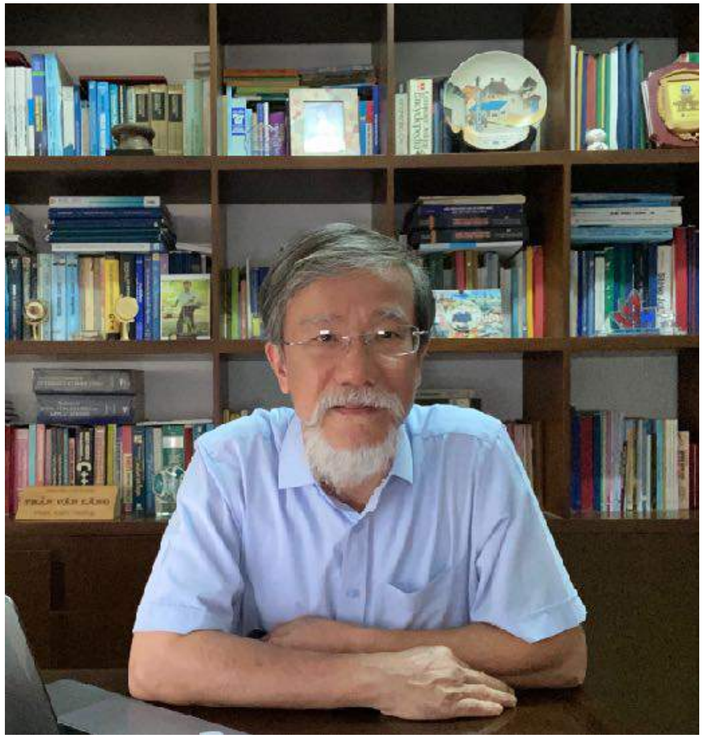
# Trí tuệ tính toán và ứng dụng

## Computational Intelligence and Its Applications



langtv@vast.gov.vn

A.Prof. Tran Van Lang, PhD - Vietnam Academy of Science and Technology

# Mô tả về môn học

- Môn học bao gồm các kiến thức trong những lĩnh vực lấy cảm hứng từ tự nhiên, cũng như mô phỏng hoạt động của trí tuệ con người; qua đó thực hiện các tính toán thông minh như:
  - Tư duy logic với các hệ thống điều khiển mờ (Fuzzy Control System).
  - Học máy với các thuật toán dự đoán, phân lớp, gom cụm, ...
- Đồng thời cũng bao gồm một số ứng dụng dùng trí tuệ tính toán (Computational Intelligence - CI)
- Và quan trọng là phải hiện thực được ứng dụng bằng một ngôn ngữ lập trình nào đó (Python) và sử dụng công nghệ sẵn có (TensorFlow).

# Tài liệu học tập và tham khảo

- Sách giáo trình chính:
  - Andries P. Engelbretch, *Computational Intelligence: An Introduction*, 2nd Edition, John Wiley & Sons Ltd, 598 p, 2007
  - Aurélien Géron, Hands-On Machine Learning with Scikit-Learn and TensorFlow, O'Reilly Media, Inc., 542 p, 2017
  - Vũ Hữu Tiệp, *Machine Learning cơ bản*, Nxb. Khoa học và Kỹ thuật, 422 tr, 2018
- Sách tham khảo
  - Plamen Parvanov Anlgelov, *Hanbook on Computational Intelligence*, World Scientific Publishing Company, 958 p, 2016

# Giới thiệu chung

## General Introduction (1/3)

- Các hệ thống thông minh có thể dựa trên các tính toán mang tính thông minh. Nên có thể sử dụng những tri thức và công nghệ sau đây để xây dựng hệ thống:
  - Tư duy logic với hệ thống điều khiển dùng Python
  - Machine Learning, Deep Learning với TensorFlow

langtv@gmail.com

# Giới thiệu chung

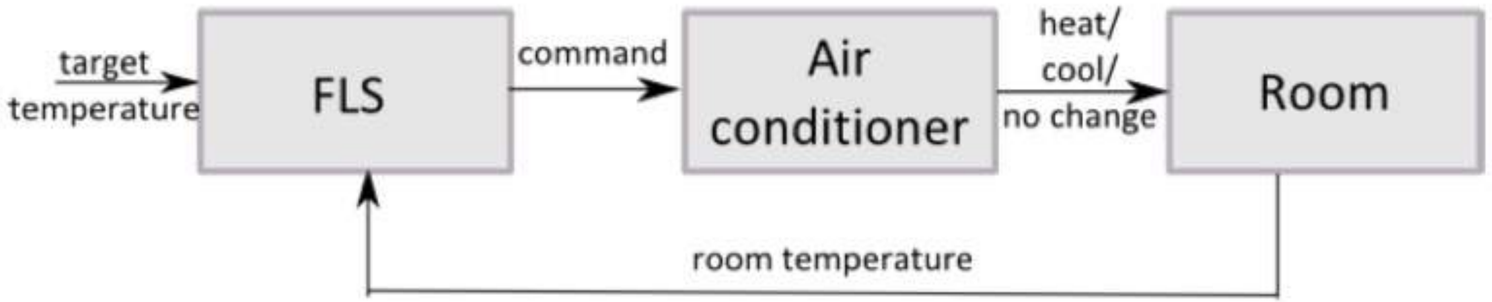
## 2/3

- Tư duy logic trong thời đại ngày nay có 2 dạng:
  - Tư duy với logic cổ điển
  - Với logic mờ
- Trong đó logic mờ là sự mở rộng của logic cổ điển (nói cách khác logic cổ điển là một trường hợp riêng của logic mờ)

# Giới thiệu chung

## 3/3

- Có thể sử dụng logic mờ trong việc xây dựng các hệ thống điều khiển,
  - Chẳng hạn điều khiển hệ thống lạnh trong phòng



# Khái quát về Logic mờ (1/12)

## An Overview of Fuzzy Logic

- Logic mờ
- Tập hợp mờ

# Khái quát về Logic mờ (2/12)

## Logic mờ  (Fuzzy Logic) (1/3)

- Logic cổ điển có 2 giá trị chân trị là sai (false) và đúng (true) ứng với 2 số nguyên là 0 và 1
- Trong khi đó Logic mờ là logic có giá trị chân trị là những số thuộc [0,1]
  - Nó là logic chính xác nhằm làm việc với các dữ liệu không chính xác.
  - Ví dụ: học giỏi, lương cao, lớn tuổi

# Khái quát về Logic mờ (3/12)

## Logic mờ (2/3)

- Hoặc một ví dụ thực tế khác
  - Độ ma sát trên đường khi lái xe theo quy chuẩn như sau bao gồm quy định về hệ số ma sát và độ nguy hiểm:

| Hệ số ma sát | Độ nguy hiểm |
|---|---|
| > 0.8 | Rất an toàn |
| 0.60 - 0.79 | Tương đối an toàn |
| 0.40 - 0.59 | Nguy hiểm |
| 0.35 - 0.39 | Rất nguy hiểm |
| 0.00 - 0.34 | Vô cùng nguy hiểm |

# Khái quát về Logic mờ (4/12)

## Logic mờ (3/3)

- Nhận xét:
  - Với một hệ số ma sát là 0,595 giữa nguy hiểm và tương đối an toàn nên đó là gì: ***tương đối an toàn*** hay ***nguy hiểm***
- Logic cổ điển không lý giải được mệnh đề "*Hệ số ma sát là 0,595 nên đoạn đường này tương đối an toàn*" là đúng hay sai.

# Khái quát về Logic mờ (5/12)

## Tập hợp mờ (Fuzzy Set) (1/8)

- Logic mờ được định nghĩa dựa trên **tập hợp mờ** hay **tập mờ** (fuzzy set)
  - Ví dụ: *X* là tập hợp những nghiên cứu viên của Viện Hàn lâm Khoa học và Công nghệ Việt Nam (VAST)
  - Tập hợp *A* gồm những **nghiên cứu viên** trong Lĩnh vực Công nghệ thông tin của VAST, *A* là một tập con của *X*. Như vậy *A* là một tập hợp bình thường - gọi là tập hợp rõ hay tập rõ (Crisp set)
  - Tập hợp *B* những **nghiên cứu viên giỏi** trong Lĩnh vực Công nghệ thông tin của VAST. Ở đây *B* là một khái niệm không rõ bởi chưa định nghĩa thế nào là “giỏi”

.

# Khái quát về Logic mờ (6/12)

## Tập hợp mờ (Fuzzy Set) (2/8)

- Qua đó ta thấy nếu tập hợp (ký hiệu là $A$) chứa một số phần tử của tập $X$ thì $A$ là một tập con bình thường; nên có thể ký hiệu $A = \{x \in X | \ x \ satisfies \ some \ property\}$
- Tuy nhiên, do có một số phần tử trong tập hợp (ký hiệu là $B$) ***không hoàn toàn*** thuộc về $X$, nên buộc phải đưa thêm cho mỗi phần tử của tập hợp $B$ này một mức độ phụ thuộc là bao nhiêu (hay còn gọi là độ thuộc) của nó vào tập $X$.
- Khi đó, $B$ được viết như là tập hợp của một cặp $(x, \mu_B(x))$ bao gồm cả $x$ và mức độ phụ thuộc $\mu_B(x)$ của nó vào tập hợp $X$, trong đó $\mu_B(x) \in [0,1], \forall x \in X$ khi đó ta ký hiệu $B = \{(x, \mu_B(x) | x \in X\}$

# Khái quát về Logic mờ (7/12)

## Định nghĩa Tập hợp mờ (3/8)

- Cho $X$ là một ***không gian nền***, hay còn gọi là một ***vũ trụ*** (Universe) hay là ***tập tham chiếu***), $A$ là được gọi là ***tập mờ*** trên không gian nền $X$ nếu (và chỉ nếu) $A$ được xác định bởi $x \in X$ và hàm $\mu_A(x) \in [0,1]$.
- Khi đó tập mờ $A$ được ký hiệu theo những dạng khác nhau như sau:
  - Khi $X$ là tập các giá trị liên tục:

$$A = \left\{ \frac{\mu_A(x)}{x} \,|\, x \in X \right\}, \text{ hay } A = \left\{ (\mu_A(x), x) \,|\, x \in X \right\}, \text{ hay } A = \left\{ (x, \mu_A(x) \,|\, x \in X \right\}$$

# Khái quát về Logic mờ (8/12)

## Định nghĩa Tập hợp mờ (4/8)

- Khi $X$ là tập các giá trị rời rạc như là $\{x_1, x_2, \ldots x_n\}$, $A$ được ký hiệu là

$$A = \sum_{i=1}^{n} \frac{\mu_A(x)}{x}$$

- Nếu $X$ là vô hạn không đếm được, ký hiệu

$$A = \int \frac{\mu_A(x)}{x} dx$$

- Trong đó $\mu_A$ được gọi là ***hàm thuộc*** hay ***hàm thành viên*** (*Membership Function*) để chỉ mức độ phụ thuộc của các phần tử trong tập $X$ vào tập $A$

# Khái quát về Logic mờ (9/12)

## Ví dụ về tập hợp mờ (5/8)

- Cho một không gian nền hữu hạn *X* tương ứng với độ tuổi $X = \{14,25,40,60\}$. Một người được gọi là trẻ khi còn trong độ tuổi 25
- Khi đó, có thể xây dựng một tập mờ *A* gồm các độ tuổi trẻ như sau

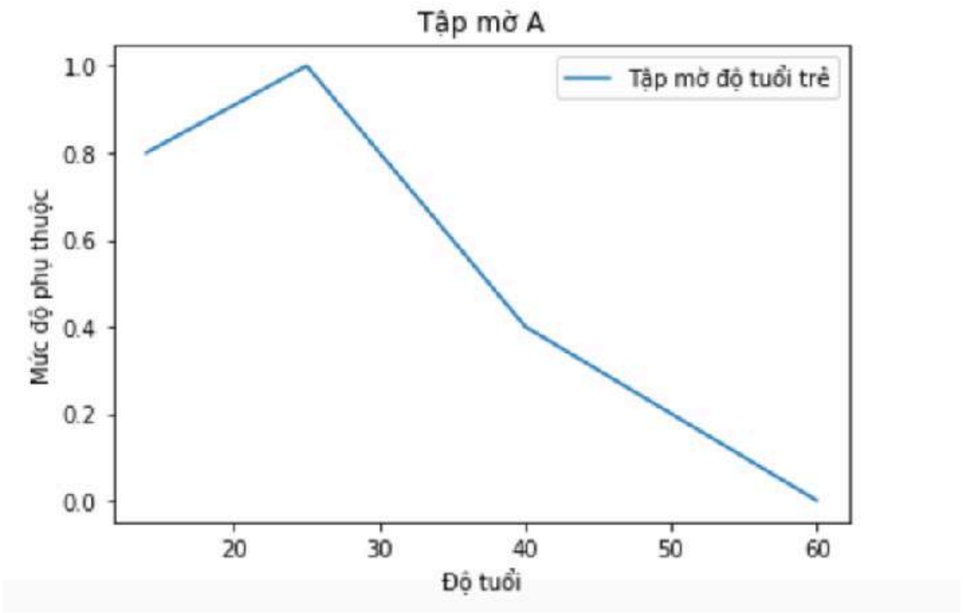
$$A = \frac{0.8}{14} + \frac{1}{25} + \frac{0.4}{40} + \frac{0}{60}$$

- Điều đó có nghĩa là giá trị thuộc của hàm thuộc $\mu_A(x)$ ứng với: *x* = 14 là 0.8, *x* = 25 là 1, *x* = 40 là 0.4 và *x* = 60 là 0

# Khái quát về Logic mờ (10/12)

## Ví dụ về tập hợp mờ (6/8)

- Cũng có thể ký hiệu:




$$A = \{(0.8,14), (1,25), (0.4,40), (0,60)\}$$

$$A = \{(14,0.8), (25,1), (40,0.4), (60,0)\}$$

$$A = \left\{ \frac{0.8}{4}, \frac{1}{25}, \frac{0.4}{40}, \frac{0}{60} \right\}$$

