**AIDS Lab**

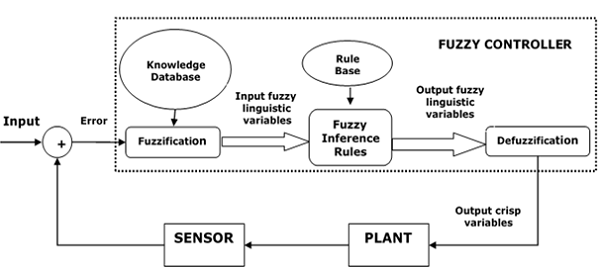
**EXPERIMENT NO. 8**

**Aim**: To design a Fuzzy control system using Fuzzy tool/library.

**Theory**:

Fuzzy logic control (FLC) is the most active research area in the application of fuzzy set theory, fuzzy reasoning, and fuzzy logic. The application of Fuzzy logic control extends from industrial process control to biomedical instrumentation and securities. Compared to conventional control techniques, Fuzzy logic control has been best utilized in complex ill-defined problems, which can be controlled by an efficient human operator without knowledge of their underlying dynamics.

The following diagram shows the architecture of Fuzzy Logic Control (FLC).



Followings are the major components of the Fuzzy logic control as shown in the above figure:

1. Fuzzifier: The role of fuzzifier is to convert the crisp input values into fuzzy values.
2. Fuzzy Knowledge Base: It stores the knowledge about all the input-output fuzzy relationships. It also has the membership function which defines the input variables to the fuzzy rule base and the output variables to the plant under control.
3. Fuzzy Rule Base: It stores the knowledge about the operation of the process of domain.
4. Inference Engine: It acts as a kernel of any Fuzzy logic control. Basically it simulates human decisions by performing approximate reasoning.
5. Defuzzifier: The role of a defuzzifier is to convert the fuzzy values into crisp values obtained from a fuzzy inference engine.

Following are the steps involved in designing Fuzzy logic control:

1. Identification of variables: Here, the input, output and state variables must be identified of the plant which is under consideration.
2. Fuzzy subset configuration: The universe of information is divided into a number of fuzzy subsets and each subset is assigned a linguistic label. Always make sure that these fuzzy subsets include all the elements of the universe.
3. Obtaining membership function: Now obtain the membership function for each fuzzy subset that we get in the above step.
4. Fuzzy rule base configuration: Now formulate the fuzzy rule base by assigning a relationship between fuzzy input and output.
5. Fuzzification: The fuzzification process is initiated in this step.
6. Combining fuzzy outputs: By applying fuzzy approximate reasoning, locate the fuzzy output and merge them.
7. Defuzzification: Finally, initiate the defuzzification process to form a crisp output.

Fuzzy logic control systems find a wide range of applications in various industrial and commercial products and systems. In several applications- related to nonlinear, time-varying, ill-defined systems and also complex systems, Fuzzy logic control systems have proved to be very efficient in comparison with other conventional control systems.

**Code and Output**:

| !pip install scikit-fuzzy from skfuzzy import control as ctrl import skfuzzy as fuzz import numpy as np |
| --- |

Using scikit-fuzzy we will generate a Control System that will estimate how long it will take to wash one load of clothes. Our inputs will be known as Antecedents and Outputs will be known as Consequents in a scikit-fuzzy controller.

Antecedents (Inputs):

1. type\_of\_dirtiness:

* Universe (crisp value range): Type of dirtiness in terms of percentage 1 to 100
* Fuzzy set (fuzzy value range): NonFat, Medium, Fat

1. degree\_of\_dirtiness:

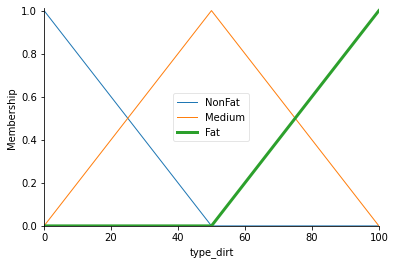
* Universe (crisp value range): Degree of dirtiness in terms of percentage 1 to 100
* Fuzzy set (fuzzy value range): Low, Medium, Fat

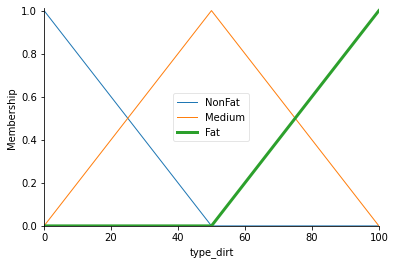
Consequents (Outputs):

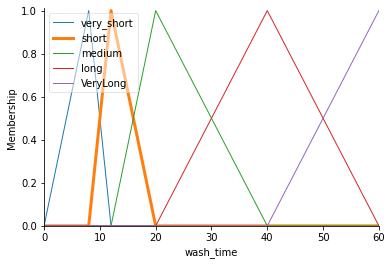
1. wash\_time:

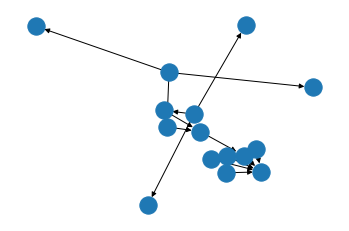
* Universe: According to type\_of\_dirtiness and degree\_of\_dirtiness program will determine how long it would take to wash one load of clothes. Output is generated in the format of minutes between (1 to 60)
* Fuzzy set (fuzzy value range): VeryShort, Short, Medium, Long, VeryLong

| class washing\_machine:  degree\_dirt = ctrl.Antecedent(np.arange(0, 101, 1), 'degree\_dirt')  type\_dirt = ctrl.Antecedent(np.arange(0, 101, 1), 'type\_dirt')  wash\_time = ctrl.Consequent(np.arange(0, 61, 1), 'wash\_time')  degree\_names = ['Low', 'Medium', 'High']  type\_names = ['NonFat', 'Medium', 'Fat']   *# Outputting them into auto-membership functions*  degree\_dirt.automf(names=degree\_names)  type\_dirt.automf(names=type\_names)    *# You can see how these look with .view()*  degree\_dirt['Medium'].view()  type\_dirt['Fat'].view()   *# Washing Time Universe*  wash\_time['very\_short'] = fuzz.trimf(wash\_time.universe, [0, 8, 12])  wash\_time['short'] = fuzz.trimf(wash\_time.universe, [8, 12, 20])  wash\_time['medium'] = fuzz.trimf(wash\_time.universe, [12, 20, 40])  wash\_time['long'] = fuzz.trimf(wash\_time.universe, [20, 40, 60])  wash\_time['VeryLong'] = fuzz.trimf(wash\_time.universe, [40, 60, 60])  wash\_time['short'].view()    *# Rule Application*  rule1 = ctrl.Rule(degree\_dirt['High'] | type\_dirt['Fat'], wash\_time['VeryLong'])  rule2 = ctrl.Rule(degree\_dirt['Medium'] | type\_dirt['Fat'], wash\_time['long'])  rule3 = ctrl.Rule(degree\_dirt['Low'] | type\_dirt['Fat'], wash\_time['long'])  rule4 = ctrl.Rule(degree\_dirt['High'] | type\_dirt['Medium'], wash\_time['long'])  rule5 = ctrl.Rule(degree\_dirt['Medium'] | type\_dirt['Medium'], wash\_time['medium'])  rule6 = ctrl.Rule(degree\_dirt['Low'] | type\_dirt['Medium'], wash\_time['medium'])  rule7 = ctrl.Rule(degree\_dirt['High'] | type\_dirt['NonFat'], wash\_time['medium'])  rule8 = ctrl.Rule(degree\_dirt['Medium'] | type\_dirt['NonFat'], wash\_time['short'])  rule9 = ctrl.Rule(degree\_dirt['Low'] | type\_dirt['NonFat'], wash\_time['very\_short'])  rule1.view()    *# Washing Control Simulation*  washing\_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4, rule5, rule6, rule7, rule8, rule9])  washing = ctrl.ControlSystemSimulation(washing\_ctrl) |
| --- |







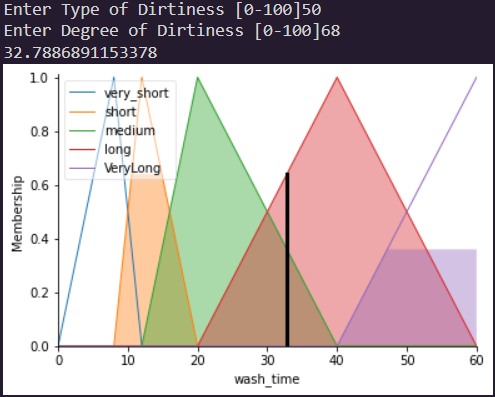


| def fuzzify\_laundry(fuzz\_type, fuzz\_degree):  washing\_machine.washing.input['type\_dirt'] = fuzz\_type  washing\_machine.washing.input['degree\_dirt'] = fuzz\_degree  washing\_machine.washing.compute()  washing\_machine.wash\_time.view(sim=washing\_machine.washing)  return washing\_machine.washing.output['wash\_time'] |
| --- |

| def compute\_washing\_parameters(type\_of\_dirt, degree\_of\_dirt):  if type\_of\_dirt < 0.0 or type\_of\_dirt > 100.0:  raise Exception("Invalid Type of Dirtiness: %lf" %type\_of\_dirt)  if degree\_of\_dirt < 0.0 or type\_of\_dirt > 100.0:  raise Exception("Invalid Degree of Dirtiness: %lf" %degree\_of\_dirt)  type\_fuzzy = fuzzify\_laundry(type\_of\_dirt, degree\_of\_dirt)  return type\_fuzzy |
| --- |

Once the output is computed all together, we can visualize it.

| if \_\_name\_\_ == "\_\_main\_\_":  type\_of\_dirt = float(input("Enter Type of Dirtiness [0-100]"))  degree\_of\_dirt = float(input("Enter Degree of Dirtiness [0-100]"))  washing\_parameters = compute\_washing\_parameters(type\_of\_dirt, degree\_of\_dirt)  print(washing\_parameters) |
| --- |



Inputs that we put in were type\_of\_dirtiness and degree\_of\_dirtiness around 50 and 68 respectively, according to that washing time is generated around approximately 32.7886891153378 minutes, we can round that up to 33 minutes.

**Conclusion**:

Thus we studied an overview of what Fuzzy Logic Control is, implemented the Fuzzy control system on a Washing Machine and found out the washing time generated after inputting type and degree of dirtiness.