



人工智能导论

模糊逻辑与模糊系统

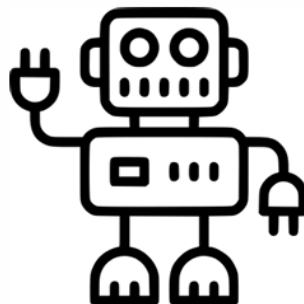


Dr Leo Chen

leo.chen@ieee.org

课程目录

1. 导论
2. 进化计算
3. 人工神经网络
4. **模糊逻辑与模糊系统**
5. 更多AI子集
6. AI与工业4.0



章节内容

引言

什么是模糊逻辑

为什么使用模糊逻辑?

模糊逻辑基础

模糊系统与应用

课堂讨论

阅读列表

常见问题解答

附录

参考文献

引言

- 模糊的

形容词 不清楚或含糊的

/ˈfazi/

模糊逻辑类似于人类决策方法，处理模糊和不精确的信息。[1]

这是对现实世界问题的严重简化，基于真实度，而不是通常的真/假或 1/0 布尔逻辑。

引言

- 一个持续变化的事件、过程或函数不能总是被定义为真或假，这意味着我们需要以模糊的方式定义此类活动。
在现实世界中，当我们无法确定状态是真还是假时，模糊逻辑为推理提供了非常有价值的灵活性。
模糊逻辑由 Lofti A. Zadeh 于 1965 年在其研究论文《模糊集》中提出。他被认为是模糊逻辑之父。

Lotfi A. Zadeh (1921–2017) [2-4]

- "随着复杂性的增加，精确的陈述失去意义，而有意义的陈述失去精确性。"
-- Lotfi A. Zadeh ('模糊逻辑'之父)



课堂讨论

- 问问自己或你的朋友：
今天冷吗？
今天天气好吗？
你有多爱我？（一万年？）
我学一门新语言（精灵语）年纪太大了吗？
为什么快乐的时候时间过得那么快？你有多快乐？

什么是模糊逻辑

- 模糊逻辑不是指逻辑本身是模糊的，而是用于描述模糊性的逻辑。
- 模糊逻辑为推理提供了非常有价值的灵活性。
- 当将任务/系统视为黑盒（过于复杂或难以处理）时，模糊逻辑学习人类操作。
- 人类处理模糊和不精确数据的能力使我们能够轻松完成此类任务。
- 必须构建模糊逻辑，使其能够处理模糊信息。

为什么使用模糊逻辑？

- 我们的目标是复制人类操作员的控制动作，我们必须能够对操作员的活动进行建模，而不是对工厂本身进行建模。
- 许多任务对人类来说很简单，但对机器来说却构成了持续的挑战：
 - 驾驶汽车
 - 停车
 - 在杂乱的环境中行走
 - 搬运易碎物品

模糊逻辑基础 [5,6]

- 清晰集与模糊集
隶属函数
架构与模糊推理系统
优点与缺点

清晰集

- 经典逻辑基于清晰集，其中一组不同的对象被视为一个集合。
示例01，颜色：白色和红色都是不同的对象，但可以使用记号 {红色, 白色} 将它们视为一个集合。
按照惯例，清晰集用大写字母 F 表示，因此上述示例可以描述为：

$$F = \{\text{red, white}\}$$

清晰集

- 可以从一个更广泛的集合中定义一个清晰子集，其中该集合的元素根据某些条件属于该子集。

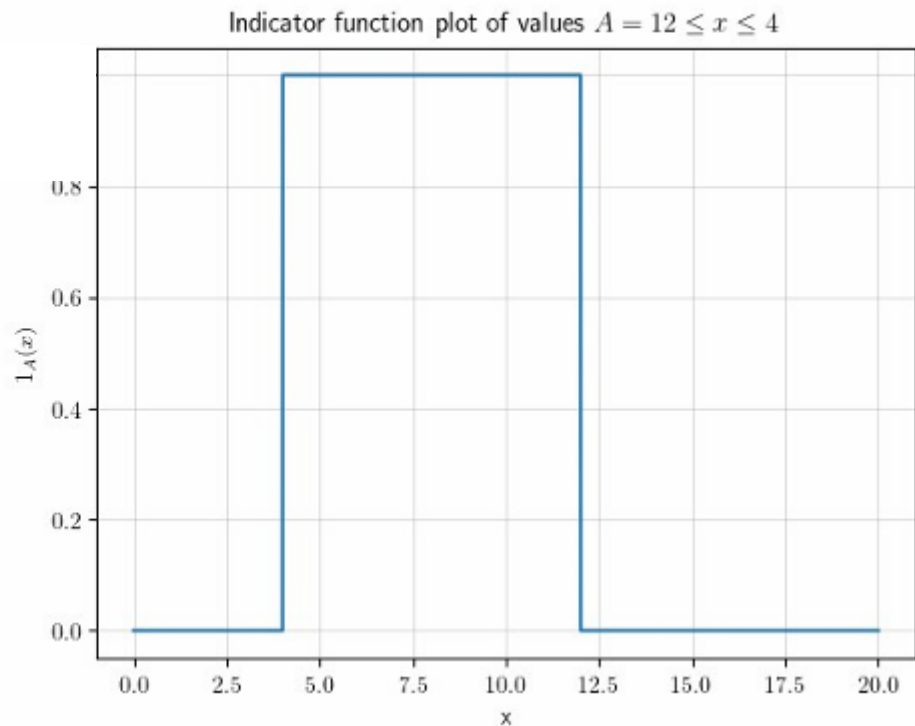
示例02，集合 A 可以定义为大于等于4且小于等于12的数字集合。这个陈述可以使用以下记号来描述：

