ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH

ĐỒ ÁN MÔN HỌC TRÍ TUỆ NHÂN TẠO:

XÂY DỰNG CHƯƠNG TRÌNH GIẢI BÀI TOÁN TAM GIÁC TRÊN MÁY TÍNH

Nhóm thực hiện: Lưu Thanh Sơn – MSSV:14520772

Đỗ Phú An – MSSV: 14520002

Giảng viên hướng dẫn: PGS.TS Đỗ Văn Nhơn

ThS Nguyễn Thị Ngọc Diễm

Lớp: Trí tuệ Nhân tạo (CS106.G21.KHTN)

Thành phố Hồ Chí Minh, ngày 15 tháng 6 năm 2016

MỤC LỤC

CHƯƠNG 1: MỞ ĐẦU 3
1. Giới thiệu: 3
2. Ứng dụng khả năng của máy tính vào học tập: 3
3. Mục tiêu cụ thể của Đồ án: 4
CHƯƠNG 2: TỔNG QUAN VỀ BIỂU DIỄN TRI THỨC 5
1. Mở đầu 5
2. Phân loại tri thức 5
3. Biểu diễn tri thức trên máy tính 5
a. Logic Mệnh đề 5
b. Logic Vị từ 6
c. Luật sinh 7
d. Mạng ngữ nghĩa: 8
e. Frame: 9
CHƯƠNG 3: CƠ SỞ THUẬT TOÁN 10
1. Cơ chế suy diễn tiến 10
2. Cơ chế suy diễn lùi 11
3. Tối ưu tập luật 12
a. Rút gọn bên phải 12
b. Rút gọn bên trái 12
c. Luật thừa 12
d. Tối ưu tập luật dẫn 12
CHƯƠNG 4: CẤU TRÚC CHƯƠNG TRÌNH 13
1. Giao diện chính của chương trình 13
2. Mã nguồn chương trình 14
a. Lớp Rule.cs 14
b. Lớp ForwardReasoning.cs 15
c. Lớp Calculate.cs 19
d. Lớp Process.cs 24
3. Đánh giá về Chương trình 29
TÀI LIỆU THAM KHẢO 30

CHƯƠNG 1: MỞ ĐẦU

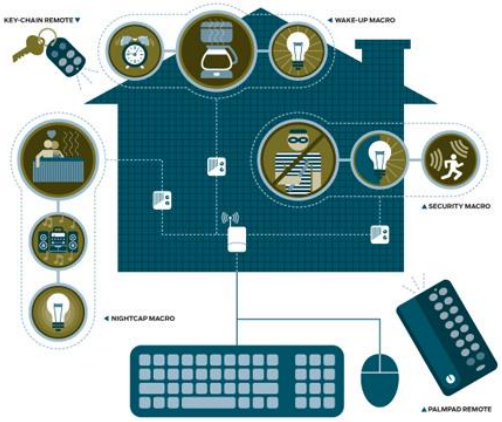
1. Giới thiệu:

- Hiện nay, khả năng của máy tính ngày càng được mở rộng về tốc độ và hiệu năng, máy tính được ứng dụng trên khắp các lĩnh vực trên đời sống như: thương mại, giao dịch ngân hàng, quảng cáo, điều khiển hệ thống, lưu trữ, giải trí, học tập, nghiên cứu,…. Vì vậy, việc xây dựng các ứng dụng phục vụ cho các nhu cầu trên cũng trở nân cần thiết hiện nay.
- Mục tiêu của Trí tuệ nhân tạo là nghiên cứu các phương pháp, kỹ thuật cần để cho máy tính có thể hiểu và suy nghĩ như một con người, từ đó, ta áp dụng khả năng đó của máy tính vào những ứng dụng cụ thể trong đời sống.
- Trong số các ứng dụng trong đời sống, học tập là một lĩnh vực đòi hỏi sự áp dụng của máy tính nhiều nhất và phổ biến nhất.
- Một số ứng dụng cụ thể của máy tính:

Thương mại điện tử

Giao dịch ngân hàng

Học tập (E-Learning)

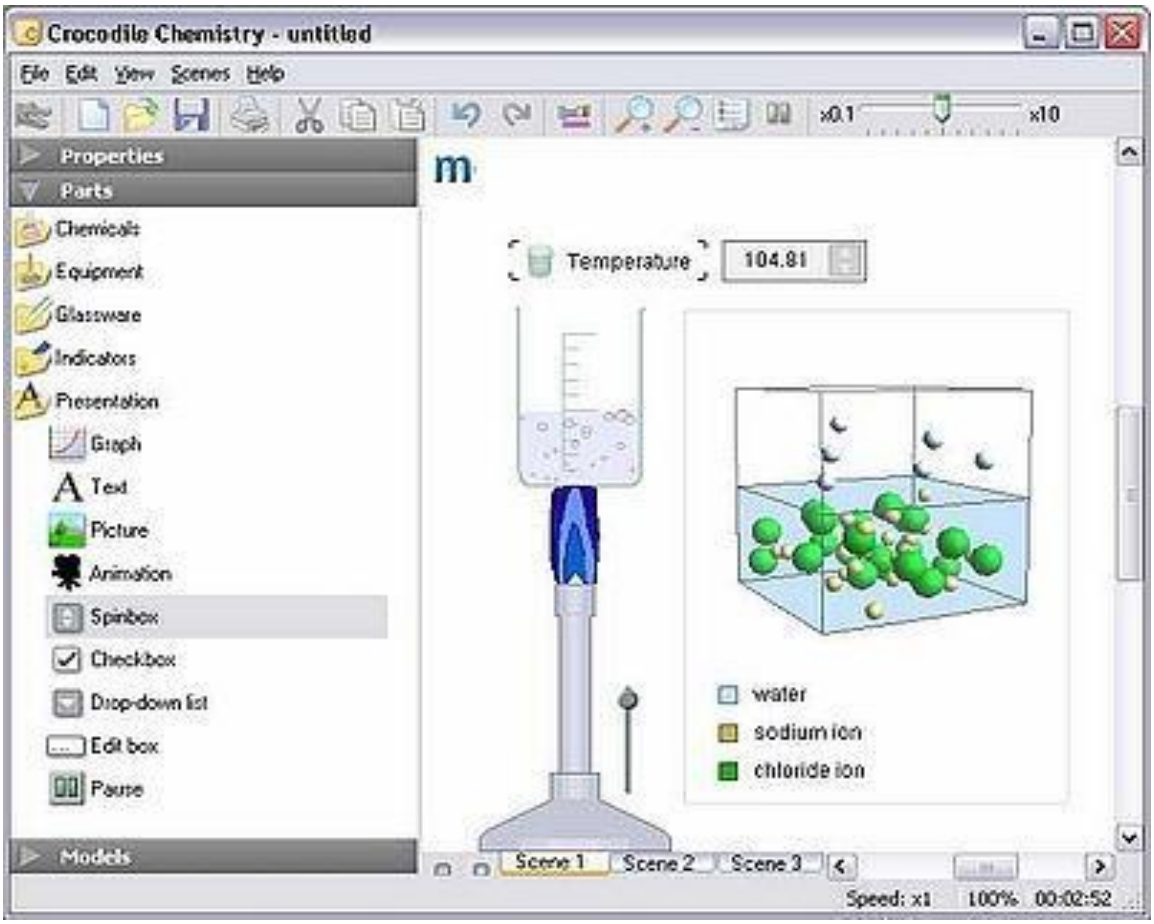


Điều khiển tự động (Smart Hoiuse)

2. Ứng dụng khả năng của máy tính vào học tập:

- Máy tính có khả năng xử lý, tính toán nhanh và chính xác, vì vậy, nhu cầu áp dụng nó vào việc học tập, cụ thể là giải bài tập, cũng được phát triển. Vấn đề là, khả năng máy tính không chỉ giới hạn ở việc tính toán đơn thuần (như các phép toán cộng, trừ, nhân, chia cơ bản với input là do con người nhập và chỉ biết tính toán đơn thuần!) mà nó còn có khả năng "suy nghĩ" như một con người, nghĩa là biết rút trích thông tin từ những tri thức có sẵn để giải quyết một bài toán cụ thể.

