ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA TOÁN - CƠ - TIN HỌC

PHƯƠNG PHÁP NGHIÊN CỨU KHOA HỌC

Mã lớp học phần: MAT2315 1

Giảng viên: TS. Nguyễn Ngọc Phan

Ứng dụng Logic mờ trong tự động hóa

Nguyễn Long Vũ - 21000714

Mai Thanh Tùng - 21000711

Tạ Quang Tùng - 21000712

Hà Nội, 2023

ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA TOÁN - CƠ - TIN HỌC

PHƯƠNG PHÁP NGHIÊN CỨU KHOA HỌC

Mã lớp học phần: MAT2315 1

Giảng viên: TS. Nguyễn Ngọc Phan

Ứng dụng Logic mờ trong tự động hóa

Nguyễn Long Vũ - 21000714

Mai Thanh Tùng - 21000711

Tạ Quang Tùng - 21000712

Hà Nội, 2023

NHÓM 6

Thành viên 1:

Họ và tên	Nguyễn Long vũ
Mã sinh viên	21000714
Lớp	K66A2 - Toán tin
Email	nguyenlongvu_t66@hus.edu.vn

Thành viên 2:

Họ và tên	Mai Thanh Tùng
Mã sinh viên	21000711
Lớp	K66A2 - Toán tin
Email	maithanhtung_t66@hus.edu.vn

Thành viên 3:

Họ và tên	Tạ Quang Tùng
Mã sinh viên	21000712
Lớp	K66A2 - Toán tin
Email	taquangtung_t66@hus.edu.vn

Mục lục

Lời nói đầu

Trong cuộc sống hàng ngày, chúng ta thường đối diện với những thông tin, các vấn đề mơ hồ trong thế giới thực, ví dụ, một câu hỏi như "Ngày hôm nay có lạnh không?" có thể nhận được các câu trả lời khác nhau như "Có", "Không", "Hơi lạnh", "Rất lạnh", tùy thuộc vào cảm nhận cá nhân và trạng thái cảm xúc của mỗi người. Logic mờ cho phép chúng ta xử lý các câu trả lời này một cách linh hoạt hơn so với logic nhị phân truyền thống.

Tập mờ và logic mờ dựa trên suy luận của con người về các thông tin "không chính xác" hoặc "không đầy đủ" về hệ thống để hiểu biết và điều khiển hệ thống một cách chính xác. Điều khiển mờ chính là bắt chước cách xử lý thông tin và điều khiển của con người đối với các đối tượng. Do vậy, bộ điều khiển mờ thích hợp để điều khiển những đối tượng phức tạp mà các phương pháp kinh điển không cho được kết quả mong muốn.

Khái niệm về logic mờ được giáo sư Lotfi A. Zadeh đưa ra lần đầu tiên năm 1965, tại trường Đại học Berkeley, bang California - Mỹ. Từ đó lý thuyết mờ đã được phát triển và ứng dụng rộng rãi. Năm 1970 tại trường Mary Queen, London – Anh, Ebrahim Mamdani đã dùng logic mờ để điều khiển một máy hơi nước mà ông không thể điều khiển được bằng kỹ thuật cổ điển. Tại Nhật logic mờ được ứng dụng vào nhà máy xử lý nước của Fuji Electronic vào 1983, hệ thống xe điện ngầm của Hitachi vào 1987. Lý thuyết mờ ra đời ở Mỹ, ứng dụng đầu tiên ở Anh nhưng phát triển mạnh mẽ nhất là ở Nhật.

Trong lĩnh vực tự động hoá logic mờ ngày càng được ứng dụng rộng rãi. Nó thực sự hữu dụng với các đối tượng phức tạp mà ta chưa biết rõ hàm truyền, logic mờ có thể giải quyết các vấn đề mà điều khiển kinh điển không làm được.

1 Các khái niệm cơ bản trong logic mờ

1.1 Định nghĩa tập mờ

Tập mờ A xác định trên tập X là một tập mà mỗi phần tử của nó là một cặp các giá trị $(x, \mu_A(x))$, trong đó $x \in X$ và μ_A là ánh xạ:

$$\mu_A : X \to [0, 1]$$

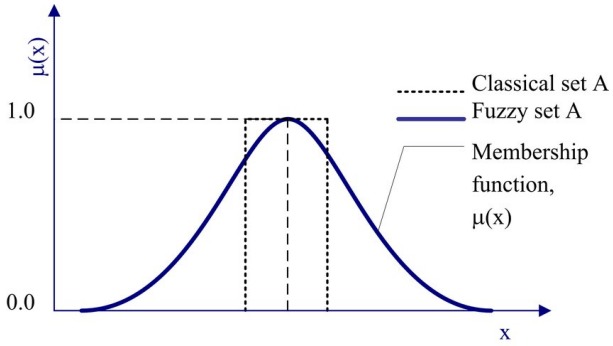
Ánh xạ μ_A được gọi là hàm thuộc hoặc hàm liên thuộc (hoặc hàm thành viên - membership function) của tập mờ A. Tập X được gọi là cơ sở của tập mờ A.

$\mu_A(x)$ là độ phụ thuộc của x vào tập mờ A, sử dụng hàm thuộc để tính độ phụ thuộc của một phần tử X nào đó, có hai cách:

- Tính trực tiếp nếu $\mu_A(x)$ ở dạng công thức tường minh.
- Tra bảng nếu $\mu_A(x)$ ở dạng bảng.

Kí hiệu:

$$A = \{(x, \mu_A(x)) | x \in X\}$$



Hình 1: Tập mờ - Hàm thuộc

Hàm thuộc $\mu_A(x)$ thỏa mãn các điều kiện sau:

$$\mu_A(x) \geq 0 \quad \forall x \in X$$

$$Sup_{x \in X}\, \mu_A(x) = 1$$

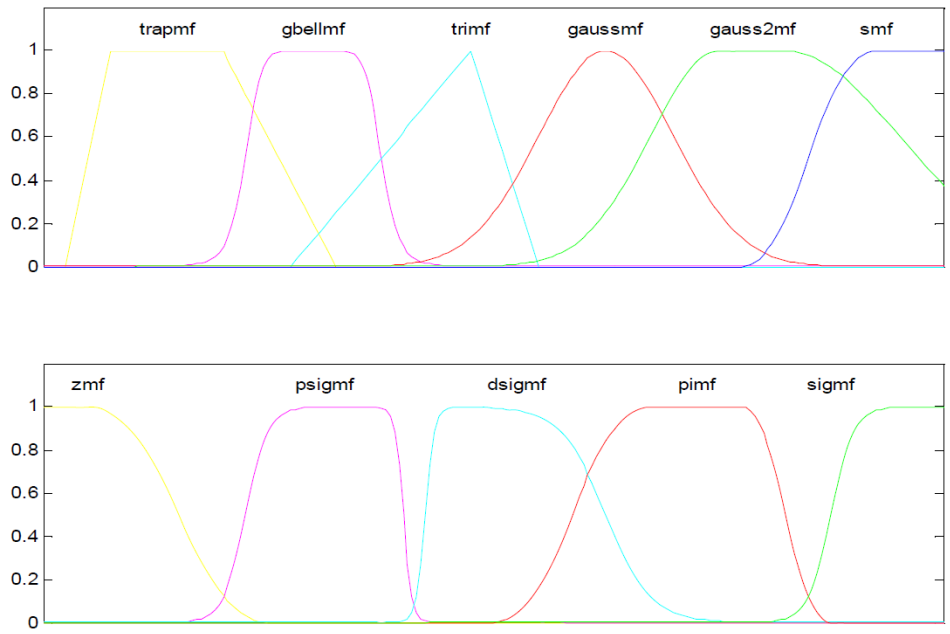
Miền tin cậy của tập mờ A trên tập nền X là một tập T là tập con của X thỏa mãn:

$$T = \{x \in X / \mu_A(x) = 1\}$$

Miền xác định của tập mờ F trên X là một tập S là tập con của X thỏa mãn:

$$S = \{x \in X / \mu_A(x) > 0\}$$

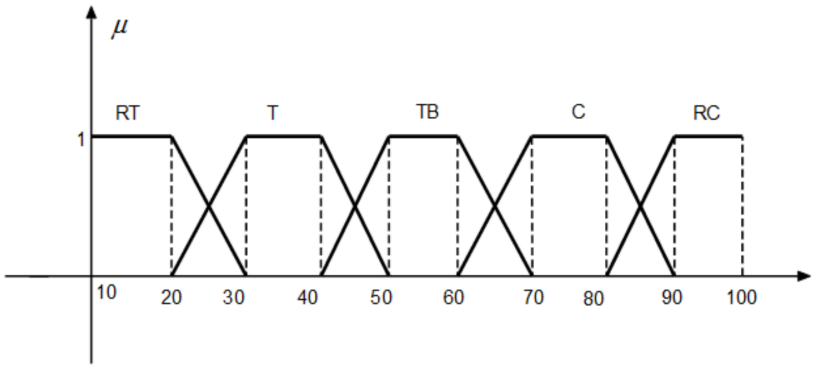
Các dạng hàm thuộc (membership function) trong logic mờ: Gaussian, trapmf, trimf, sigmoid ...



Hình 2: Các dạng hàm thuộc

1.2 Ví dụ về tập mờ

Nhiệt độ của một lò:



- Rất thấp (RT)
- Thấp (T)
- Trung bình (TB)
- Cao (C)
- Rất cao (RC)

Hình 3: Nhiệt đồ của một lò

- Hàm thuộc: $\mu_{RT}(x), \mu_T(x), \mu_{TB}(x), \mu_C(x), \mu_{RC}(x)$
- Giá trị ngôn ngữ: rất thấp, thấp, trung bình, cao, rất cao
- Mỗi x là nhiệt độ ta có:

$$\mu(x) = \{\mu_{RT}(x), \mu_T(x), \mu_{TB}(x), \mu_C(x), \mu_{RC}(x)\}$$

- Tại $x = 47.5°C \rightarrow \mu(47.5) = \{0; 0.25; 0.75; 0; 0\}$

1.3 Phép toán trên tập mờ

Cho A và B là hai tập mờ trên không gian nền X, có các hàm thuộc μ_A, μ_B, theo Zadeh ta có các phép toán:

Phép toán trên tập mờ	Định nghĩa hàm thuộc
$A \subseteq B$	$\mu_A(x) \leq \mu_B(x)$
$A \cup B$	$\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$
$A \cap B$	$\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}$
$\neg A$	$\mu_{\neg A}(x) = 1 - \mu_A(x)$
$A \oplus B$	$\mu_{A\oplus B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$
X	$\mu_X(x) = 1$
$\emptyset$	$\mu_\emptyset(x) = 0$

Ta định nghĩa:

- Phép hợp hai tập mờ: $A \cup B$

  + Luật Max: $\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$

  + Luật Sum: $\mu_{A\cup B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\}$

  + Tổng trực tiếp: $\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x).\mu_B(x)$

- Phép giao hai tập mờ: $A \cap B$

  + Luật Min: $\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}$

  + Luật Lukasiewics: $\mu_{A\cap B}(x) = \max\{0, \mu_A(x) + \mu_B(x) - 1\}$

  + Luật Prod: $\mu_{A\cap B}(x) = \mu_A(x).\mu_B(x)$

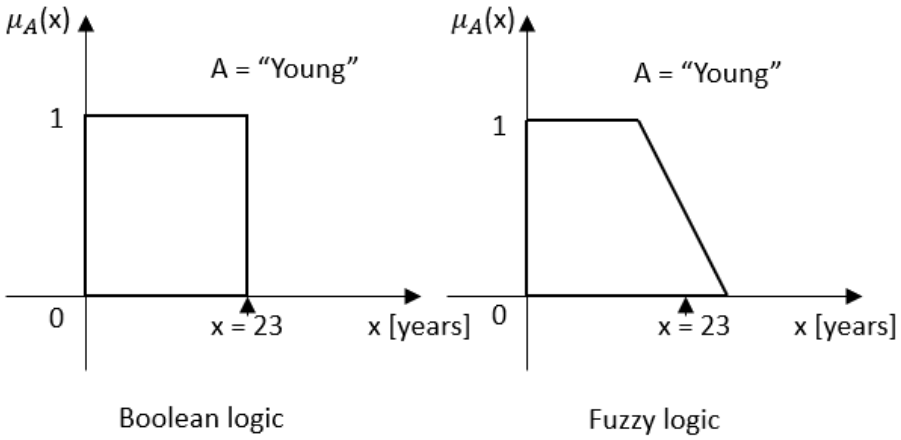
1.4 Logic mờ

Logic mờ (fuzzy logic) là một hệ thống logic dùng công cụ chính là lý thuyết tập mờ, nó tập trung trên biến ngôn ngữ trong ngôn ngữ tự nhiên nhằm cung cấp nền tảng cho lập luận xấp xỉ với những vấn đề không chính

xác, nó phản ánh cả tính đúng đắn lẫn sự mơ hồ của ngôn ngữ tự nhiên trong lập luận theo cảm tính. Điều này có thể hữu ích trong nhiều tình huống thực tế khi các giá trị không chỉ là True (đúng) hoặc False (sai), mà có thể thuộc một phạm vi giá trị giữa hai trạng thái này.

Logic mờ cho phép độ liên thuộc có giá trị trong khoảng đóng 0 và 1, và ở hình thức ngôn từ, các khái niệm không chính xác như "hơi hơi", "gần như", "khá là" và "rất". Cụ thể, nó cho phép quan hệ thành viên không đầy đủ giữa thành viên và tập hợp. Tính chất này có liên quan đến tập mờ và lý thuyết xác suất.

Ví dụ minh họa cho sự mềm dẻo của Logic mờ là việc xác định lứa tuổi:



Hình 4: Ví dụ về logic mờ

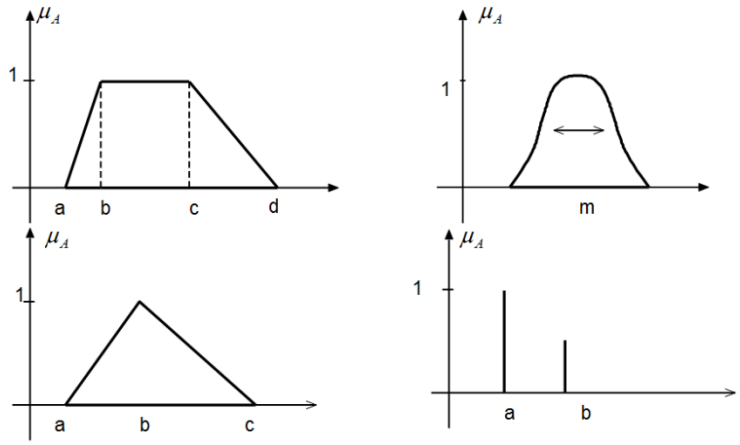
1.5 Số mờ

1.5.1 Khái niệm số mờ

Số mờ hay khoảng mờ dùng để diễn tả khái niệm một số hay một khoảng xấp xỉ hay gần bằng một số thực hay một khoảng số thực cho trước. Số mờ hay khoảng mờ là tập mờ xác định trên tập số thực.

Gọi A là một số mờ, A là một tập mờ trên tập tổng là tập số thực R.

