**Các phương pháp biểu diễn tri thức cơ bản**

* Logic vị từ

p(n) = “n là số nguyên tố”

p(5) = 1

p(10) = 0

us(a,b) = “a là ước số của b”

us(5, 10) = 1

us(5,22) = 0

us(5,b) = vị từ phụ thuộc 1 biến b.

gioi(x) = “x là giỏi”

**+ mô hình:** (**Predicates**, **Clauses**)  
**Predicates** là tập gồm các **vị từ**, mỗi vị từ biểu diễn cho phát biểu nói về một tính chất của đối tượng hay một quan hệ giữa các đối tượng. mỗi vị từ xác định bởi tên vị từ và các kiểu tham biến.  
Ví dụ: gioi(x:sinhvien)  
 Ví dụ: vg(v: vector, P: plane).  
**Clauses** là tập gồm các **biểu thức vị từ** gồm 2 dạng:

***fact: <vị từ>(các đối tượng)***

r***ule***: fact :- fact1, fact2, …, factk.

“if fact1, fact2, …, factk then fact”

Vd:

p(5).

Forall a, forall b, forall c, us(a,b) ∧ us(b,c) → us(a,c).

Viết thành: us(A, B) ∧ us(B, C) → us(A, C).

Final là: **us(A, C) :- us(A, B), us(B, C).**

**Cách thực hiện BDTT theo logic vị từ:**

* Có tri thức thực (tế) K được thu thập và ghi chép.
* Muốn biểu diễ K thành dạng logic vị từ: (P,C)

B1: phân tích các phát biểu tự nhiên trong K, để xác lập được các phát biểu cơ bản thuộc 2 loại: (1) nói về tính chất của đối tượng, (2) quan hệ giữa các đối tượng

* Xác định được các vị từ cần thiết cho biểu diễn: tập P.

B2: Liệt kê và phân loại các phát biểu trong K.

B3: Viết các phát biểu trong (B2) thành các biểu thức vị từ.

(B2 và B3 có thể lặp lại nhiều lần 🡪 C)

Predicates

nguyen\_to(integer)

us(integer, integer)

...

Clauses

Nguyen\_to(5).

us(5, 10).

us(A,C) :- us(A,B) , us(B,C).

**+ Tổ chức CSTT cụ thể trên máy ?**

+ Nên dùng **PROLOG**, công cụ xử lý biểu diễn theo vị từ.

Predicates

us(integer, integer)

uscln(integer, integer, integer)

clauses

uscln(A,0,A).

uscln(0,A,A).

uscln(A,B,D) :- A > B, A1 = A-B, uscln(A1,B,D).

uscln(A,B,D) :- A <= B, B1 = B-A, uscln(A,B1,D).

goal

uscln(24,36, What)

Run 🡺 What = 12

* Mạng ngữ nghĩa

+ Mô hình tri thức dạng đồ thị: (**Nodes**, **Arcs**)  
**Nodes** gồm các *yếu tố* hay các *bộ phận* cấu thành tri thức. Các node có thể là khái niệm, đối tượng, sự kiện, cấu trúc trừu tượng, …  
**Arcs** gồm các liên kết biểu diễn cho các quan hệ giữa các nodes. Các quan hệ có thể là: IS\_A, HAS\_A, …

**Cách biểu diễn:**

* Có tri thức thực (tế) K được thu thập và ghi chép.
* Muốn biểu diễn K thành dạng đồ thị: (Nodes,Arcs)

B1: …

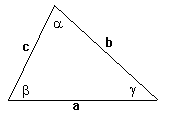
B2: …

+ **Tổ chức lưu trữ**: Dựa trên các kỹ thuật biểu diễn và tổ chức lưu trữ xử lý đồ thị: biểu diễn ma trận, biểu diễn dạng danh sách kề hay danh sách cạnh, …

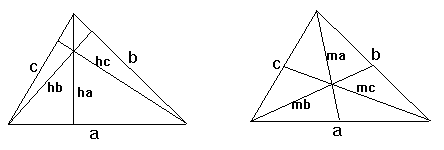
* Ví dụ: “Mạng tính toán trên tam giác” bao gồm 2 loại node là các “yếu tố” và các “công thức”, mỗi cung liên kết một công thức với một yếu tố có liên quan (có trong công thức).