- Khi Công nghệ thông tin phát triển như vũ bão hiện nay, hình thức học tập cũng có sự thay đổi. Ngày xưa, ta học ở Thầy, Cô. Ngày nay, ngoài Thầy, Cô ra, ta còn có thể học qua máy tính, nhờ sự hỗ trợ của Máy tính giúp ta trong việc học tập.
  Vd: Với chương trình hỗ trợ hỏi đáp về Toán Hình Học Phẳng, khi ta nhập một câu hỏi về Hình học mà ta muốn tìm hiểu, giả dụ như: "Hình bình hành có 1 góc vuông có là hình chữ nhật hay không ?", máy tính sẽ tìm hiểu trong cơ sở tri thức có sẵn và cho ta kết luận "Đúng, Hình bình hành có một góc vuông là Hình chữ nhật". Đó là một ví dụ về khả năng hỗ trợ của máy tính trong việc học tập.
  Hay một ví dụ khác là về Hóa học, ta có nhu cầu mô phỏng một thí nghiệm phản ứng hóa học của các chất với nhau, Máy tính có thể hỗ trợ việc này như là một phòng thí nghiệm hóa học ảo.



  Một ứng dụng cụ thể của máy tính trong học tập là phần mềm Crocodile Chemistry, tạo ra phòng thí nghiệm hóa học ảo hỗ trợ học sinh học tập môn Hóa học.

3. Mục tiêu cụ thể của Đồ án:

- Và mục tiêu của Đồ án này là xây dựng một chương trình hỗ trợ việc học tập của Học sinh, cụ thể là Hỗ trợ về Chủ đề giải toán về Tam giác phẳng (Tính toán các yếu tố của tam giác (Diện tích, Chu vi, ….) dựa vào các yếu tố có sẵn (Góc, Cạnh, Đường cao).

CHƯƠNG 2: TỔNG QUAN VỀ BIỂU DIỄN TRI THỨC

1. Mở đầu

- Để máy tính có thể suy luận từ các dữ kiện ban đầu để ra được kết quả, chúng ta phải sử dụng các kỹ thuật và công nghệ để cho máy tính có được khả năng đó. Kỹ thuật này được gọi là Biểu diễn tri thức.
- Nhưng trước tiên, chúng ta cần hiểu: Tri thức là gì ? Nó là một khái niệm rất trừu tượng. Tri thức (Knowledge) là tập hợp các thông tin, dữ kiện, mô tả có được qua trải nghiệm, nghiên cứu hoặc học tập. Trong tiếng Việt, "tri" hay "thức" điều có nghĩa là "sự hiểu biết".
  Vd: với dữ liệu ban đầu chỉ là con chuồn chuồn bay, cha ông ta, qua nghiên cứu nhiều năm, nhiều thế kỷ, đã kết luận rằng: "Chuồn chuồn bay thấp thì mưa, bay cao thì nắng, bay vừa thì râm". Từ những thông tin ban đầu, qua kinh nghiệm, nghiên cứu và tổng hợp, cha ông ta đã có được tri thức để dự báo thời tiết thông qua sự bay của con chuồn chuồn.
- Vì vậy, giữa hay khái niệm Thông tin (Information) và Tri thức (Knowledge) rất khác nhau. Để hiểu rõ hơn, ta hãy xem câu nói của nhà Bác học Karan Sing: "Chúng ta đang ngập chìm trong biển thông tin nhưng lại khát tri thức". Quả thật, Thông tin xung quanh chúng ta rất nhiều và đa dạng, nhưng tri thức thì rất ít so với lượng thông tin trên. Tập hợp rất nhiều thông tin, qua chọn lọc, quan sát, sàng lọc kỹ lưỡng, mới có được một mẫu tri thức.

2. Phân loại tri thức

- Người ta chia tri thức thành các loại như sau:
  - Tri thức Sự kiện: Là một khẳng định về Khái niệm, sự kiện nào đó trong cuộc sống: các định luật về Vật lý, hóa học. Các tiên đề….
    VD: Nước sôi ở 100 độ, Trái đất quay quanh Mặt trời, ….
  - Tri thức thủ tục: Thường dùng để điễn tả các phương pháp, các bước tiến hành để giải quyết một vấn đề: các công thức toán học, các thuật toán, thuật giải.
    VD: Công thức Hê-rông, thuật toán Euclide (Tìm Ước số chung lớn nhất), thuật giải A* (tìm đường đi trong đồ thị).
  - Tri thức mô tả: Cho biết một đối tượng, sự vật, sự việc có cấu tạo, cảm nhận, được thấy như thế nào.
    VD: Đồng (Cu) là kim loại dẻo, có màu đỏ, dẫn điện, dẫn nhiệt tốt.
  - Tri thức Heuristic: Là một dạng tri thức cảm tính. Các tri thức loại này thường có dạng ước lượng, phỏng đoán và được hình thành phần lớn nhờ vào kinh nghiệm
    VD: Xem thiên văn dự báo thời tiết.
- Và để máy tính hiểu được các tri thức, chúng ta phải có các cách biểu diễn hợp lý.

3. Biểu diễn tri thức trên máy tính

a. Logic Mệnh đề

- Là cách biểu diễn tri thức đơn giản và gần gũi. Mệnh đề là một khẳng định, một phát biểu mà giá trị của nó có thể là Đúng (True) hoặc Sai (False).
  Vd: "1+1=2". Đúng
- Có 3 phép nối cơ bản để tạo ra những mệnh đề mới từ những mệnh đề cơ sở: phép hội ($\vee$), giao ($\wedge$) và phủ định ($\neg$).
- Các cơ chế suy diễn:

- Modus Ponens:

$$\frac{A \to B \quad A}{B}$$

- Modus Tollens:

$$\frac{A \to B \quad \neg B}{\neg A}$$

- Các thuật toán liên quan đến Logic mệnh đ
  - Thuật giải Vương Hạo:

> - **Bước 1**: Phát biểu lại giả thiết và kết luận theo dạng chuẩn sau: GT1, GT2, …., GTn $\to$ KL1, KL2, …., KLn.
> - **Bước 2**: Chuyển về các GT và KL có dạng phủ định.
> - **Bước 3**: Nếu GT có phép ^ thì thay thế bằng dấu ",", nếu KL có phép $\vee$ thì thay thế bằng dấu ",".
> - **Bước 4**: Nếu GT có phép $\vee$ thì tách làm 2 dòng con, nếu KL có phép ^ thì tách làm 2 dòng con.
> - **Bước 5**: Một dòng được chứng minh nếu tồn tại chung 1 mệnh đề ở cả hai phía.
> - **Bước 6**:
>   - Nếu một dòng không còn phép nối ^ hoặc $\vee$ ở cả 2 vế và ở 2 vế không có chung một biến mệnh đề thì dòng đó kho6nh được chứng minh.
>   - Một vấn đề được chứng minh nếu tất cả các dòng dẫn xuất từ dạng chuẩn ban đầu đều được chứng minh.

