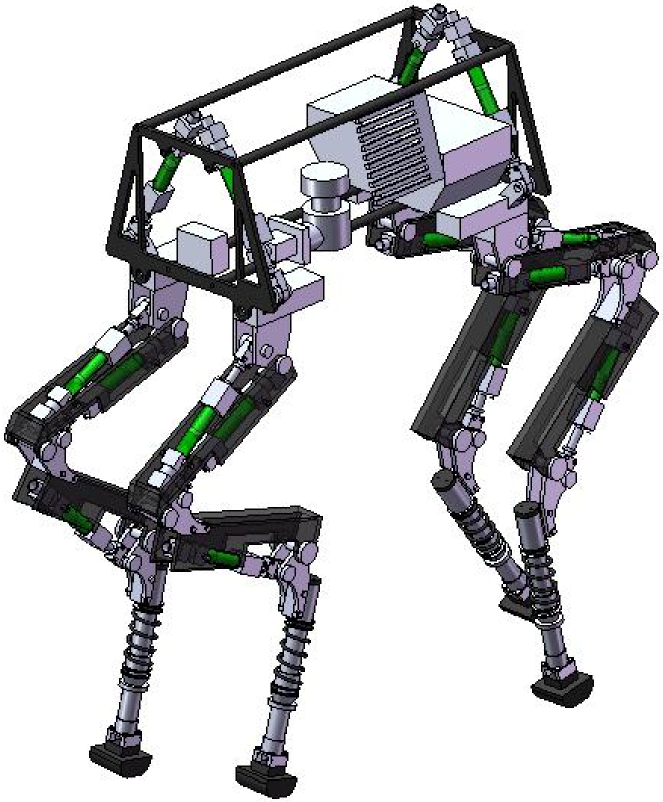


Fuzzy Logic Control

- an introduction

Dr Leo Chen

leo.chen@ieee.org

02/Jan/2019

Contents

- **What is Fuzzy Logic**
- **Foundations of Fuzzy Logic**
- **Fuzzy Logic Control**
- **Fuzzy Logic in MATLAB/SIMULINK**
- **Tutorials and Courseworks**
- **Labs**
- Reference

Contents

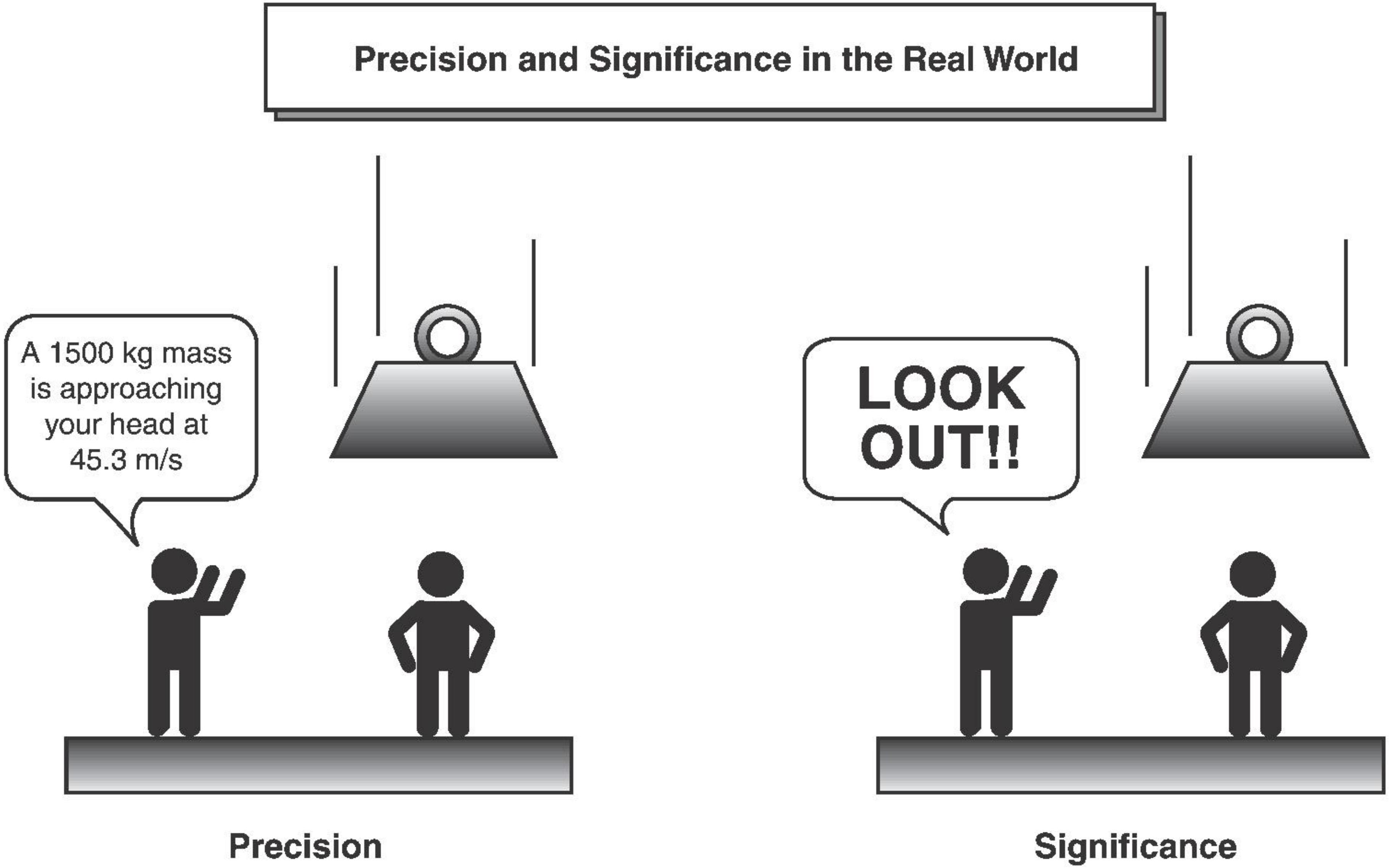
What is Fuzzy Logic (FL)

Fuzzy logic has two different meanings:

- In a **narrow** sense, fuzzy logic is a logical system, which is an extension of multivalued logic.

- In a **wider** sense fuzzy logic (FL) is almost synonymous with the theory of fuzzy sets, a theory which relates to classes of objects with unsharp boundaries in which membership is a matter of degree.

The **basic concept of FL** is that of a linguistic variable, that is, a variable whose values are **words** rather than **numbers**.



Fuzzy logic is all about the relative importance of precision: <u>How important is it to be exactly right when a rough answer will do?</u>[1]

Why Use Fuzzy Logic? (1/4)

A list of general observations about fuzzy logic:

1 Fuzzy logic is conceptually easy to understand.

*The mathematical concepts behind fuzzy reasoning are very **simple**. Fuzzy logic is a more intuitive approach without the far-reaching complexity.*

2 Fuzzy logic is flexible.

With any given system, it is easy to layer on more functionality without starting again from scratch.

Why Use Fuzzy Logic? (2/4)

3 Fuzzy logic is tolerant of imprecise data.

Everything is imprecise if you look closely enough, but more than that, most things are imprecise even on careful inspection. Fuzzy reasoning builds this understanding into the process rather than tacking it onto the end.

4 Fuzzy logic can model nonlinear functions of arbitrary complexity.

You can create a fuzzy system to match any set of input-output data. This process is made particularly easy by adaptive techniques like Adaptive Neuro-Fuzzy Inference Systems (ANFIS), which are available in Fuzzy Logic Toolbox software.

Why Use Fuzzy Logic? (3/4)

5 Fuzzy logic can be built on top of the experience of experts.

In direct contrast to neural networks, which take training data and generate opaque, impenetrable models, fuzzy logic lets you rely on the experience of people who already understand your system.

6 Fuzzy logic can be blended with conventional control techniques.

Fuzzy systems don't necessarily replace conventional control methods. In many cases fuzzy systems augment them and simplify their implementation.

Why Use Fuzzy Logic? (4/4)

7 Fuzzy logic is based on natural language.

The basis for fuzzy logic is the basis for human communication. This observation underpins many of the other statements about fuzzy logic. Because fuzzy logic is built on the structures of qualitative description used in everyday language, fuzzy logic is easy to use.

The last statement is perhaps the most important one and deserves more discussion. ***Natural language****, which is used by ordinary people on a daily basis, has been shaped by thousands of years of human history to be convenient and efficient. Sentences written in ordinary language represent a triumph of efficient communication.*

When Not to Use Fuzzy Logic

Fuzzy logic is **not a cure-all**. When should you not use fuzzy logic?

fuzzy logic is a convenient way to map an input space to an output space.

If you find it's not convenient, try something else. If a simpler solution already exists, use it. Fuzzy logic is the codification of common sense - usecommon sense when you implement it and you will probably make the right decision.

Contents

Foundations of Fuzzy Logic - Overview

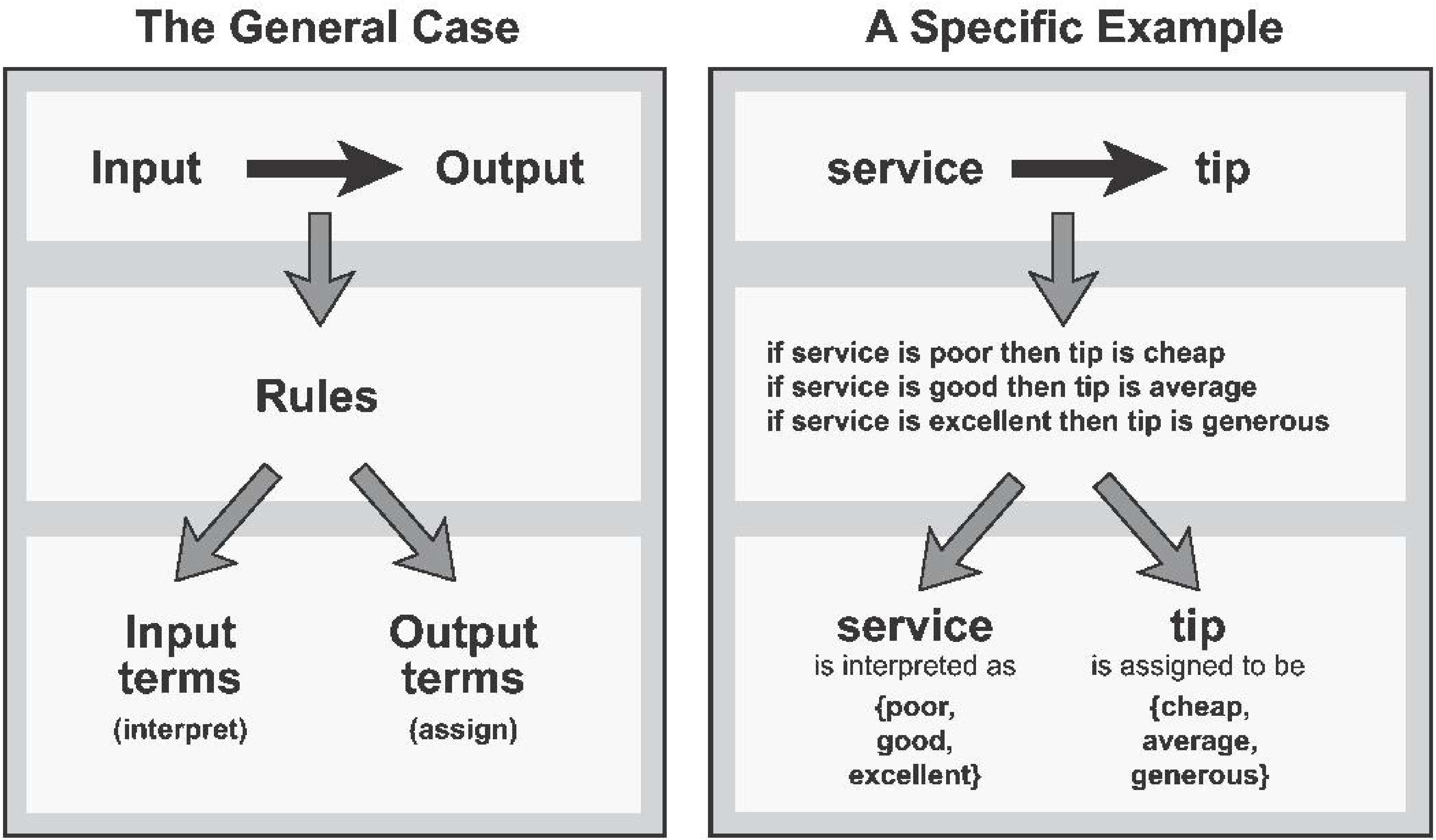
The point of fuzzy logic is to **map** an *input* space to an *output* space

the primary mechanism for doing this is *a list of if-then statements* called **rules**.