# Khái quát về Logic mờ (11/12)

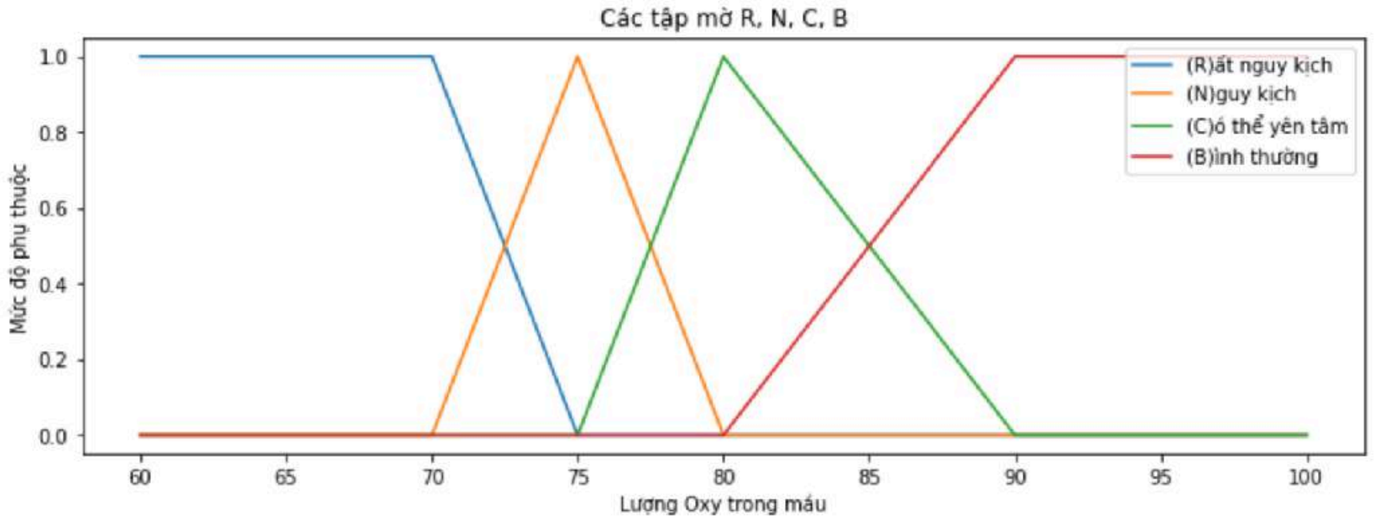
## Ví dụ về tập hợp mờ (7/8)

- Một ví dụ khác liên quan đến việc căn cứ vào kiến thức và ngữ cảnh trực quan để xây dựng các tập mờ.
- Chẳng hạn, khi xem xét lượng Oxy trong máu để nói lên mức độ nguy kịch của một người, có thể xây dựng 4 tập mờ $R$, $N$, $C$, $B$ như sau:
  - (R)ất nguy kịch (lượng Oxy xấp xỉ 70%): $R = \frac{1}{60} + \frac{1}{70} + \frac{0}{75} + \frac{0}{80} + \frac{0}{90} + \frac{0}{100}$
  - (N)guy kịch (trong khoảng gần 75%): $N = \frac{0}{60} + \frac{0}{70} + \frac{1}{75} + \frac{0}{80} + \frac{0}{90} + \frac{0}{100}$

# Khái quát về Logic mờ (12/12)

## Ví dụ về tập hợp mờ (8/8)

- (C)ó thể yên tâm (xấp xỉ 80%): $C = \frac{0}{60} + \frac{0}{70} + \frac{0}{75} + \frac{1}{80} + \frac{0}{90} + \frac{0}{100}$
- (B)ình thường (xung quanh 90%): $B = \frac{0}{60} + \frac{0}{70} + \frac{0}{75} + \frac{0}{80} + \frac{1}{90} + \frac{1}{100}$



# Dùng ngôn ngữ Python (1/7)

## Dùng gói phần mềm scikit-fuzzy

- Để sử dụng được các hàm và phương thức liên quan đến logic mờ  khi viết chương trình bằng Python, cần install thêm các gói thư viện ***scikit-fuzzy*** (https://scikit-fuzzy.readthedocs.io/en/latest/)
- Việc install được thực hiện bằng lệnh **pip** hoặc lệnh **python -m pip** tại dòng lệnh của hệ điều hành với tên gói tương ứng
  - pip install scikit-fuzzy  # fuzzy logic
  - pip install numpy  # tính toán số
  - pip install scipy  # thư việc các tính toán khoa học
  - pip install matplotlib  # để vẽ đồ thị
- Với Python3: dùng lệnh **pip3** hoặc **python3 -m pip**

# Dùng ngôn ngữ Python (2/7)

## Một số hàm thuộc thông dụng trong scikit-fuzzy

- Có một số hàm thuộc (hàm thành viên) thông dụng như
  - Hàm thuộc tam giác: trimf(x,abc)
  - Hàm thuộc hình thang: trapmf(x,abcd)
  - Hàm dạng chữ Z, chữ S: zmf(x,a,b), smf(x,a,b)
- Và một số hàm khác có thể tìm thấy trong module có tên **membership** tại https://scikit-fuzzy.readthedocs.io/en/latest/api/skfuzzy.membership.html

# Dùng ngôn ngữ Python (3/7)

## Các hàm thuộc có trong scikit-fuzzy

### Module: `membership` ¶

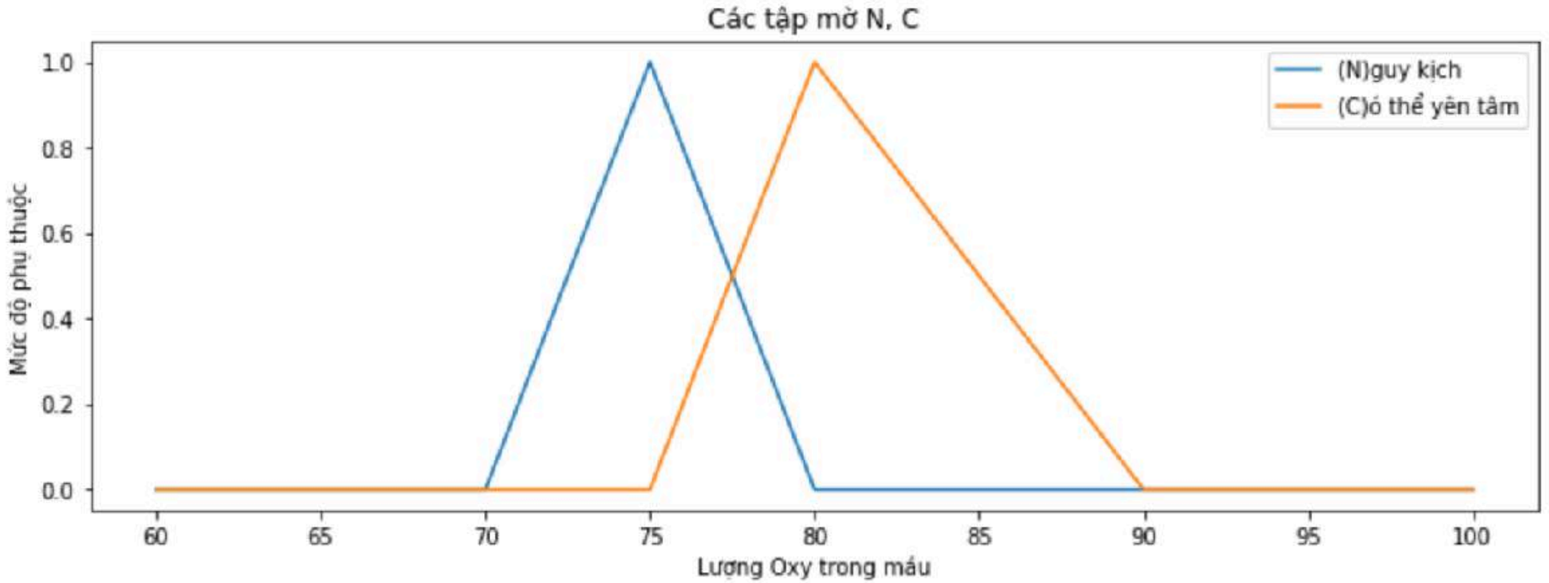
skfuzzy.membership : fuzzy membership function generators

| | |
|---|---|
| `skfuzzy.membership.dsigmf` (x, b1, c1, b2, c2) | Difference of two fuzzy sigmoid membership functions. |
| `skfuzzy.membership.gauss2mf` (x, mean1, ...) | Gaussian fuzzy membership function of two combined Gaussians. |
| `skfuzzy.membership.gaussmf` (x, mean, sigma) | Gaussian fuzzy membership function. |
| `skfuzzy.membership.gbellmf` (x, a, b, c) | Generalized Bell function fuzzy membership generator. |
| `skfuzzy.membership.piecemf` (x, abc) | Piecewise linear membership function (particularly used in FIRE filters). |
| `skfuzzy.membership.pimf` (x, a, b, c, d) | Pi-function fuzzy membership generator. |
| `skfuzzy.membership.psigmf` (x, b1, c1, b2, c2) | Product of two sigmoid membership functions. |
| `skfuzzy.membership.sigmf` (x, b, c) | The basic sigmoid membership function generator. |
| `skfuzzy.membership.smf` (x, a, b) | S-function fuzzy membership generator. |
| `skfuzzy.membership.trapmf` (x, abcd) | Trapezoidal membership function generator. |
| `skfuzzy.membership.trimf` (x, abc) | Triangular membership function generator. |
| `skfuzzy.membership.zmf` (x, a, b) | Z-function fuzzy membership generator. |

# Dùng ngôn ngữ Python (4/7)

## Một số ví dụ về hàm thuộc

- Hàm dạng tam giác, `trimf(x,v)`, trong đó $x$, $v$ là mảng 1 chiều; $x$ là không gian nền, $v$ có 3 phần tử $v = [a, b, c], with\ a \leq b \leq c$
- Tập mờ (N)guy kịch và (C)ó thể yên tâm trong ví dụ trước có hàm thuộc dạng tam giác.



# Dùng ngôn ngữ Python (5/7)

## Vẽ đồ thị hàm thuộc dạng tam giác

```
# import thư viện để sử dụng
from skfuzzy import trimf
import numpy as np
import matplotlib.pyplot as plt
```

```
# Tạo mảng một chiều cho X
# R và C là các tập mờ với hàm thuộc
# tam giác
X = np.array( [60,70,75,80,90,100] )
N = trimf( X, [70,75,80] )
C = trimf( X, [75,80,90] )
```

```
# Tạo đồ thị trong khung chiều rộng 12
# chiều cao 4
# Kết quả như hình vẽ trước
plt.figure( figsize=(12,4) )
plt.title( "Các tập mờ R, N, C, B" )
plt.plot( X, N, label="(N)guy kịch" )
plt.plot( X, C, label="(C)ó thể yên tâm" )
plt.xlabel( "Lượng Oxy trong máu" )
plt.ylabel( "Mức độ phụ thuộc" )
plt.legend( loc="upper right" )
plt.show()
```

# Dùng ngôn ngữ Python (6/7)

## Hàm thuộc dạng hình thang

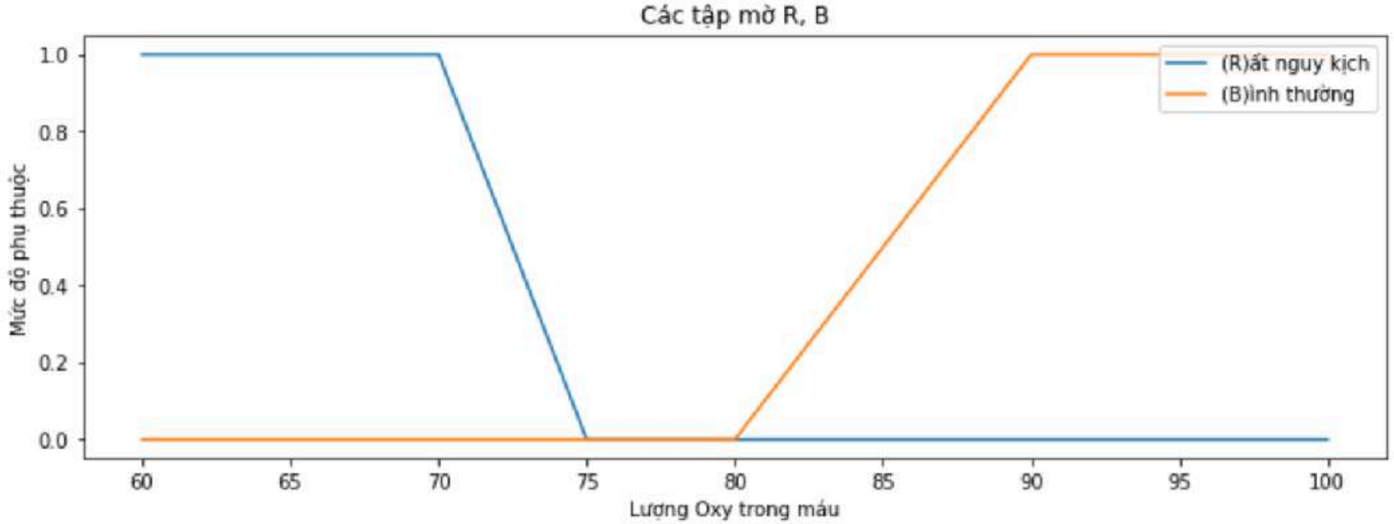
- Hàm dạng tam giác, `trapmf(x,v)`, trong đó *x*, *v* là mảng 1 chiều; *x* là không gian nền, *v* có 4 phần tử $v = [a, b, c, d], with\ a \leq b \leq c \leq d$
- Tập mờ (R)ất nguy kịch và (B)ình thường trong ví dụ trước có hàm thuộc dạng hình thang.

```
from skfuzzy import trapmf
# Có thể viết from skfuzzy import *
X = np.array( [60,70,75,80,90,100] )
R = trapmf( X, [60,60,70,75] )
B = trapmf( X, [80,90,100,100] )
```

```
plt.figure( figsize=(12,4) )
plt.title( "Các tập mờ R, B" )
plt.plot( X, R, label="(R)ất nguy kịch" )
plt.plot( X, B, label="(B)ình thường" )
plt.xlabel( "Lượng Oxy trong máu" )
plt.ylabel( "Mức độ phụ thuộc" )
plt.legend( loc="upper right" )
plt.show()
```

# Dùng ngôn ngữ Python (7/7)

## Vẽ đồ thị hàm thuộc dạng hình thang




- Ngoài các tập mờ dựa vào hàm thuộc có sẵn, chúng ta cũng có thể định nghĩa tập mờ bằng cách tạo mảng 1 chiều ứng với các giá trị lần lượt là độ thuộc tương ứng với các phần tử trong không gian nền

```
X = np.array( [14,25,40,60] )
A = np.array( [0.8,1,0.4,0] )
```

Với

$$A = \frac{0.8}{14} + \frac{1}{25} + \frac{0.4}{40} + \frac{0}{60}$$

# Điều khiển mờ

**Fuzzy Control**

langtv@gmail.com

- Hệ thống điều khiển mờ
- Xây dựng hệ thống điều khiển mờ tổng quát
- Giải quyết bài toán điều khiển mờ bằng Python

*Trong phần này chủ yếu trình bày cách thức dùng ngôn ngữ Python để giải quyết bài toán thực tế theo tư duy của con người.*