  - Thuật giải Robinson:

> - **Bước 1**: Phát biểu lại giả thiết và kết luận theo dạng chuẩn sau: GT1, GT2, …., GTn $\to$ KL1, KL2, …., KLn
> - **Bước 2**: Nếu GT có phép ^ thì thay thế bằng dấu ",", nếu KL có phép $\vee$ thì thay thế bằng dấu ","
> - **Bước 3**: Biến đổi dòng chuẩn ở Bước 1 về thành danh sách mệnh đề như sau:
>
> {GT1, GT2, …., GTn, $\neg$ KL1, $\neg$ KL2, …., $\neg$ KLn}
>
> - **Bước 4**: Nếu danh sách trong Bước 2 có 2 mệnh đề đối ngẫu nhau thì bài toán đước chứng minh, Ngược lại chuyển sang Bước 4.
> - **Bước 5**: Xây dựng lại mệnh đề mới bằng cách tuyển 1 cặp mệnh đề trong danh sách mệnh đề ở Bước 2. Nếu mệnh đề mới có các biến mệnh đề đối ngẫu nhau thì các biến đó được loại bỏ.

b. <u>Logic Vị từ</u>

- Là phương pháp biểu diễn tri thức sử dụng Logic vị từ. Trong Logic vị từ, một mệnh đề được cấu tạo bởi hai thành phần: Các đối tượng tri thức và mối quan hệ giữa chúng (vị từ) dưới dạng như sau:

Vị từ (đối tượng 1, đối tượng 2,…..,đối tượng n)

Vd: Tam giác ABC có A=90 => Góc(A,90) hay Vuông(A)

x>y =>Lớn hơn(x,y)

3 là ước số chung lớn nhất của 6 => USCLN(6,3)

- Công cụ Logic vị từ đã được nghiên cứu và phát triển thành ngôn ngữ lập trình PROLOG, là một công cụ mạnh, thường dùng trong Xử lý ngôn ngữ tự nhiên.

c. <u>Luật sinh</u>

- Là một phương pháp biểu diễn tri thức Có cấu trúc. Tri thức trong cách biểu diễn này được xây dựng thành một cặp **điều kiện – hành động**: NẾU điều kiện xảy ra THÌ hành động thi hành
- Luật sinh được biểu diễn một cách tổng quát như sau:

  P1 ^ P2 ^ … ^ Pn $\rightarrow$ Q

  Vd:
  Luật: Tam giác có 2 cạnh kề bằng nhau là tam giác cân.
  Giả thiết: Tam giác ABC có a=12, b=12
  Kết luận: Hỏi tam giác ABC có cân không ?

  Ta có: value(a,12), value(b,12) => Bằng(a,b)
  Nên tamgiaccan(ABC)
- Các thành phần cơ bảnh của hệ luật dẫn:
  - Tập sự kiện (FACT)
  - Tập luật (RULES)

VD: Cho hệ luật dẫn như sau:

FACT(A, B, C, D, E, F, G, H, K)

RULES (R1: A $\rightarrow$ E,

R2: B $\rightarrow$ D,

R3: H $\rightarrow$ A,

R4: E ^ G $\rightarrow$ C,

R5: E ^ K $\rightarrow$ B,

R6: D^E^K $\rightarrow$ C,

R7: G^K^F $\rightarrow$ A)

- Các cơ chế suy diễn trên luật sinh:
  - Suy diễn tiến: Là quá trình suy luận xuất phát từ một số sự kiện ban đầu, xác định các sự kiện có thể được sinh ra từ các sự kiện này.
  - Suy diễn lùi: Là quá trình suy luận ngược xuất phát từ một số sự kiện ban đầu, ta tìm kiếm những sự kiện đã sinh ra những sự kiện này.

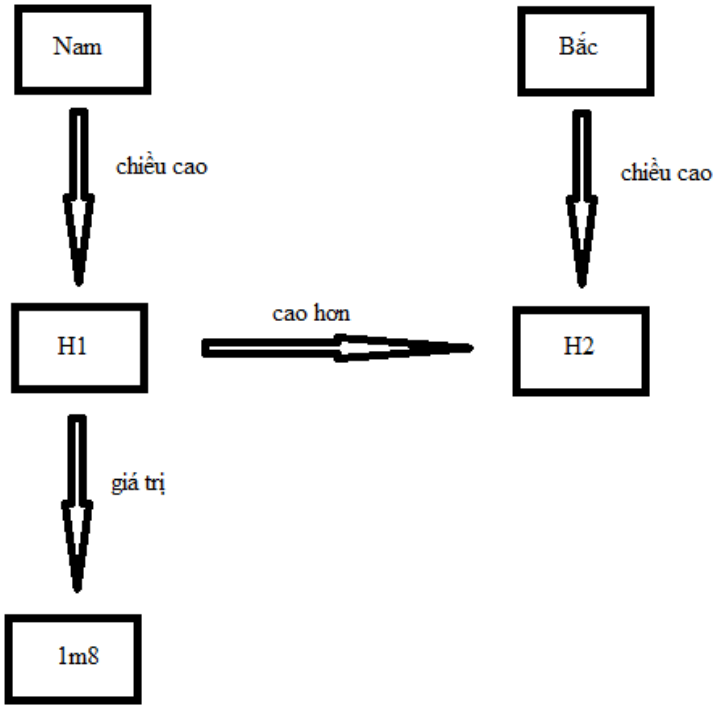
(Chương 3 sẽ nói rõ hơn về 2 cơ chế suy diễn này)

- Ưu điểm và nhược điểm của biểu diễn tri thức bằng luật sinh:
  - Ưu điểm:
    - Các luật rất dễ hiểu nên có thể dễ dàng trao đổi với người dùng.
    - Việc hiệu chỉnh và bảo trì dễ dàng.
    - Có thể cải tiến để tích hợp các luật mờ.
    - Các luật thường ít phụ thuộc vào nhau.

- Nhược điểm
  - Các tri thức phức tạp sẽ đòi hỏi nhiều luật (hàng nghìn đến hàng triệu luật). Điều này dẫn đến chi phí thực hiện luật.
  - Cơ sở tri thức luật sinh lớn sẽ làm giới hạn khả năng tìm kiếm của chương trình.

d. <u>Mạng ngữ nghĩa:</u>

- Là một phương pháp biểu diễn tri thức có cấu trúc, trong đó tri thức sẽ được biểu diễn dưới dạng đồ thị. Các đỉnh của đồ thị là các đối tượng (Khái niệm) và các cung cho biết mối quan hệ giữa các đối tượng.
  Vd: Biểu diễn tri thức sau bằng Mạng ngữ nghĩa: "Nam cao 1m8 và cao hơn Bắc".



- Do mạng ngữ nghĩa là một loại đồ thị nên nó thừa hưởng tất cả các thế mạnh của công cụ này. Ta có thể dùng những thuật toán của đồ thị cho mạng ngữ nghĩa như: thuật toán tìm liên thông, thuật toán tìm đường đi ngắn nhất, ….
- Mạng ngữ nghĩa có tính kế thừa.
- Ưu điểm và nhược điểm của mạng ngữ nghĩa:
  - Ưu điểm:
    - Mạng ngữ nghĩa rất linh động. Ta có thể dễ dàng thêm vào mạng các đỉnh hoặc cung mới để bổ sung các tri thức cần thiết.
    - Mạng ngữ nghĩa có tính trực quan cao.
    - Mạng ngữ nghĩa hoạt động khá tự nhiên theo cách thức con người ghi nhận thông tin.
  - Nhược điểm:
    - Chưa có một tiêu chuẩn nào qui định giới hạn giữa các đỉnh và cung trong mạng.
    - Tính kế thừa của mạng ngữ nghĩa sẽ dẫn đến mâu thuẫn trong tri thức.