**Các yếu tố của tam giác:**

* a, b, c : 3 cạnh của tam giác (Hình 1.1).
* α, β, γ : 3 góc đối diện với 3 cạnh tương ứng trong tam giác (Hình 1.1).
* ha, hb, hc : 3 đường cao tương ứng với 3 cạnh của tam giác (Hình 1.2a).
* ma, mb, mc : 3 đường trung tuyến tương ứng với 3 cạnh của tam giác (Hình 1.2b).
* pa, pb, pc : 3 đường phân giác trong.
* S : diện tích tam giác.
* p : nữa chu vi của tam giác.
* R : bán kính đường tròn ngoại tiếp tam giác.
* r : bán kính đường tròn nội tiếp tam giác.
* ra, rb, rc : các bán kính của các đường tròn bàng tiếp tam giác.



**Hình 1.1**



**Hình 1.2a**. 3 đường cao **Hình 1.2b**. 3 đường trung tuyến

**Các công thức tính toán thể hiện các luật liên hệ trên các yếu tố:**

* Liên hệ giữa 3 góc:

f1 : α + β + γ = π (radian).

* Định lý cosin :

f2 : a2 = b2 + c2 - 2.b.c.cosα

f3 : b2 = a2 + c2 - 2.a.c.cosβ

f4 : c2 = a2 + b2 - 2.a.b.cosγ

* Định lý Sin :

f5 :



f6 :



f7 :



f8 :



f9 :



f10 :



* Liên hệ giữa nửa chu vi và 3 cạnh :

f11 : 2.p = a + b + c

* Các công thức tính diện tích :

f12 : S = a.ha/2

f13 : S = b.hb/2

f14 : S = c.hc/2

f15 : S = p.r

f16 : S =



f17 : S = b.c.sinα / 2

f18 : S = c.a.sinβ / 2

f19 : S = a.b.sinγ / 2

* Caùc coâng thöùc tính ñöôøng cao theo caïnh vaø goùc :

f20 : ha = b.sinγ

f21 : ha = c.sinβ

f22 : hb = a.sinγ

f23 : hb = c.sinα

f24 : hc = a.sinβ

f25 : hc = b.sinα

* Các công thức tính các đường trung tuyến :

f26 : 4.ma2 = 2.b2 + 2.c2 - a2

f27 : 4.mb2 = 2.a2 + 2.c2 - b2

f28 : 4.mc2 = 2.a2 + 2.b2 - c2

* Các công thức tính các đường phân giác trong :

f29 : pa =



f30 : pb =



f31 : pc =



* Một số công thức khác :

f32 : R =



f33 : ra =



f34 : rb =



f35 : rc =



f36 : 4.R = ra + rb + rc - r

* Định lý Pitago:
* Định lý Ta-lét:
* Định lý liên quan đến “Tam giác đồng dạng”
* Định lý về “tam giác bằng nhau”
* Định lý về “đường trung bình”
* Bất đẳng thức tam giác
* Định lý liên hệ cạnh và góc (trường hợp tam giác cân)

**Chọn lọc và tỉnh chỉnh hay diễn đạt lại tri thức cần thiết đưa vào CSTT:**

* QĐ: bỏ ra các định lý dạng luật dẫn.
* Chỉ giữ lại những luật thể hiện dưới dạng các “hằng đẳng thức”
* Giữ lại các đk ràng buộc dạng bất đẳng thức
* Vấn đề tìm kiếm với thuật giải lan truyền

Phát biểu vấn đề:

Trên một mạng ngữ nghĩa (**Nodes**, **Arcs**), ta có khái niệm “*kích hoạt*” đối với các node và “*quy tắc lan truyền kích hoạt*”.