# Điều khiển mờ

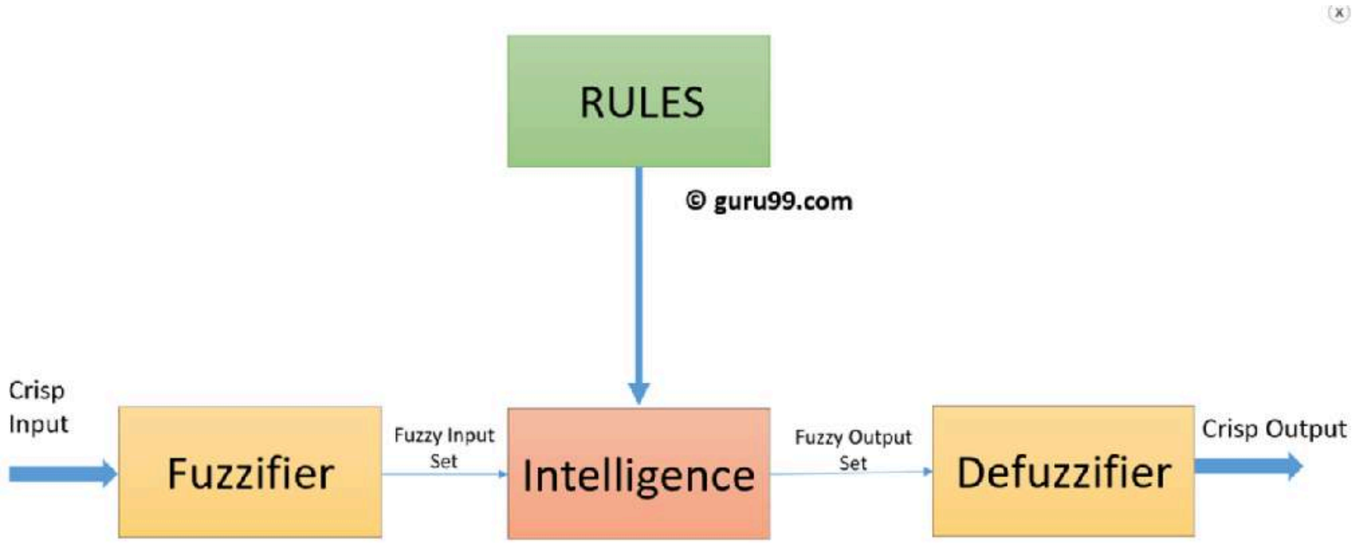
## Hệ thống điều khiển mờ (Fuzzy Control System)

- Hệ thống điều khiển mờ, hay hệ điều khiển logic mờ (FLC - Fuzzy Logic Control) là một hệ thống điều khiển dựa trên logic mờ bằng cách phân tích những trạng thái vật lý theo giá trị liên tục thuộc đoạn [0,1].
- Có thể coi FLC là một cách thông minh để điều khiển một tiến trình dựa trên các luật dạng IF - THEN.
- Điều khiển logic mờ là một cách tiếp cận heuristic mà có thể dễ dàng đưa kiến thức và các đặc trưng trong suy nghĩ của con người vào việc thiết kế các bộ điều khiển phi tuyến phức tạp, mà không cần thiết lập dưới dạng mô hình toán học hay thuật toán máy tính.

# Điều khiển mờ

## Sơ đồ hệ điều khiển mờ

- Bước đầu tiên trong quy trình xây dựng bộ điều khiển mờ là xác định các biến đầu vào và đầu ra của bộ điều khiển (là những dữ liệu rõ)
- Sau đó là các luật hay quy tắc điều khiển được xây dựng theo thuật ngữ của ngôn ngữ tự nhiên.
- Cuối cùng là giải mờ
- Sơ đồ của một hệ thống điều khiển mờ như hình



# Xây dựng một hệ thống điều khiển mờ

## Các giai đoạn tổng quát

- **Giai đoạn 1**: xác định các biến đầu vào và đầu ra của hệ điều khiển, cùng với các miền giá trị của chúng như là các không gian nền. Từ đó xác định đâu là tiền dề (Antecendent), đâu là kết quả (Consequent)
- **Giai đoạn 2**: dựa trên các biến này và ngôn ngữ tự nhiên để xây dựng các tập mờ tương ứng sao cho thể hiện được các ý tưởng chính của hệ điều khiển
- **Giai đoạn 3**: xây dựng những luật if then mờ thể hiện mối quan hệ giữa đầu vào và đầu ra trên các biến ngôn ngữ này. Chẳng hạn, thông qua việc khai khoáng dữ liệu (data mining) để tìm luật kết hợp với một ngưỡng nào đó.
- **Giai đoạn 4**: xây dựng hệ thống để tính toán
- **Giai đoạn 5**: giải mờ và xuất kết quả

# Dùng ngôn ngữ Python

**Bài toán 1:** Xây dựng một bộ điều chỉnh máy lạnh tự động căn cứ vào số người có trong phòng và nhiệt độ trong phòng và nhiệt độ bên ngoài phòng

- **Giai đoạn 1**: Dữ liệu nhập và miền giá trị để tạo bộ tiền đề và kết quả
  - Nhập
    - Số người trong phòng $N = [1,10] \in \mathbb{N}$
    - Nhiệt độ trong phòng $T_{in} = [18,30] \in \mathbb{N}$
    - Nhiệt độ ngoài trời $T_{out} = [20,40] \in \mathbb{N}$
  - Xuất:
    - Mức điều chỉnh $L = [18,27] \in \mathbb{N}$

```
# Cách viết import kiểu khác
import skfuzzy as fz
import numpy as np
from skfuzzy import control as ctrl

N    = ctrl.Antecedent([1,2,3,4,5,6,7,8,9,10],"So nguoi")
Tin  = ctrl.Antecedent(np.arange(18,31),"Nhiet do trong phong")
Tout = ctrl.Antecedent(np.arange(20,41),"Nhiet do ngoai troi")
L    = ctrl.Consequent(np.arange(18,27),"Muc dieu chinh")
```

# Dùng ngôn ngữ Python

- Những phương thức như **Antecedent()**, **Consequent()** có trong module có tên control của gói scikit-fuzzy nên cần phải import bởi lệnh **from skfuzzy import control**
- Ngoài ra, do muốn dùng chữ ctrl thay cho control nên bổ sung thêm **as ctrl** trong lệnh import này.
- Trong module này còn có các phương thức khác như bảng


- Trong module này còn có các phương thức khác như bảng


## Module: `control`

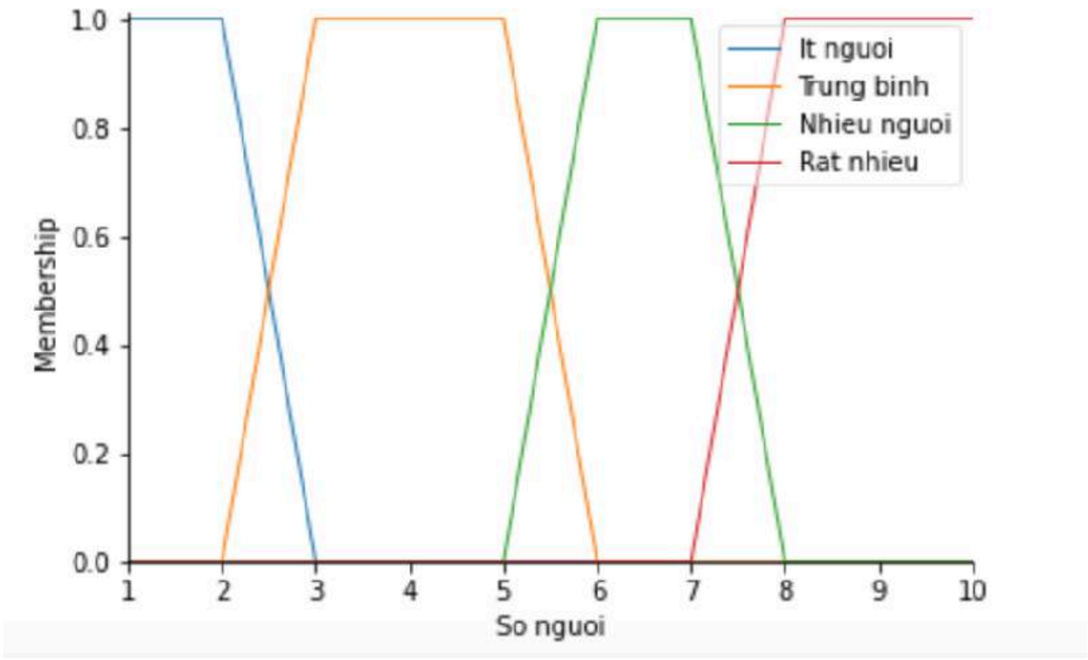
skfuzzy.control subpackage, providing a high-level API for fuzzy system design.

| | |
|---|---|
| `skfuzzy.control.Antecedent` (universe, label) | Antecedent (input/sensor) variable for a fuzzy control system. |
| `skfuzzy.control.Consequent` (universe, label) | Consequent (output/control) variable for a fuzzy control system. |
| `skfuzzy.control.ControlSystem` ([rules]) | Base class to contain a Fuzzy Control System. |
| `skfuzzy.control.ControlSystemSimulation` (...) | Calculate results from a ControlSystem. |
| `skfuzzy.control.Rule` ([antecedent, ...]) | Rule in a fuzzy control system, connecting antecedent(s) to consequent(s). |

# Bài toán 1

- **Giai đoạn 2**: xây dựng các tập mờ
  - ***Số người***: có 4 tập mờ
    - Ít người
    - Trung bình
    - Nhiều người
    - Rất nhiều

```
N["It nguoi"] = fz.trapmf( N.universe,[1,1,2,3] )
N["Trung binh"] = fz.trapmf( N.universe,[2,3,5,6] )
N["Nhieu nguoi"] = fz.trapmf( N.universe,[5,6,7,8] )
N["Rat nhieu"] = fz.trapmf( N.universe,[7,8,10,10] )
N.view()
```

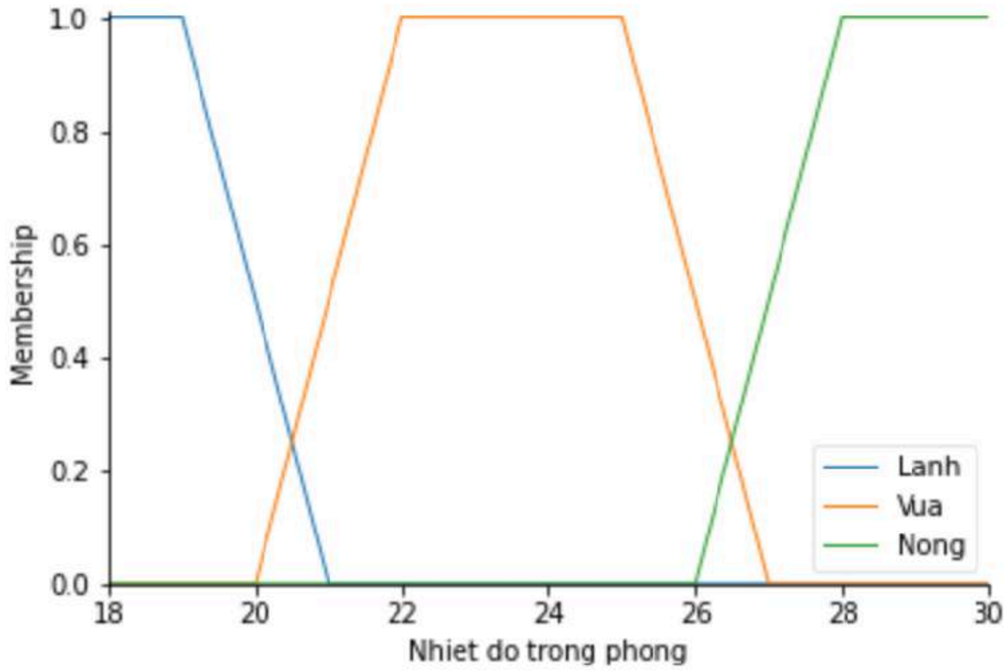


# Bài toán 1

## Tập mờ với hàm thuộc

- ***Nhiệt độ trong phòng***: có 3 tập mờ:
  - Lạnh
  - Vừa
  - Nóng

```
Tin["Lanh"] = fz.trapmf( Tin.universe,[18,18,19,21] )
Tin["Vua"] = fz.trapmf( Tin.universe,[20,22,25,27] )
Tin["Nong"] = fz.trapmf( Tin.universe,[26,28,30,30] )
Tin.view()
```

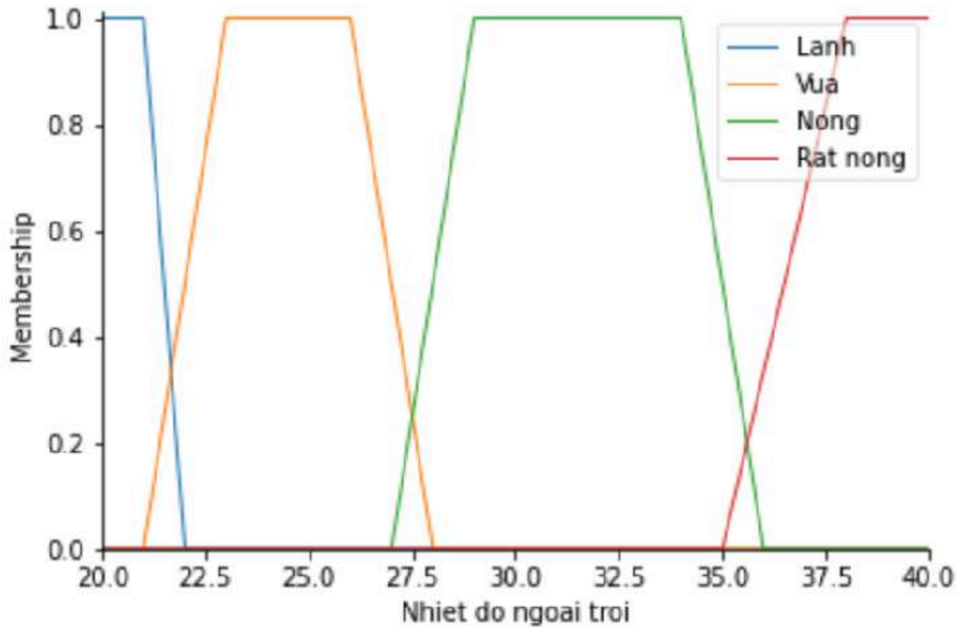


# Bài toán 1

## Biểu diễn tập mờ

- ***Nhiệt độ ngoài trời***: có 4 tập mờ:
  - Lạnh
  - Vừa
  - Nóng
  - Rất nóng

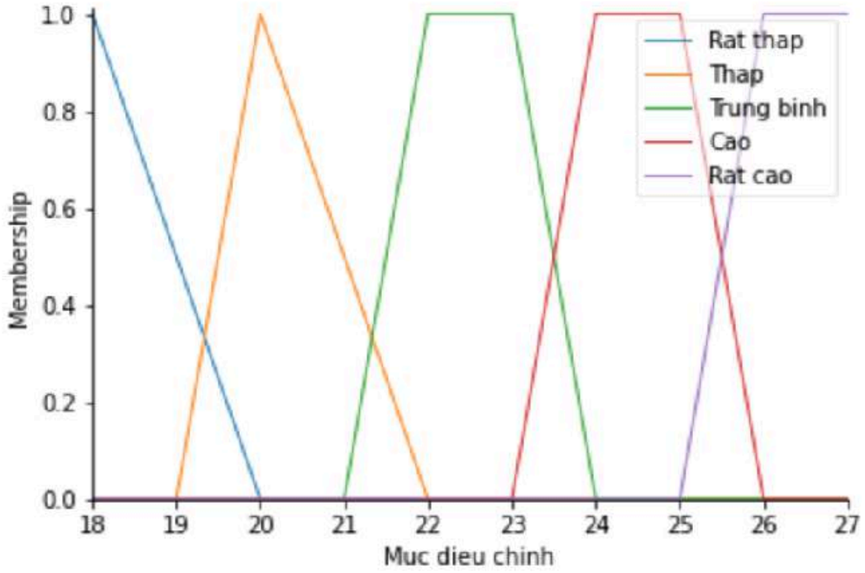
```
Tout["Lanh"] = fz.trapmf( Tout.universe,[20,20,21,22] )
Tout["Vua"] = fz.trapmf( Tout.universe,[21,23,26,28] )
Tout["Nong"] = fz.trapmf( Tout.universe,[27,29,34,36] )
Tout["Rat nong"] = fz.trapmf( Tout.universe,[35,38,40,40] )
Tout.view()
```