e. *Frame:*

- Là một cấu trúc dữ liệu chứa đựng tất cả những tri thức liên quan đến một đối tượng cũ thể nào đó. Frame có liên hệ chặt chẽ đến Hướng đối tưo75ng (Frame chính là nguồn gốc của Hướng đối tượng).
- Cấu trúc của Frame:
    - Slot: Chứa các đối tượng được đặc tả
    - Facet: Chứa thông tin đặc tả của đối tượng. Facet có nhiều loại khác nhau như: Value(giá trị), Default(giá trị mặc định), Range(miền giá trị), If added, If needed…

  Vd: Frame Hình chữ nhật

  a: cạnh dài

  b: cạnh ngắn

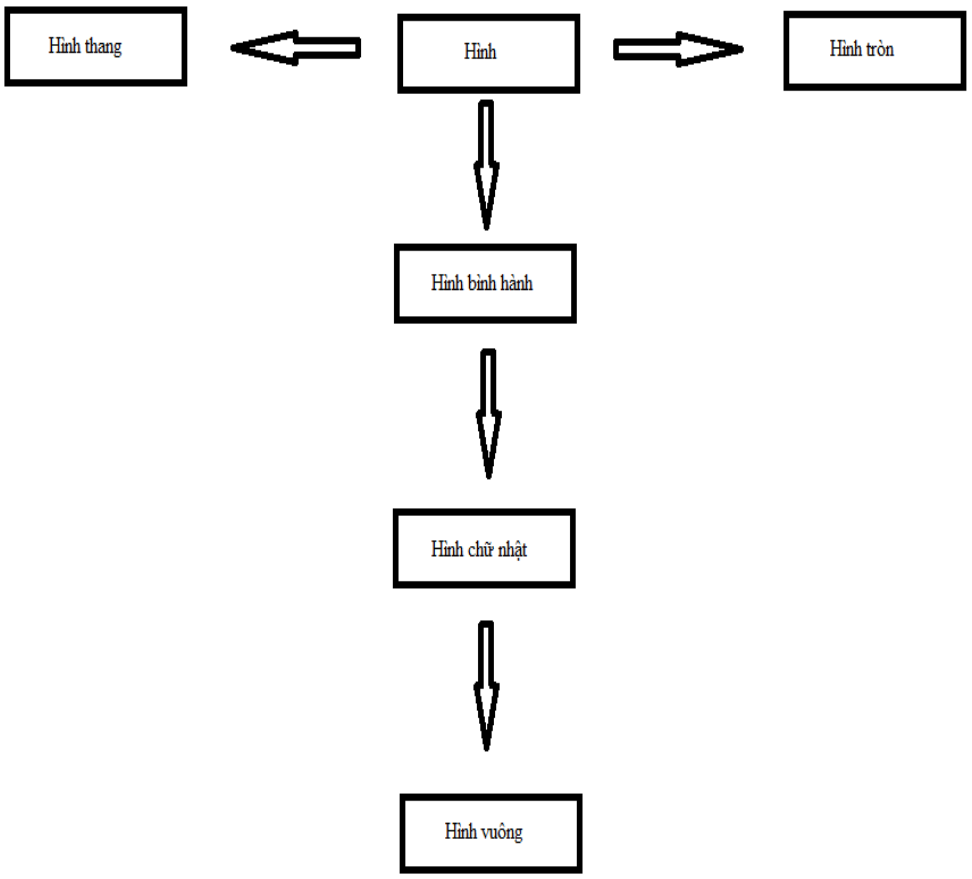
  s: diện tích

  p: chu vi

  s = a * b

  p = (a + b) * 2

- Frame có đặc tính phân cấp. Do đó, nó cho phép sự kế thừa tính chất giữa các Frame.



CHƯƠNG 3: CƠ SỞ THUẬT TOÁN

Để máy tính có thể hiểu và vận dụng các tri thức đã biết vào việc giải một bài toán cụ thể, chúng ta phải áp dụng các phương pháp, cơ chế thích hợp để máy tính có thể làm được việc đó. Phương pháp đó được gọi là suy luận (hay suy diễn). Trong đồ án này, chúng ta sẽ dùng cơ chế suy diễn tiến áp dụng cho tập tri thức được biểu diễn bằng hệ luật dẫn.

1. Cơ chế suy diễn tiến

- Là quá trình suy luận xuất phát từ một số sự kiện ban đầu, xác định các sự kiện có thể được sinh ra từ các sự kiện này.

VD: Cho hệ cơ sở tri thức sau:

R = {
R1: A $\rightarrow$ E
R2: B $\rightarrow$ D
R3: H $\rightarrow$ A
R4: E ^ G $\rightarrow$ C
R5: E ^ K $\rightarrow$ B
R6: D ^ E ^ K $\rightarrow$ C
R7: G ^ K ^ F $\rightarrow$ A }

Hỏi H , K $\rightarrow$ C ?

BÀI LÀM

Ta có:

R3: H $\rightarrow$ A	{A,H,K}=Q
R1: A $\rightarrow$ E	{A,E,H,K}=Q
R5: E ^ K $\rightarrow$ B	{A,B,E,H,K}=Q
R2: B $\rightarrow$ D	{A,B,D,E,H,K}=Q
R6: D ^ E ^ K $\rightarrow$ C	{A,B,C,D,E,H,K}=Q

Ta có C $\subset$ Q => H , K $\rightarrow$ C

- Thuật toán suy diễn tiến Tổng quát:

Function FOL-FC-ASK (KB,a)
Input: **KB**: cơ sở tri thức
a: câu truy vấn nguyên tố
biến cục bộ new: câu mới được suy dẫn ra từ mỗi lần lặp
repeat until new là rỗng
new {} $\leftarrow$ {}
for each câu r in KB do
(p1 ^ p2 ^ … ^ pn => q) STANDARDIZE-APART(r)
For each θ sao cho SUBST(θ,p1^…^pn) = SUBST(θ,p1'^…^pn') với p1'^…^pn' trong **KB**
q $\leftarrow$ SUBST(θ,q)
if q' không là 1 câu được đổi tên từ các câu đã có trong KB hay new **then do**
thêm q' vào new
$\phi \leftarrow$ Đồng nhất (q',a)
If ϕ không *thất bại* **then retur**n ϕ
Thêm new vào KB
Return false

Thuật toán suy diễn tiến đối với Logic bậc nhất là đầy đủ vì mỗi suy diễn đều là sự áp dung của tam đoạn luận tổng quát. Thuật toán này cũng đầy đủ với các cơ sở tri thức mệnh đề xác định.

Tuy nhiên, khi cơ sở tri thức lớn, thuật toán này phải vét cạn hết các trạng thái có thể sinh ra từ trạng thái đầu nên chi phí thực hiện lớn.

2. Cơ chế suy diễn lùi

- Là quá trình suy luận ngược xuất phát từ một số sự kiện ban đầu, ta tìm kiếm những sự kiện đã sinh ra những sự kiện này.

VD: Cho hệ cơ sở tri thức sau:

R = {

R1: A $\rightarrow$ E

R2: B $\rightarrow$D

R3: H $\rightarrow$ A

R4: E ^ G$\rightarrow$C

R5: E ^ K $\rightarrow$B

R6: D ^ E ^ K $\rightarrow$C

R7: G ^ K ^ F $\rightarrow$A }

Hỏi C,D $\rightarrow$H,K ?