$$A = \{i \mid i \text{ is an integer and } 4 \leq i \leq 12\}$$

清晰集

图形化表示:

$$1_A(x) = \begin{cases} 1 & \text{if } 4 \leq x \leq 12 \\ 0, & \text{otherwise} \end{cases}$$

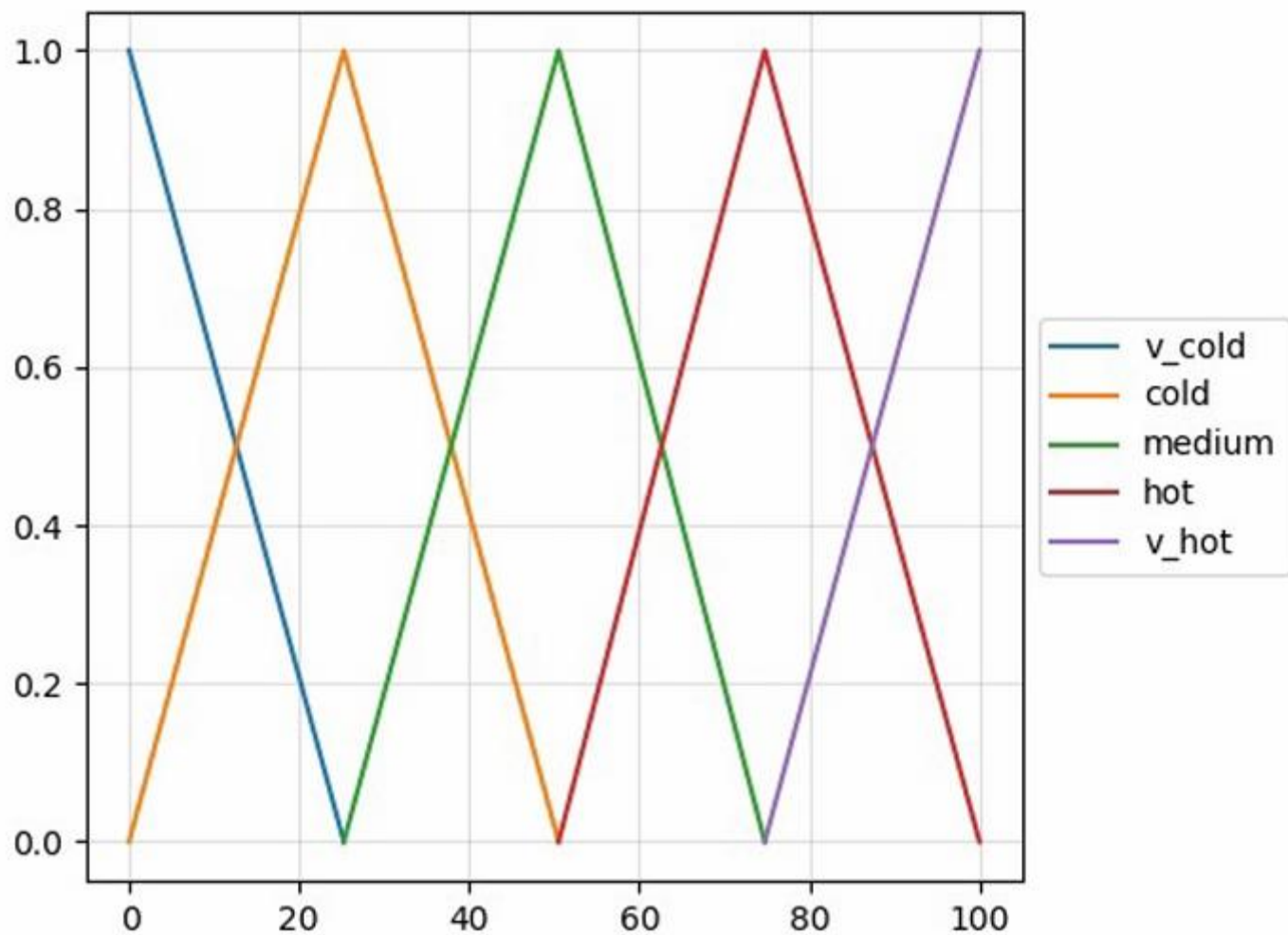


模糊集

- 模糊集由 Lotfi Zadeh 于 1965 年提出。
- 与清晰集不同，模糊集允许部分属于一个集合，这由隶属度定义，记为 μ ，可以取从 0（元素完全不属于该集合）到 1（元素完全属于该集合）的任何值。