Giả sử có một tập các node H đã được kích hoạt trước (giả thiết), và

G (mục tiêu) là một tập node được quan tâm để được kích hoạt.

Hãy tìm cách kích hoạt G.

Thuật giải lan truyền:

Bước 1: Known = H; // **node**s được **kích hoạt**

T = []; // **Vết** lan truyền

Bước 2:

While not ( G in Known) do

2.1: Tìmkiếm node có thể được kích hoạt tiếp theo

dựa trên **quy tắc lan truyền**.

2.2: if ( tìm kiếm ở bước 2.1 thành công)

Cập nhật node kích hoạt mới vào Known;

Ghi nhận thông tin mới về vết lan truyền vào T;

Else

Kết thúc: Tìm kiếm thất bại

Bước 3: Tìm kiếm thành công.

Vđ 2: so khớp (matching)

* Trong lý thuyết đồ thị, có định nghĩa về sự “đẳng cấu” (hay đẳng hình) giữa 2 đồ thị.

Vđ 3: phân rã, tích hợp

Ví dụ 1: “Mạng tính toán trên tam giác”.

Tamgiac.txt:

Begin\_variables

a : cạnh a của tam giác.

b : cạnh b của tam giác.

…

S : diện tích tam giác.

…

End\_variables

Begin\_formulae

A + B + C = pi : công thức về góc tam giác.

S = sqrt(p\*(p-a)\*(p-b)\*(p-c)) : Công thức Hê-rông.

…

End\_formulae

Ví dụ 2: tự cho.

* Hệ luật dẫn  
  - Khái niệm luật dẫn:  
   + luật dẫn có dạng “**if … then …**”

- Ví dụ:

\* Định lý Pitago.

\* Định lý về 2 tam giác bằng nhau

\* Định lý Euler trong lý thuyết đồ thị

\* Ref: LVThS của N.T.T. Ly về chẩn đoán biến chứng

bệnh đái tháo đường.

Nếu bệnh nhân đái tháo đƣờng type 1 trẻ tuổi, kiểm soát đường máu kém, mắt mờ nhanh khám khúc xạ thấy đục thủy tinh thể nghĩ nhiều đến biến chứng mắt của bệnh đái tháo đường.

Trên bệnh nhân lớn tuổi mờ mắt khám thấy đục thủy tinh thể thì có thể không do biến chứng đái tháo đường. Trong trƣờng hợp này cần soi đáy mắt xem có dấu hiệu tăng sinh mạch máu không, nếu có mới xác định là đục thủy tinh thể do biến chứng đái tháo đường.

Đối với bệnh nhân đau nhứt mắt. Nếu giác mạc bình thường, cũng mạc bình thường mà nhãn áp cao thì chẩn đoán glaucoma (tăng nhãn áp ). Đây cũng là biến chứng trên mắt của bệnh đái tháo đường.

\* Ref: LVThS của T.T.N. Vân về “Hệ tư vấn kiến trúc nhà ở TPHCM”