# Bài toán 1

- ***Mức điều chỉnh***: có 5 tập mờ:
  - Rất thấp
  - Thấp
  - Trung bình
  - Cao
  - Rất cao

```
L["Rat thap"] = fz.trimf( L.universe,[18,18,20] )
L["Thap"] = fz.trimf( L.universe,[19,20,22] )
L["Trung binh"] = fz.trapmf( L.universe,[21,22,23,24] )
L["Cao"] = fz.trimf( L.universe,[23,24,25,26] )
L["Rat cao"] = fz.trapmf( L.universe,[25,26,27,27] )
L.view()
```



# Bài toán 1

## Luật mờ (Fuzzy Rules)

- **Giai đoạn 3**: Luật mờ dạng if then được rút ra theo quy tắc của "chuyên gia" như sau:
  - Nếu nhiệt độ càng nóng thì mức điều chỉnh phải thấp.
  - Nhiệt độ (cả bên ngoài lẫn bên trong) tỷ lệ thuận với số người có trong phòng

| N | $T_{in}$ | $T_{out}$ | L |
|---|---|---|---|
| It nguoi | Lanh | Lanh | Rat cao |
| | Lanh | Vua | Rat cao |
| | Lanh | Nong | Cao |
| | Lanh | Rat nong | Cao |
| It nguoi | Vua | Lanh | Rat cao |
| | Vua | Vua | Cao |
| | Vua | Nong | Cao |
| | Vua | Rat nong | Trung binh |

| N | $T_{in}$ | $T_{out}$ | L |
|---|---|---|---|
| It nguoi | Nong | Lanh | Cao |
| | Nong | Vua | Trung binh |
| | Nong | Nong | Trung binh |
| | Nong | Rat nong | Thap |
| Trung binh | Lanh | Lanh | Rat cao |
| | Lanh | Vua | Rat cao |
| | Lanh | Nong | Cao |
| | Lanh | Rat nong | Trung binh |

# Bài toán 1

| N | $T_{in}$ | $T_{out}$ | L |
|---|---|---|---|
| Trung binh | Vua | Lanh | Cao |
| | Vua | Vua | Trung binh |
| | Vua | Nong | Thap |
| | Vua | Rat nong | Thap |
| Trung binh | Nong | Lanh | Cao |
| | Nong | Vua | Trung binh |
| | Nong | Nong | Thap |
| | Nong | Rat nong | Thap |
| Nhieu nguoi | Lanh | Lanh | Cao |
| | Lanh | Vua | Trung binh |
| | Lanh | Nong | Trung binh |
| | Lanh | Rat nong | Thap |
| Nhieu nguoi | Vua | Lanh | Trung binh |
| | Vua | Vua | Trung binh |
| | Vua | Nong | Thap |
| | Vua | Rat nong | Thap |

| N | $T_{in}$ | $T_{out}$ | L |
|---|---|---|---|
| Nhieu nguoi | Nong | Lanh | Trung binh |
| | Nong | Vua | Thap |
| | Nong | Nong | Thap |
| | Nong | Rat nong | Rat thap |
| Rat nhieu | Lanh | Lanh | Trung binh |
| | Lanh | Vua | Trung binh |
| | Lanh | Nong | Thap |
| | Lanh | Rat nong | Thap |
| Rat nhieu | Vua | Lanh | Trung binh |
| | Vua | Vua | Thap |
| | Vua | Nong | Thap |
| | Vua | Rat nong | Rat thap |
| Rat nhieu | Nong | Lanh | Thap |
| | Nong | Vua | Thap |
| | Nong | Nong | Rat thap |
| | Nong | Rat nong | Rat thap |

# Dùng ngôn ngữ Python

- Bằng ngôn ngữ Python (không phải ngôn ngữ tự  nhiên như bình thường) các luật này được viết dưới dạng như sau:

R = ctrl.Rule( N[""] & Tin[""] & Tout[""], L[""] )

```
R1_2   = ctrl.Rule( N["It nguoi"] & Tin["Lanh"] & (Tout["Lanh"] | Tout["Vua"]), L["Rat cao"] )
R3_4   = ctrl.Rule( N["It nguoi"] & Tin["Lanh"] & (Tout["Nong"] | Tout["Rat nong"]), L["Cao"] )
R5     = ctrl.Rule( N["It nguoi"] & Tin["Vua"] & Tout["Lanh"], L["Rat cao"] )
R6_7   = ctrl.Rule( N["It nguoi"] & Tin["Vua"] & (Tout["Vua"] | Tout["Nong"]), L["Cao"] )
R8     = ctrl.Rule( N["It nguoi"] & Tin["Vua"] & Tout["Rat nong"], L["Trung binh"] )
R9     = ctrl.Rule( N["It nguoi"] & Tin["Nong"] & Tout["Lanh"], L["Cao"] )
R10_11 = ctrl.Rule( N["It nguoi"] & Tin["Nong"] & (Tout["Vua"] | Tout["Nong"]), L["Trung binh"] )
R12    = ctrl.Rule( N["It nguoi"] & Tin["Nong"] & Tout["Rat nong"], L["Thap"] )
```

# Dùng ngôn ngữ Python

```
R13_14 = ctrl.Rule( N["Trung binh"] & Tin["Lanh"] & (Tout["Lanh"] | Tout["Vua"]), L["Rat cao"] )
R15    = ctrl.Rule( N["Trung binh"] & Tin["Lanh"] & Tout["Nong"], L["Cao"] )
R16    = ctrl.Rule( N["Trung binh"] & Tin["Lanh"] & Tout["Rat nong"], L["Trung binh"] )
R17    = ctrl.Rule( N["Trung binh"] & Tin["Vua"] & Tout["Lanh"], L["Cao"] )
R18    = ctrl.Rule( N["Trung binh"] & Tin["Vua"] & Tout["Vua"], L["Trung binh"] )
R19_20 = ctrl.Rule( N["Trung binh"] & Tin["Vua"] & (Tout["Nong"] | Tout["Rat nong"]), L["Thap"] )
R21    = ctrl.Rule( N["Trung binh"] & Tin["Nong"] & Tout["Lanh"], L["Cao"] )
R22    = ctrl.Rule( N["Trung binh"] & Tin["Nong"] & Tout["Vua"], L["Trung binh"] )
R23_24 = ctrl.Rule( N["Trung binh"] & Tin["Nong"] & (Tout["Nong"] | Tout["Rat nong"]), L["Thap"] )
```

```
R25    = ctrl.Rule( N["Nhieu nguoi"] & Tin["Lanh"] & Tout["Lanh"], L["Cao"] )
R26_27 = ctrl.Rule( N["Nhieu nguoi"] & Tin["Lanh"] & (Tout["Vua"] | Tout["Nong"] ), L["Trung binh"])
R28    = ctrl.Rule( N["Nhieu nguoi"] & Tin["Lanh"] & Tout["Rat nong"], L["Thap"] )
R29_30 = ctrl.Rule( N["Nhieu nguoi"] & Tin["Vua"] & (Tout["Lanh"] | Tout["Vua"] ), L["Trung binh"] )
R31_32 = ctrl.Rule( N["Nhieu nguoi"] & Tin["Vua"] & (Tout["Nong"] | Tout["Rat nong"]), L["Thap"] )
R33    = ctrl.Rule( N["Nhieu nguoi"] & Tin["Nong"] & Tout["Lanh"], L["Trung binh"] )
R34_35 = ctrl.Rule( N["Nhieu nguoi"] & Tin["Nong"] & (Tout["Vua"] | Tout["Nong"]), L["Thap"] )
R36    = ctrl.Rule( N["Nhieu nguoi"] & Tin["Nong"] & Tout["Rat nong"], L["Rat thap"] )
```

# Dùng ngôn ngữ Python

```
R37_38   = ctrl.Rule( N["Rat nhieu"] & Tin["Lanh"] & (Tout["Lanh"] | Tout["Vua"]), L["Trung binh"])
R39_40   = ctrl.Rule( N["Rat nhieu"] & Tin["Lanh"] & (Tout["Nong"] | Tout["Rat nong"]), L["Thap"] )
R41      = ctrl.Rule( N["Rat nhieu"] & Tin["Vua"] & Tout["Lanh"], L["Trung binh"] )
R42_43   = ctrl.Rule( N["Rat nhieu"] & Tin["Vua"] & (Tout["Vua"] | Tout["Nong"]), L["Thap"] )
R44      = ctrl.Rule( N["Rat nhieu"] & Tin["Vua"] & Tout["Rat nong"], L["Rat thap"] )
R45_46   = ctrl.Rule( N["Rat nhieu"] & Tin["Nong"] & (Tout["Lanh"] | Tout["Vua"]), L["Thap"] )
R47_48   = ctrl.Rule( N["Rat nhieu"] & Tin["Nong"] & (Tout["Nong"] | Tout["Rat nong"]), L["Rat thap"])
```

- **Giai đoạn 4**: xây dựng hệ thống mô phỏng dựa trên các luật đã có

```
rules = [R1_2,R3_4,R5,R6_7,R8,R9,R10_11,R12,R13_14,R15,R16,R17,R18,R19_20,R21,R22,R23_24,
R25,R26_27,R28,R29_30,R31_32,R33,R34_35,R36,R37_38,R39_40,R41,R42_43,R44,R45_46,R47_48]


system = ctrl.ControlSystemSimulation( ctrl.ControlSystem(rules) )
```

# Dùng ngôn ngữ Python

- **Giai đoạn 5**: nhập dữ liệu đầu vào là 3 yếu tố: $L$, $T_{in}$, $T_{out}$ sau đó giải mờ và xuất kết quả là $L$.

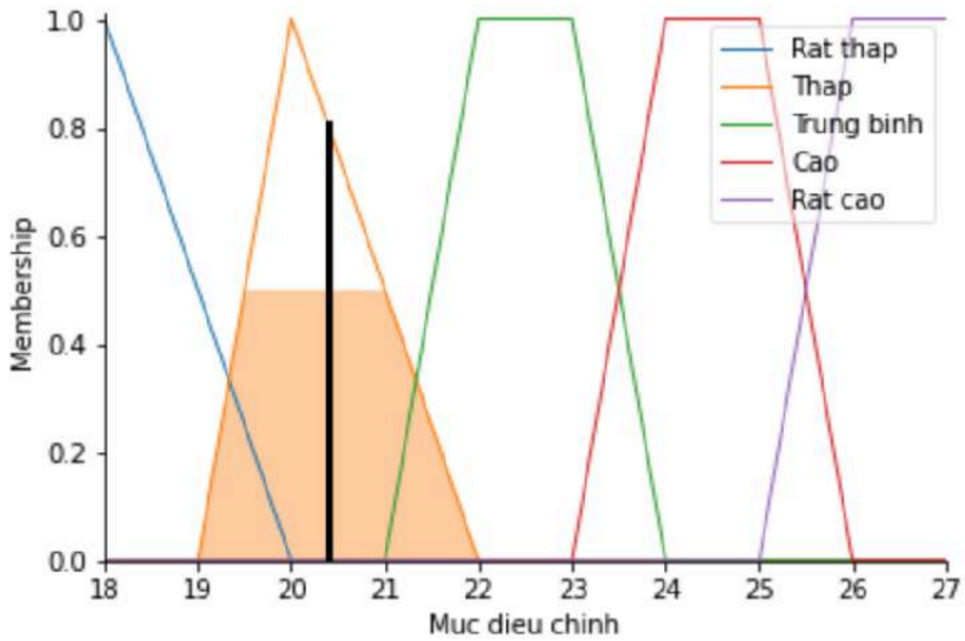
```
iN = int( input("Số người: ") )
iTin = float( input("Nhiệt độ trong phòng: ") )
iTout = float( input("Nhiệt độ bên ngoài: ") )


system.input["So nguoi"] = iN
system.input["Nhiet do trong phong"] = iTin
system.input["Nhiet do ngoai troi"] = iTout
```

```
system.compute()
oL = system.output["Muc dieu chinh"]
L.view( system )


print( "Với %d người ở trong phòng, nhiệt độ trong phòng
là %.1f và ngoài trời là %.1f thì mức điều chỉnh là %d" %
(iN,iTin,iTout,oL) )
```

# Bài toán 1

- Nhập dữ liệu, giả sử với $L$, $T_{in}$, $T_{out}$

```
Số người: 4
Nhiệt độ trong phòng: 27
Nhiệt độ bên ngoài: 30
```




- Kết quả

Với 4 người ở trong phòng, nhiệt độ trong phòng là 27.0 và ngoài trời là 30.0 thì mức điều chỉnh là 20

# Dùng ngôn ngữ Python

**Bài toán 2:** Dự báo mức độ lây lan của bệnh viêm phổi Wuhan sẽ lây lan ra (Spread) sao trên các vùng địa lý căn cứ vào số người nhiễm bệnh (Confirmed Cases), số người tử vong và nguy kịch (Deaths & Serious Critical) cũng như mức độ tiếp giáp với vùng địa lý (Border) có nguy cơ cao.