BÀI LÀM

R6: D ^ E ^ K $\rightarrow$C {C,D,E,K} = Q

R1: A $\rightarrow$ E {C,D,E,K,A} = Q

R3: H $\rightarrow$ A {C,D,E,K,A,H} = Q

Ta có H,K $\subset$Q => C, D $\rightarrow$ H, K

- Thuật toán suy diễn lùi Tổng quát:

Function FOL-BC-ASK(KB, truy_vấn, θ)
Input: KB: cơ sở tri thức
truy_vấn: một danh sách truy vấn dạng nối liền
θ: phép thể hiện tại (ban đầu là 1 phép thế rỗng)
Biến cục bộ: trả_lời: một tập phép thế (khởi tạo là rỗng)

If (truy_vấn rỗng) **then return** {θ}
q' $\leftarrow$ SUBST(θ,Câu đầu(truy_vấn))
for each câu r in KB **trong đó** STANDADIZE_APART(r) = (p1 ^ p2 ^ … ^ pn => q) và θ' $\leftarrow$ Đồng nhất (q,q') thành công
truy_vấn_mới $\leftarrow$[p1,…pn[Phần_Còn_Lại(truy_vấn)]
trả lời $\leftarrow$ FOL-BC-ASK(KB,truy_vấn_mới, Tổng hợp (θ,θ') $\cup$ trả_lời
return trả_lời

Suy diễn lùi là một thuật toán tìm kiếm theo chiều sâu. Điều này có nghĩa là nó cần không gian tuyến tính với kích thước của phép chứng minh (không gian cần cho tích lũy đáp án).

3. Tối ưu tập luật

a. *Rút gọn bên phải*

- Quy tắc rút gọn: có thể loại bỏ những sự kiện bên vế phải nếu những sự kiện đó đã xuất hiện bên vế trái. Nếu sau khi rút gọn mà luật trở thành rỗng thì đó là luật hiển nhiên. Ta có thể loại các luật hiển nhiên ra khỏi tri thức.
  VD:
  Luật A ^ B $\rightarrow$ A hiển nhiên đúng.
  Luật A ^ B $\rightarrow$ A ^ C tương đương A ^ B $\rightarrow$ C

b. *Rút gọn bên trái*

- Quy tắc rút gọn tương tự như Rút gọn bên vế phải.
  VD:
  R1: A, B $\rightarrow$C
  R2: A $\rightarrow$ X
  Ta thấy A, B $\rightarrow$C có thể thay thế bằng A $\rightarrow$ C mà không làm ảnh hưởng đến kết luận trong mọi trường hợp. Sự kiên B trong R1 là dư thừa và có thể loại bỏ ra khỏi R1.

c. *Luật thừa*

- Một luật dẫn A $\rightarrow$ B gọi là luật thừa nếu có thể suy ra luật này từ những luật còn lại.
  VD:
  R1: A $\rightarrow$ B
  R2: B $\rightarrow$ C
  R3: A $\rightarrow$ C
  Luật R3 là luật thừa vì có thể suy ra từ R1 và R2: A $\rightarrow$ B ^ B $\rightarrow$ C = A $\rightarrow$ C

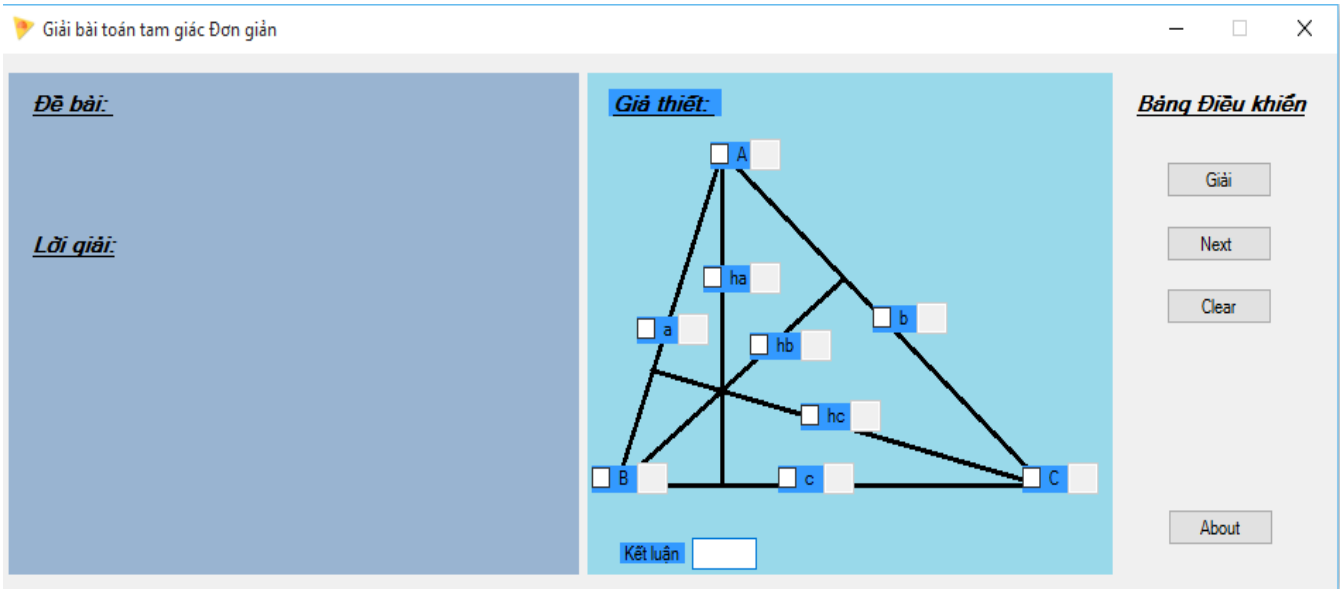
d. *Tối ưu tập luật dẫn*

- *Bước 1*: Rút gọn vế phải.
- *Bước 2*: Phân rã các luật.
- *Bước 3*: Loại bỏ luật thừa.
- *Bước 4*: Rút gọn vế trái.

CHƯƠNG 4: CẤU TRÚC CHƯƠNG TRÌNH

1. Giao diện chính của chương trình

- Chương trình được xây dựng bằng Windows Form trên nền DotNet 3.5.



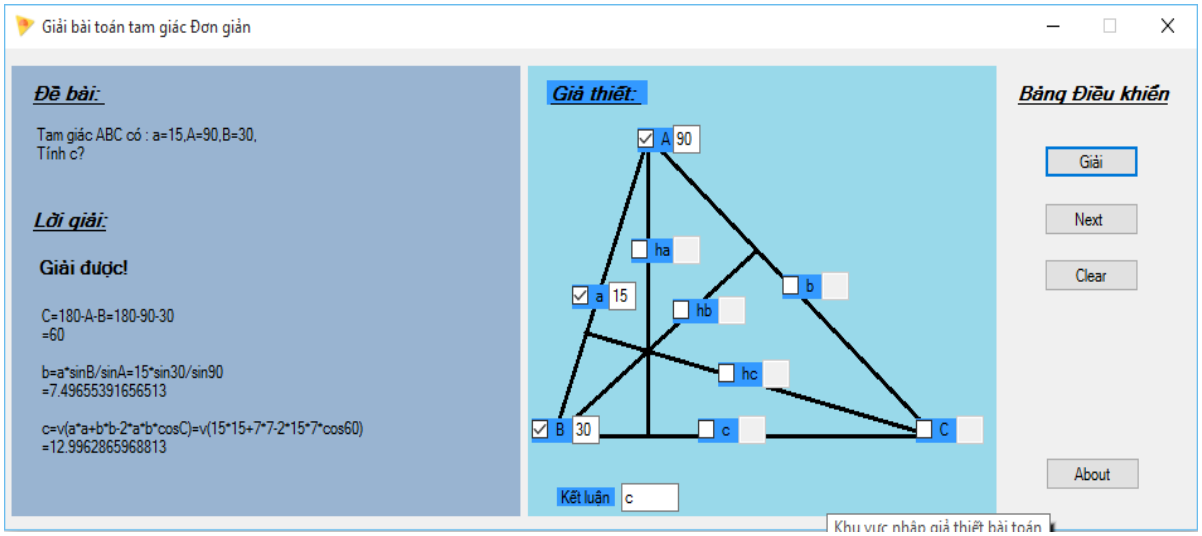
- Chức năng các button:

Button	Chức năng
Giải	Giải bài toán (khi có đủ dữ kiện)
Next	Giữ lại giả thiết, nhập kết luận mới cho bài toán
Clear	Xóa hết giả thiết và kết luận
About	Thông tin về Chương trình

- Phần giả thiết có các yếu tố cần thiết của một tam giác (góc, cạnh, đường cao) và cho người dùng nhập vào các yếu tố theo đề bài cùng với giá trị của nó.
- Phần kết luận bao gồm đề bài (suy luận ra từ các giả thiết do người dùng nhập) và phần kết luận xem bài toán có thể giải được không với những giả thiết đó. Nếu giải được thì in ra lời giải (Đáp số đầy đủ).