* Theo quy định về luật xây dựng áp dụng ở thành phố Hồ Chí Minh, thì *nhà ở mặt tiền phải lớn hơn hay bằng 36 m2, đồng thời chiều ngang và chiều dài nhà tối thiểu là 3m*. Nhà trong hẻm tối thiểu là 9m2 và chiều ngang nhà và chiều dài nhà lớn hơn 2m [9,tr.6]. Ngòai các luật xây dựng được ban hành trong quyết định QĐ135/2007-UBND [9], thì ở mỗi quận , huyện đều có những quy định riêng trong xây dựng. Và đề tài này chỉ tư vấn dự trên quy định về luật xây dựng của Ủy Ban Nhân Dân thành phố Hồ Chí Minh.
* Cũng theo quy định về luật xây dựng hiện nay trên địa bàn thành phố Hồ Chí Minh thì nhà ở có chiều cao tầng từ 1 đến 8 tầng tùy thuộc vào khu vục xây dựng nhà ở[8]. Các công trình có chiều cao lớn hơn 8 tầng là không thuộc phạm vi tư vấn của đề tài này. Một công trình nhà ở được đánh số tầng bắt đầu là 1 cho tầng trệt, và các tầng trên được đánh số tiếp theo là 2,3,4,5,6,7,8. Ngoài ra, trong phạm vi đề tài, xin tạm đánh số tầng hầm là -1, và tầng lửng được đánh số là 1.5
* Một giới hạn nữa cũng cần phải xem xét trong phạm vi của đề tài đó là mỗi công trình nhà ở chỉ tối đa 3 lộ giới. Lộ giới là khỏang cách tính từ tim đường đến mặt tiền tầng trệt. Lộ giới này cũng chỉ giới hạn trong phạm vi lớn hơn hay bằng 4.5 m. Và mỗi công trình nhà ở sẽ có mặt tiền chính quay về một lộ giới.
* Mô hình Hệ luật dẫn: **(Facts, Rules)**

Facts là tập các *sự kiện* hay *tác vụ* trong phạm vi tri thức.

Rules là tập các luật dẫn, mỗi r thuộc Rules có dạng

r: gt(r) kl(r)



* Sự kiện (fact) ?  
  NX: sự kiện trong thực tế thường là có cấu trúc như: (1) phát biểu về tính chất của 1 phần tử, (2) phát biểu về quan hệ (2 ngôi hay nhiều ngôi) giữa các phần tử.

Ví dụ: một phần kiến thức về một tam giác trong hình học

1. Các yếu tố của tam giác, mỗi sự kiện là một phát biểu nói lên tính xác định của yếu tố 🡪 ký hiệu cho các sự kiện: a, b, c, A, B, C, S, p, R, ha, hb, hc, …
2. Các luật nói lên liên hệ “dẫn xuất” giữa các sự kiện:

* A+B+C = pi:  
  if A, B then C;  
  if B, C then A;  
  if A, C then B;
* a/sin(A) = b/sin(B) = c/sin(C) = 2R:  
  if a, A then R;  
  v.v …

+ Mô hình: (**Facts**, **Rules**)  
**Facts** gồm các phát biểu chỉ các sự kiện hay tác vụ nào đó.  
**Rules** gồm luật dẫn có dạng “**if … then …**”.

Phần tri thức trên có thể biểu diễn theo mô hình hệ luật dẫn (Facts, Rules) gồm:

**Facts = {a, b, c, A, B, C, S, p, R, ha, hb, hc, …}**

**Rules = {**

**r1:** **{A, B} → {C}, và C = pi – A – B;**

**r2: {A, C} → {B}, và B = pi – A – C;**

**v.v….  
 }**

Hãy xây dựng cách tổ chức lưu trữ kiến thức diễn đạt bởi mô hình trên thành các tập tin trên đĩa.

+ Tổ chức lưu trữ ?: được xác lập cụ thể dựa trên các dạng facts. Thường là ta sử dụng các cấu dữ liệu đã biết như struct, frames, classes, …

Lưu trên đĩa: sử dụng 2 tập tin dạng “text có cấu trúc”: Facts.txt và Rules.txt

Cấu trúc file Facts.txt:

Begin

a : cạnh a của tam giác.

b : …

v.v…

End

Cấu trúc file **Rules.txt**:

Begin

{A, B} => {C}, C = pi – A - B : luật về góc tam giác.

v.v…

End

Với tri thức lưu trữ bởi 2 tập tin trên, để có thể đọc kiến thức vào chương trình viết theo C/C++ ta cần phải định nghĩa những **kiểu dữ liệu** gì? Có cần biến toàn cục gì không?