- **Giai đoạn 1**: dữ liệu
  - Dữ liệu nhập:
    - Tổng số người nhiễm bệnh (dương tính): $X_C = [1,100000] \in \mathbb{N}$
    - Số người đã tử vong: $X_D = [1,10000] \in \mathbb{N}$
    - Mức độ tiếp giáp: $X_B = \{1,2,3\}$
  - Kết quả
    - Mức độ lây lan: $X_S = [1,10] \in \mathbb{N}$

```
# Xây dựng 4 không gian nền tương ứng
XC = np.arange(1,100001)
XD = np.arange(1,10001)
XB = [1,2,3]
XS = np.arange(1,11)
```

# Bài toán 2

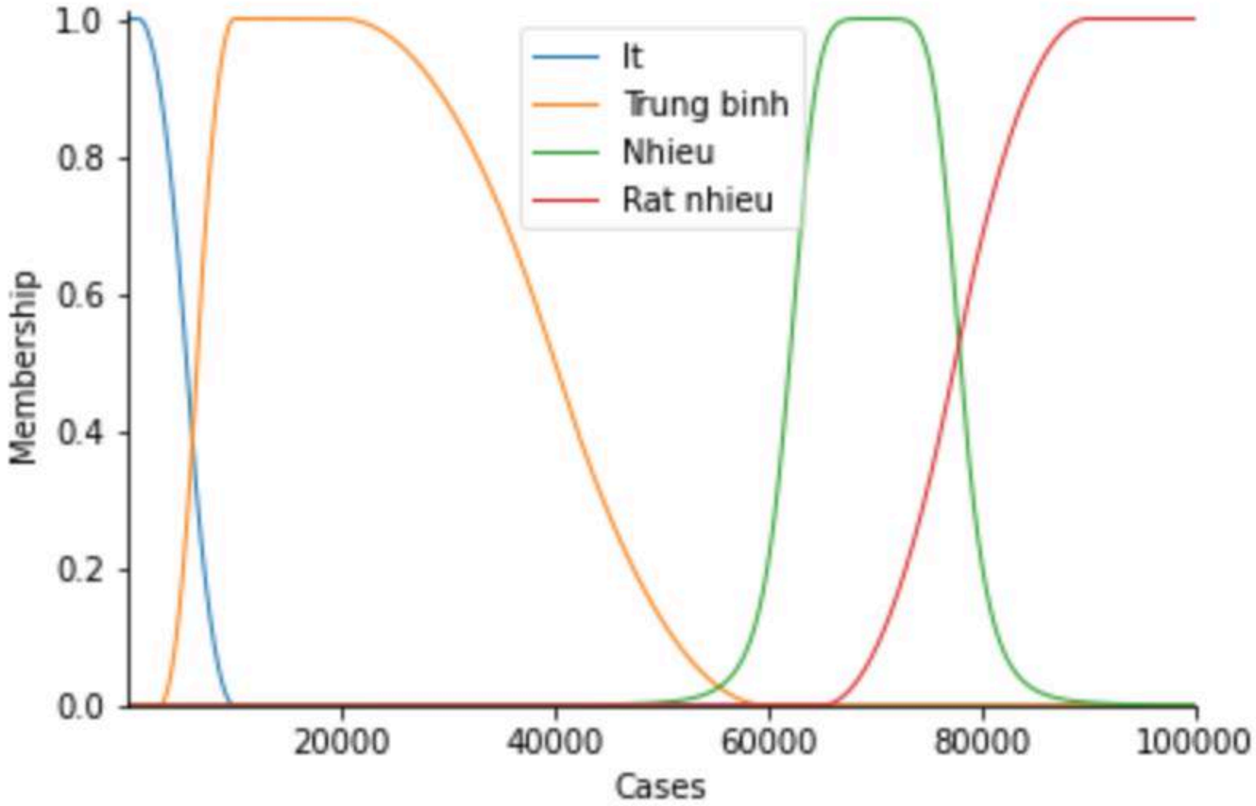
```
# Định nghĩa tiền đề và kết luận dựa trên các biến ngôn ngữ
C = ctrl.Antecedent( XC,"Cases" )
D = ctrl.Antecedent( XD,"Deaths" )
B = ctrl.Antecedent( XB,"Border" )
S = ctrl.Consequent( XS,"Spread" )
```

- **Giai đoạn 2**: tập mờ
  - ***Số người nhiễm***
    - Ít
    - Trung bình
    - Nhiều
    - Rất nhiều

```
C["It"] = fz.zmf( C.universe,1000,10000 )
C["Trung binh"] = fz.pimf( C.universe,3000,10000,20000,60000 )
C["Nhieu"] = fz.gbellmf( C.universe,8000,3,70000 )
C["Rat nhieu"] = fz.smf( C.universe,65000,90000 )
C.view()
```

# Bài toán 2

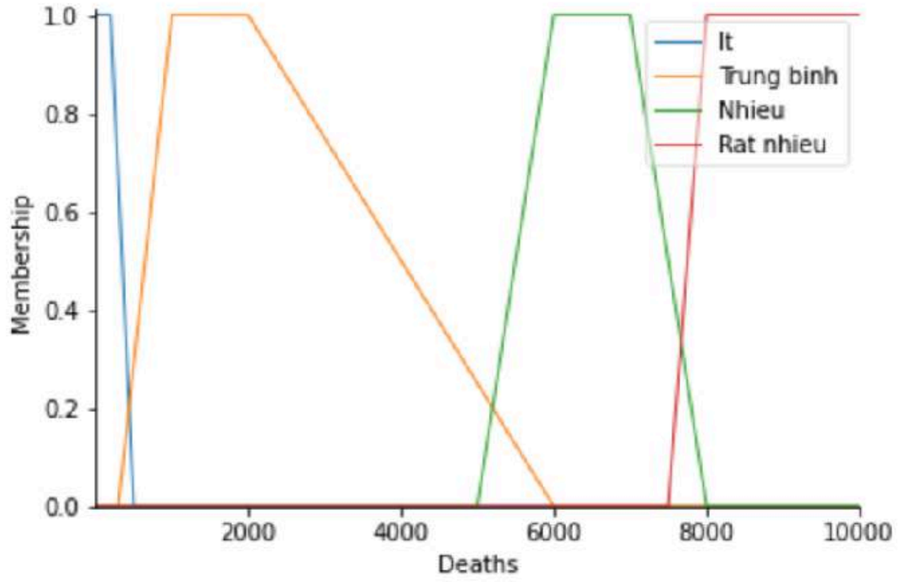
- Ở đây có hàm thành viên dạng:
  - Chữ Z: zmf()
  - Chữ Π: pimf()
  - Quả chuông: gbellmf()
  - Chữ S: smf()



# Bài toán 2

- ***Số người tử vong***
  - Ít
  - Trung bình
  - Nhiều
  - Rất nhiều

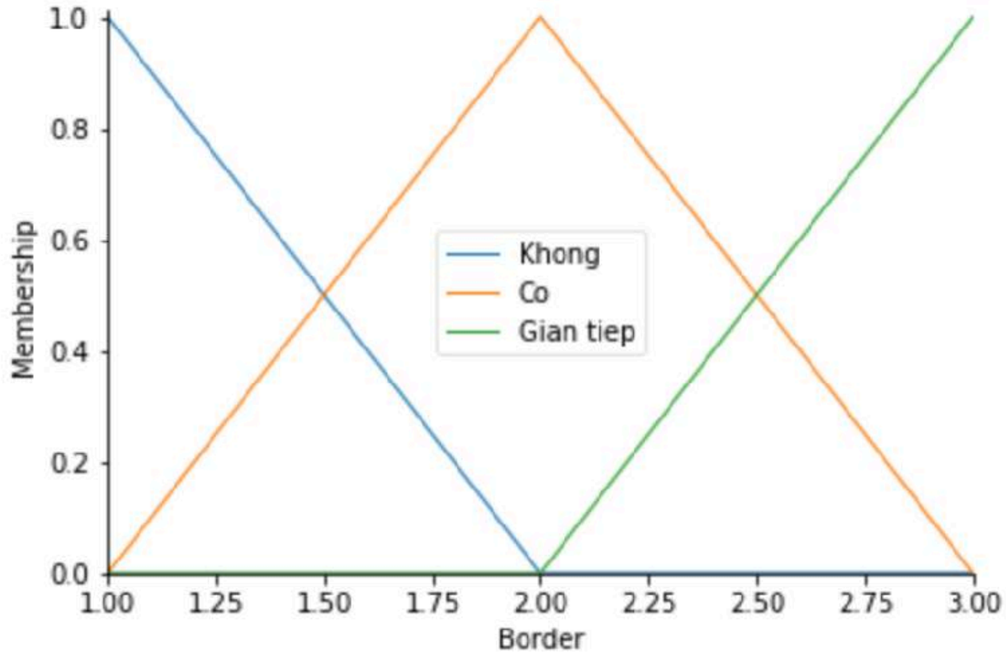
```
D["It"] = fz.trapmf( D.universe,[1,1,200,500] )
D["Trung binh"] = fz.trapmf( D.universe,[300,1000,2000,6000] )
D["Nhieu"] = fz.trapmf( D.universe,[5000,6000,7000,8000] )
D["Rat nhieu"] = fz.trapmf( D.universe,[7500,8000,10000,10000] )
D.view()
```



# Bài toán 2

- **Mức độ tiếp giáp**
  - Không
  - Có
  - Gián tiếp

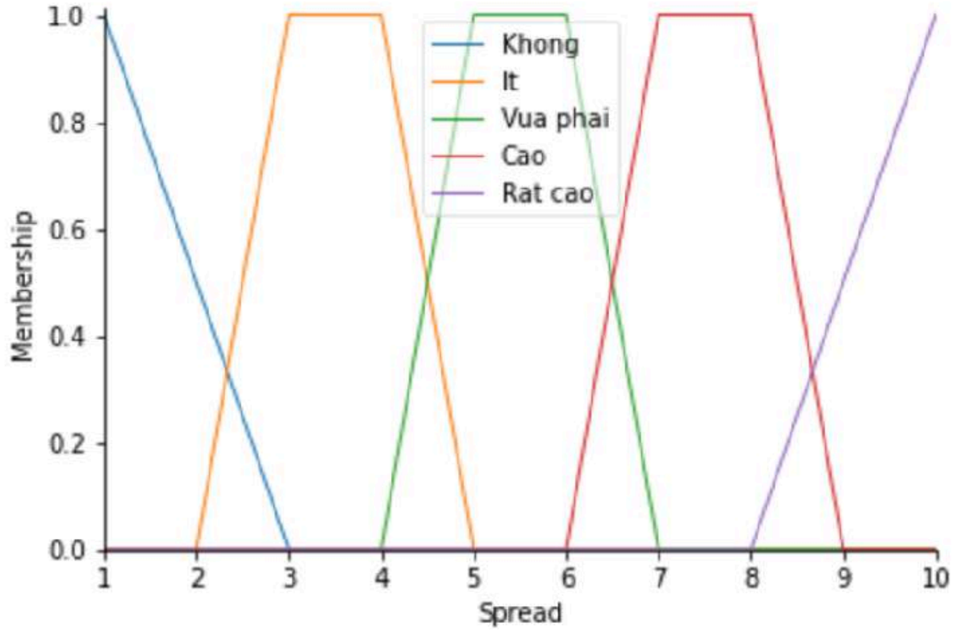
```
B["Khong"] = fz.trimf( B.universe,[1,1,2] )
B["Co"] = fz.trimf( B.universe,[1,2,3] )
B["Gian tiep"] = fz.trimf( B.universe,[2,3,3] )
B.view()
```



# Bài toán 2

- **Mức độ lây lan**
  - Không
  - Ít
  - Vừa phải
  - Cao
  - Rất cao

```
S["Khong"] = fz.trimf( S.universe,[1,1,3] )
S["It"] = fz.trapmf( S.universe,[2,3,4,5] )
S["Vua phai"] = fz.trapmf( S.universe,[4,5,6,7] )
S["Cao"] = fz.trapmf( S.universe,[6,7,8,9] )
S["Rat cao"] = fz.trimf( S.universe,[8,10,10] )
S.view()
```



# Bài toán 2

## Luật mờ (Fuzzy Rules)

- **Giai đoạn 3**: Luật mờ
  - Số người nhiễm, hoặc số người tử vong càng nhiều thì mức độ lây lan càng cao
  - Tiếp giáp với vùng dịch thì dễ lây nhiễm

| C | D | B | S |
|---|---|---|---|
| It | It | Khong | Khong |
| | It | Co | It |
| | It | Gian tiep | Khong |
| It | Trung binh | Khong | Khong |
| | Trung binh | Co | It |
| | Trung binh | Gian tiep | It |
| It | Nhieu | Khong | Vua phai |
| | Nhieu | Co | Vua phai |
| | Nhieu | Gian tiep | Vua phai |

| C | D | B | S |
|---|---|---|---|
| It | Rat nhieu | Khong | Vua phai |
| | Rat nhieu | Co | Vua phai |
| | Rat nhieu | Gian tiep | Vua phai |
| Trung binh | It | Khong | Khong |
| | It | Co | It |
| | It | Gian tiep | It |
| Trung binh | Trung binh | Khong | It |
| | Trung binh | Co | Vua phai |
| | Trung binh | Gian tiep | Vua phai |

# Bài toán 2

| C | D | B | S |
|---|---|---|---|
| Trung binh | Nhieu | Khong | Vua phai |
| | Nhieu | Co | Cao |
| | Nhieu | Gian tiep | It |
| Trung binh | Rat nhieu | Khong | Vua phai |
| | Rat nhieu | Co | Cao |
| | Rat nhieu | Gian tiep | Vua phai |
| Nhieu | It | Khong | Vua phai |
| | It | Co | Vua phai |
| | It | Gian tiep | It |
| Nhieu | Trung binh | Khong | Vua phai |
| | Trung binh | Co | Cao |
| | Trung binh | Gian tiep | It |
| Nhieu | Nhieu | Khong | It |
| | Nhieu | Co | Cao |
| | Nhieu | Gian tiep | Vua phai |

| C | D | B | S |
|---|---|---|---|
| Nhieu | Rat nhieu | Khong | Cao |
| | Rat nhieu | Co | Rat cao |
| | Rat nhieu | Gian tiep | Cao |
| Rat nhieu | It | Khong | Vua phai |
| | It | Co | Vua phai |
| | It | Gian tiep | Vua phai |
| Rat nhieu | Trung binh | Khong | Vua phai |
| | Trung binh | Co | Cao |
| | Trung binh | Gian tiep | Vua phai |
| Rat nhieu | Nhieu | Khong | Cao |
| | Nhieu | Co | Rat cao |
| | Nhieu | Gian tiep | Cao |
| Rat nhieu | Rat nhieu | Khong | Cao |
| | Rat nhieu | Co | Rat cao |
| | Rat nhieu | Gian tiep | Cao |