Hàm thuộc của số mờ A là: $\mu_A : R \to [0, 1]$, thường có dạng hình thang, hình tam giác, hình chuông hay hình thẳng đứng như sau:



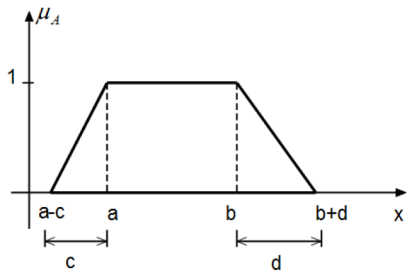
Hình 5: Các dạng hàm thuộc

1.5.2 Dạng số mờ thường dùng

Trong điều khiển, với mục đích sử dụng các hàm thuộc sao cho khả năng tích hợp chúng là đơn giản, người ta thường chỉ quan tâm đến hai dạng số mờ hình thang và số mờ hình tam giác.

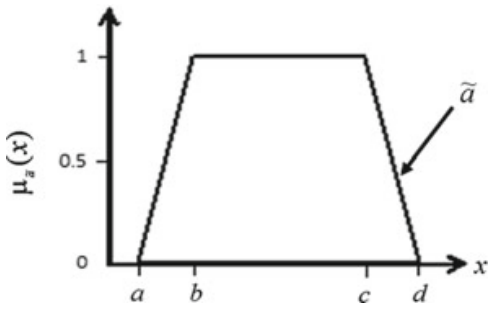
- Số mờ hình thang
  Hàm thuộc có dạng:

$$\mu_A(x) = \begin{cases} 0 & x < a - c \\ (x - a + c)/c & a - c \le x < a \\ 1 & a \le x \le b \\ (b + d - x)/d & b < x \le b + d \\ 0 & b + d < x \end{cases}$$



Hình 6: Số mờ hình thang

Ta xây dựng hàm tfnm(na, nb, nc, x) để tìm tập mờ (sử dụng hình thang có dạng):



```python
def tfnm(na, nb, nc, x):
    if x < na:
        membrsp = 0
    elif x >= na and x <= nb:
        membrsp = (x-na)/(nb-na)
    elif x >= nb and x <= nc:
        membrsp = (nc-x)/(nc-nb)
    elif x > nc:
        membrsp = 0
    return membrsp

values = input("Enter the value of a, b and c for TFN (a,b,c)\n").split()
a = float(values[0])
b = float(values[1])
c = float(values[2])

print("\nThe generated TFN in the form of fuzzy set (x,mu(x)) is")
i = a
while i <= c:
    mem = tfnm(a, b, c, i)
    print(f"({i},{mem})")
    i += 0.1
```

INPUT

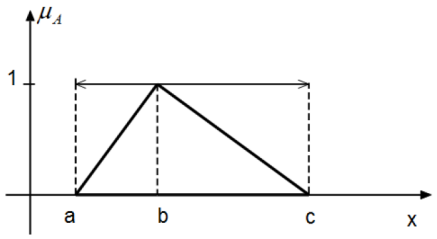
```
Enter the value of a, b, c and d for TrFN (a,b,c,d)
-1 2 4 6
```

OUTPUT

```
The generated TFN in the form of fuzzy set (x,mu(x)) is
(-1.000000,0.000000)
(-0.500000,0.166667)
(0.000000,0.333333)
(0.500000,0.500000)
(1.000000,0.666667)
(1.500000,0.833333)
(2.000000,1.000000)
(2.500000,1.000000)
(3.000000,1.000000)
(3.500000,1.000000)
(4.000000,1.000000)
(4.500000,0.750000)
(5.000000,0.500000)
(5.500000,0.250000)
(6.000000,0.000000)
```

- Số mờ hình tam giác
  Hàm thuộc có dạng:

$$\mu_A(x) = \begin{cases} \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} & b \le x \le c \\ 0 & \text{khác} \end{cases}$$



Hình 7: Số mờ tam giác

```
def tfnm(na, nb, nc, x):
    if x < na:
        membrsp = 0
    elif x >= na and x <= nb:
        membrsp = (x-na)/(nb-na)
    elif x >= nb and x <= nc:
        membrsp = (nc-x)/(nc-nb)
    elif x > nc:
        membrsp = 0
    return membrsp

values = input("Enter the value of a, b and c for TFN
   (a,b,c)\n").split()
a = float(values[0])
b = float(values[1])
c = float(values[2])

print("\nThe generated TFN in the form of fuzzy set
   (x,mu(x)) is")
i = a
while i <= c:
    mem = tfnm(a, b, c, i)
    print(f"({i},{mem})")
    i += 0.1
```

INPUT

```
Enter the value of a, b and c for TFN (a,b,c)
-2 1 2
```

OUTPUT

```
The generated TFN in the form of fuzzy set (x,mu(x)) is
(-2.0,0.0)
(-1.9,0.03333333333333336)
(-1.7999999999999998,0.06666666666666672)
(-1.6999999999999997,0.10000000000000009)
(-1.5999999999999996,0.13333333333333344)
(-1.4999999999999996,0.16666666666666682)
(-1.3999999999999995,0.20000000000000018)
(-1.2999999999999994,0.23333333333333353)
(-1.1999999999999993,0.2666666666666669)
(-1.0999999999999992,0.30000000000000027)
(-0.9999999999999992,0.33333333333333365)
(-0.8999999999999992,0.3666666666666669)
(-0.7999999999999993,0.4000000000000002)
(-0.6999999999999993,0.43333333333333357)
(-0.5999999999999993,0.46666666666666695)
(-0.49999999999999933,0.5000000000000002)
(-0.39999999999999936,0.5333333333333335)
(-0.2999999999999994,0.5666666666666669)
(-0.19999999999999937,0.6000000000000002)
(-0.09999999999999937,0.6333333333333335)
(6.38378239159465e-16,0.6666666666666669)
(0.10000000000000064,0.7000000000000002)
(0.20000000000000065,0.7333333333333335)
(0.30000000000000066,0.7666666666666669)
(0.4000000000000007,0.8000000000000003)
(0.5000000000000007,0.8333333333333336)
(0.6000000000000006,0.8666666666666668)
(0.7000000000000006,0.9000000000000002)
(0.8000000000000006,0.9333333333333336)
(0.9000000000000006,0.9666666666666668)
(1.0000000000000007,0.9999999999999993)
(1.1000000000000008,0.8999999999999992)
(1.2000000000000008,0.7999999999999992)
(1.300000000000001,0.6999999999999991)
(1.400000000000001,0.599999999999999)
(1.500000000000001,0.4999999999999989)
(1.6000000000000012,0.3999999999999988)
(1.7000000000000013,0.2999999999999987)
(1.8000000000000014,0.19999999999999862)
(1.9000000000000015,0.09999999999999853)
```

1.6 Biến ngôn ngữ

Khái niệm **biến ngôn ngữ** đã được Zadeh đưa ra năm 1975 như sau:

- Một biến ngôn ngữ được xác định bởi bộ (x, T, U, M)
  Trong đó: x là tên biến. Ví dụ: "nhiệt độ", "tốc độ", "độ ẩm", ...

- T là tập các từ là các giá trị ngôn ngữ tự nhiên mà x có thể nhận.

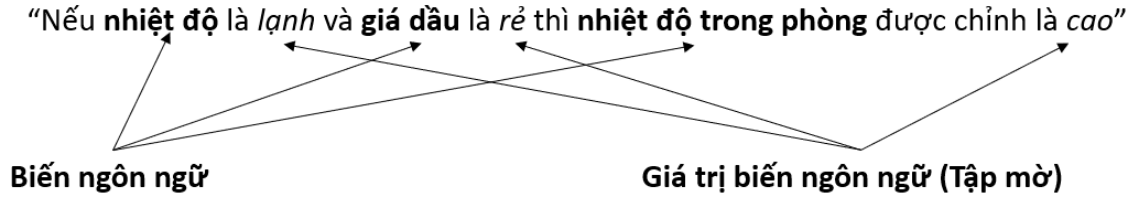
Ví dụ: x là "tốc độ" thì T có thể là "chậm", "trung bình", "nhanh"

- U là miền các giá trị vật lý mà x có thể nhận.
  Ví dụ: x là "tốc độ" thì U có thể là 0km/h, 1km/h, ..., 150km/h

- M là quy tắc ngữ nghĩa, ứng mỗi từ trong T với một tập mờ A trong U.