* “sự kiện”: dạng chuỗi ghi lại ký hiệu của yếu tố được xem xét hay quan tâm
* Tập các sự kiện: cấu trúc dãy với array 1 chiều cấp phát động.
* Luật dẫn: Cấu trúc gồm: (1) tập sự kiện (giả thiết), (2) tập sự kiện (kết luận), công thức tính sự kiện kết quả.
* Tập các luật dẫn: cấu trúc dãy với array 1 chiều cấp phát động.
* Công thức dạng “<biến> = <biểu thức>”: dạng chuỗi ghi lại công thức.  
  VD: “C = pi – A – B”

**Vấn đề suy diễn (suy luận) trên hệ luật dẫn:**

Cho trước hệ luật dẫn **K** = (Facts, Rules). Giả sử có một tập sự kiện **GT** đã xác định, ta xét một tập sự kiện mục tiêu **KL**. Hỏi có thể suy ra KL từ GT dựa trên tri thức K hay không ?

Ký hiệu bài toán: GT 🡪 KL

**Thuật giải suy diễn tiến:**

Các biến trong thuật giải:

* Tập sự kiện đã biết: **Known**
* Danh sách các luật được áp dụng: **Solution**

B1: Solution = [];  
 Known = GT;

B2: while (KL chưa nằm trong Known) do  
 2.1: Tìm luật r để áp dụng trên Known nhằm sinh ra sự kiện mới:

gt(r) ⊆ Known, và kl(r) không nằm trong Known.

2.2: if (không có r) then

Dừng: không tìm được lời giải

2.3: Thêm r vào Solution; thêm kl(r) vào Known;

End while

B3: Tìm được lời giải sử dụng danh sách luật Solution

**Ghi chú:**

* Cần loại bỏ luật thừa trong Solution.

**Bài tập:**

**Thuật giải suy diễn lùi:**

…

Next:

* Frames và classes
* Scripts

**Vận dụng:**

1. **Nắm vững** mỗi mô hình hay phương pháp: Cấu trúc của mô hình, biết tổ chức tổ chức lưu trữ và xử lý mô hình trên máy.
2. Kết hợp, phối hợp các mô hình để tạo nên mô hình phức tạp đáp ứng cho việc biểu diễn, tổ chức tri thức của ứng dụng thực tế.
3. Vận dụng mô hình tổng quát vào vấn đề hay tri thức cụ thể 🡪 dạng mô hình đặc biệt hơn.  
   Ví dụ 1: Mạng suy diễn - tính toán (Computational Network for knowledge representation)  
   Ví dụ 2: Đồ thị khái niệm (Conceptual Graph) 🡪 ứng dụng trong semantic web, biểu diễn ngữ nghĩa của các tài liệu văn bản.  
   Ví dụ 3: Sơ đồ tư duy (Mind map)

**Vấn đề suy diễn: (dạng phổ biến)**

**Cho trước (giả thiết) một tập các sự kiện liên quan đến một số đối tượng nhất định. Hãy suy ra một tập các sự kiện (kết luận) được quan tâm, dựa trên một cơ sở tri thức K trong đó có tập luật Rules.**

**Thuật giải suy diễn tiến ở dạng tổng quát:**

**Bước 1:**

**A = gt;**

**Sol = []; // danh sách luật được áp dụng**

**Bước 2:**

**While ( A không chứa kl )**

**2.1: Tìm luật r (trong Rules) để áp dụng trên tập A**

**(chỉ sử dụng một tập sự kiện A(r) trong A) sinh ra sự kiện mới (tập sự kiện mới được ký hiệu là B(r)).**

**2.2: if (tìm r bị thất bại)**

**Dừng: bị bế tắc.**

**2.3: Thêm r vào Sol; A = A B(r)**



**Bước 3:**

**Kết luận về lời giải Sol.**

**Ghi chú:**

1. **Sol có thể được ghi nhận dưới dạng danh sách các “*bước giải*”, mỗi bước giải ít nhất gồm r, A(r), B(r).**
2. **Sol có thể (thường là) bị thừa theo nghĩa …**
3. **Để giải quyết việc loại các bước thừa ta có thể dựa trên “dạng cây” của phép suy diễn tiến. Tuy nhiên, ta thường loại bước thừa trong Sol bằng cách duyệt ngược danh sách Sol để loại bước thừa 🡪thuật giải ?**