# Dùng ngôn ngữ Python

- Để phủ khắp, số luật cần có $4 \times 4 \times 3 = 48$ luật như trên

```
R1 = ctrl.Rule( C["It"] & D["It"] & (B["Khong"]), S["Khong"] )
R2 = ctrl.Rule( C["It"] & D["It"] & (B["Co"]), S["It"] )
R3 = ctrl.Rule( C["It"] & D["It"] & (B["Gian tiep"]), S["Khong"] )
R4 = ctrl.Rule( C["It"] & D["Trung binh"] & (B["Khong"]), S["Khong"] )
R5_6 = ctrl.Rule( C["It"] & D["Trung binh"] & (B["Co"] | B["Gian tiep"]), S["It"] )
R7_8_9 = ctrl.Rule( C["It"] & D["Nhieu"] & (B["Khong"] | B["Co"] | B["Gian tiep"]), S["Vua phai"] )
R10_11_12 = ctrl.Rule( C["It"] & D["Rat nhieu"] & (B["Khong"]|B["Co"]|B["Gian tiep"]), S["Vua phai"])
R13 = ctrl.Rule( C["Trung binh"] & D["It"] & (B["Khong"]), S["Khong"] )
R14_15 = ctrl.Rule( C["Trung binh"] & D["It"] & (B["Co"] | B["Gian tiep"]), S["It"] )
R16 = ctrl.Rule( C["Trung binh"] & D["Trung binh"] & (B["Khong"]), S["It"] )
R17_18 = ctrl.Rule( C["Trung binh"] & D["Trung binh"] & (B["Co"] | B["Gian tiep"]), S["Vua phai"] )
R19 = ctrl.Rule( C["Trung binh"] & D["Nhieu"] & (B["Khong"]), S["Vua phai"] )
R20 = ctrl.Rule( C["Trung binh"] & D["Nhieu"] & (B["Co"]), S["Cao"] )
R21 = ctrl.Rule( C["Trung binh"] & D["Nhieu"] & (B["Gian tiep"]), S["It"] )
```

```
R22 = ctrl.Rule( C["Trung binh"] & D["Rat nhieu"] & (B["Khong"]), S["Vua phai"] )
R23 = ctrl.Rule( C["Trung binh"] & D["Rat nhieu"] & (B["Co"]), S["Cao"] )
R24 = ctrl.Rule( C["Trung binh"] & D["Rat nhieu"] & (B["Gian tiep"]), S["Vua phai"] )
R25_26 = ctrl.Rule( C["Nhieu"] & D["It"] & (B["Khong"] | B["Co"]), S["Vua phai"] )
R27 = ctrl.Rule( C["Nhieu"] & D["It"] & (B["Gian tiep"]), S["It"] )
R28 = ctrl.Rule( C["Nhieu"] & D["Trung binh"] & (B["Khong"]), S["Vua phai"] )
R29 = ctrl.Rule( C["Nhieu"] & D["Trung binh"] & (B["Co"]), S["Cao"] )
R30 = ctrl.Rule( C["Nhieu"] & D["Trung binh"] & (B["Gian tiep"]), S["It"] )
R31 = ctrl.Rule( C["Nhieu"] & D["Nhieu"] & (B["Khong"]), S["It"] )
R32 = ctrl.Rule( C["Nhieu"] & D["Nhieu"] & (B["Co"]), S["Cao"] )
R33 = ctrl.Rule( C["Nhieu"] & D["Nhieu"] & (B["Gian tiep"]), S["Vua phai"] )
R34 = ctrl.Rule( C["Nhieu"] & D["Rat nhieu"] & (B["Khong"]), S["Cao"] )
R35 = ctrl.Rule( C["Nhieu"] & D["Rat nhieu"] & (B["Co"]), S["Rat cao"] )
R36 = ctrl.Rule( C["Nhieu"] & D["Rat nhieu"] & (B["Gian tiep"]), S["Cao"] )
R37_38_39 = ctrl.Rule( C["Rat nhieu"] & D["It"] & (B["Khong"] | B["Co"]|B["Gian tiep"]), S["Vua phai"])
R40 = ctrl.Rule( C["Rat nhieu"] & D["Trung binh"] & (B["Khong"]), S["Vua phai"] )
R41 = ctrl.Rule( C["Rat nhieu"] & D["Trung binh"] & (B["Co"]), S["Cao"] )
R42 = ctrl.Rule( C["Rat nhieu"] & D["Trung binh"] & (B["Gian tiep"]), S["Vua phai"] )
R43 = ctrl.Rule( C["Rat nhieu"] & D["Nhieu"] & (B["Khong"]), S["Cao"] )
R44 = ctrl.Rule( C["Rat nhieu"] & D["Nhieu"] & (B["Co"]), S["Rat cao"] )
R45 = ctrl.Rule( C["Rat nhieu"] & D["Nhieu"] & (B["Gian tiep"]), S["Cao"] )
R46 = ctrl.Rule( C["Rat nhieu"] & D["Rat nhieu"] & (B["Khong"]), S["Cao"] )
R47 = ctrl.Rule( C["Rat nhieu"] & D["Rat nhieu"] & (B["Co"]), S["Rat cao"] )
R48 = ctrl.Rule( C["Rat nhieu"] & D["Rat nhieu"] & (B["Gian tiep"]), S["Cao"] )
```

# Dùng ngôn ngữ Python

- **Giai đoạn 4**: xây dựng hệ thống mô phỏng

```
rules = [R1,R2,R3,R4,R5_6,R7_8_9,R10_11_12,R13,R14_15,R16,
R17_18,R19,R20,R21,R22,R23,R24,R25_26,R27,R28,R29,R30,R31,R32,
R33,R34,R35,R36,R37_38_39,R40,R41,R42,R43,R44,R45,R46,R47,R48]


system = ctrl.ControlSystemSimulation( ctrl.ControlSystem(rules) )
```

- **Giai đoạn 5**: nhập dữ liệu và giải mờ

```
iC = int( input("Số người bị nhiễm: ") )
iD = int( input("Tổng số tử vong: ") )
iB = int( input("Mức độ tiếp giáp  (1:không,
2:có, 3:gián tiếp): ") )
```

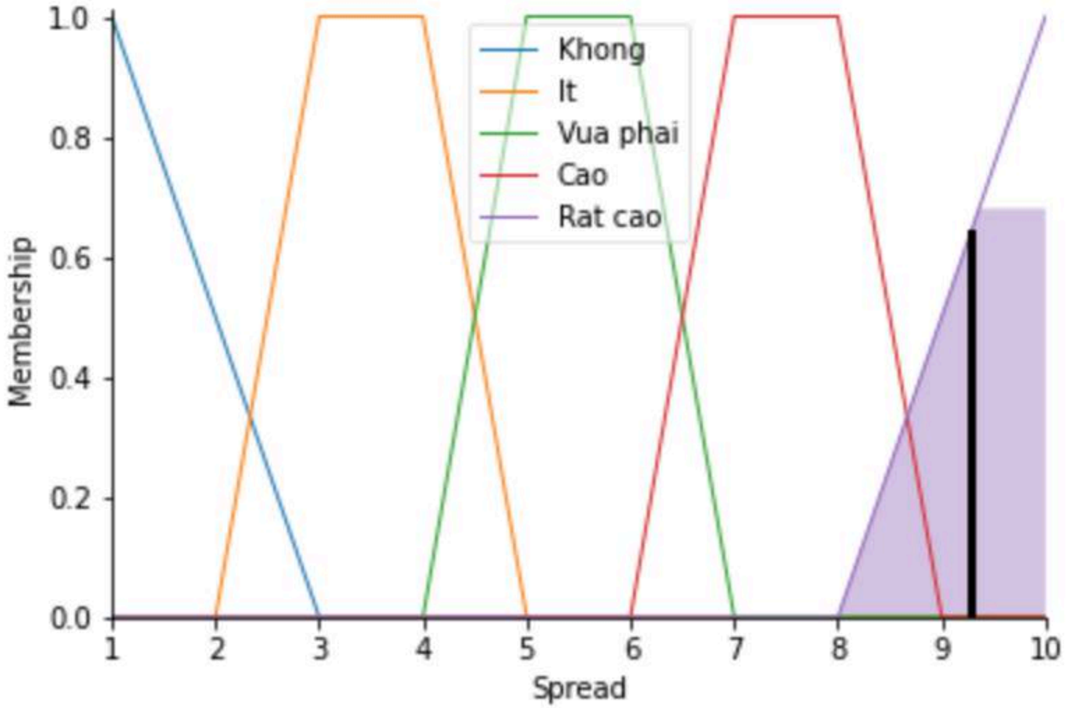
```
system.input["Cases"] = iC
system.input["Deaths"] = iD
system.input["Border"] = iB
```

# Bài toán 2

```
system.compute()
oS = system.output["Spread"]
print("Số người nhiễm %d, tử vong %d; tiếp giáp cấp độ %d. Khả năng lây lan cấp độ %d" % (iC,iD,iB,oS))
S.view( system )
```

```
Số người bị nhiễm: 80000
Tổng số tử vong: 8000
Mức độ tiếp giáp (1:không, 2:có, 3:gián tiếp): 2
```

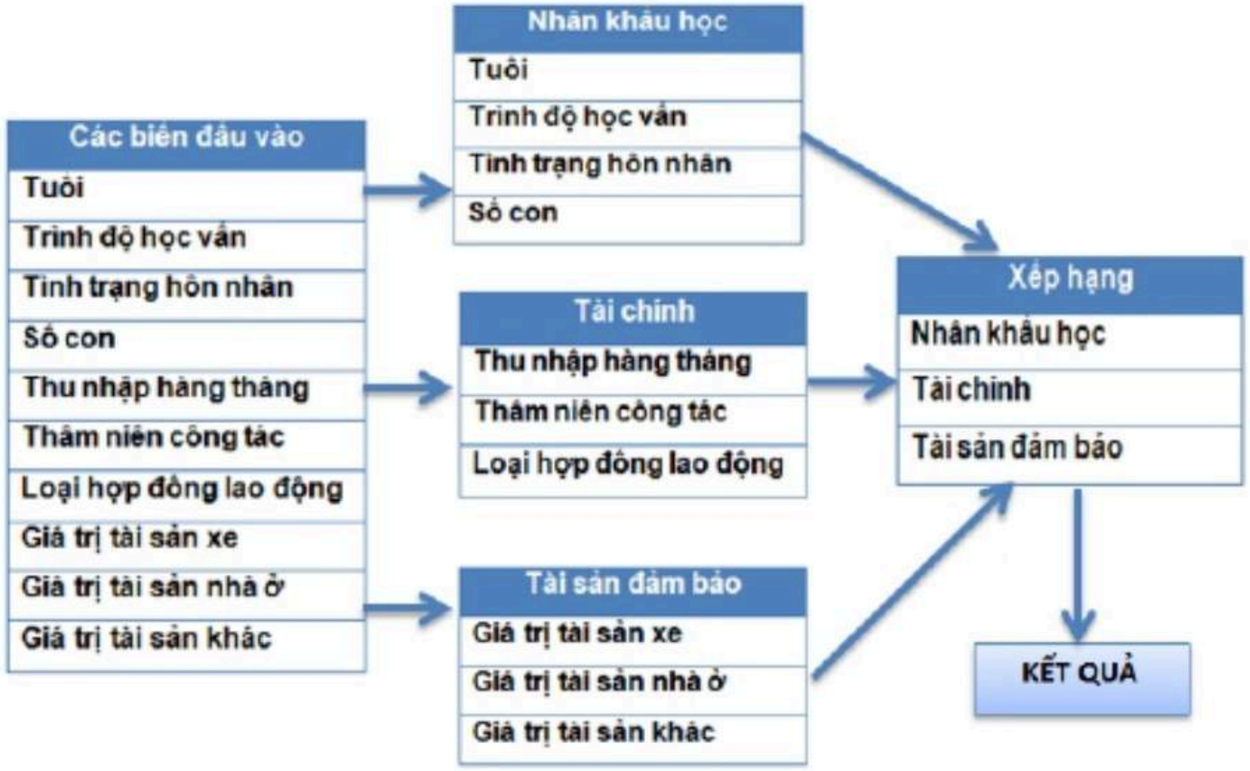
```
Số người nhiễm 80000, tử vong 8000; tiếp giáp
cấp độ 2. Khả năng lây lan cấp độ 9
```



# Dùng ngôn ngữ Python

**Bài toán 3:** Xếp hạng tín dụng (xem "Logic mờ và các bài toán ứng dụng trong lĩnh vực tài chính", Tập san Tin học Quản lý, Tập 3, Số 1&2, 2014, tr.27-44, https://www.academia.edu/24400043/LOGIC_MỜ_VÀ_CÁC_BÀI_TOÁN_ỨNG_DỤNG_TRONG_LĨNH_VỰC_TÀI_CHÍNH).
Trong đó nhân thân, khả năng tài chính và tài sản như là tiền đề để xếp hạng tín dụng

- Để đơn giản, việc xếp hạng tín dụng ở đây được suy đoán từ *nhân thân* (nhân khẩu học), *tài chính* và *tài sản đảm bảo* (như hình bên)