1.7 Luật mờ

Một luật mờ là một biểu thức IF - THEN (Nếu - Thì) được phát biểu ở dạng ngôn ngữ tự nhiên thể hiện sự phụ thuộc nhân quả giữa các biến.



Giá dầu \ Nhiệt độ	Lạnh	Ấm	Nóng
Rẻ	Cao	Cao	Trung bình
Bình thường	Cao	Trung bình	Thấp
Đắt	Trung bình	Thấp	Thấp

1.8 Luật hợp thành mờ

Luật hợp thành là tên gọi chung của mô hình biểu diễn một hay nhiều hàm thuộc cho một hay nhiều mệnh đề hợp thành (mệnh đề có cấu trúc chung IF A THEN B)

Các luật hợp thành cơ bản

+ Luật Max - Min

+ Luật Max - Prod

+ Luật Sum - Min

+ Luật Sum - Prod

$\Rightarrow$ Ta thường sử dụng luật Max - Min và luật Sum - Min trong điều khiển.

2 Bộ điều khiển mờ

2.1 Giới thiệu bộ điều khiển mờ

Lý thuyết logic mờ đã có nhiều áp dụng thành công trong lĩnh vực điều khiển. Bộ điều khiển dựa trên lý thuyết logic mờ gọi là bộ điều khiển mờ. Trái với kỹ thuật điều khiển kinh điển, kỹ thuật điều khiển mờ thích hợp với các đối tượng phức tạp, không xác định mà người vận hành có thể điều khiển bằng kinh nghiệm. Đặc điểm của bộ điều khiển mờ là không cần biết mô hình toán học mô tả đặc tính động của hệ thống mà chỉ cần biết đặc tính của hệ thống dưới dạng các phát biểu ngôn ngữ. Chất lượng của bộ điều khiển mờ phụ thuộc rất nhiều vào kinh nghiệm của người thiết kế.

Về nguyên tắc, hệ thống điều khiển mờ cũng không có gì khác so với hệ thống điều khiển tự động thông thường khác. Sự khác biệt ở đây là bộ điều khiển mờ làm việc có tư duy như "bộ não" dưới dạng trí tuệ nhân tạo. Nếu khẳng định với bộ điều khiển mờ có thể giải quyết mọi vấn đề từ trước đến nay chưa giải quyết được theo phương pháp kinh điển thì không hoàn toàn chính xác, vì hoạt động của bộ điều khiển phụ thuộc vào kinh nghiệm và phương pháp rút ra kết luận theo tư duy con người, sau đó được cài đặt vào máy tính dựa trên cơ sở logic mờ.

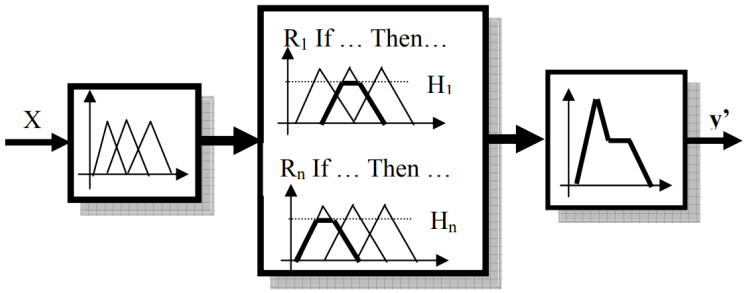
2.2 Cấu trúc bộ điều khiển mờ

Bộ điều khiển mờ bao gồm 3 khâu chính:

- Khâu mờ hóa

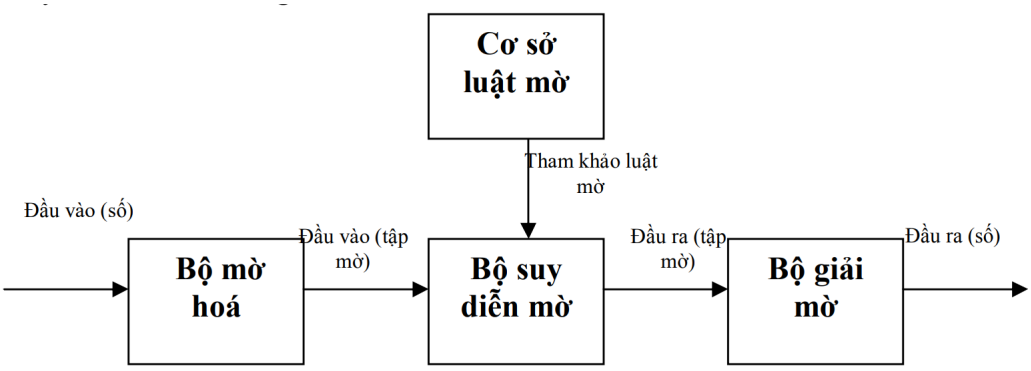
- Khâu thực hiện luật suy diễn (cơ sở luật mờ)

- Khâu giải mờ

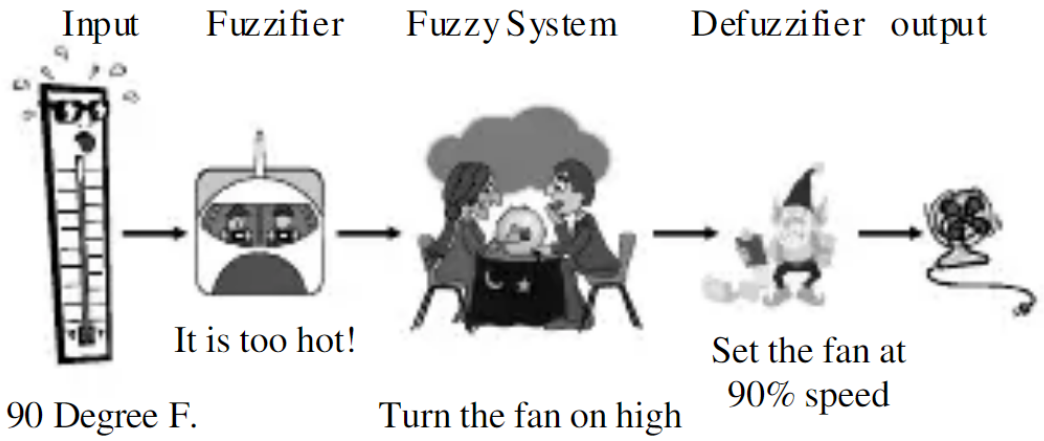


Hình 8: Các khâu trong bộ điều khiển mờ

Trong đó bao gồm 4 thành phần cơ bản: Bộ mờ hóa, cơ sở luật mờ, bộ suy diễn mờ, bộ giải mờ.



Hình 9: Cấu trúc bộ điều khiển mờ



Hình 10: Minh họa bộ điều khiển mờ

2.3 Bộ mờ hóa

Vì các luật cho dưới dạng dùng các biến ngôn ngữ với các từ thông thường. Như vậy với những giá trị (rõ) quan sát được, đo được cụ thể, để có thể tham gia vào quá trình điều khiển thì cần thiết phải mờ hóa.

Có thể định nghĩa, mờ hóa là một ánh xạ từ không gian các giá trị quan sát được vào không gian của các từ - tập mờ trên không gian nền của các biến ngôn ngữ đầu vào.