All rules are evaluated in *parallel*, and the order of the rules is unimportant.

Before you can build a system that *interprets rules*, you must ***define all the terms*** you plan on using and the adjectives that describe them.

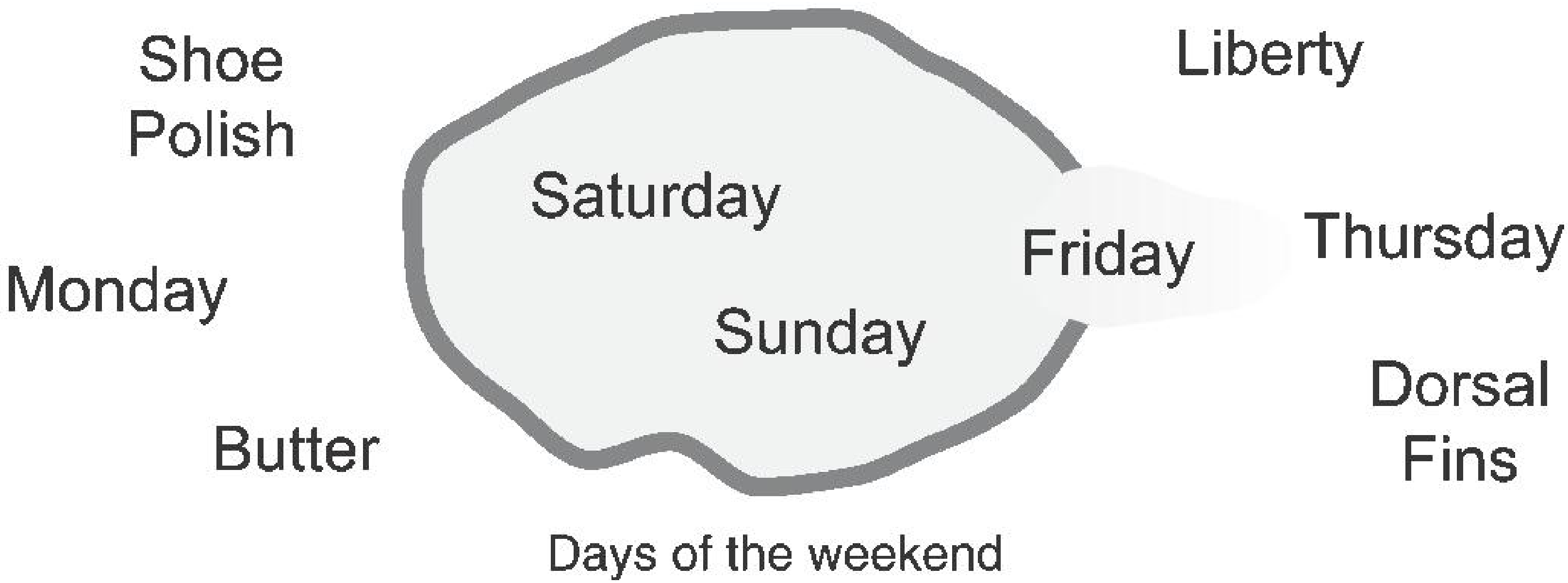
Roadmap for the fuzzy inference process.



Fuzzy Sets

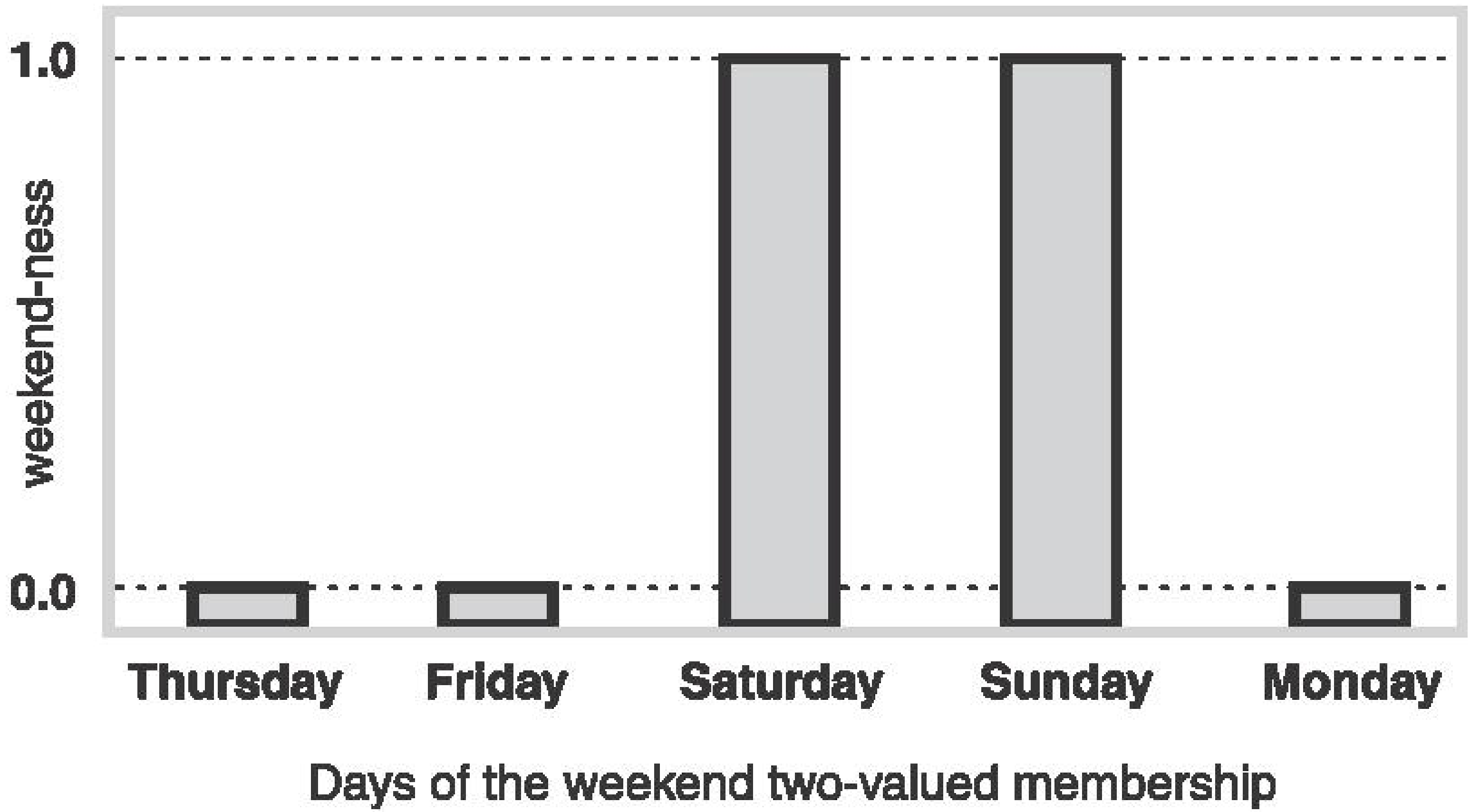
A fuzzy set is a set without a crisp, clearly defined boundary. It can contain elements with only a partial degree of membership.

A classical set is a container that wholly includes or wholly excludes any given **element**. In fuzzy logic, the truth of any statement becomes a **matter of degree**.



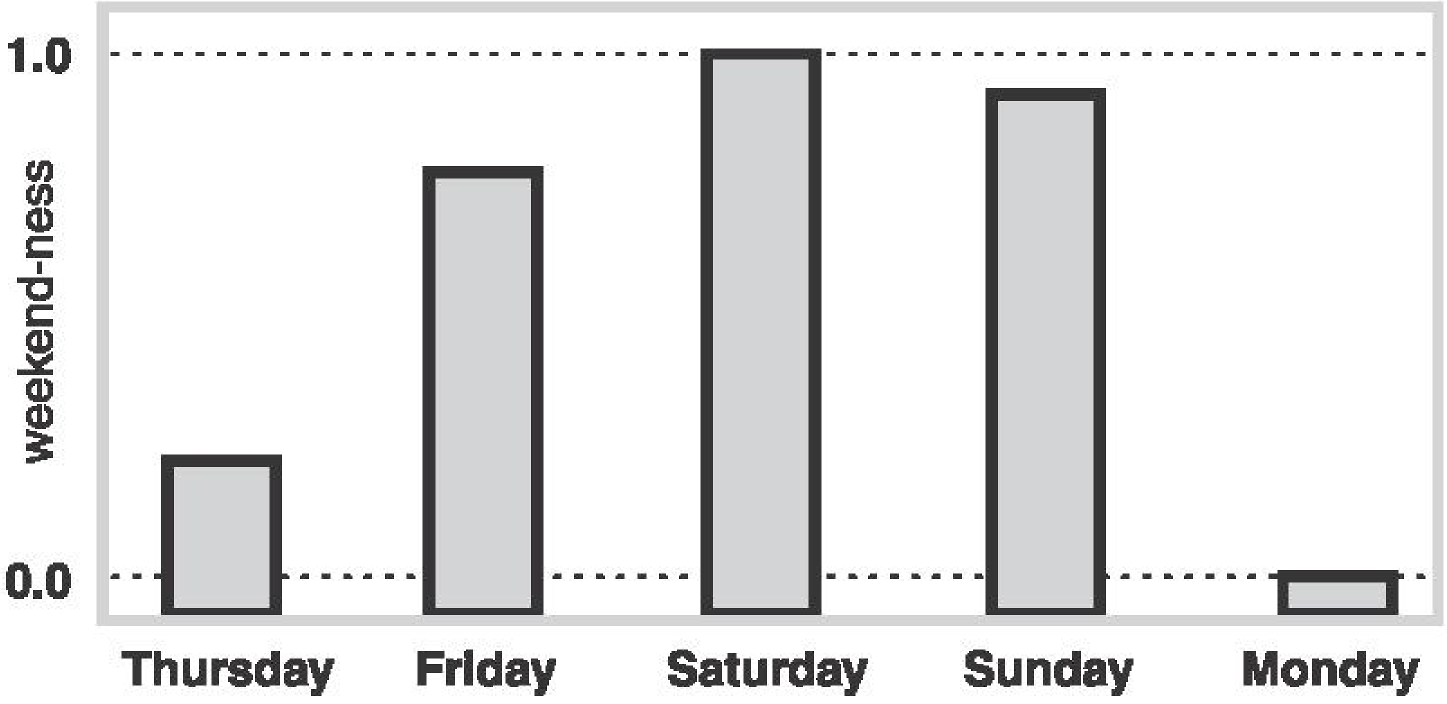
How does it work?

Reasoning in fuzzy logic is just a matter of generalizing the familiar **yes-no** (Boolean) logic. If you give true the **numerical value** of **1** and false the numerical value of **0.**



How does it work?

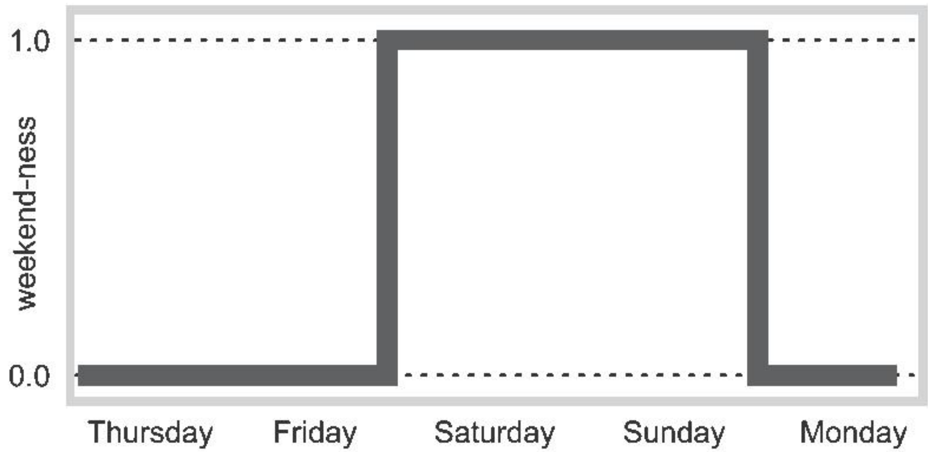
this value indicates that **fuzzy logic** also permits in-between values like 0.2 and 0.7453.