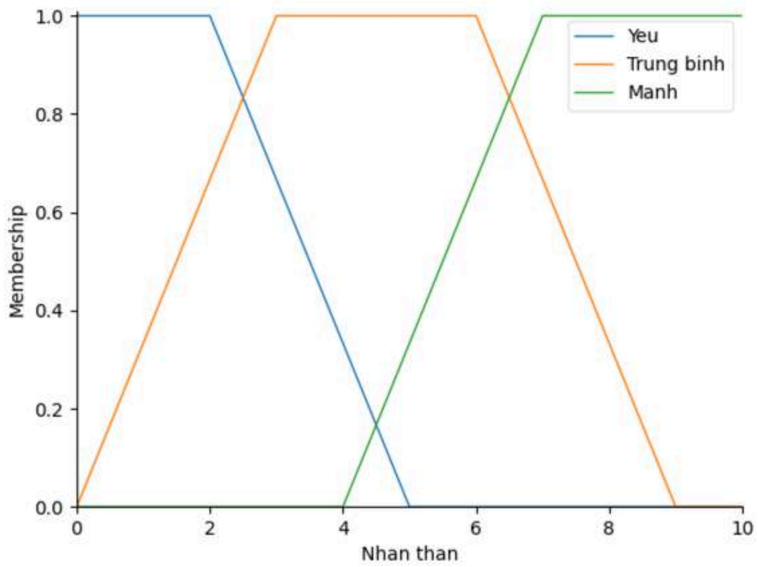


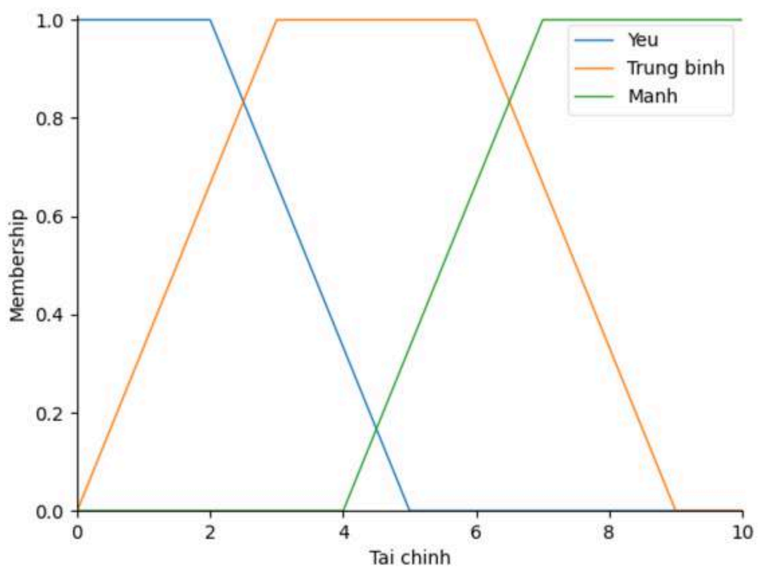
# Dùng ngôn ngữ Python

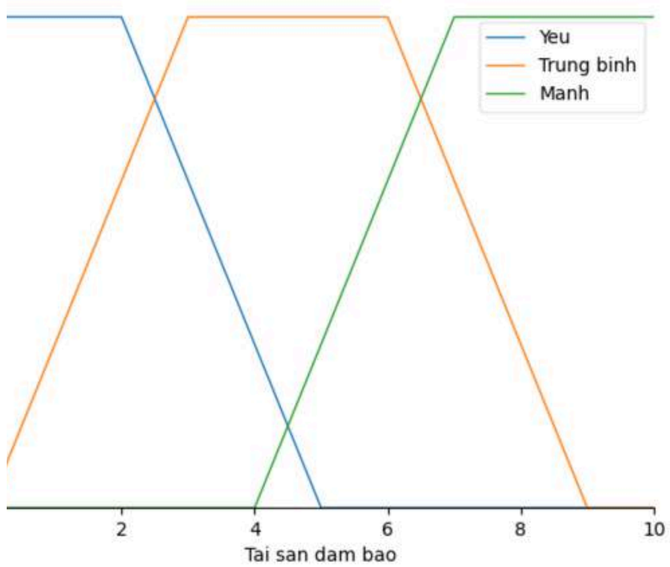
- ***Xếp hạng tín dụng*** được phân thành 3 mức trong khoảng điểm [0,10]
  - Rủi ro thấp
  - Rủi ro trung bình
  - Rủi ro cao
- ***Nhân thân***, ***tài chính*** và ***tài sản*** được phân thành 3 loại trong khoảng [0,10]
  - Yếu
  - Trung bình
  - Mạnh

# Dùng ngôn ngữ Python

- Chương trình được viết bằng Python với thư việc scikit-fuzzy có trong tập tin Bankline.py với các tập mờ sau

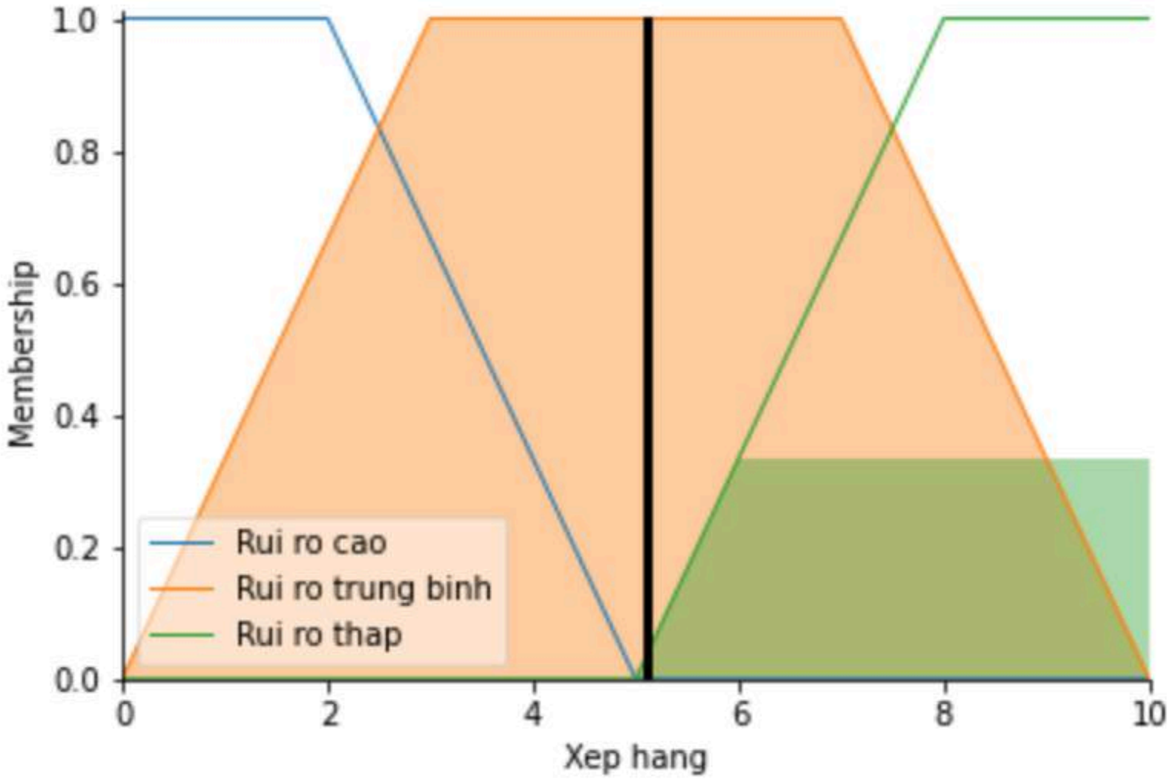


# Dùng ngôn ngữ Python

- Với số liệu
  - Nhan than [0,10]: 10
  - Tai chinh [0,10]: 3
  - Tai san dam bao [0,10]: 5
- Kết quả
  - Với nhân thân 10 điểm, tài chính 3 điểm, tài sản đảm bảo 5 điểm; xếp hạng tính dụng là 5 điểm



# Bài tập chính xác hoá Bài toán 2

**Dùng 3 kết quả suy đoán này để đưa vào như là giá trị nhập cho bài toán xem xét khả năng tín dụng ở trên**

- Suy đoán về ***nhân thân*** căn cứ vào 4 tiền đề là:
  - Tuổi
  - Trình độ học vấn
  - Tình trạng hôn nhân
  - Số con

| | |
|---|---|
| Tuổi (từ 18 đến 65 tuổi) | **Trẻ**: nhỏ hơn 33<br>**Trung niên**: từ 27 đến 53<br>**Già**: lớn hơn 48 |
| Trình độ học vấn (từ 1 đến 3: 0- phổ thông, 1- công nhân kỹ thuật, 2- cao đẳng, 3- đại học và sau đại học) | **Thấp**: nhỏ hơn 1<br>**Trung bình**: từ 0.8 đến 2.25<br>**Cao**: hơn 1.5 |
| Tình trạng hôn nhân (từ 0 đến 1: 0 – độc thân, 1 – có giá đình; từ 0-1: là tình nhân hoặc góa phụ,…) | **Độc thân**: nhỏ hơn 0.7<br>**Có gia đình**: lớn hơn 0.7 |
| Số con (từ 1 đến 5 con) | **Ít**: nhỏ hơn 2<br>**Vừa**: từ 1 đến 3.7<br>**Nhiều**: lớn hơn 3 |

# Bài tập

- Suy đoán về ***tài chính*** căn cứ vào:
  - Thu nhập hằng tháng
  - Thâm niên công tác
  - Loại hợp đồng lao động

| | |
|---|---|
| Thu nhập hàng tháng (từ 800 đến 5000) | **Thấp**: nhỏ hơn 2900<br>**Trung bình**: từ 1850 đến 3950<br>**Cao**: lớn hơn 2950 |
| Thâm niên công tác (từ 0 đến 15 năm) | **Ngắn**: nhỏ hơn 7.5 năm<br>**Trung bình**: từ 3.7 đến 11.25 năm<br>**Dài**: trên 7.5 năm |
| Loại hợp đồng lao động (từ 0 đến 2: 0- thời vụ, 1- có thời hạn, 2-không thời hạn) | **Thời vụ**: nhỏ hơn 1<br>**Có thời hạn**: từ 0.5 đến 1.5<br>**Không thời han**: lớn hơn 1 |

# Bài tập

- ***Tài sản*** đảm bảo căn cứ vào giá trị tài sản:
  - Xe hơi
  - Nhà ở
  - Các loại khác

| | |
|---|---|
| Giá trị tài sản xe hơi (từ 10000 đến 100000) | **Rẻ**: nhỏ hơn 55000<br>**Vừa**: từ 33000 đến 77500<br>**Đắt**: lớn hơn 55000 |
| Giá trị tài sản nhà ở (từ 0 đến 500000) | **Thấp**: nhỏ hơn 325000<br>**Trung bình**: từ 237500 đến 412500<br>**Cao**: lớn hơn 325000 |
| Giá trị tài sản khác (từ 1000 đến 20000) | **Thấp**: nhỏ hơn 4500<br>**Trung bình**: từ 2700 đến 15250<br>**Cao**: lớn hơn 10500 |

# Dùng ngôn ngữ Python

**Bài toán 4:** Điều khiển tự động tốc độ xe (xem trang 35 của luận văn thạc sĩ "Tìm hiểu logic mờ và xây dựng ứng dụng điều khiển tự động tốc độ xe ô tô" (https://www.slideshare.net/itsky7792/tm-hiu-logic-m-v-xy-dng-ng-dng-iu-khin-t-ng-tc-xe-t)

- Tốc độ (Speed) của xe được suy đoán căn cứ vào:
  - Khí hậu (Temperature)
  - Mây bao phủ (Cloud Cover)
- Trong đó, Temperature (thuộc [0,110]) có 4 tình trạng:
  - Freezing
  - Cool
  - Warm
  - Hot
- Cloud Cover (thuộc [0,100]) có 3 trường hợp
  - Sunny
  - Partly Cloudy
  - Overcast
- Còn Speed (thuộc [0,100]) chỉ có:
  - Fast
  - Slow

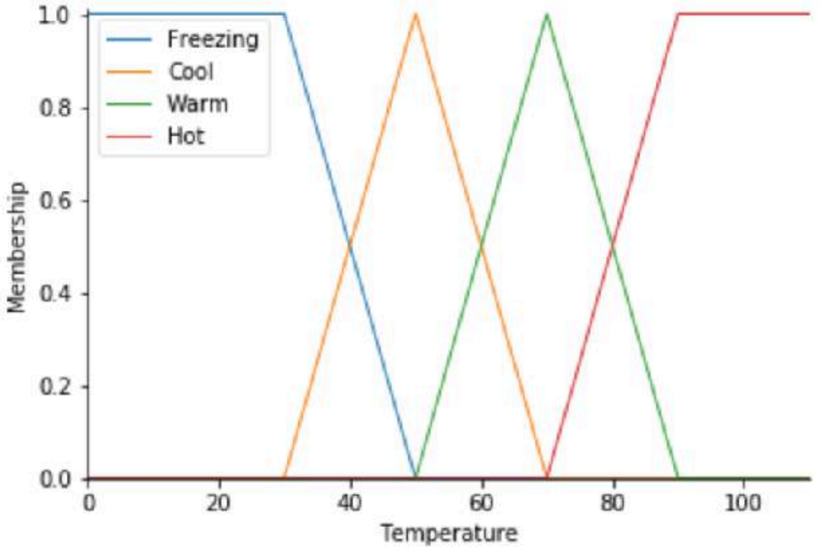
# Dùng ngôn ngữ Python

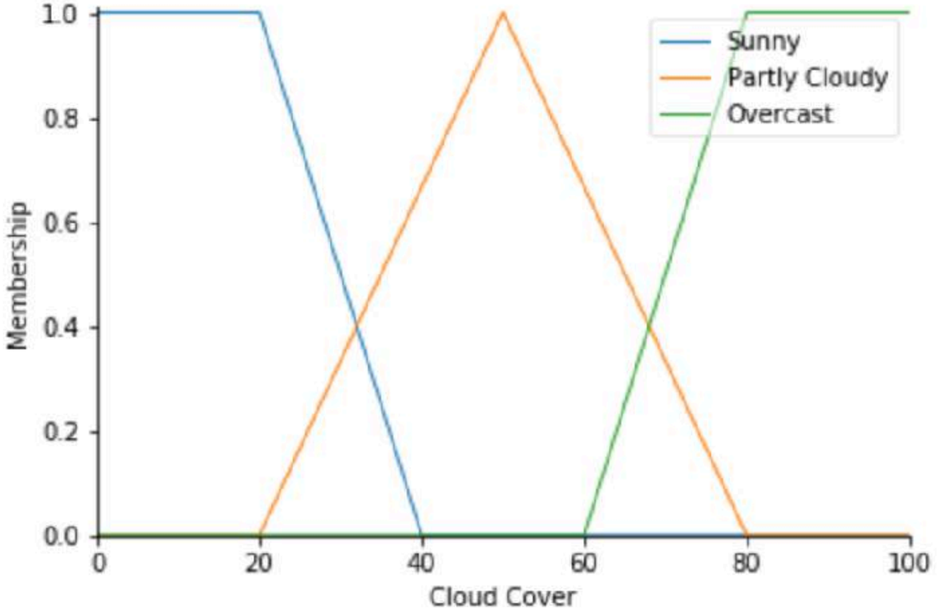
```
import numpy as np
import skfuzzy as fz
from skfuzzy import control as ctrl

X = np.arange(0,120,10)
Y = np.arange(0,110,10)
Z = np.arange(0,105,5)

T = ctrl.Antecedent( X, "Temperature" )
C = ctrl.Antecedent( Y, "Cloud Cover" )
S = ctrl.Consequent( Z, "Speed" )

T["Freezing"] = fz.trapmf( T.universe, [0,0,30,50] )
T["Cool"] = fz.trimf( T.universe, [30,50,70] )
T["Warm"] = fz.trimf( T.universe, [50,70,90] )
T["Hot"] = fz.trapmf( T.universe, [70,90,110,110] )
T.view()

C["Sunny"] = fz.trapmf( C.universe, [0,0,20,40] )
C["Partly Cloudy"] = fz.trimf( C.universe, [20,50,80] )
C["Overcast"] = fz.trapmf( C.universe, [60,80,100,100] )
C.view()

S["Fast"] = fz.trapmf( S.universe, [0,0,25,75] )
S["Slow"] = fz.trapmf( S.universe, [25,75,100,100] )
```