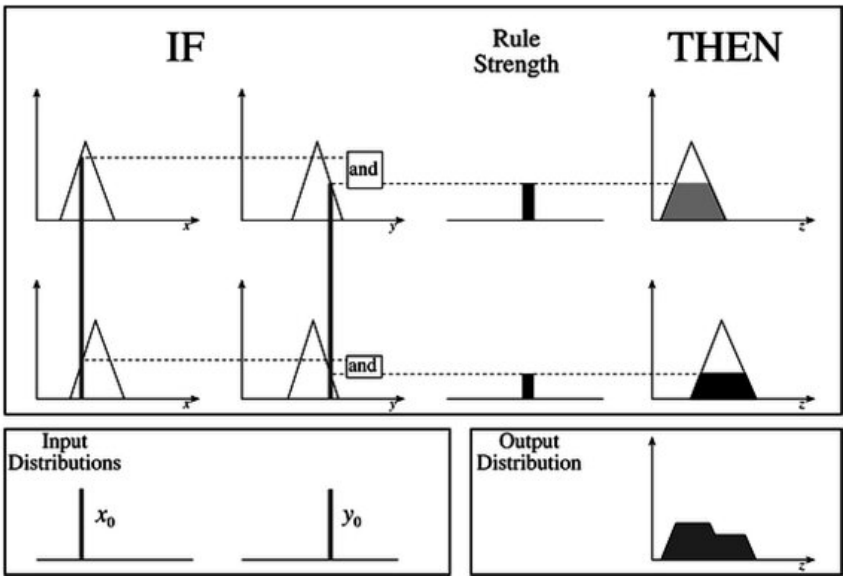
2.4 Cơ sở luật mờ

Có nhiều phương pháp để xác định các luật mờ để đưa vào cơ sở luật mờ. Các phương pháp thông dụng là nhờ các chuyên gia trong lĩnh vực áp dụng, hoặc từ quan sát, thực nghiệm thống kê để có được các tập dữ liệu mẫu đầu vào và ra tương ứng, từ đó dùng các kỹ thuật khai mở dữ liệu để rút ra các luật.

2.5 Bộ suy diễn mờ

Đây là phần cốt lõi nhất của bộ điều khiển mờ trong quá trình mô hình hóa các bài toán điều khiển và chọn quyết định của con người trong khuôn khổ vận dụng logic mờ và lập luận xấp xỉ.

Giá trị rõ sau khi qua bộ mờ hóa thu được tập mờ đầu vào sử dụng trong bộ suy diễn mờ, kết hợp cơ sở luật mờ và lựa chọn luật hợp thành phù hợp thì bộ suy diễn mờ sẽ đưa ra tập mờ đầu ra và được sử dụng trong bộ giải mờ.



Hình 11: Mô tả bộ suy diễn mờ

2.6 Bộ giải mờ

Đây là khâu thực hiện quá trình xác định một giá trị rõ có thể chấp nhận được làm đầu ra từ hàm thuộc của giá trị mờ đầu ra. Có nhiều

phương pháp giải mờ: phương pháp điểm cực đại và phương pháp điểm trọng tâm, phương pháp độ cao ...

Các phương pháp giải mờ

2.6.1 Phương pháp điểm cực đại

Các bước thực hiện:

- Xác định miền chứa giá trị rõ y_o thỏa mãn:

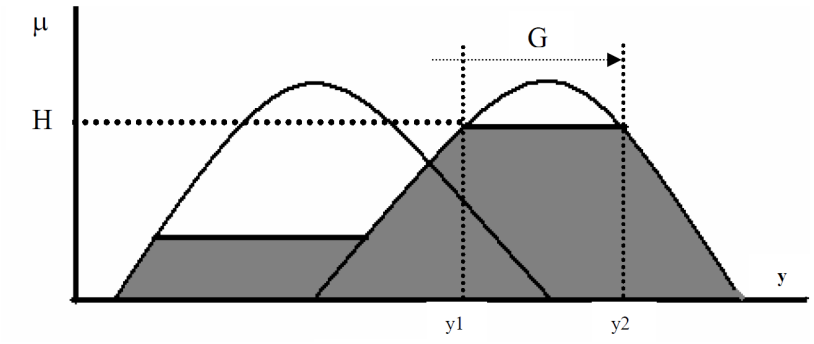
$$G = \{y \in Y \mid \mu_R(y) = H\} \quad H\text{:độ thuộc cực đại của hàm thuộc}$$

- Xác định y_o: G là đoạn $[y_1, y_2]$ của tập nền R, ta có 3 cách tìm y_o:
  - Nguyên lý trung bình: $y_o = \dfrac{y_1 + y_2}{2}$

    $\Rightarrow$Nếu các hàm thuộc đều có dạng hình tam giác hoặc hình thang thì điểm y_0 xác định theo phương pháp này sẽ không quá bị nhạy cảm với sự thay đổi của giá trị đầu vào rõ x_0. Do đó rất thích hợp với các bài toán có nhiễu biên độ nhỏ tại đầu vào.

  - Nguyên lý cận trái: $y_o = y_1$
  - Nguyên lý cận phải: $y_o = y_2$

    $\Rightarrow$Nếu các hàm thuộc đều có dạng hình tam giác hoặc hình thang thì điểm y_0 sẽ phụ thuộc tuyến tính vào giá trị rõ x_0 tại đầu vào.



Hình 12: Minh họa

2.6.2 Phương pháp điểm trọng tâm

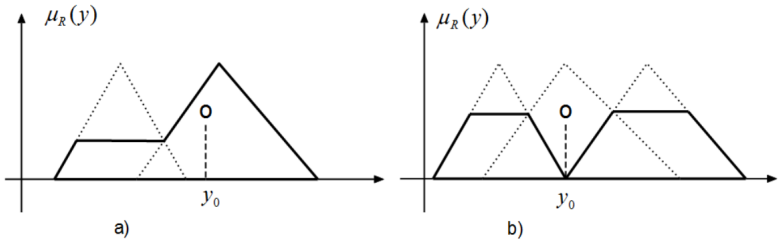
Phương pháp điểm trọng tâm sẽ cho kết quả y_o là hoành độ của điểm trọng tâm, miền được bao phủ bởi trục hoành và hàm thuộc.

Công thức để tính được giá trị y_o được cung cấp như sau:

$$y_o = \frac{\int\limits_S y\mu_R(y)\,dy}{\int\limits_S \mu_R(y)\,dy}$$

Với $S = \{y \mid \mu_R(y) \neq 0\}$ là miền xác định của tập mờ R.

Đây là phương pháp ưa được sử dụng nhất, cho phép ta xác định giá trị y_0 với sự tham gia của tất cả các tập mờ đầu ra của luật mờ một cách bình đẳng, chính xác. Tuy nhiên một trong những nhược điểm cơ bản của phương pháp này là giá trị y_0 xác định được lại có độ thuộc nhỏ nhất, thậm chí bằng 0.



Hình 13: Giải mờ bằng phương pháp điểm trọng tâm.

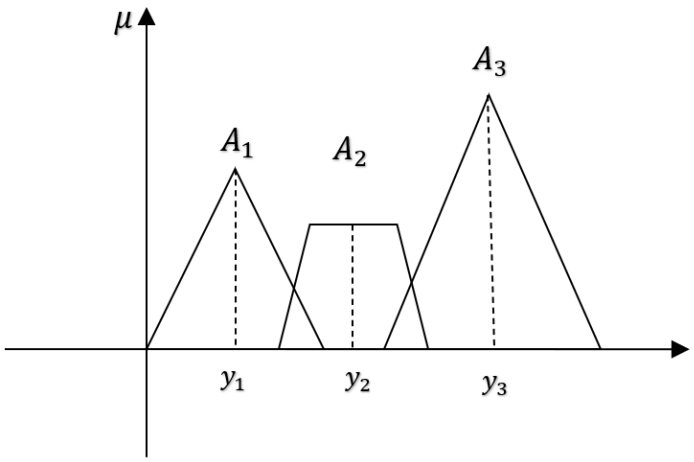
2.6.3 Phương pháp độ cao

Phương pháp độ cao được hình thành bằng cách tính trọng số của từng hàm ở đầu ra theo độ thuộc tối đa tương ứng của nó.