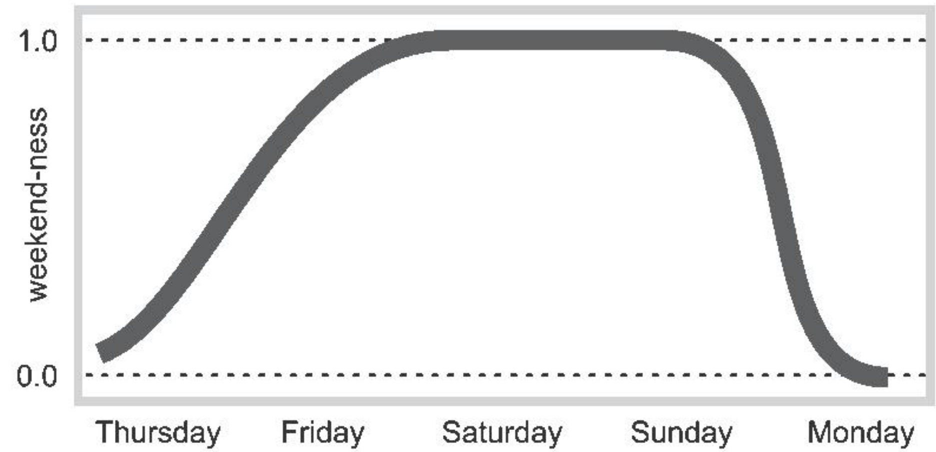
Days of the weekend multivalued membership

Multivalued logic vs. two-valued (or bivalent yes-no) logic

Consider a continuous scale time plot of weekend-ness shown in the following plots.



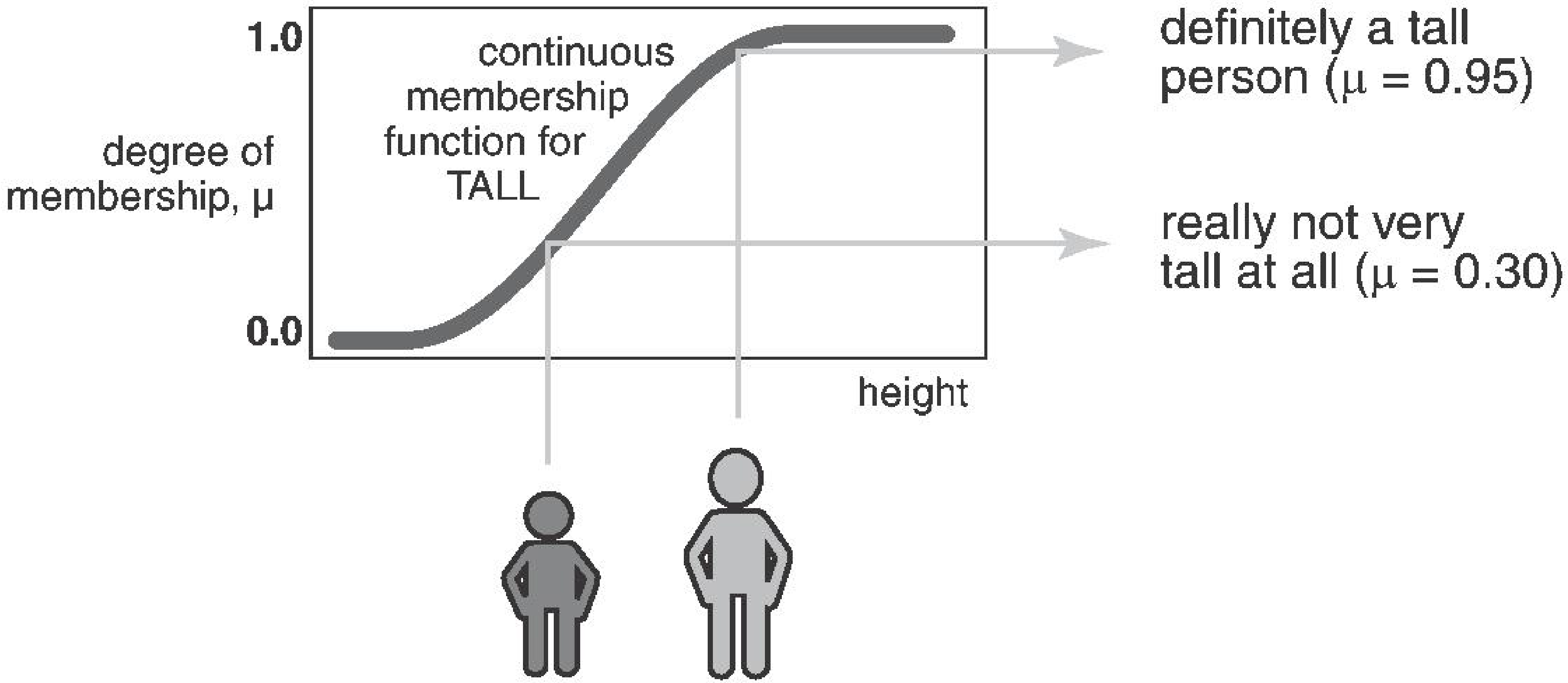
Days of the weekend two-valued membership



Days of the weekend multivalued membership

Membership Functions

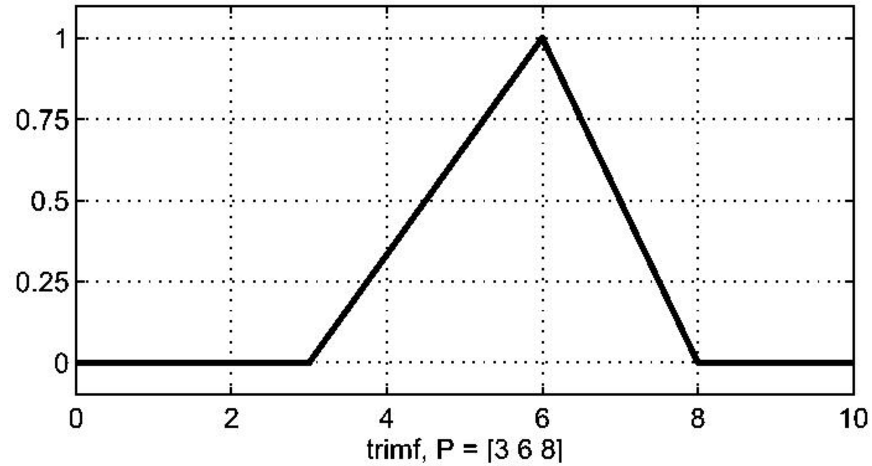
A membership function (**MF**) is a *curve* that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1.



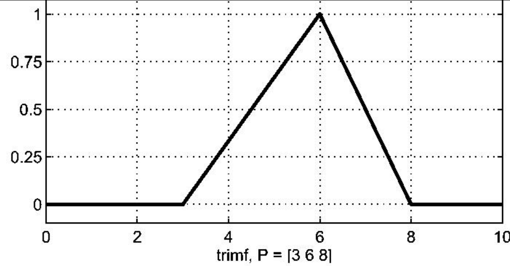
Membership Functions in Fuzzy Logic Toolbox

The toolbox includes **11 built-in membership function types**. These 11 functions are, in turn, built from several basic functions:

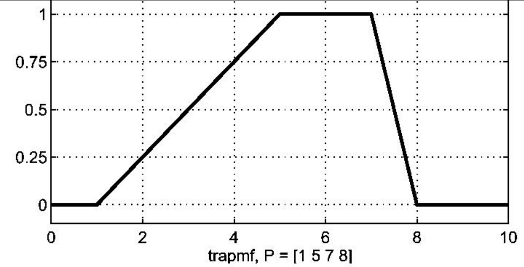
- Piece-wise linear functions
- Gaussian distribution function
- Sigmoid curve
- Quadratic and cubic polynomial curves