VD: Tam giác ABC có góc A=90^o, góc B=30^o, cạnh a=15. Tính cạnh b ?

Chương trình giải ra cạnh b với bài giải như trong hình:



2. Mã nguồn chương trình

a. Lớp Rule.cs

- Đây là lớp định nghĩa các luật trong hệ luật dẫn bao gồm: giả thiết (vế trái), kết luận (vế phải), biểu thức tính toán, biểu thức thế số và Kết quả tính toán

Mã nguồn lớp:

```
class Rules
   {
      String theory="";              //vế trái
      String conclude="";            //vế phải
      String express = "";           //biểu thức (công thức tính)
      String result = "";            //biểu thức thế số
      String Value = "";             //giá trị của biểu thức
      public void inputA(char c)
      {
         theory += c;
      }
      public void inputB(char c)
      {
         conclude += c;
      }
      public void inputC(char c)
      {
         express += c;
      }
      public void inputD(char c)
      {
         result += c;
      }
      public void inputE(char c)
      {
         Value += Value;
      }
      public String returnA()
      {
         return theory;
```

```
        }
        public String returnB()
        {
           return conclude;
        }
        public String returnC()
        {
           return express;
        }
        public String returnD()
        {
           return result;
        }
        public String returnE()
        {
           return Value;
        }
        public void setA(String a)
        {
           theory = a;
        }
        public void setB(String b)
        {
           conclude = b;
        }
        public void setC(String b)
        {
           express = b;
        }
        public void setD(String b)
        {
           result = b;
        }
        public void setE(String b)
        {
           Value = b;
        }
    }
}
```

b. *Lớp ForwardReasoning.cs*

- Là lớp thực hiện thuật toán suy diễn tiến.

Mã nguồn lớp:

```
class ForwardReasoning
   {
      String src;              //trạng thái đầu
      String dest;             //trạng thái đích
      List<Rules>lRules=new List<Rules>();
      List<Rules> lResult = new List<Rules>();
      public ForwardReasoning(String src,String dest,List<Rules>lR)
      {
```

```
            this.src = src;
            this.dest = dest;
            this.lRules = lR;
        }
-------------------------------------------------------------------------
        public bool isInLeft(String s,List<Rules>lR)
        {
            for(int i=0;i<lR.Count;i++)
                if(isExist(s,lR[i].returnA())==true)
                    return true;
            return false;
        }
-------------------------------------------------------------------------
//tối ưu hóa trạng thái.
        public void OptimizeResult()
        {
            bool k = true;
            do
            {
                k = true;
                for (int i = 0; i < lResult.Count; i++)
                {
                    if (isInLeft(lResult[i].returnB(), lResult) == false)  //bỏ đi các trạng thái chỉ tồn tại
bên phải
                        if (lResult[i].returnB() != dest)
                        {
                            lResult.RemoveAt(i);
                            k = false;
                            break;
                        }
                    if(isExist(lResult[i].returnB(),src)) //bỏ đi các trạng thái bên vế phải đã có trong
tập dest (tập kết luận)
                    {
                        lResult.RemoveAt(i);
                        k = false;
                        break;
                    }
                }
                for (int i = 0; i < lResult.Count - 1; i++)  //bỏ đ các trạng thái có vế phải trùng nhau
                    for (int j = i + 1; j < lResult.Count; j++)
                        if (lResult[i].returnB() == lResult[j].returnB())
                        {
                            lResult.RemoveAt(j);
                            k = false;
                            break;
                        }
            }
            while (k == false);
        }

-------------------------------------------------------------
```

```
    public bool isInString(char a,String b)
    {
        for (int i = 0; i < b.Length; i++)
            if (a == b[i])
                return true;
        return false;
    }
-------------------------------------------------------------
    public bool isExist(String a,String b)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)
            if (isInString(a[i], b) == true)
                    count++;
        if (count == a.Length)
            return true;
        else
            return false;
    }
-------------------------------------------------------------
    public int findInString(char a, String b)
    {
        for (int i = 0; i < b.Length; i++)
            if (a == b[i])
                return i;
        return -1;
    }
-------------------------------------------------------------
```

//chuẩn hóa lại luật (thay thế các ký tự "ha, hb, hc" trong tập tri thức thành "x, y, z) cho dễ xử lý.

```
    public void normalizeResult()
    {
        for (int i = 0; i < lResult.Count; i++)
        {
            if (lResult[i].returnB() == "x")
                lResult[i].setB("ha");
            if (lResult[i].returnB() == "y")
                lResult[i].setB("hb");
            if (lResult[i].returnB() == "z")
                lResult[i].setB("hc");
        }
        for (int i = 0; i < lResult.Count; i++)
        {
            if (isInString('x',lResult[i].returnA()))
            {
                String tmp = lResult[i].returnA();
                tmp = tmp.Insert(findInString('x', tmp), "ha");
                tmp = tmp.Remove(findInString('x', tmp));
                lResult[i].setA(tmp);
            }
            if (isInString('y', lResult[i].returnA()))
```

```
            {
               String tmp = lResult[i].returnA();
               tmp = tmp.Insert(findInString('y', tmp), "hb");
               tmp=tmp.Remove(findInString('y', tmp));
               lResult[i].setA(tmp);
            }
            if (isInString('z', lResult[i].returnA()))
            {
               String tmp = lResult[i].returnA();
               tmp = tmp.Insert(findInString('z', tmp), "hc");
               tmp = tmp.Remove(findInString('z', tmp));
               lResult[i].setA(tmp);
            }
         }
      }
-----------------------------------------------------
      public String AddtoTemp(String b, String s)
      {
         String tmp=b;
         for (int i = 0; i < s.Length; i++)
            if (isInString(s[i], tmp))
               continue;
            else
               tmp += s[i];
         return tmp;
      }
-----------------------------------------------------
//hàm thực thi suy diễn tiến.
      public void Processing()
      {
         String temp = "";
         temp = temp + src;
         bool k = false;
         do
         {
            k = true;
               for (int i = 0; i < lRules.Count; i++)
                  if (isExist(lRules[i].returnA(), temp) == true)
                  {
                     if (isExist(dest, temp) == true || lRules.Count <= 0)
                     {
                        k = true;
                        break;
                     }
                     temp = AddtoTemp(temp, lRules[i].returnB());
                     lResult.Add(lRules[i]);
                     lRules.RemoveAt(i);
                     k = false;
                  }
         }
         while (k == false);
```

```
            if (isExist(dest, temp) == false)
                lResult.Clear();
        }
-------------------------------------------------------------
//hàm trả về kết quả suy diễn tiến.
        public List<Rules> returnResult()
        {
            OptimizeResult();
            normalizeResult();
            return this.lResult;
        }
    }
```

c. *Lớp Calculate.cs*

- Lớp thực hiện công việc tính toán biểu thức dựa trên Ký pháp Ba Lan.