Phương pháp này áp dụng cho các tập mờ có hàm thuộc đầu ra đối xứng và cho kết quả khá giống với phương pháp điểm trọng tâm, tuy nhiên ít tính toán hơn. Độ thuộc tối đa được sử dụng để tính trọng số cho từng hàm thuộc.

Công thức:

$$y_o = \frac{\sum \mu_{A_i}(y_i)(y_i)}{\sum \mu_{A_i}(y_i)}$$



Hình 14: Giải mờ bằng phương pháp độ cao

3 Thiết kế bộ điều khiển mờ

3.1 Nguyên lý làm việc của bộ điều khiển mờ

+ Giao diện đầu vào gồm: Mờ hóa và các khâu hiệu chỉnh như tỷ lệ, tích phân, vi phân ...

+ Bộ suy diễn mờ: triển khai luật hợp thành mờ

+ Giao diện đầu ra gồm: Khâu giải mờ và các khâu giao diện trực tiếp với đối tượng.

3.2 Các bước thiết kế bộ điều khiển mờ

- **Bước 1**: Định nghĩa tất cả các biến ngôn ngữ vào/ra, đó cũng chính là các tín hiệu vào/ra của bộ điều khiển.

- **Bước 2**: Xác định các tập mờ (giá trị ngôn ngữ) cho từng biến vào/ra, tức là thực hiện công việc mờ hóa:

  + Miền giá trị vật lý của các biến ngôn ngữ.

  + Xác định số lượng tập mờ.

  + Xác định hàm thuộc.

  + Rời rạc hóa tập mờ.

- **Bước 3**: Xây dựng luật điều khiển.

- **Bước 4**: Chọn cơ chế suy diễn mờ.

- **Bước 5**: Giải mờ.

3.3 Lưu ý và phân loại nhóm thiết kế bộ điều khiển mờ

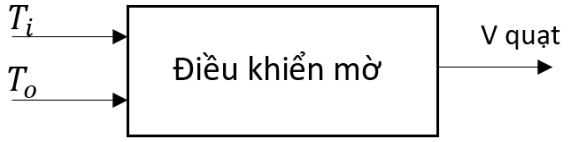
- Những lưu ý khi thiết kế bộ điều khiển mờ
  - Không bao giờ dùng điều khiển mờ để giải quyết bài toán mà có thể dễ dàng thực hiện bằng bộ điều khiển kinh điển (sử dụng logic nhị phân).
  - Không nên dùng bộ điều khiển mờ cho các hệ thống cần độ an toàn cao.
  - Thiết kế bộ điều khiển mờ phải được thực hiện qua thực nghiệm.
- Phân loại nhóm bộ điều khiển mờ:
  - Nhóm bộ điều khiển SISO: Chỉ có một đầu vào và một đầu ra.
  - Nhóm bộ điều khiển MIMO: Nhiều đầu vào và nhiều đầu ra.
  - Nhóm bộ điều khiển SIMO: Chỉ có một đầu vào nhưng nhiều đầu ra.
  - Nhóm bộ điều khiển MISO: Nhiều đầu và một đầu ra.

3.4 Ví dụ ứng dụng

* **Thiết kế điều khiển tự động máy điều hòa:**

- 2 đầu vào: T_i đo nhiệt độ trong nhà, T_o đo nhiệt độ bên ngoài
- 1 đầu ra: Tốc độ quạt
- Thông số:
  + Tầm nhiệt quan tâm: $[0°C, 50°C]$
  + Tốc độ quạt: $[0, 600\,\text{vòng/phút}]$
- Tính tốc độ quạt với:
  + $T_i = 27°C$
  + $T_o = 32°C$

+ **Bước 1: Xác định biến ngôn ngữ Input/Output**



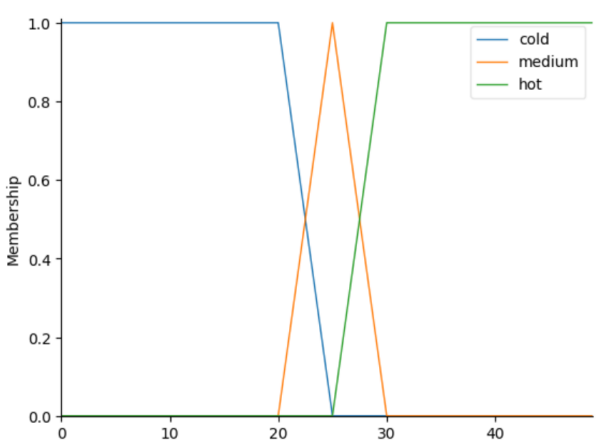
+ **Bước 2: Xác định tập mờ, hàm thuộc**

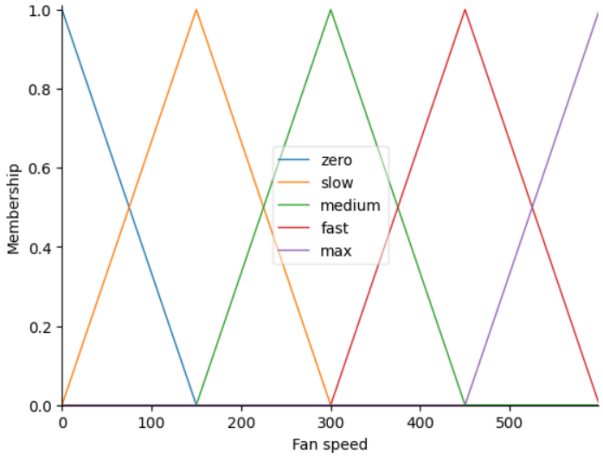
  – T_i, T_o : Lạnh, vừa, nóng tương ứng $\{20°C, 25°C, 30°C\}$

  – V : {Zero, Chậm, Trung bình, Nhanh, Max} tương ứng {0, 150, 300, 450, 600}

*** Hàm thuộc**

– Chọn hàm thuộc tam giác





– Xét $T_i* = 27°C$ và $T_o* = 32°C$

- $\mu(T_i*) = \mu(27°C) = \{0; 0.6; 0.4\}$
- $\mu(T_o*) = \mu(32°C) = \{0; 0; 1\}$

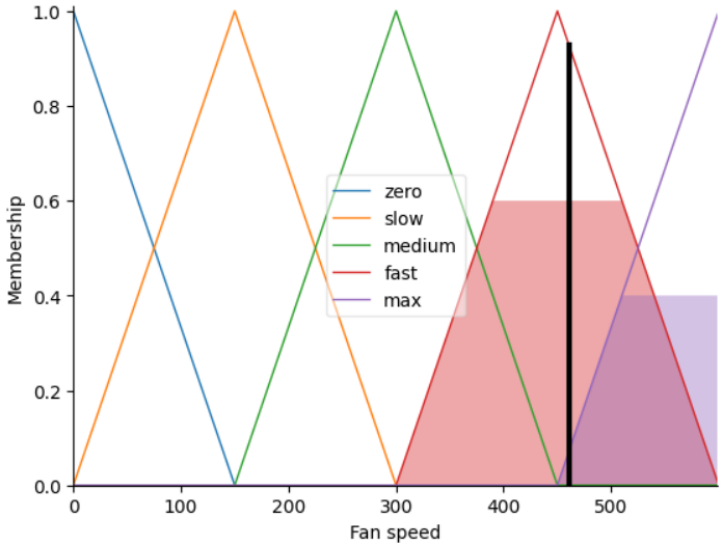
+ **Bước 3: Xây dựng luật điều khiển**

$\mathbf{T_i}$ / $\mathbf{T_o}$	**Lạnh**	**Vừa**	**Nóng**
Lạnh	Zero	Chậm	Trung bình
Vừa	Chậm	Trung bình	Nhanh
Nóng	Trung bình	Nhanh	Max