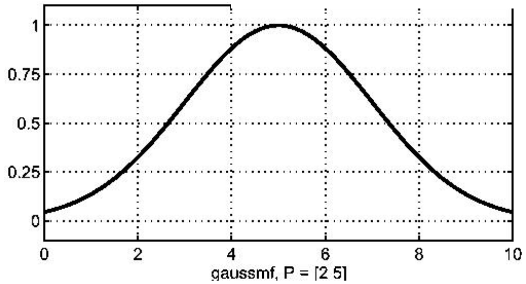
trimf



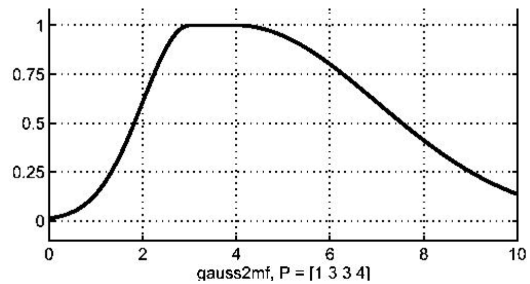
trimf



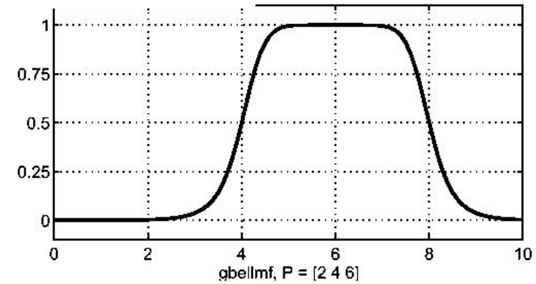
trapmf



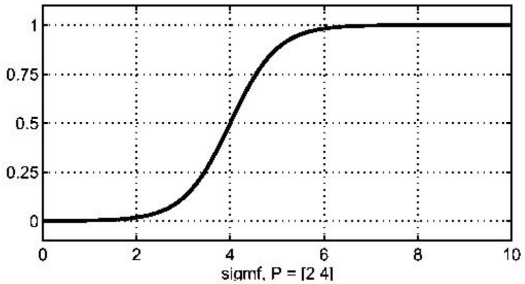
gaussmf



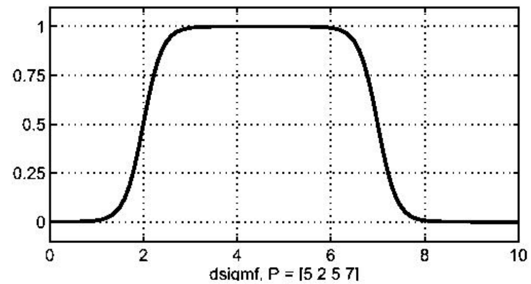
gauss2mf



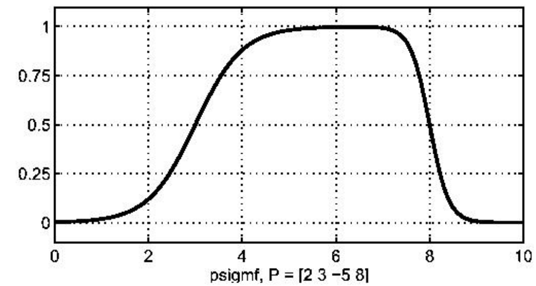
gbellmf



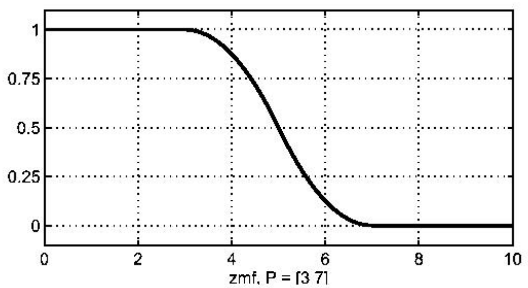
sigmf



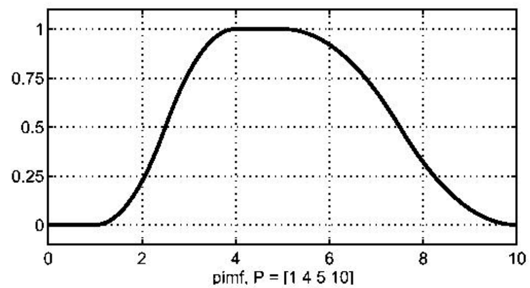
dsigmf



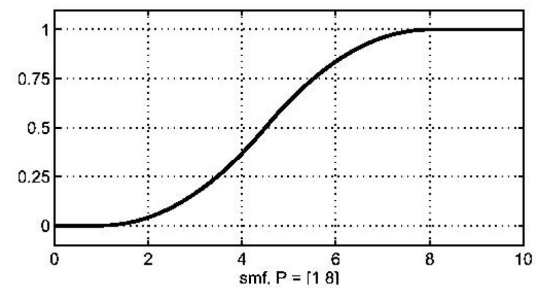
psigmf



zmf

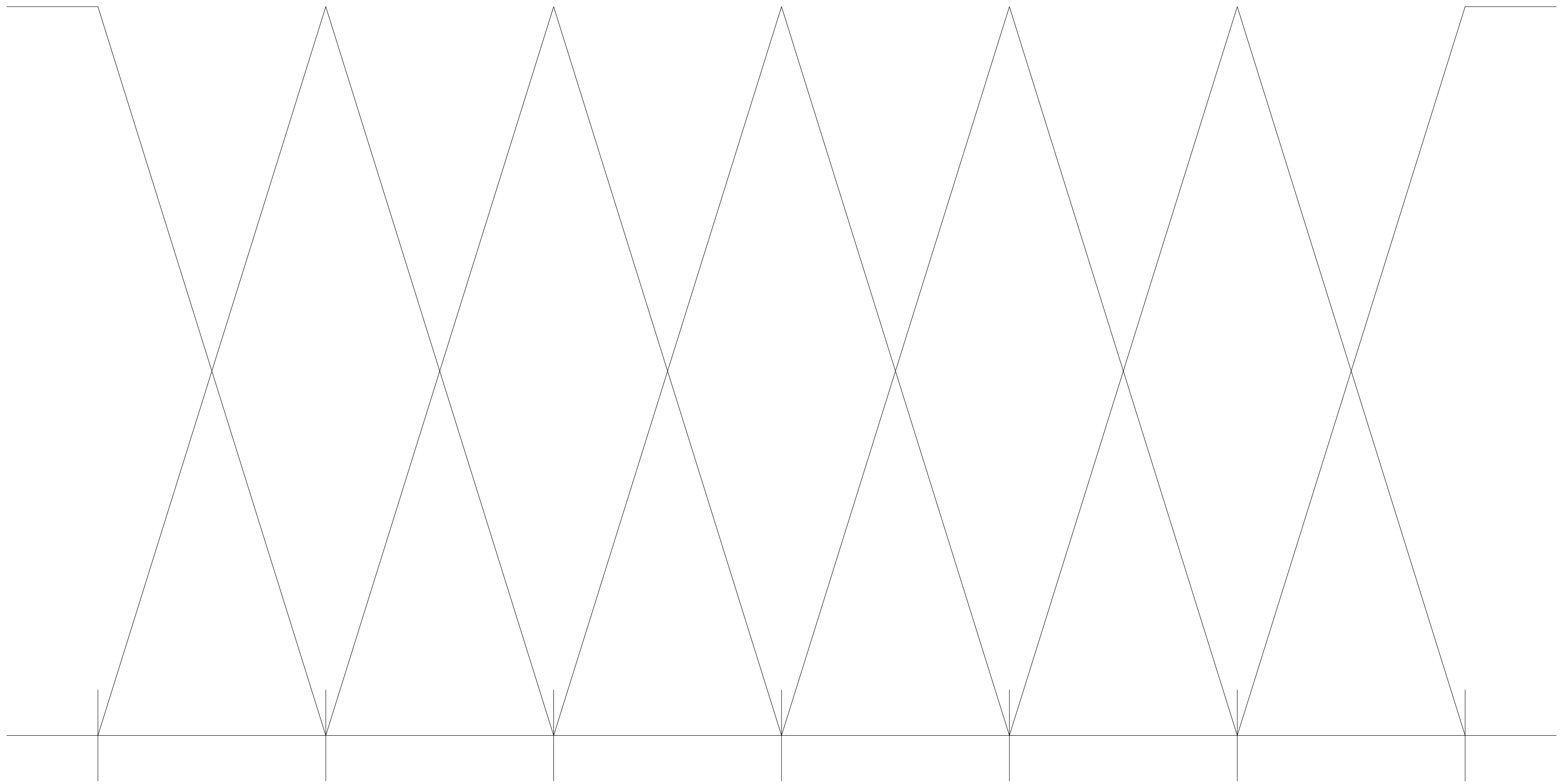


pimf



smf

Symmetrical membership functions of the seven elements



Summary of Membership Functions

- Fuzzy sets **describe** vague concepts (e.g., fast runner, hot weather, weekend days).

- A fuzzy set **admits** the possibility of partial membership in it. (e.g., Friday is sort of a weekend day, the weather is rather hot).

- The **degree** an object belongs to a fuzzy set is denoted by a membership value between 0 and 1. (e.g., Friday is a weekend day to the degree 0.8).

- A membership function **associated** with a given fuzzy set maps an input value to its appropriate membership value.

Logical Operations

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

AND

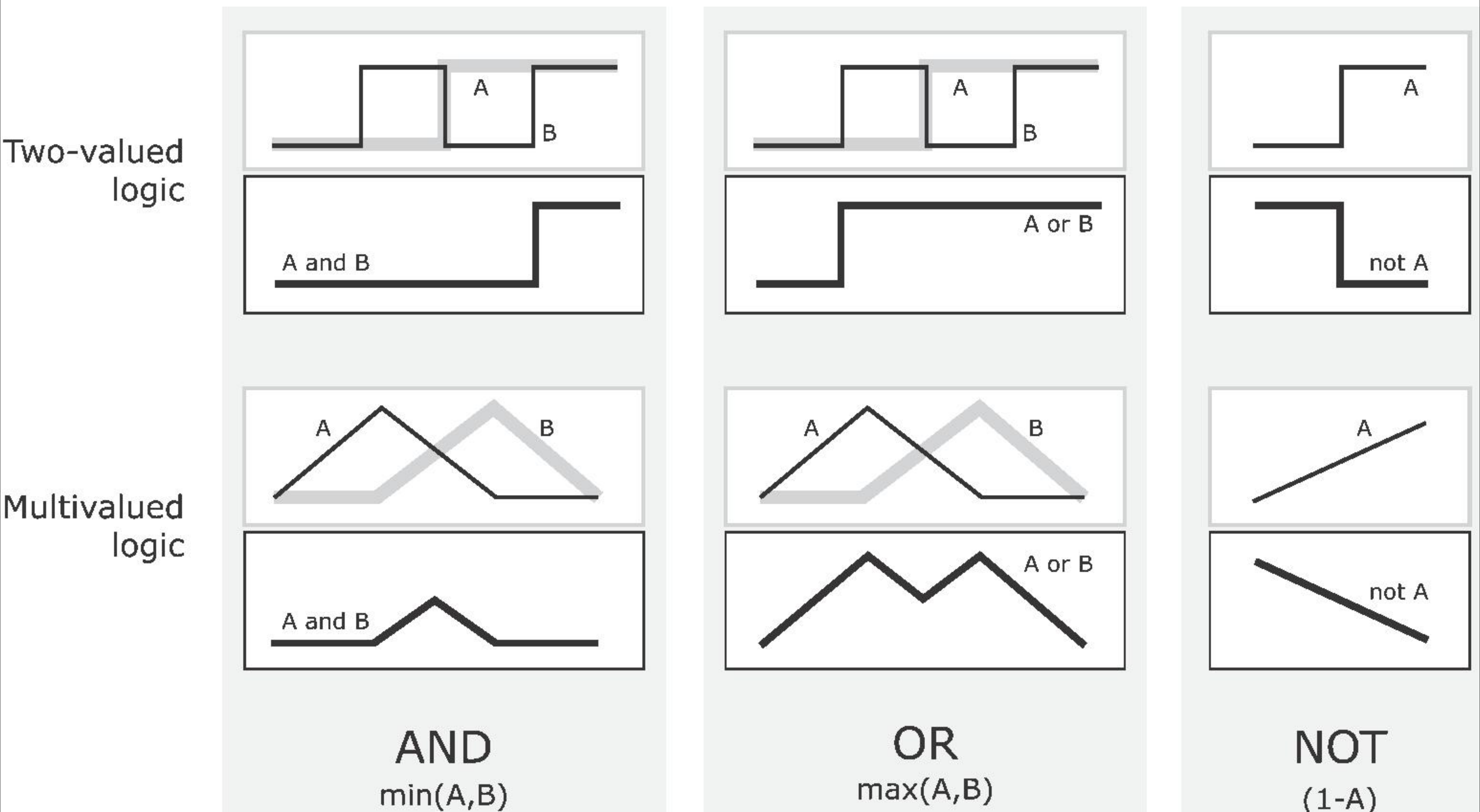
A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

OR

A	not A
0	1
1	0

NOT

Logical Operations



Additional Fuzzy Operators

① fuzzy intersection or conjunction (AND)

② fuzzy union or disjunction (OR)

③ fuzzy complement (NOT)

The classical operators for these functions are: AND = min, OR = max, and NOT = additive complement.

T-conorm (or S-norm)

If-Then Rules: If x is A, then y is B

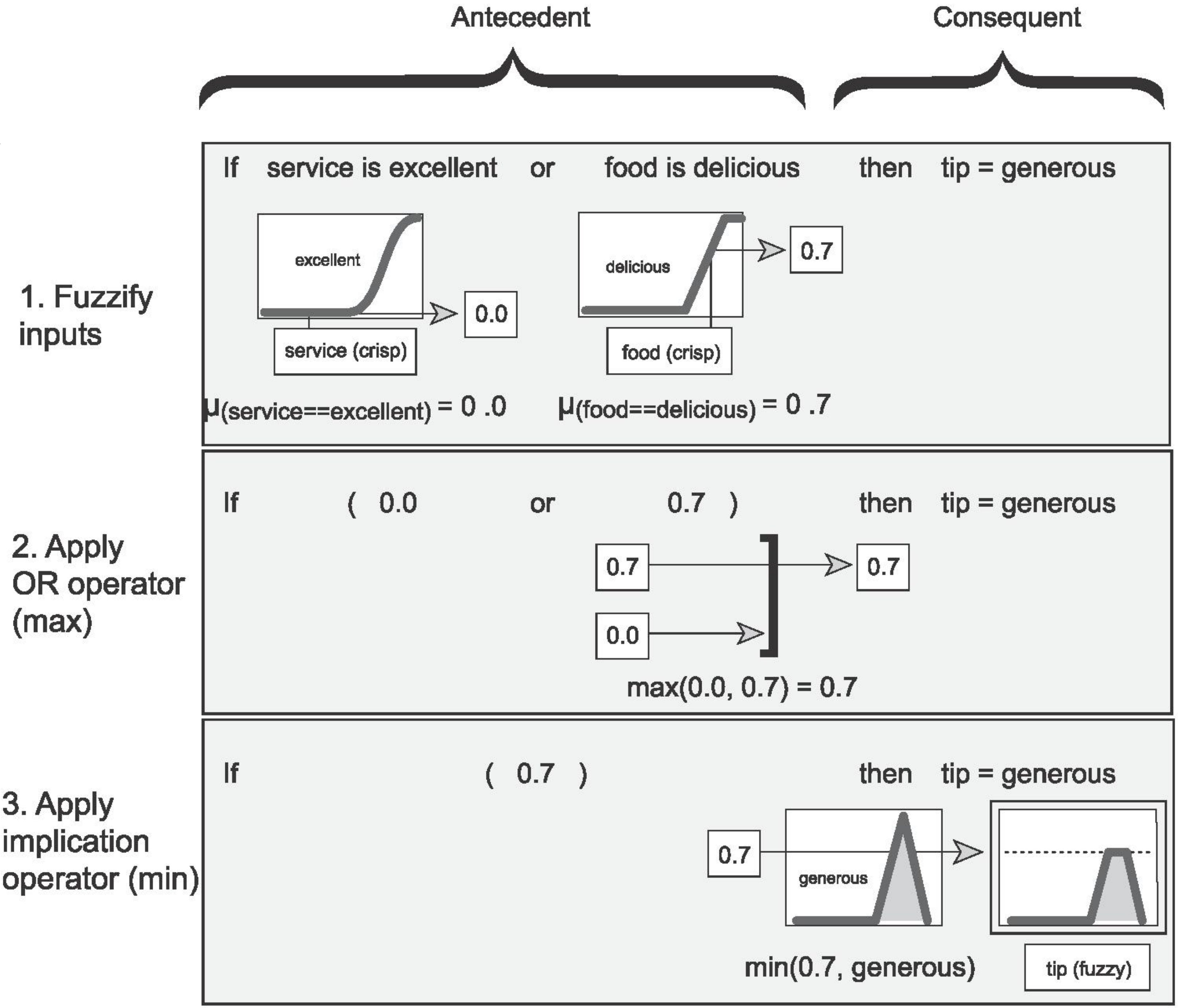
1 Fuzzify inputs:

Resolve all fuzzy statements in the antecedent to ***a degree of membership*** *between 0 and 1. If there is only one part to the antecedent, then this is the degree of support for the rule.*

2 Apply fuzzy operator to multiple part antecedents: *If there are multiple parts to the antecedent, apply fuzzy logic operators and resolve the antecedent to a single number between 0 and 1. This is the degree of support for the rule.*

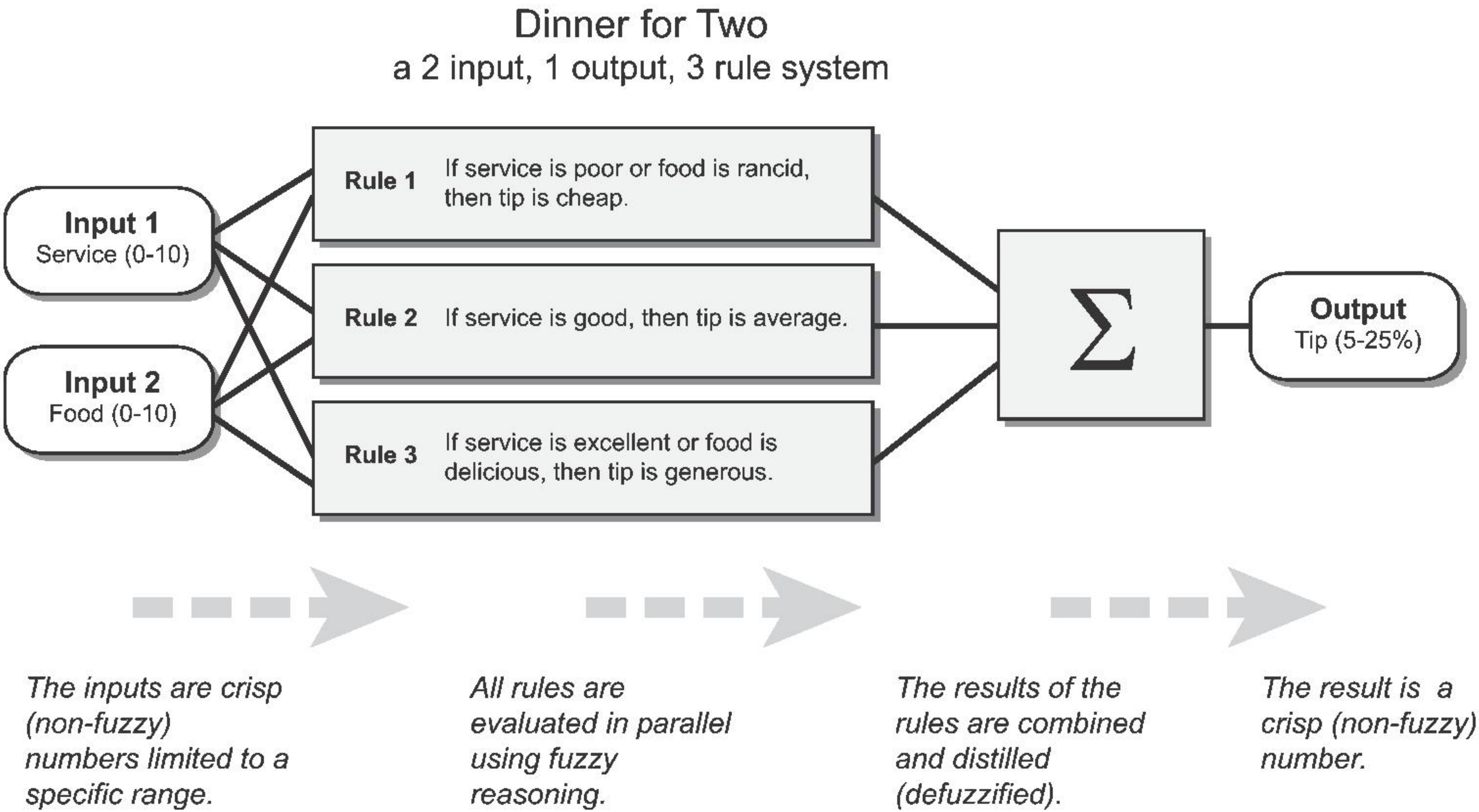
3 Apply implication method:

Use the degree of support for the entire rule to shape the output fuzzy set. The consequent of a fuzzy rule assigns an entire fuzzy set to the output. This fuzzy set is represented by a membership function that is chosen to indicate the qualities of the consequent. If the antecedent is only partially true, (i.e., is assigned a value less than 1), then the output fuzzy set is truncated according to the implication method.



Antecedent
Consequent
1. Fuzzify inputs
If service is excellent or food is delicious then tip = generous
excellent
service (crisp)
0.0
delicious
food (crisp)
0.7
μ(service==excellent) = 0 .0
μ(food==delicious) = 0 .7
2. Apply OR operator (max)
If (0.0 or 0.7) then tip = generous
0.7
0.0
0.7
max(0.0, 0.7) = 0.7
3. Apply implication operator (min)
If (0.7) then tip = generous
0.7
generous
min(0.7, generous)
tip (fuzzy)

Fuzzy Inference Process

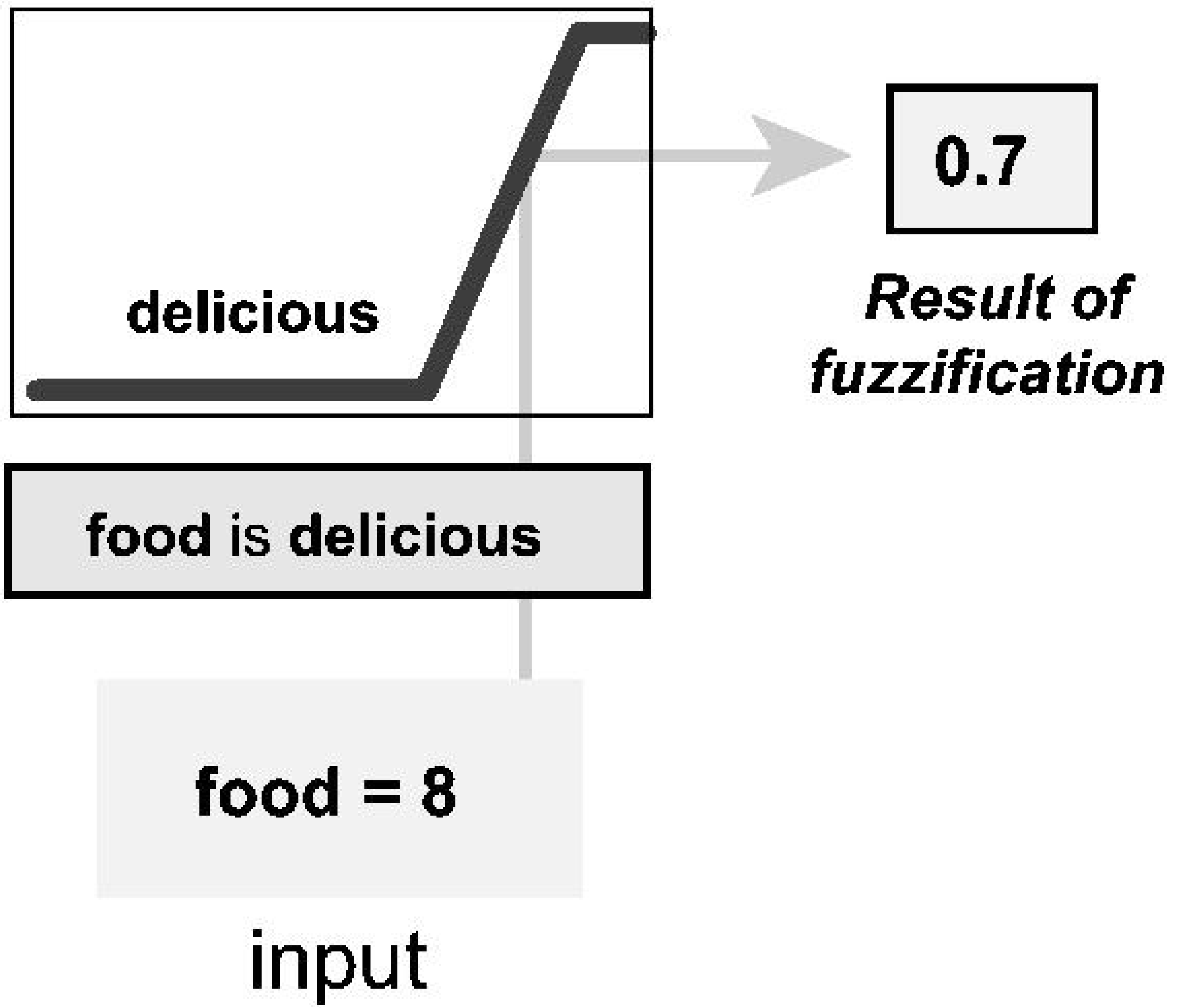


Fuzzy inference process comprises of five parts:

1. **Fuzzification** of the input variables
2. Application of the **fuzzy operator** (AND or OR) in the antecedent
3. **Implication** from the antecedent to the consequent
4. **Aggregation** of the consequents across the rules
5. **Defuzzification**

1 Fuzzify Inputs

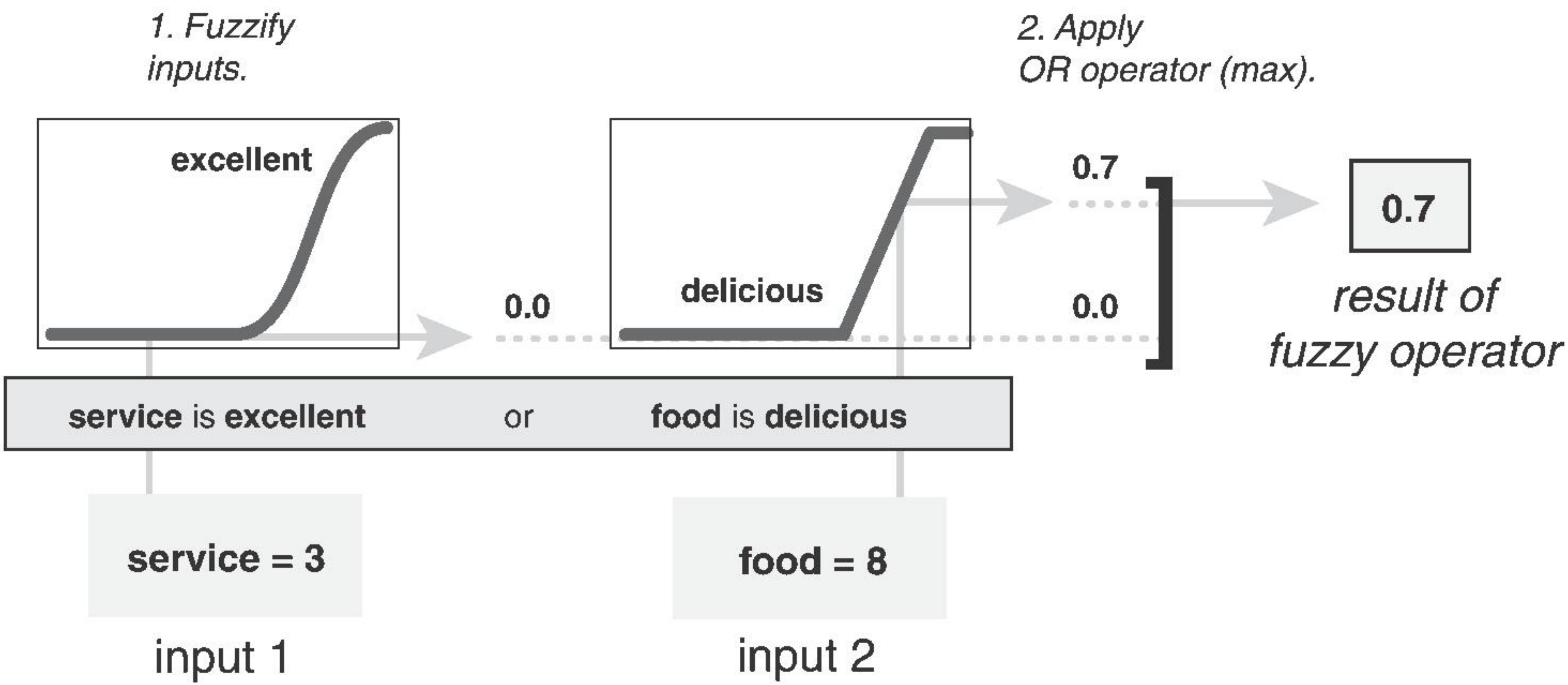
1. Fuzzify inputs.