Mã nguồn lớp:

```
class Calculate
    {
        double[] c;
        int dem = 0;
        String s = "";
        Double Result = 0;
        public Calculate(String st)
        {
            s = st;
        }
-----------------------------------------------------------
        public static bool IsEmpty(Stack c)
        {
            if (c.Count == 0)
                return true;
            else
                return false;
        }
-------------------------------------------------------------
//độ ưu tiên toán tử.
        public int prio(char a)
        {
            if (a == '*' || a == '/')
                return 2;
            if (a == 's' || a == 'c' || a == 'a' || a == 'r' || a == 'v')
                return 3;
            if (a == '+' || a == '-')
                return 1;
            return 0;
        }
-------------------------------------------------------------
//chuyển thành biểu thức tiền tố.
        public void chuyenkyphap()
        {
            Stack a = new Stack();
```

```
c = new double[s.Length];
for (int i = 0; i < s.Length; i++)
{
   if (s[i] >= '0' && s[i] <= '9')
   {
      int temp = 0;
      char ch;
      ch = s[i];
      while (ch >= '0' && ch <= '9')
      {
         temp = temp * 10 + ch - 48;

         if (i == s.Length-1)
            break;
         else
         {
            i++;
            ch = s[i];
         }
      }
      c[dem] = Convert.ToInt32(temp);
      dem++;
   }
   if (s[i] == '+' || s[i] == '-' || s[i] == '*' || s[i] == '/' || s[i] == 'v')
   {
      if (IsEmpty(a) == true)
         a.Push((char)(s[i]));
      else
         if (prio(s[i]) <= prio((char)(a.Peek())))
         {
            c[dem] = (char)(a.Pop());
            dem++;
            a.Push((char)(s[i]));
         }
         else
            a.Push((char)(s[i]));
      continue;
   }
   if (s[i] == '(')
   {
      a.Push(s[i]);
   }
   if (s[i] == ')')
   {
      char ch;
      do
      {
         ch = Convert.ToChar(a.Pop());
         if (ch != '(')
         {
            c[dem] = ch;
```

```
                dem++;
            }
        }
        while (ch != '(');
    }
    if(s[i]=='s')
    {
        if (IsEmpty(a) == true)
            a.Push((char)(s[i]));
        else
            if (prio(s[i]) <= prio((char)(a.Peek())))
            {
                c[dem] = (char)(a.Pop());
                dem++;
                a.Push((char)(s[i]));
            }
            else
                a.Push((char)(s[i]));
        i += 2;
    }
    if (s[i] == 'c')
    {
        if (IsEmpty(a) == true)
            a.Push((char)(s[i]));
        else
            if (prio(s[i]) <= prio((char)(a.Peek())))
            {
                c[dem] = (char)(a.Pop());
                dem++;
                a.Push((char)(s[i]));
            }
            else
                a.Push((char)(s[i]));
        i += 2;
    }
    if(s[i]=='a')
    {
        if(s[i+3]=='s')
        {
            if (IsEmpty(a) == true)
                a.Push('a');
            else
                if (prio(s[i]) <= prio((char)(a.Peek())))
                {
                    c[dem] = (char)(a.Pop());
                    dem++;
                    a.Push((char)('a'));
                }
                else
                    a.Push((char)('a'));
            i += 5;
```

```
                continue;
            }
            if(s[i+3]=='c')
            {
                if (IsEmpty(a) == true)
                    a.Push(('r'));
                else
                    if (prio(s[i]) <= prio((char)(a.Peek())))
                    {
                        c[dem] = (char)(a.Pop());
                        dem++;
                        a.Push('r');
                    }
                    else
                        a.Push(('r'));
                i += 5;
                continue;
            }

        }
    }
    while (IsEmpty(a) != true)
    {
        c[dem] = Convert.ToChar(a.Pop());
        dem++;
    }
}
//tính toán dựa trên biểu thức tiền tố.
    public void ketqua()
    {
        Stack b = new Stack();
        for (int i = 0; i < dem; i++)
        {
            switch ((int)c[i])
            {
                case '+':
                    {
                        double a, k;
                        a = (double)(b.Pop());
                        k = (double)(b.Pop());
                        b.Push(a + k);
                        break;
                    }
                case '-':
                    {
                        double a, k;
                        a = (double)(b.Pop());
                        k = (double)(b.Pop());
                        b.Push(k - a);
                        break;
                    }
```

```
            case '*':
               {
                  double a, k;
                  a = (double)(b.Pop());
                  k = (double)(b.Pop());
                  b.Push(a * k);
                  break;
               }
            case '/':
               {
                  double a, k;
                  a = (double)(b.Pop());
                  k = (double)(b.Pop());
                  b.Push(k / a);
                  break;
               }
            case 's':
               {
                  double a;
                  a = (double)(b.Pop());
                  b.Push(Math.Sin((a / 180) * 3.14));
                  break;
               }
            case 'c':
               {
                  double a;
                  a = (double)(b.Pop());
                  b.Push(Math.Cos((a / 180) * 3.14));
                  break;
               }
            case 'a':
               {
                  double a;
                  a = (double)(b.Pop());
                  b.Push((Math.Asin(a)/3.14)*180);
                  break;
               }
            case 'r':
               {
                  double a;
                  a = (double)(b.Pop());
                  b.Push((Math.Acos(a) / 3.14) * 180);
                  break;
               }
            case 'v':
               {
                  double a;
                  a = (double)(b.Pop());
                  b.Push(Math.Sqrt(a));
                  break;
               }
```

```
                default:
                    {
                        b.Push((double)c[i]);
                        break;
                    }
                }
            }
            Result = Convert.ToDouble(b.Pop());
        }
-------------------------------------------------------------
//hảm thực thi
        public void Processing()
        {
            chuyenkyphap();
            ketqua();
        }
-------------------------------------------------------------
//trả về kết quả
        public double returnKq()
        {
            return this.Result;
        }
    }
```

d. *Lớp Process.cs*

- Lớp thực hiện tổng hợp các công việc (đọc file, tính toán, suy diễn, đưa ra kết quả…) và cho kết quả chính thức.

Mã nguồn lớp:

```
class Process
    {
        String src="";
        String dest="";
        List<Rules> lRules;
        List<Rules> lResult;
        ForwardReasoning f_Reason;
        public Process(String s,String d)
        {
            this.src = s;
            this.dest = d;
        }
-------------------------------------------------------------
//đọc cơ sở tri thức từ file.
        public void ReadRulesFromFile()
        {
            lRules = new List<Rules>();
           try
            {
               StreamReader inRule = File.OpenText("TriangleKnowldge.txt");
                while(!inRule.EndOfStream)
                {
                    String buffer="";
```

```
                Rules r = new Rules();
                buffer=inRule.ReadLine();
                r = GetRules(buffer);
                lRules.Add(GetRules(buffer));
            }
        }
        catch(FileNotFoundException e)
        {
            MessageBox.Show("File Not Found!");
        }
    }
-----------------------------------------------------
//phân tích cơ sở tri thức ra các luật.
    public Rules GetRules(String s)
    {
        Rules r=new Rules();
        for (int i = 0; i < s.Length; i++)
        {
            while(s[i]!='-')
            {
                if (s[i] == '^')
                    i++;
                else
                    if(s[i]=='h')
                    {
                        String tmp = "h" + s[i + 1];
                        if(tmp=="ha")
                            r.inputA('x');
                        if (tmp == "hb")
                            r.inputA('y');
                        if (tmp == "hc")
                            r.inputA('z');
                        i += 2;
                    }
                else
                {
                    r.inputA(s[i]);
                    i++;
                }
            }
            i+=2;
            while(s[i]!='=')
            {
                if (s[i] == '^')
                    i++;
                else
                    if (s[i] == 'h')
                    {
                        String tmp = "h" + s[i + 1];
                        if (tmp == "ha")
                            r.inputB('x');
```

```
                    if (tmp == "hb")
                        r.inputB('y');
                    if (tmp == "hc")
                        r.inputB('z');
                    i += 2;
                }
                else
                {
                    r.inputB(s[i]);
                    i++;
                }
            }
            i++;
            while(i<s.Length)
            {
                r.inputC(s[i]);
                i++;
            }
        }
        return r;
    }
```