+ **Bước 4: Suy diễn mờ:**

→ Chọn cơ chế suy diễn (Luật hợp thành) MAX - MIN

T_i / T_o	**Lạnh** 0	**Vừa** 0.6	**Nóng** 0.4
Lạnh 0	Zero 0	Chậm 0	Trung bình 0
Vừa 0	Chậm 0	Trung bình 0	Nhanh 0
Nóng 1	Trung bình 0	Nhanh 0.6	Max 0.4



Hình 15: Tập mờ output sau khi suy diễn mờ

+ **Bước 5: Giải mờ**

Sử dụng phương pháp điểm trọng tâm để giải mờ:

$$y' = \frac{\int\limits_S y\mu_R(y)\,dy}{\int\limits_S \mu_R(y)\,dy} = 460.9(\text{vòng/phút})$$

4 Công cụ thực thiện logic mờ

Các công cụ hỗ trợ thực hiện:

- Matlab (Fuzzy Logic Toolbox)

- FuzzyTech

- Winfact
- C++ (Fuzzy lite)
- Delphi
- C# (Accord.Fuzzy, DotFuzzy, ...)
- Python (pyfuzzylite, Scikit-fuzzy, ...)

* Sử dụng thư viện skfuzzy trong Python để mô hình ví dụ thiết kế điều khiển tự động điều hòa.

```
import matplotlib.pyplot as plt
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

# Antecedents
tempIn = ctrl.Antecedent(np.arange(0, 50), 'Temperature indoor')
tempOut =  ctrl.Antecedent(np.arange(0, 50), 'Temperature
   outdoor')

# Consequents
fanSpeed = ctrl.Consequent(np.arange(0, 600), 'Fan speed')

# Temperature indoor memberships
tempIn['cold'] = np.where(tempIn.universe < 20, 1
   ,fuzz.trimf(tempIn.universe, [15, 20, 25]))
tempIn['medium'] = fuzz.trimf(tempIn.universe, [20, 25, 30])
tempIn['hot'] = np.where(tempIn.universe > 30, 1
   ,fuzz.trimf(tempIn.universe, [25, 30, 35]))

# Temperature outdoor memberships
tempOut['cold'] = np.where(tempOut.universe < 20, 1
   ,fuzz.trimf(tempOut.universe, [15, 20, 25]))
tempOut['medium'] = fuzz.trimf(tempOut.universe, [20, 25, 30])
tempOut['hot'] = np.where(tempOut.universe > 30, 1
   ,fuzz.trimf(tempOut.universe, [25, 30, 35]))

# Fan speed memberships
fanSpeed['zero'] = fuzz.trimf(fanSpeed.universe, [-150, 0, 150])
fanSpeed['slow'] = fuzz.trimf(fanSpeed.universe, [0, 150, 300])
fanSpeed['medium'] = fuzz.trimf(fanSpeed.universe, [150, 300,
   450])
fanSpeed['fast'] = fuzz.trimf(fanSpeed.universe, [300, 450, 600])
fanSpeed['max'] = fuzz.trimf(fanSpeed.universe, [450, 600, 750])

# Rule system
```

```
31 # Rules for fan speed
32 rule1 = ctrl.Rule(tempIn['cold'] & tempOut['cold'],
      fanSpeed['zero'])
33 rule2 = ctrl.Rule(tempIn['medium'] & tempOut['cold'],
      fanSpeed['slow'])
34 rule3 = ctrl.Rule(tempIn['hot'] & tempOut['cold'],
      fanSpeed['medium'])
35 rule4 = ctrl.Rule(tempIn['cold'] & tempOut['medium'],
      fanSpeed['slow'])
36 rule5 = ctrl.Rule(tempIn['medium'] & tempOut['medium'],
      fanSpeed['medium'])
37 rule6 = ctrl.Rule(tempIn['hot'] & tempOut['medium'],
      fanSpeed['fast'])
38 rule7 = ctrl.Rule(tempIn['cold'] & tempOut['hot'],
      fanSpeed['medium'])
39 rule8 = ctrl.Rule(tempIn['medium'] & tempOut['hot'],
      fanSpeed['fast'])
40 rule9 = ctrl.Rule(tempIn['hot'] & tempOut['hot'],
      fanSpeed['max'])
41
42 # Control System Creation and Simulation
43 fanSpeed_output =
      ctrl.ControlSystemSimulation(ctrl.ControlSystem([rule1,
      rule2, rule3, rule4, rule5, rule6, rule7, rule8, rule9]))
44
45 # Enter values to test
46 indoorTemperature = float(input("Input Ti (Temperature indoor
      [0, 50 degrees C]): "))
47 while indoorTemperature < 0 or indoorTemperature > 50:
48     try:
49       indoorTemperature = float(input("Please input Ti between 0
            and 50: "))
50     except ValueError:
51       print("We expect you to enter a valid integer!")
52
53 outdoorTemperature = float(input("Input To (Temperature outdoor
      [0, 50 degrees C]): "))
54 while outdoorTemperature < 0 or outdoorTemperature > 50:
55     try:
56       outdoorTemperature = float(input("Please input To between
            0 and 50: "))
57     except ValueError:
58       print("We expect you to enter a valid integer!")
59
60 # Defuzzification
61 fanSpeed_output.input['Temperature indoor'] = indoorTemperature
62 fanSpeed_output.input['Temperature outdoor'] = outdoorTemperature
63
64 fanSpeed_output.compute()
65
66 # View
```

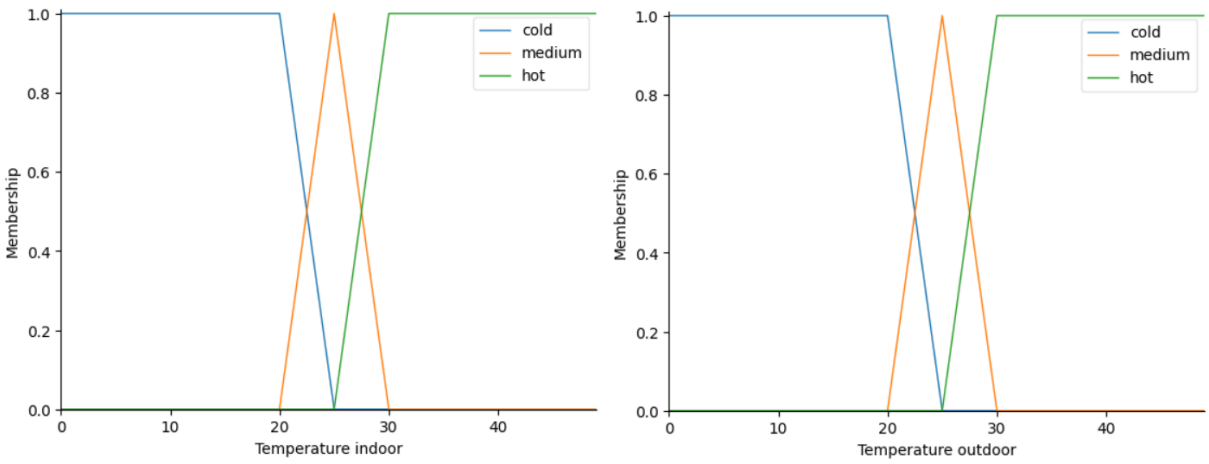
```
67 print(f"\nFan Speed after defuzzyfication (using centroid
       gravity method): {fanSpeed_output.output['Fan speed']}
       (RPM)\n")
68 tempIn.view()
69 tempOut.view()
70 fanSpeed.view(sim=fanSpeed_output)
71 plt.show()
```

