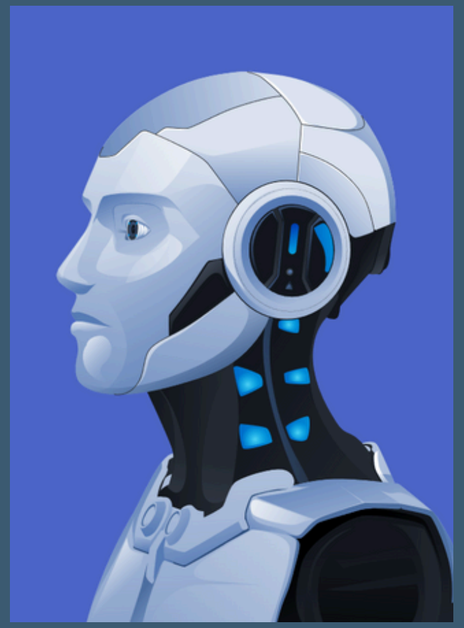
Trái ngược với suy luận quy nạp, suy luận diễn dịch bắt đầu từ một hoặc nhiều tiền đề chung và suy ra một kết luận cụ thể. Nếu các tiền đề là đúng, kết luận suy ra cũng phải đúng.

SUY LUẬN BỔ TÚC

Suy luận bổ túc là quá trình tìm ra giải thích có khả năng cao nhất cho một tập hợp các sự kiện. Loại suy luận này thường được sử dụng khi có một lượng thông tin hạn chế và cần phải đưa ra giả thuyết để giải thích các sự kiện.

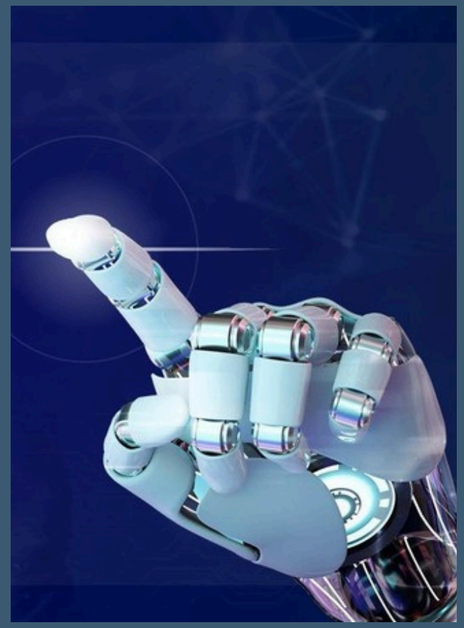
SUY LUẬN TƯƠNG QUAN

Suy luận tương quan dựa trên việc so sánh giữa các sự kiện, ý tưởng, hoặc tình huống tương tự để rút ra kết luận về một sự kiện hoặc tình huống mới.



SUY LUẬN KHẢ NĂNG

Suy luận khả năng là việc đưa ra quyết định dựa trên các xác suất của các sự kiện. Phương pháp này thường dựa trên các mô hình thống kê và được sử dụng rộng rãi trong các lĩnh vực có tính chất dự doán



SUY LUẬN NON-MONOTONIC

Trong suy luận non-monotonic, việc thêm thông tin mới có thể thay đổi kết luận trước đó. Suy luận non-monotonic phản ánh một cách tiếp cận linh hoạt hơn trong việc đối phó với thực tế phức tạp.

=> Mỗi loại suy luận này đều có vai trò và ứng dụng riêng, phù hợp với các tình huống và mục đích khác nhau, từ giải quyết các bài toán logic cho đến việc hỗ trợ ra quyết định trong cuộc sống hàng ngày.

KIỂU LOGIC TRONG SUY LUẬN LOGIC

Hệ thống suy luận logic có thể sử dụng nhiều kiểu logic khác nhau để xử lý thông tin và ra quyết định, bao gồm:

- Logic Mệnh đề (Propositional Logic)
- Logic Dự thức (Predicate Logic)
- Logic Mờ (Fuzzy Logic)
- Logic Temporal (Temporal Logic)

KIỂU LOGIC

LOGIC MỆNH ĐỀ

Sử dụng các biến mệnh đề và các phép toán logic như AND, OR, NOT để xây dựng các mệnh đề logic.

LOGIC DỰ THỨC

Mở rộng logic mệnh đề bằng cách thêm các hàm, các biến, và các lượng giác.

LOGIC MỜ

Được thiết kế để xử lý các thông tin không chắc chắn hoặc mơ hồ, logic mờ cho phép các giá trị thuộc về không chỉ đơn giản là đúng hoặc sai mà có thể nằm trong một phạm vi giữa hai giá trị này.

LOGIC TEMPORAL

Dùng để biểu diễn và suy luận về các mệnh đề có liên quan đến thời gian, như các sự kiện xảy ra trước hoặc sau sự kiện khác.

CÁC THÁCH THỨC

KHI TRIỂN KHAI HỆ THỐNG SUY LUẬN LOGIC, CÓ MỘT SỐ THÁCH THỨC CHÍNH CẦN ĐƯỢC GIẢI QUYẾT:

1. Phạm vi và Độ phức tạp: Việc xây dựng một cơ sở tri thức toàn diện và chính xác là điều không hề đơn giản. Cơ sở tri thức cần phải liên tục được cập nhật và bổ sung để phản ánh chính xác thực tế.

2. Hiệu suất và Tối ưu hóa: Cơ chế suy luận cần xử lý một lượng lớn dữ liệu và mối quan hệ. Việc tối ưu hóa các thuật toán suy luận để đạt được hiệu suất cao là rất quan trọng, đặc biệt trong các hệ thống thời gian thực.

3. Độ tin cậy và Chính xác: Suy luận dựa trên thông tin không đầy đủ hoặc sai lệch có thể dẫn đến kết quả không chính xác. Đảm bảo độ tin cậy của cơ sở tri thức và cơ chế suy luận là cực kỳ quan trọng.

THANK YOU FOR YOUR PRESENCE TODAY!

Anyone who has questions can send email for us to
hungtn2910@gmail.com