2 Apply Fuzzy Operator

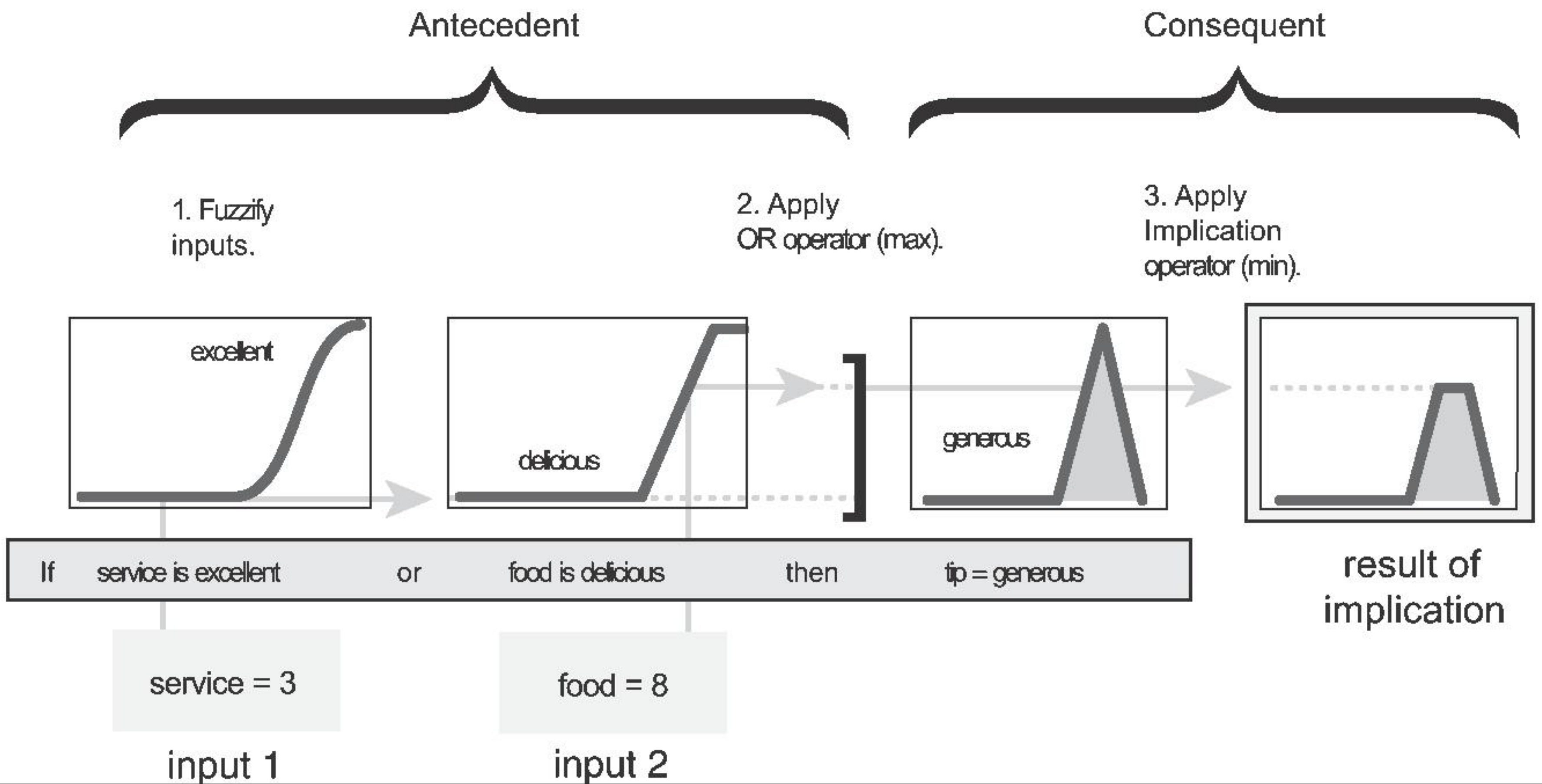
AND: *min* (minimum) and *prod* (product)

OR: *max* (maximum), and the probabilistic OR *probor*.



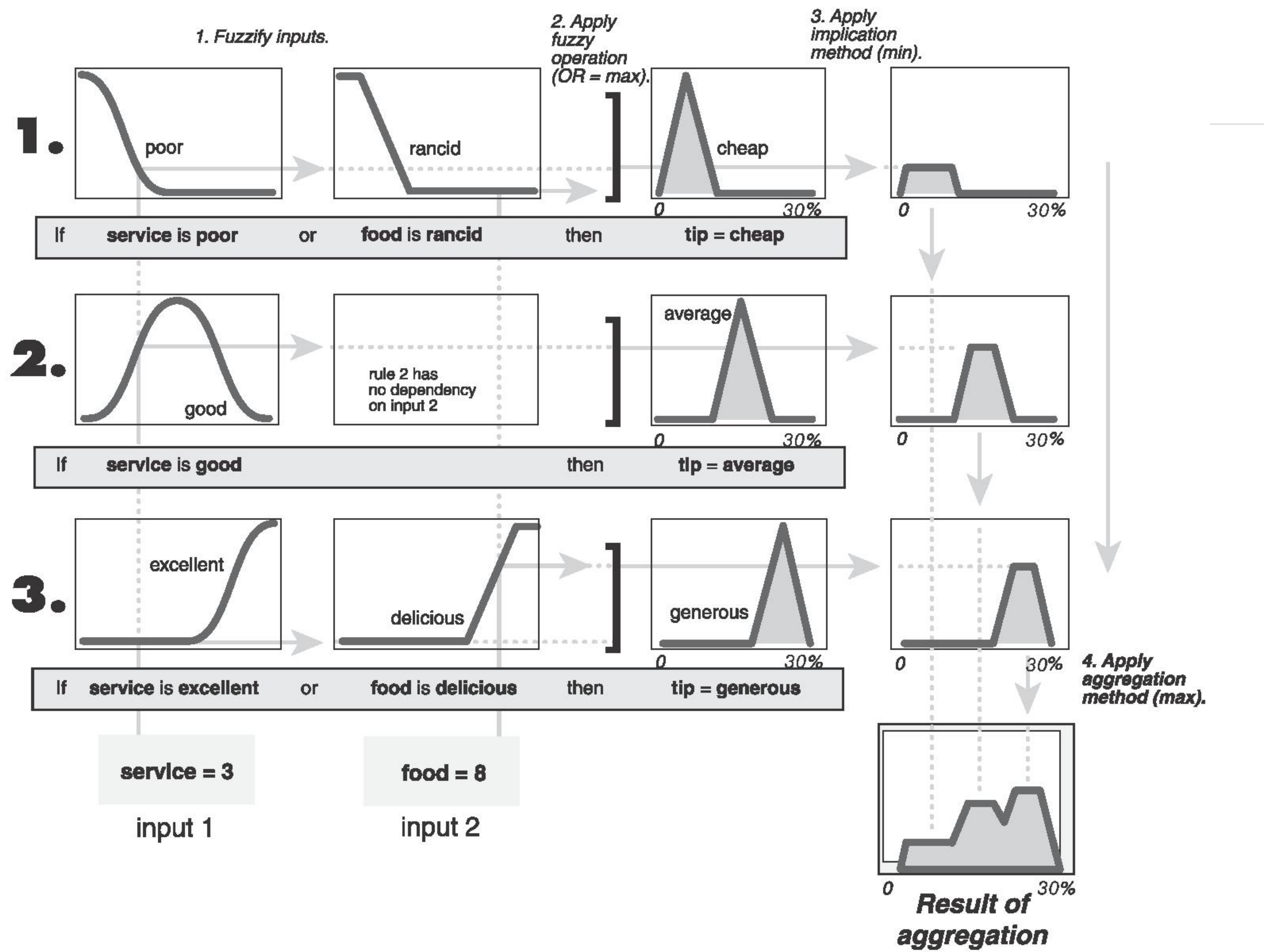
3 Apply Implication Method

The **input** for the implication process is a ***single number*** given by the antecedent, and the **output** is a ***fuzzy set***. *Implication is implemented for each rule.*



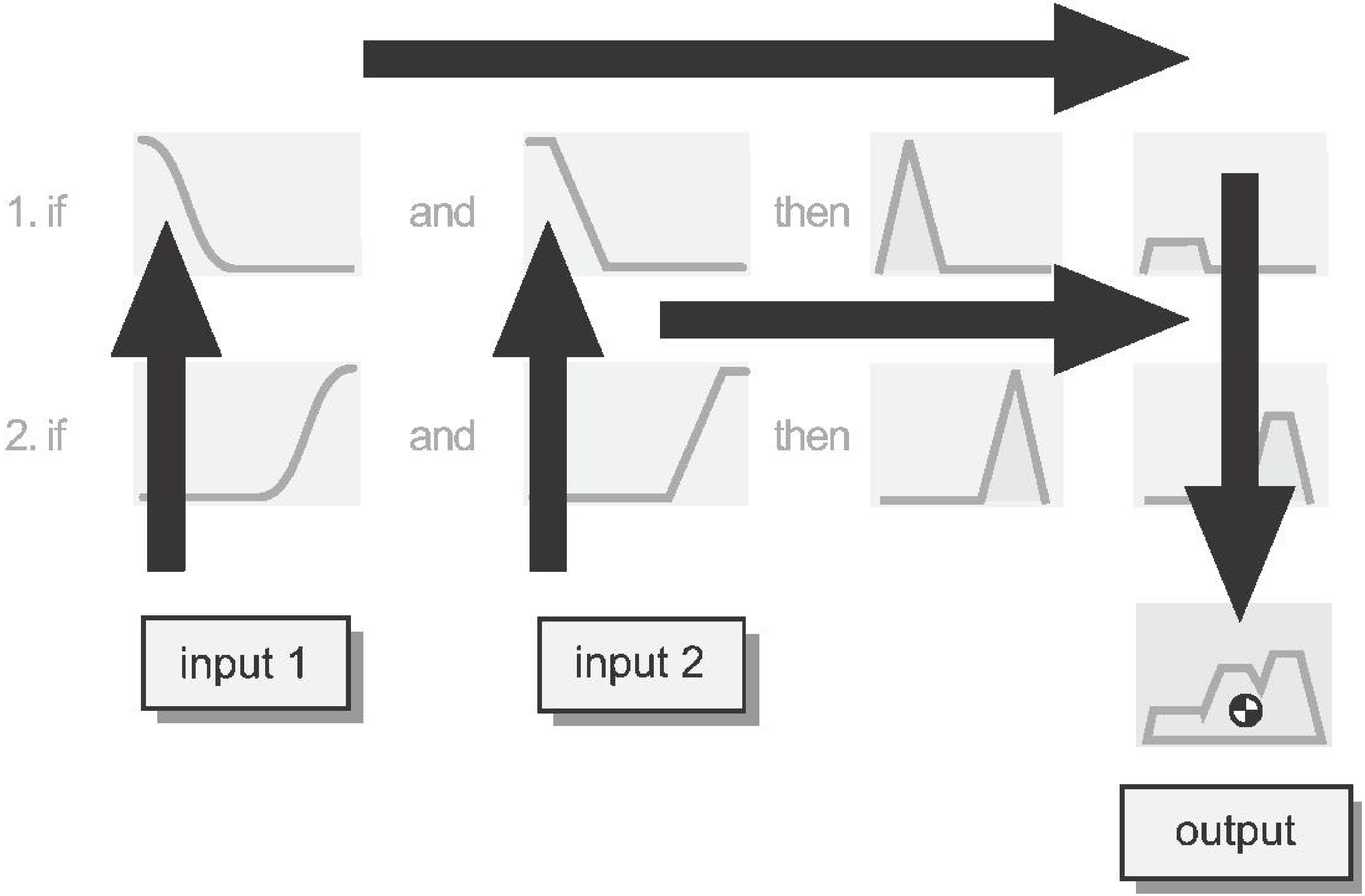
4 Aggregate All Outputs

- Aggregation is the process by which the fuzzy sets that represent the **outputs** of each rule are **combined** into a single fuzzy set.
- Aggregation only occurs once for **each output** variable, which is before the final defuzzification step.
- The **input** of the aggregation process is the list of **truncated output functions** returned by the implication process for each rule.
- The output of the aggregation process is one **fuzzy set** for **each output variable**.