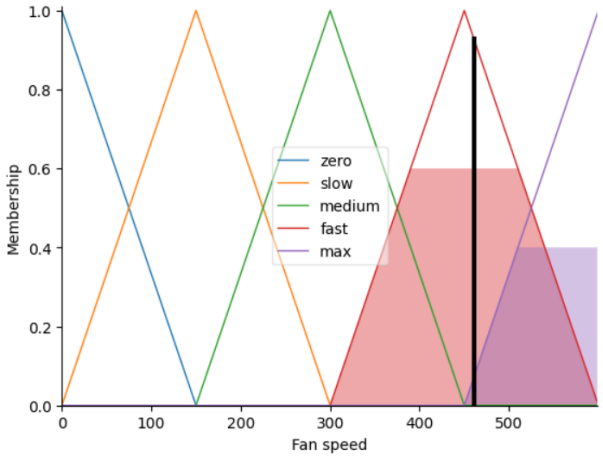
INPUT

Input T_i (Temperature indoor [0, 50°C]): 27
Input T_o (Temperature outdoor [0, 50°]): 32

OUTPUT

Fan Speed after defuzzyfication (using centroid of gravity method)
460.9026162790696 (RPM)

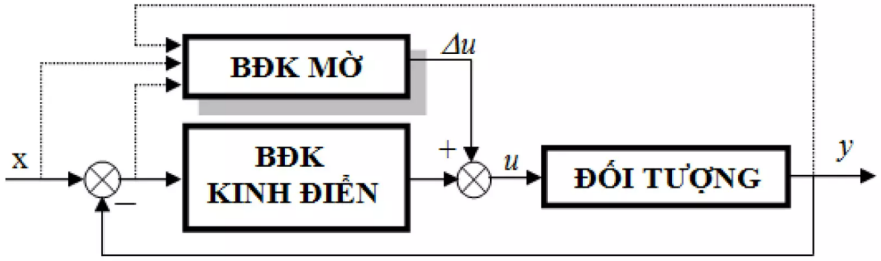




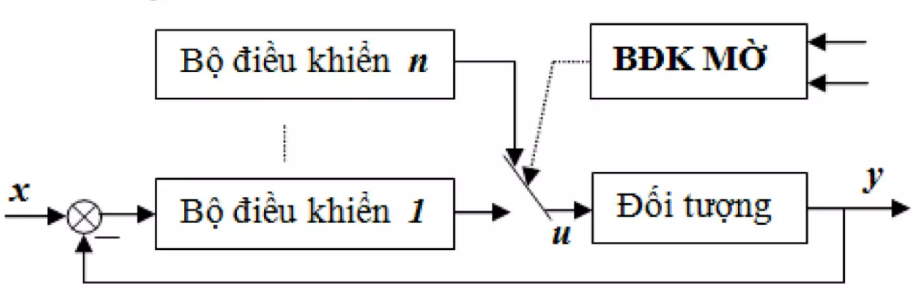
5 Mô hình ứng dụng điều khiển mờ

- Hệ mờ lai không thích nghi

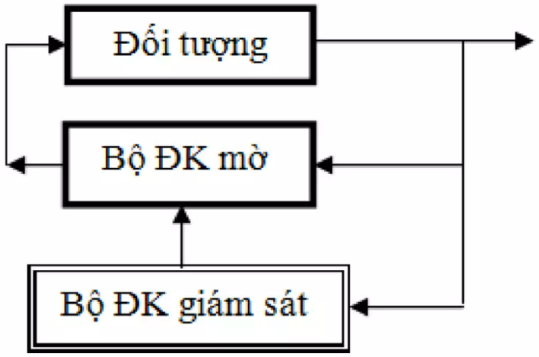
- Hệ mờ lai cascade



- Công tắc mờ



- Điều khiển mờ có hệ thống giám sát



6 Một số ứng dụng của logic mờ

Logic mờ vẫn luôn bị phê phán ở một số cộng đồng nghiên cứu, nhưng một số nhà thống kê khẳng định rằng hệ thống logic mờ là mô tả toán học chặt chẽ duy nhất về sự không chắc chắn (uncertainty).

Tuy nhiên chúng vẫn tạo nên nhiều ứng dụng thành công và được chấp nhận rộng rãi.

- Các hệ thống con của ô tô và các phương tiện giao thông khác, chẳng hạn các hệ thống con như ABS và quản lý hơi (ví dụ Tokyo monorail)
- Máy điều hòa nhiệt độ
- Camera
- Xử lý ảnh số (Digital image processing), chẳng hạn như phát hiện biên (edge detection)
- Nhận dạng mẫu trong Cảm nhận từ xa (Remote Sensing)
- Thang máy
- Nồi cơm điện, máy rửa bát, máy giặt và các thiết bị gia dụng khác
- Trí tuệ nhân tạo trong trò chơi điện tử
- Các bộ lọc ngôn ngữ tại các bảng tin (message board) và phòng chat để lọc bỏ các đoạn văn bản khiếm nhã
- Logic mờ cũng đã được tích hợp vào một số bộ vi điều khiển và vi xử lý

  Và logic mờ còn được ứng dụng trong rất nhiều lĩnh vực khác.

Kết luận

Trong phần này, chúng ta đã tìm hiểu về các khái niệm quan trọng trong logic mờ. Điều này bao gồm tập mờ (fuzzy set), hàm thuộc (membership function) và các phép toán logic mờ ... Những khái niệm đó là nền tảng phát triển logic mờ. Logic mờ đã chứng minh khả năng xử lý thông tin không chắc chắn và mơ hồ, điều này rất quan trọng trong môi trường thực tế nơi mà dữ liệu thường không hoàn toàn chính xác hoặc đầy đủ. Khái niệm cơ bản như tập mờ, hàm thuộc, giúp chúng ta mô phỏng và đo lường mức độ không rõ ràng trong quyết định, tạo ra sự linh hoạt và chính xác hơn trong các ứng dụng tự động hóa.

Ta cũng đã xem xét về bộ điều khiển mờ và cách nó được áp dụng trong tự động hóa. Bộ điều khiển mờ sử dụng các quy tắc logic mờ để điều chỉnh các biến đầu vào và đạt được các giá trị đầu ra mong muốn. Điều này cho phép chúng ta xử lý các tình huống không rõ ràng và không chắc chắn một cách hiệu quả Bộ điều khiển mờ, như một phần quan trọng của logic mờ, được thiết kế để xử lý đầu vào mờ và tạo ra đầu ra mờ, giúp giải quyết vấn đề mô phỏng logic không chắc chắn. Trong quá trình thiết kế bộ điều khiển mờ, các bước cụ thể như xác định biến đầu vào, xây dựng tập quy tắc mờ, lựa chọn hàm ánh xạ mờ, và điều chỉnh các tham số đều đóng vai trò quan trọng. Ứng dụng của logic mờ trong bộ điều khiển giúp hệ thống tự động hóa đạt được hiệu suất cao và khả năng đáp ứng linh hoạt đối với biến động và không chắc chắn trong môi trường.

Phần báo cáo của nhóm đã cố gắng truyền tải nội dung một cách chặt chẽ, đầy đủ, tuy có nhiều phần còn thiếu sót. Trong tương lai nhóm sẽ tiếp tục hoàn thiện hơn để mang đế cho các bạn đọc những bài báo cáo chỉn chu nhất.

Xin chân thành cảm ơn!

Hà Nội, 2023

Tài liệu tham khảo

[1] Zadeh L.A. Fuzzy Sets, "Information and Control", vol.8, p338-353 (1965)

[2] Guanrong Chen & Trung Tat Pham. "Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems" (2000)

[3] Jager M. "Fuzzy Logic in Control" (1995)

[4] Jan Jantzen. "Foundations of Fuzzy Control" (2007)

[5] Snehashish Chakraverty, Deepti Moyi Sahoo, Nisha Rani Mahato. "Concepts of Soft Computing: Fuzzy and ANN with Programming" (2019)