**Thuật giải suy diễn lùi ở dạng tổng quát:**

**??**

**FRAMES VÀ CLASSES**

1. **Class :**

* Giới thiệu chung.
* Class :

1. **Trong OOP :**

* khái niệm :   
  Cấu trúc trừu tượng có các thành phần gồm 2 loại :  
   (1) các biến thể hiện các thuộc tính dữ liệu hay thông tin  
   (2) các phương thức thể hiện (biểu diễn) cho các hành vi của đối tượng, và được cụ thể hóa (về mặt cài đặt trong lập trình) bởi các hàm và toán tử.
* Công cụ hỗ trợ lập trình: C ++
* Công dụng biểu diễn : Classes thường biểu diễn cho các khái niệm trong miền tri thức của người.
* Còn có thể biểu diễn các quan hệ.
* Ví dụ áp dụng.

Khẳng định : « class » là một công cụ hữu ích cho việc BDTT. Why ?

+ Tăng khả năng biểu diễn dữ liệu, thông tin, tri thức.

+ Nền tảng của tri thức là các khái niệm và các liên hệ (dựa trên các thể hiện cụ thể của các khái niệm, đó là các thực thể ) 🡺 sử dụng « class » để biểu diễn cho các khái niệm là điều tất nhiên.

1. **Trong « BDTT » :**
   * + - « class » sẽ được hiểu theo nghĩa trừu tượng hơn :  
         + cấu trúc thành phần : các thuộc tính, các quan hệ trên các thuộc tính, các tính chất, các luật nội tại ;   
         + các phương thức, các hành vi trừu tượng khác như suy diễn giải quyết vấn đề, giao tiếp, …  
         + có cả khái niệm về các thuốc tính đặc trưng, slot.
       - Các thuộc tính đặc trưng thường có giá trị default.
2. **Frame : (**[**xem**](file:///C:\Users\nhondv\AppData\Roaming\Microsoft\Word\frame_based_sys.ppt)**)**

(1) **BDTT** cho hệ giải bài toán trên tam giác với giả thiết có thể gồm 2 loại sự kiện: <biến>=giá trị, liên hệ các biến bởi đẳng thức.

Mẫu 1:

GT = { a=5, b = 4, c = 3}

KL = { S, R }

Mẫu 2:

GT = { a=5, A = 2B, C = A+B }

KL = { S, R }

Chương trình khi thực thi cho:

Nhập: bài toán

* Giả thiết
* Kết luận (mục tiêu)

Xuất: lời giải

* Lời giải chi tiết

**Kiến thức thu thập cần thiết cho ứng dụng:**

* Danh sách yếu tố 🡪 các biến (số thực dương)
* Tập các công thức (xem ơ trên)
* Các định lý khác như: Pitago

**Tiếp cận dùng hệ luật dẫn:**

* Sự kiện: 2 loại  
  (1) <biến> = <giá trị hằng>  
  <biến> thuộc { a, b, …, r }  
  (2) <bt> = <bt> (xét sau)
* Luật dẫn với :  
  \* giả thiết là tập các sự kiện, được ghi theo cách …  
  \* kết luật là tập các sự kiện

If {A, B} then { C = pi – A – B}

**Mô hình tri thức như sau:**

(V, F, Rules)

V = { a, b, …, r }

F = {fact : fact có dạng (1) <biến> = <giá trị hằng>,   
 (2) <bt> = <bt>}

Rules = { luật dẫn dạng:  
 if {biến thuộc V với giá trị đã biết} then { <biến> = bt theo các biến trong giả thiết của luật} }

Ví dụ:

If {A, B} then { C } với C = pi – A – B.