1. Fuzzify inputs.
2. Apply fuzzy operation (OR = max).
3. Apply implication method (min).
1.
poor
rancid
cheap
0
30%
If service is poor or food is rancid then tip = cheap
2.
good
rule 2 has no dependency on input 2
average
If service is good then tip = average
3.
excellent
delicious
generous
If service is excellent or food is delicious then tip = generous
4. Apply aggregation method (max).
service = 3
input 1
food = 8
input 2
Result of aggregation

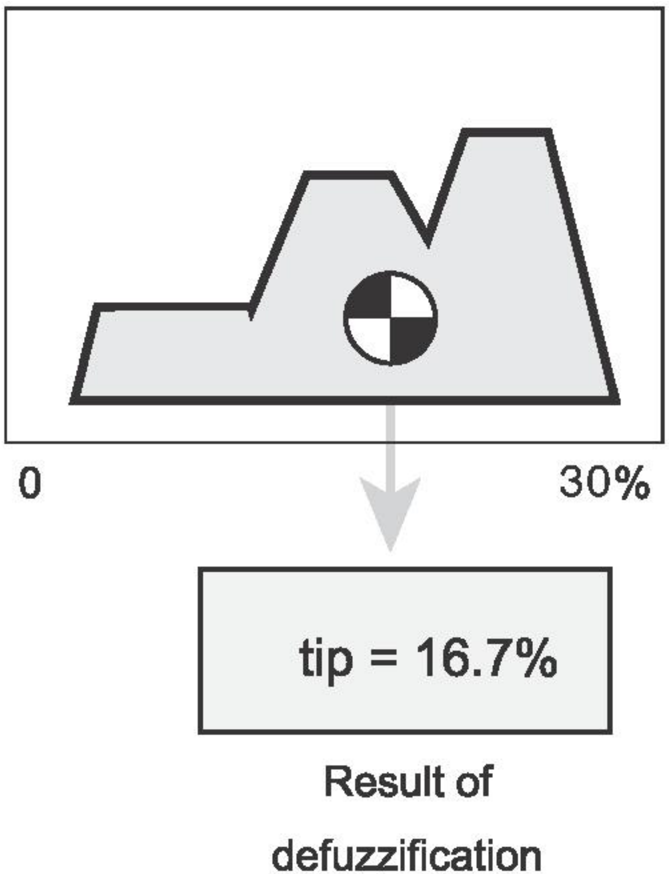
Interpreting the fuzzy inference diagram

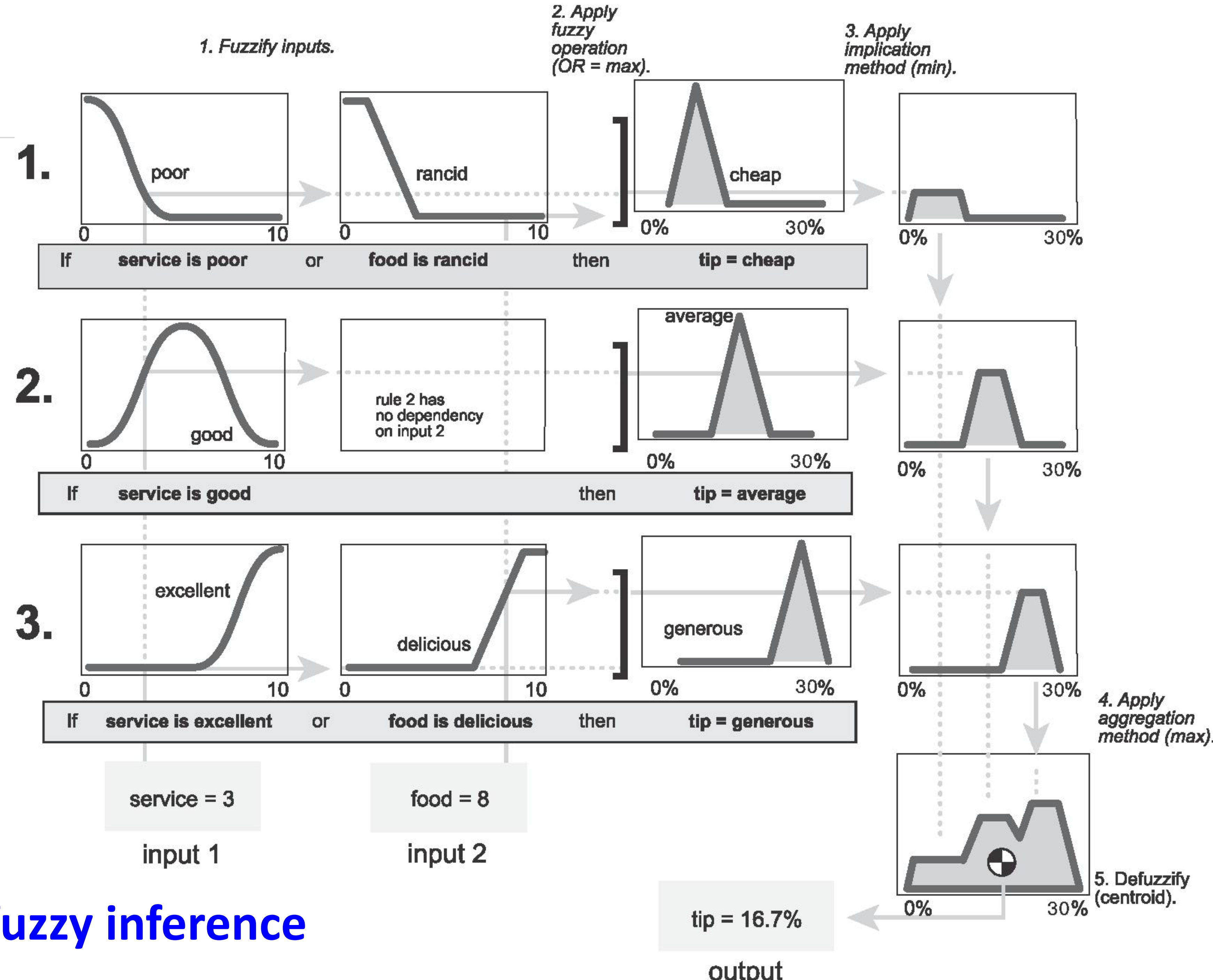


Defuzzify

The **input** for the defuzzification process is a *fuzzy set* (the aggregate **output** fuzzy set) and the output is a *single number*.

Perhaps the most popular **defuzzification** method is the *centroid calculation*, which returns the center of area under the curve





fuzzy inference

Contents

- What is Fuzzy Logic
- **Foundations of Fuzzy Logic**
- **Fuzzy Logic Control**
- Fuzzy Logic in MATLAB/SIMULINK
- Tutorials and Courseworks
- Labs
- Reference

Fuzzy Logic Control

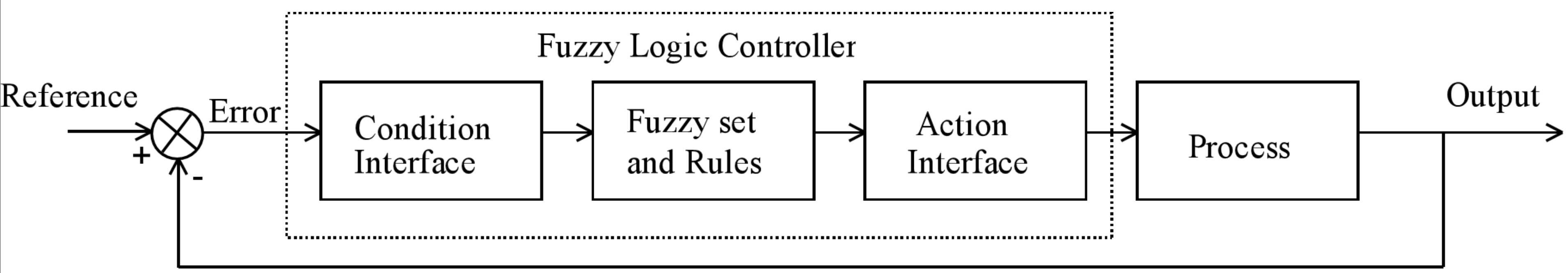
A fuzzy controller consists of a set of *control rules* and each rule is a linguistic statement about the *control action* to be taken for a given process condition given by the following rule structure:

IF <condition> ***THEN*** <control action>

The <condition> is termed as the antecedent and the <control action> is the consequence.

Fuzzy Logic Control

A block diagram of fuzzy logic control systems



Essential steps in designing fuzzy controllers

(a) defining **input** and **output** **decision variables** for fuzzy controllers;

(b) specifying all the **fuzzy sets** and their **membership functions** defined for each input and output **decision variables**;

(c) **converting** the input decision variables to fuzzy sets by a fuzzification technique;

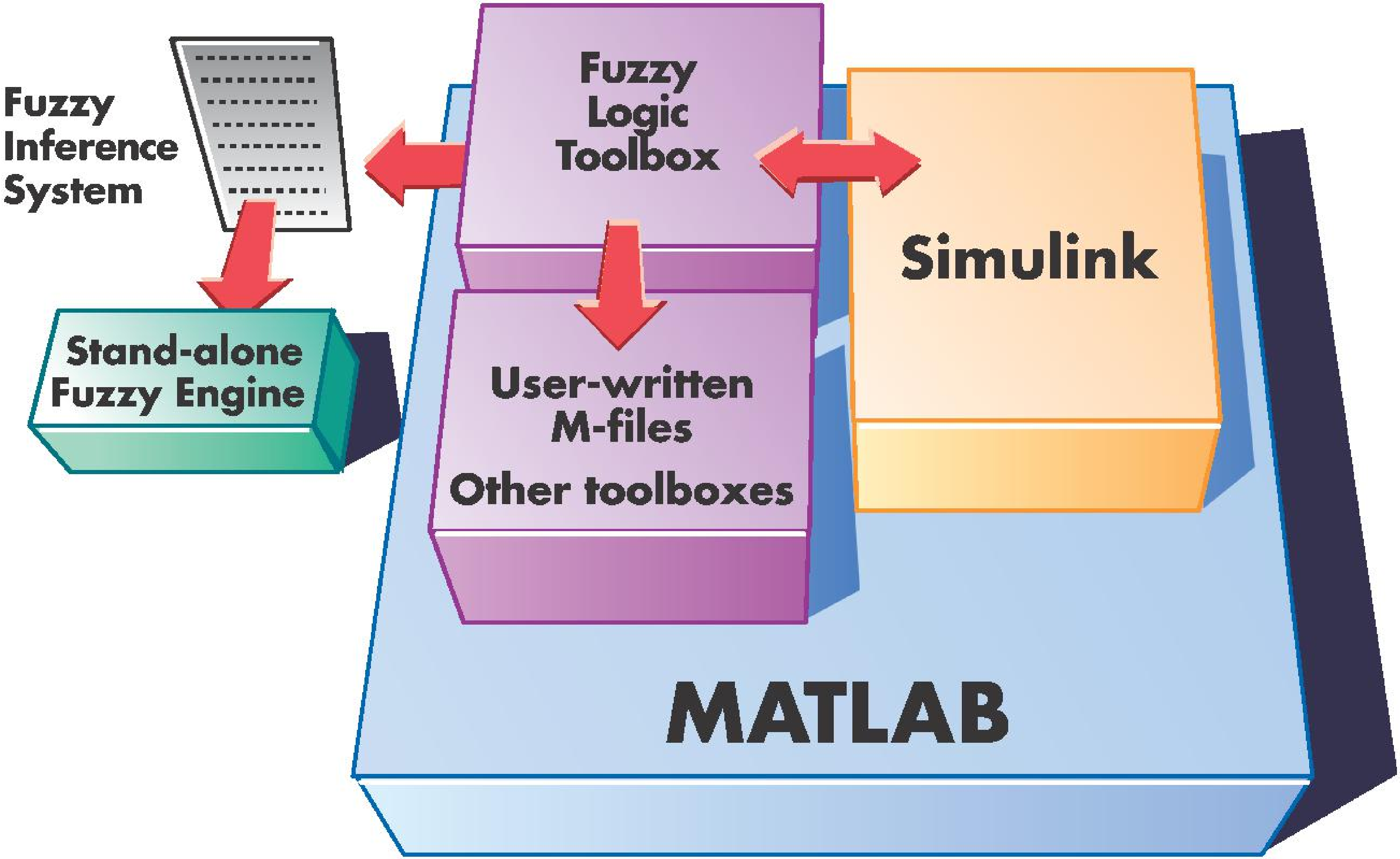
(d) **compilation** of an **appropriate and sufficient** set of control rules that operate on these fuzzy sets, i.e. formulating the fuzzy rule-base which used as an inference engine;

(e) **devising** a method that computes for a single resultant fuzzy control action;

(f) **devising** a transformation method for converting fuzzy control action to crisp value.