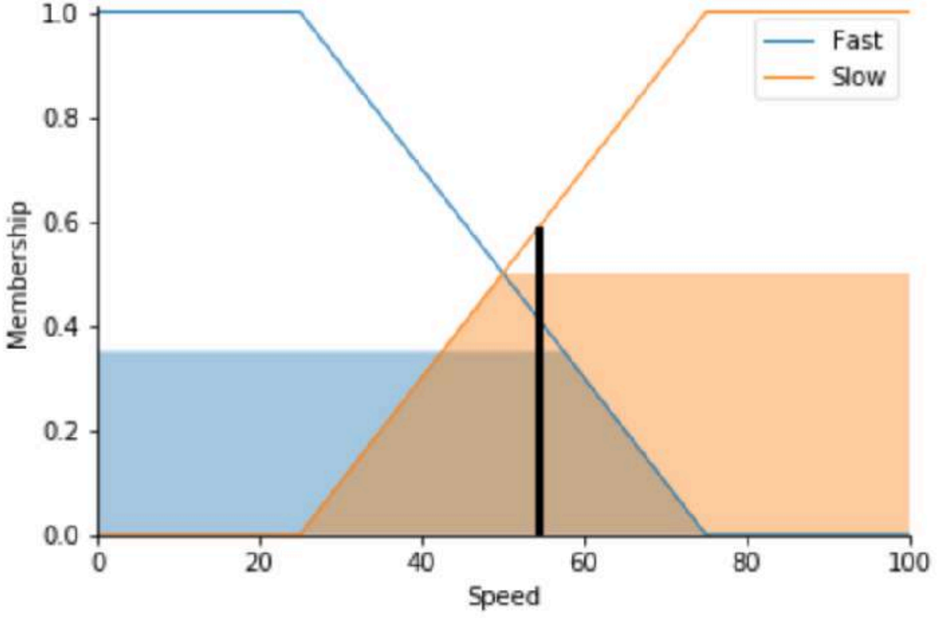
```
R1 = ctrl.Rule( C["Sunny"] & T["Freezing"], S["Slow"] )
R2 = ctrl.Rule( C["Sunny"] & T["Cool"], S["Fast"] )
R3 = ctrl.Rule( C["Sunny"] & T["Warm"], S["Fast"] )
R4 = ctrl.Rule( C["Sunny"] & T["Hot"], S["Fast"] )
R5 = ctrl.Rule( C["Partly Cloudy"] & T["Freezing"], S["Slow"] )
R6 = ctrl.Rule( C["Partly Cloudy"] & T["Cool"], S["Slow"] )
R7 = ctrl.Rule( C["Partly Cloudy"] & T["Warm"], S["Fast"] )
R8 = ctrl.Rule( C["Partly Cloudy"] & T["Hot"], S["Fast"] )
R9  = ctrl.Rule( C["Overcast"] & T["Freezing"], S["Slow"] )
R10 = ctrl.Rule( C["Overcast"] & T["Cool"], S["Slow"] )
R11 = ctrl.Rule( C["Overcast"] & T["Warm"], S["Slow"] )
R12 = ctrl.Rule( C["Overcast"] & T["Hot"], S["Slow"] )

system = ctrl.ControlSystem( [R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12] )
simulation = ctrl.ControlSystemSimulation( system )

A = float(input( "Temperature [0,100oF]: " ))
B = float(input( "Cloud Cover [0,100%]: " ))

simulation.input["Temperature"] = A
simulation.input["Cloud Cover"] = B

simulation.compute()
print( "Nhiệt độ %2.0f độ F, Tỷ lệ phần trăm mây bao phủ %2.0f; tốc độ là %2.0f mph\n"
% (A, B, simulation.output["Speed"]) )
S.view(simulation)
```

```
%run SpeedAuto.py
```

```
Temperature [0,100oF]: 57
Cloud Cover [0,100%]: 35
Nhiệt độ 57 độ F, Tỷ lệ phần trăm mây bao phủ 35; tốc độ là 54 mph
```



# Dùng ngôn ngữ Python

**Bài toán 5:** Điều khiển tín hiệu đèn giao thông (xem "Ứng dụng logic mờ điều khiển đèn tín hiệu giao thông thông minh", Kỷ yếu FAIR'2016, Cần Thơ, 4-5/8/2016 http://proceeding.vap.ac.vn/index.php/proceedingvap/article/download/2016.00097/304)

- Khi đến giao lộ có:
  - Số phương tiện đến chỗ có đèn xanh (Arrival)
  - Số phương tiện chờ ở đèn đỏ (Queue)
- Cần suy đoán:
  - Thời gian kéo dài đèn xanh (Extension)

- Với Arrival (thuộc khoảng [0,56]) và Queue (thuộc khoảng [0,32]) có các trạng thái số lượng phương tiện:
  - Rất ít
  - Ít
  - Bình thường
  - Nhiều
  - Rất nhiều
- Thời gian kéo dài đèn xanh Extension thuộc khoảng [0,20] bao gồm:
  - Không kéo dài
  - Ngắn
  - Trung bình
  - Dài
  - Rất dài

- Luật để xử lý: Nếu có quá nhiều phương tiện ở hướng đến (đèn xanh) và có một số lượng nhỏ phương tiện ở hướng chờ (đèn đỏ), thì mở rộng đèn xanh dài hơn

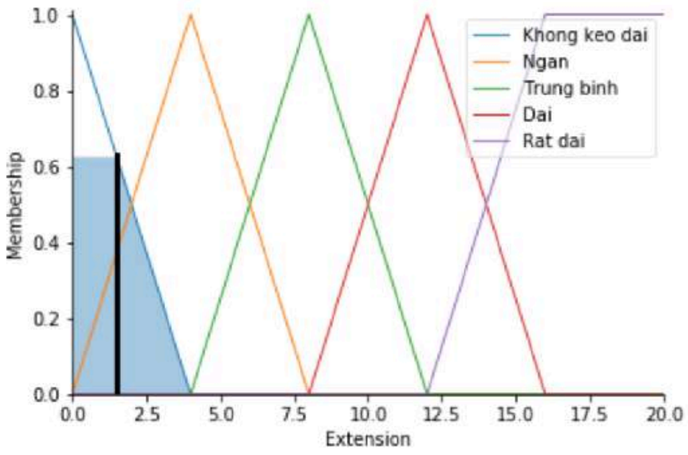
| | **Rất ít** | **Ít** | **Bình thường** | **Nhiều** | **Rất nhiều** |
|---|---|---|---|---|---|
| **Rất ít** | Không | Ngắn | Trung bình | Dài | Rất dài |
| **Ít** | Không | Ngắn | Trung bình | Dài | Dài |
| **Bình thường** | Không | Không | Ngắn | Trung bình | Dài |
| **Nhiều** | Không | Không | Không | Ngắn | Trung bình |
| **Rất nhiều** | Không | Không | Không | Không | Ngắn |

```
import numpy as np
import skfuzzy as fz
from skfuzzy import control as ctrl

X = [0,16,26,36,46,56]
Y = [0,8,14,20,26,32]
Z = np.arange(0,24,4)

A = ctrl.Antecedent( X, "Arrival" )
Q = ctrl.Antecedent( Y, "Queue" )
E = ctrl.Consequent( Z, "Extension" )

A["Rat it"] = fz.trimf( A.universe, [0,0,16] )
A["It"] = fz.trimf( A.universe, [0,16,26] )
A["Binh thuong"] = fz.trimf( A.universe, [16,26,36] )
A["Nhieu"] = fz.trimf( A.universe, [26,36,46] )
A["Rat nhieu"] = fz.trapmf( A.universe, [36,46,56,56] )
#A.view()

Q["Rat it"] = fz.trimf( Q.universe, [0,0,8] )
Q["It"] = fz.trimf( Q.universe, [0,8,14] )
Q["Binh thuong"] = fz.trimf( Q.universe, [8,14,20] )
Q["Nhieu"] = fz.trimf( Q.universe, [14,20,26] )
Q["Rat nhieu"] = fz.trapmf( Q.universe, [20,26,32,32] )
#Q.view()

E["Khong keo dai"] = fz.trimf( E.universe, [0,0,4] )
E["Ngan"] = fz.trimf( E.universe, [0,4,8] )
E["Trung binh"] = fz.trimf( E.universe, [4,8,12] )
E["Dai"] = fz.trimf( E.universe, [8,12,16] )
E["Rat dai"] = fz.trapmf( E.universe, [12,16,20,20] )
```

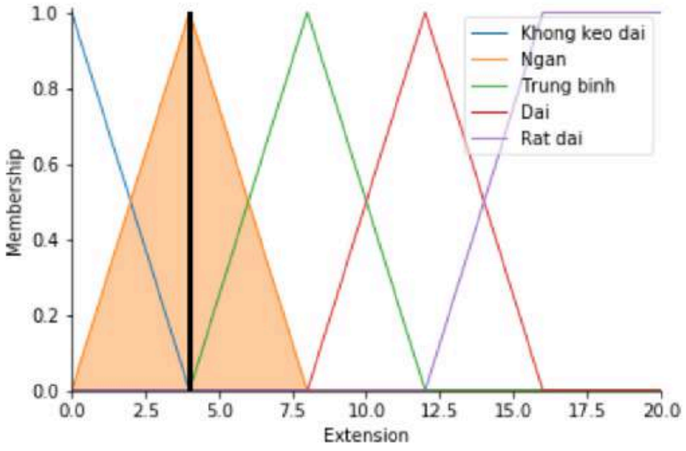
```
R1 = ctrl.Rule( Q["Rat it"] & A["Rat it"], E["Khong keo dai"] )
R2 = ctrl.Rule( Q["Rat it"] & A["It"], E["Ngan"] )
R3 = ctrl.Rule( Q["Rat it"] & A["Binh thuong"], E["Trung binh"] )
R4 = ctrl.Rule( Q["Rat it"] & A["Nhieu"], E["Dai"] )
R5 = ctrl.Rule( Q["Rat it"] & A["Rat nhieu"], E["Rat dai"] )
R6  = ctrl.Rule( Q["It"] & A["Rat it"], E["Khong keo dai"] )
R7  = ctrl.Rule( Q["It"] & A["It"], E["Ngan"] )
R8  = ctrl.Rule( Q["It"] & A["Binh thuong"], E["Trung binh"] )
R9  = ctrl.Rule( Q["It"] & A["Nhieu"], E["Dai"] )
R10 = ctrl.Rule( Q["It"] & A["Rat nhieu"], E["Dai"] )
R11 = ctrl.Rule( Q["Binh thuong"] & A["Rat it"], E["Khong keo dai"] )
R12 = ctrl.Rule( Q["Binh thuong"] & A["It"], E["Khong keo dai"] )
R13 = ctrl.Rule( Q["Binh thuong"] & A["Binh thuong"], E["Ngan"] )
R14 = ctrl.Rule( Q["Binh thuong"] & A["Nhieu"], E["Trung binh"] )
R15 = ctrl.Rule( Q["Binh thuong"] & A["Rat nhieu"], E["Dai"] )
R16 = ctrl.Rule( Q["Nhieu"] & A["Rat it"], E["Khong keo dai"] )
R17 = ctrl.Rule( Q["Nhieu"] & A["It"], E["Khong keo dai"] )
R18 = ctrl.Rule( Q["Nhieu"] & A["Binh thuong"], E["Khong keo dai"] )
R19 = ctrl.Rule( Q["Nhieu"] & A["Nhieu"], E["Ngan"] )
R20 = ctrl.Rule( Q["Nhieu"] & A["Rat nhieu"], E["Trung binh"] )
R21 = ctrl.Rule( Q["Rat nhieu"] & A["Rat it"], E["Khong keo dai"] )
R22 = ctrl.Rule( Q["Rat nhieu"] & A["It"], E["Khong keo dai"] )
R23 = ctrl.Rule( Q["Rat nhieu"] & A["Binh thuong"], E["Khong keo dai"] )
R24 = ctrl.Rule( Q["Rat nhieu"] & A["Nhieu"], E["Khong keo dai"] )
R25 = ctrl.Rule( Q["Rat nhieu"] & A["Rat nhieu"], E["Ngan"] )
```

- Chương trình bằng Python: TraficLightControl.py

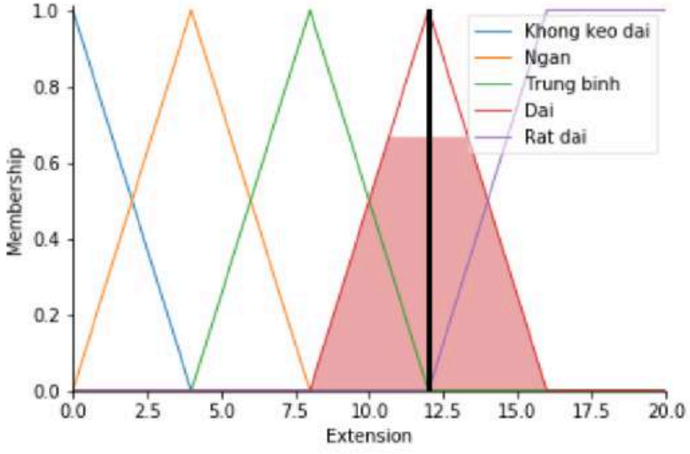
So phuong tien den cho o den xanh [0,56]: 10
So phuong tien cho o den do [0,32]: 32
Với số phương tiện đến đèn xanh 10, số phương tiện chờ đèn đỏ 32; thì thời gian kéo dài đèn xanh là  1 giây




So phuong tien den cho o den xanh [0,56]: 50
So phuong tien cho o den do [0,32]: 32
Với số phương tiện đến đèn xanh 50, số phương tiện chờ đèn đỏ 32; thì thời gian kéo dài đèn xanh là  4 giây




So phuong tien den cho o den xanh [0,56]: 50
So phuong tien cho o den do [0,32]: 10
Với số phương tiện đến đèn xanh 50, số phương tiện chờ đèn đỏ 10; thì thời gian kéo dài đèn xanh là 12 giây




**Email:** langtv@vast.vn **Homepage:** http://fair.conf.vn/~lang