//Thế số vào biểu thức tính toán

```
    public String ReplaceNumber(String tmp,String a, String b, String c, String A, String B,
String C, String ha, String hb,String hc)
    {
        String bt = "";
        for (int j = 0; j < tmp.Length; j++)
        {
            if ((tmp[j] >= '0' && tmp[j] <= '9'))
                bt += tmp[j];
            else
                if (tmp[j] == '+' || tmp[j] == '-' || tmp[j] == '*' || tmp[j] == '/' || tmp[j] == '(' || tmp[j] ==
')')
                    bt += tmp[j];
                else
                {
                    if (tmp[j] == 'v')
                        bt += "v";
                    if (tmp[j] == 'h')
                    {
                        if (tmp[j + 1] == 'a')
                        {
                            bt += ha;
                            j += 1;
                        }
                        if (tmp[j + 1] == 'b')
                        {
                            bt += hb;
                            j += 1;
                        }
```

```
                if (tmp[j + 1] == 'c')
                {
                    bt += hc;
                    j += 1;
                }

            }
            else
            {
                if (tmp[j] == 'a')
                    if (j < tmp.Length - 1)
                        if (tmp[j + 1] == 'r')
                        {
                            bt += "arc";
                            j += 3;
                        }
                        else
                            bt += a;
                    else
                        bt += a;
                if (tmp[j] == 'b')
                    bt += b;
                if (tmp[j] == 'c')
                    if (j < tmp.Length - 1)
                        if (tmp[j + 1] == 'o')
                        {
                            bt += "cos";
                            j += 2;
                        }
                        else
                            bt += c;
                    else
                        bt += c;
                if (tmp[j] == 'A')
                    bt += A;
                if (tmp[j] == 'B')
                    bt += B;
                if (tmp[j] == 'C')
                    bt += C;
                if (tmp[j] == 's')
                {
                    if (tmp[j + 1] == 'i')
                    {
                        bt += "sin";
                        j += 2;
                    }
                }
            }
        }
    }
    return bt;
```

```
    }
    public void Processing()
    {
        ReadRulesFromFile();
        f_Reason = new ForwardReasoning(src, dest, lRules);
        f_Reason.Processing();
        lResult = f_Reason.returnResult();
    }
-------------------------------------------------------------------------
//thực thi việc tính toán giá trị.
    public void Computing(String va,String vb,String vc,String vA,String vB,String vC,String
vha,String vhb,String vhc)
    {
        String a, b, c, A, B, C, ha, hb, hc;
        a = va; b = vb; c = vc;
        A = vA; B = vB; C = vC;
        ha = vha; hb = vhb; hc = vhc;
        Processing();
        for (int i = 0; i < lResult.Count; i++)
        {
            String tmp = lResult[i].returnC();
            String bt="";
            bt = ReplaceNumber(tmp, a, b, c, A, B, C, ha, hb, hc);
            lResult[i].setD(bt);

            Calculate cal = new Calculate(bt);

            cal.Processing();
            lResult[i].setE(Convert.ToString(cal.returnKq()));
            if (lResult[i].returnB() == "a")
                a = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "b")
                b = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "c")
                c = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "A")
                A = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "B")
                B = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "C")
                C = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "ha")
                ha = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "hb")
                hb = Convert.ToString(Math.Round(cal.returnKq()));
            if (lResult[i].returnB() == "hc")
                hc = Convert.ToString(Math.Round(cal.returnKq()));
        }
    }
-------------------------------------------------------------
//trả về giá trị.
```

```
    public String getResult()
    {
        String temp = "";
        if (lResult.Count > 0)
        {
            for (int i = 0; i < lResult.Count; i++)
                temp += lResult[i].returnB() + "=" + lResult[i].returnC() + "=" + lResult[i].returnD()
+"\n"+ "=" + lResult[i].returnE() + "\n\n";
        }
        return temp;
    }
  }
```

3. Đánh giá về Chương trình

- Chương trình đã giải được một số bài toán tương đối đơn giản về Tam giác dựa trên tập cơ sở tri thức được biểu diễn bằng hệ luật dẫn.
- Tuy nhiên, chương trình vẫn còn một số hạn chế. Chẳng hạn như khi ta nhập vào 1 tam giác vuông với 2 cạnh kề a và b, thì chương trình chưa thể nhận biết được đó là một tam giác vuông, do đó, khi muốn tìm cạnh huyền c, chương trình phải dùng công thức $\mathbf{c^2 = a^2 + b^2 – 2*a*b*cosC}$, trong khi ta chỉ đơn giản dùng định lý Pythagore để tính c theo công thức $\mathbf{c^2 = a^2 + b^2}$. Hay khi tính diện tích tam giác vuông, khi ta nhập vào 2 cạnh kề a và b thì chương trình không thể tính diện tích (do không có đường cao ha, hb để tính theo công thức $\mathbf{S} = \frac{1}{2}\mathbf{a*ha}$ hay $\mathbf{S} = \frac{1}{2}\mathbf{b*hb}$), trong khi đó, ta dễ dàng tính được diện tích của tam giác vuông theo công thức $\mathbf{S} = \frac{1}{2}\mathbf{a*b}$.
- Trên là một số hạn chế cơ bản của chương trình. Nhìn chung, chương trình đã đáp ứng được một số yêu cầu cơ bản của Đồ án này, đó là giải được một bài toán tam giác đơn giản từ những dữ kiện bạn đầu do người dùng nhập vào. Chương trình sẽ được cải tiến để dần hoàn thiện và tốt hơn.

TÀI LIỆU THAM KHẢO

1. Hoảng Kiếm, Đinh Nguyễn Anh Dũng – Giáo trình *Nhập môn Trí tuệ nhân tạo*, Trường Đại học Công nghệ Thông tin – ĐHQG.TPHCM, Nhà Xuất bản Đại học Quốc gia TPHCM – 2011
2. Lê Hoài Bắc, Tô Hoài Việt – Giáo trình *Cơ sở Trí tuệ nhân tạo*, Trường Đại học Khoa học Tự nhiên – ĐHQG.TPHCM, Nhà Xuất bản Khoa học và Kỹ Thuật – 2014
3. Cao Hoàng Trụ, Giáo trình *Trí tuệ nhân tạo = Thông minh + Giải thuật*, Trường Đại học Bách Khoa – ĐHQG.TPHCM, Nhà Xuất bản Đại học Quốc gia TPHCM – 2014
4. Đỗ Văn Nhơn, Giáo trình *Toán rời rạc*, Trường Đại học Công nghệ Thông tin – ĐHQG.TPHCM, Nhà Xuất bản Đại học Quốc gia TPHCM – 2014
5. Website: https://bienuit.wordpress.com/tag/**suy-dien-tien**/
6. Website: https://**msdn**.microsoft.com/
7. Website: https://vi.wikipedia.org/wiki/K%C3%AD_ph%C3%A1p_Ba_Lan