Contents

Fuzzy Logic Toolbox



Contents

- What is Fuzzy Logic
- **Foundations of Fuzzy Logic**
- Fuzzy Logic Control
- Fuzzy Logic in MATLAB/SIMULINK
- **Tutorials and Courseworks**
- Lab 1: Simulations on Fuzzy Logic Control of Robotic Vehicle using MATLAB/Simulink
- Reference

Tutorials and Courseworks

FLC - Tutorial and Coursework.PDF

FLC - Tutorial and Coursework.pdf

Contents

Labs

Lab 1: Simulations on Fuzzy Logic Control of Robotic Vehicle using MATLAB/Simulink

Lab 2: fuzzy PID controller[2,3]

Lab 1: Simulations on Fuzzy Logic Control of Robotic Vehicle using MATLAB/Simulink

- **Section A**: **2-DOF robotic vehicle** in MATLAB/SIMULINK (20mins)

- **Section B**: '**2-in-1out**' fuzzy logic control (**FLC**) in S-function (20mins)

- **Section C**: Simulation of **FLC** for **robotic vehicle**

using MATLAB/SIMULINK (20mins)

Tip: **Section A + Section B = Section C**

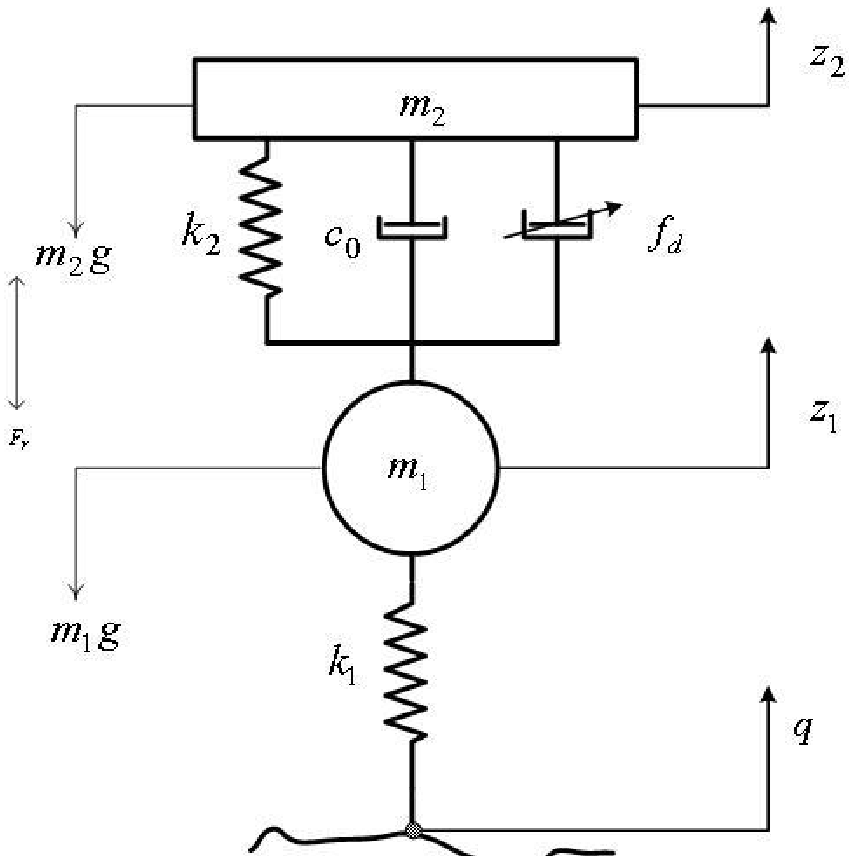
Section A: 2-DOF robotic vehicle in MATLAB/SIMULINK (20mins)

1. Vehicle Suspension System Modelling

Vehicle Suspension System Modelling.pdf

2. FLC&skyhook

FLC&skyhook.pdf



Section B: '2-in-1out' fuzzy logic control (FLC) in S-function (20mins)

What Is an S-Function?

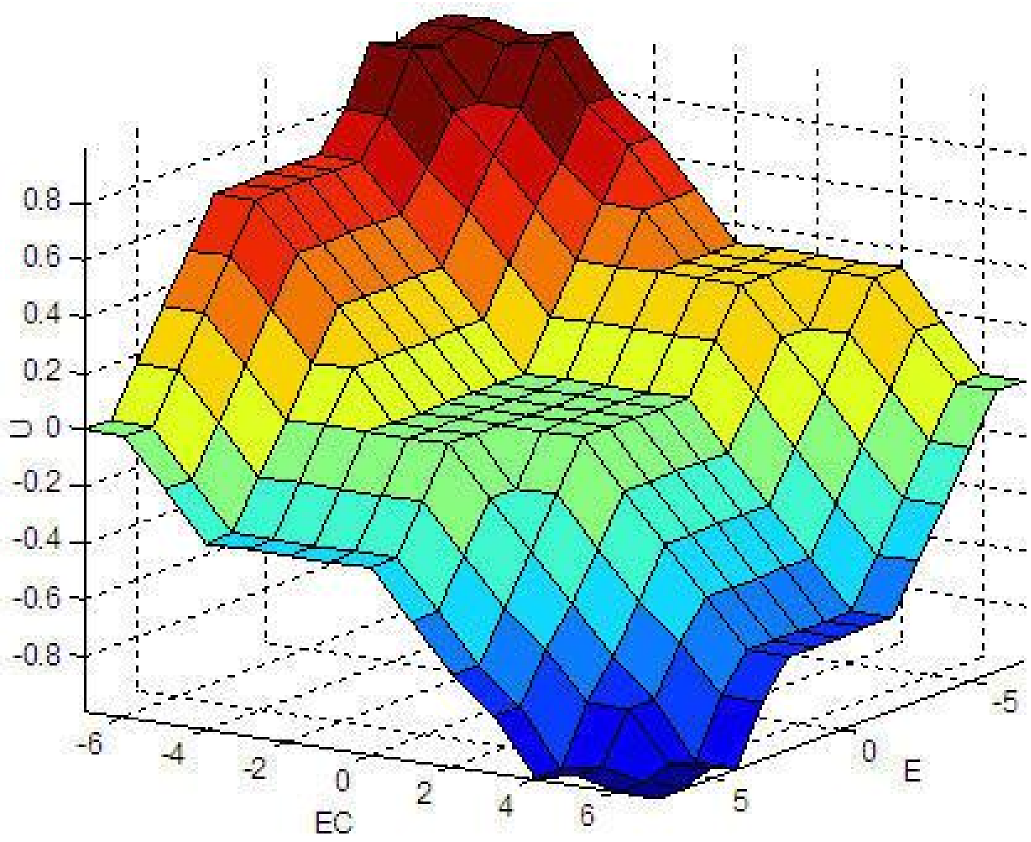
S-functions (system-functions) provide a powerful mechanism for extending the capabilities of the Simulink® environment.

An S-function is a **computer language** description of a Simulink block written in *MATLAB®, C, C++, or Fortran. C, C++, and Fortran* S-functions are compiled as MEX files using the mex utility.

As with other MEX files, S-functions are *dynamically linked* subroutines that the MATLAB execution engine can automatically load and execute.

Fuzzy Rules

SGA__suspension_flc_std_2in1out_sfunction.m



[rule_base_7rules]=...

[1 1 7 1 1

2 1 7 1 1

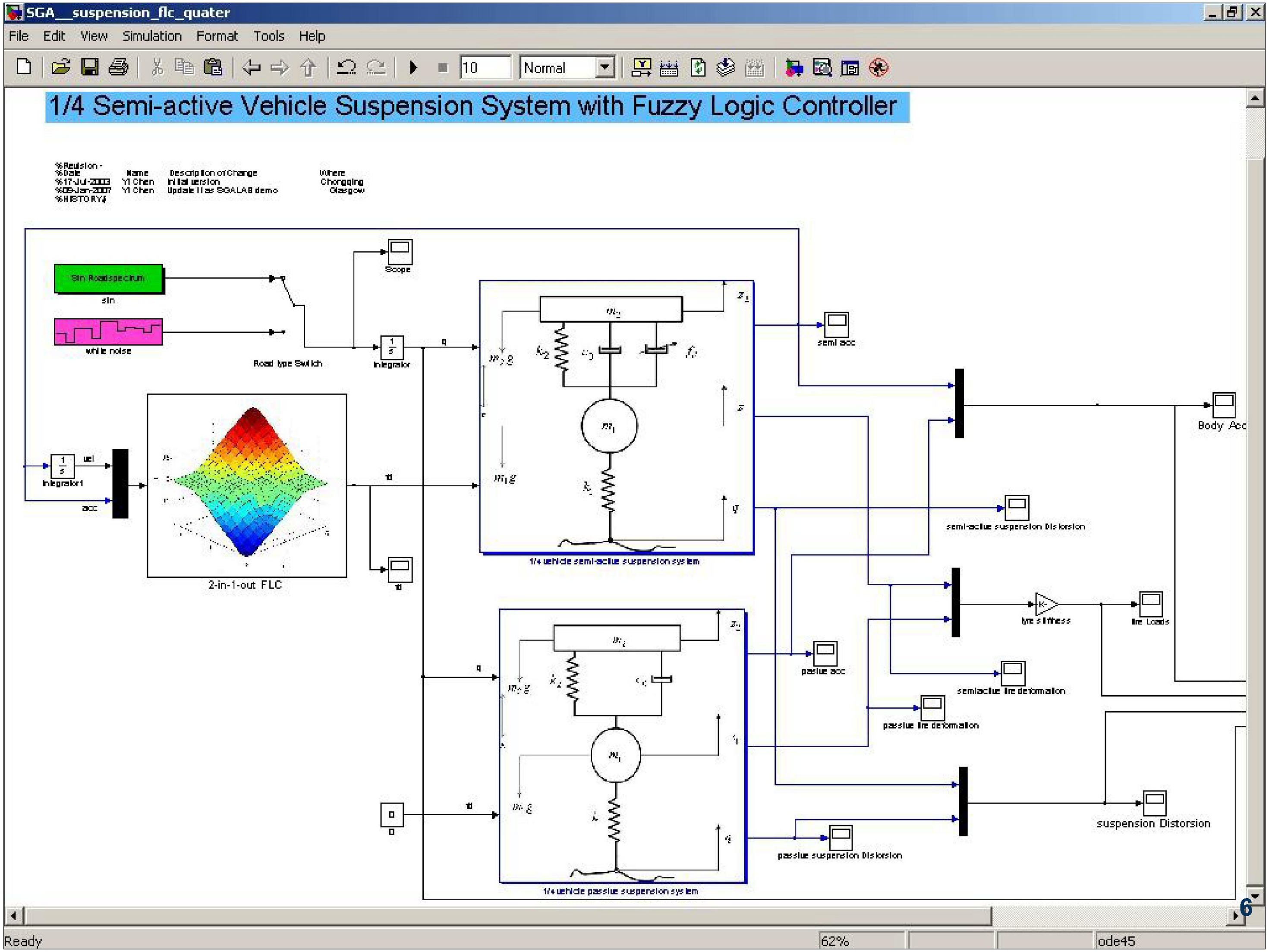
...

3 1 6 1 1

7 7 1 1 1];

Section C: Simulation of FLC for robotic vehicle using MATLAB/SIMULINK (20mins)

SGA__suspension_flc_std_2in1out.mdl



Lab 2: fuzzy PID controller[2,3]

MathWorks® Products Solutions Academia Support Community Events

PID Control with MATLAB and Simulink

Search MathWor

Design and implement PID controllers

PID control is ubiquitous. While simple in theory, design and implementation of PID controllers can be difficult and time consuming in practice.

PID control involves several tasks that include:

- Selecting an appropriate PID algorithm (P, PI, or PID)
- Tuning controller gains
- Simulating the controller against a plant model
- Implementing the controller on a target processor

Contents

Reference

[1] Fuzzy Logic Toolbox™ User's Guide

[2] PID controller

http://en.wikipedia.org/wiki/PID_controller

[3] PID Control with MATLAB and Simulink

http://uk.mathworks.com/discovery/pid-control.html

END

Fuzzy Logic Control

- an introduction

Dr Leo Chen

leo.chen@ieee.org

02/Jan/2019