Tổ chức lưu trữ:

V.txt

---------------------------------------------------------

Begin variable

a : cạnh a của tam giác, là biến có giá trị thực dương;

b : cạnh a của tam giác, …;

…

End variable

---------------------------------------------------------

Kind\_facts.txt

---------------------------------------------------------

Begin KindsOfFacts

<variable> = <value> : sự kiện cho biết <variable> có giá trị là <value>;

<bt theo các biến> = <bt theo các biến>: …

End KindsOfFacts

---------------------------------------------------------

Rules.txt

---------------------------------------------------------

Begin Rules

Begin rule <?>

{ một số biến thuộc V}; // tập biến giả thiết

{ <biến> = bt(các biến giả thiết)}

End rule

Begin rule <tính chất góc của tam giác>

{ A, B}; // tập biến giả thiết

{ C = pi – A – B }

End rule

Begin rule <Tính S theo a, b, C>

{ a, b, C}; // tập biến giả thiết

{ S = a\*b\*sin(C) / 2 }

End rule

End Rules

---------------------------------------------------------

**Thật giải suy diễn tiến cụ thể giải GT 🡪 KL**

B1: Solution = [];  
 Known1 = tập các sự kiện loại 1 trong GT;

Known2 = tập các sự kiện loại 2 trong GT;

B2: while (KL chưa nằm trong Known1) do  
 2.1: Tìm luật r để áp dụng trên Known nhằm sinh ra sự kiện mới:

gt(r) ⊆ Known1, và kl(r) không nằm trong Known1.

2.2: if (không có r) then

Dừng: không tìm được lời giải

2.3: Thêm r vào Solution;   
thêm kl(r) vào Known;

End while

B3: Tìm được lời giải sử dụng danh sách luật Solution

(2) hệ giải bài toán logic: rút gọn bt Logic, kiểm chứng suy luận logic.

**NHẬN XÉT VỀ CÁC PHƯƠNG PHÁP BIỂU DIỂN TRI THỨC CƠ BẢN**

* **Nhận xét chung:**

1. Biểu diễn được các **dạng tri thức thành phần** **khá phổ biến** trong các miền tri thức trong các lĩnh vực ứng dụng thực tế của con người.
2. Đơn giản, dễ cài đặt trên máy tính.
3. Chỉ biểu diễn được tri thức bộ phận (hay cục bộ), chưa phù hợp cho việc biểu diễn cho một **miền tri thức** hay cho **một CSTT**, vì tri thức con người thường đa dạng và phức tạp với cấu trúc nhiều thành phần như khái niệm, quan hệ, sự kiện, luật, v.v…
4. Chưa quan tâm đến cơ sở lý thuyết cho biểu diễn những thành phần tri thức cơ bản trong tri thức con người như: các khái niệm (loại thực thể) với những liên hệ cơ bản, các quan hệ giữa/trên khái niệm, sự kiện, luật, …

* **Đánh giá cụ thể một số phương pháp:**

1. Biểu diễn tri thức theo logic vị từ
2. Hệ luật dẫn
3. Mạng ngữ nghĩa
4. Frame và class

* **Định hướng mở rộng, phát triển:**
* Mở rộng bổ sung thành phần tri thức mới.
* Làm rõ hơn (hay cụ thể hóa) một số yếu tố hay thành phần trong mô hình 🡪 các mô hình đặc thù với nhiều vấn đề kỹ thuật mới được đặt ra.
* Kết hợp, phối hợp các phương pháp cơ bản.

**Một số đề xuất:**

1. Hệ luật dẫn trên cơ sở các khái niệm hay thuộc tính, và các sự kiện được mô hình hóa hay định nghĩa rõ ràng.
2. Mạng tính toán cơ bản: một dạng mạng ngữ nghĩa.
3. Đồ thị khái niệm: một dạng mạng ngữ nghĩa
4. Đối tượng tính toán cơ bản: dạng class đóng gói mô hình mạng tính toán cơ bản cùng với một số thủ tục (thuật giải).