- 显然，如果我们移除除 0 和 1 之外的所有隶属值，模糊集将退化为前一节描述的清晰集。

隶属函数

- 集合的隶属函数是集合元素与其隶属度之间的关系。
- 下面显示了如何将隶属函数应用于温度的一个示例：
- *模糊集描述了发动机的温度范围，从非常冷到非常热。*



隶属函数

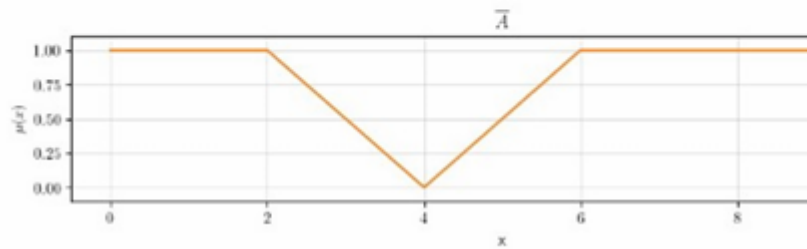
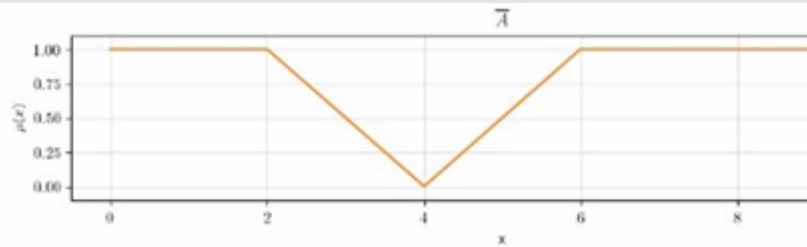
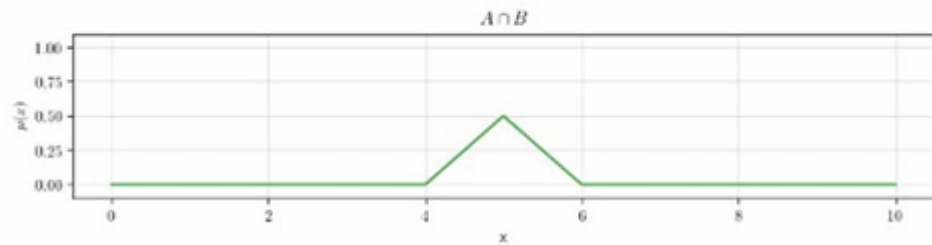
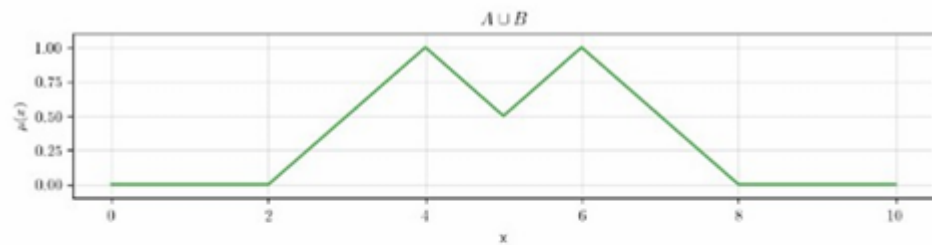
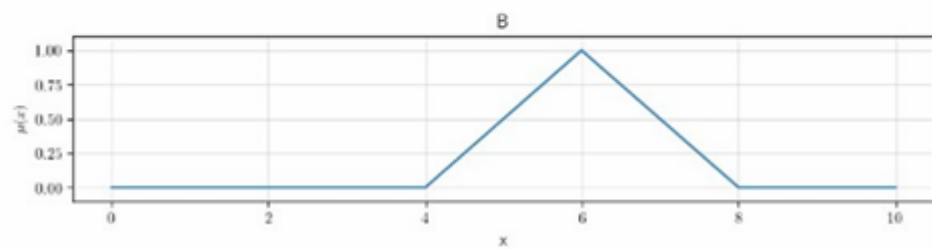
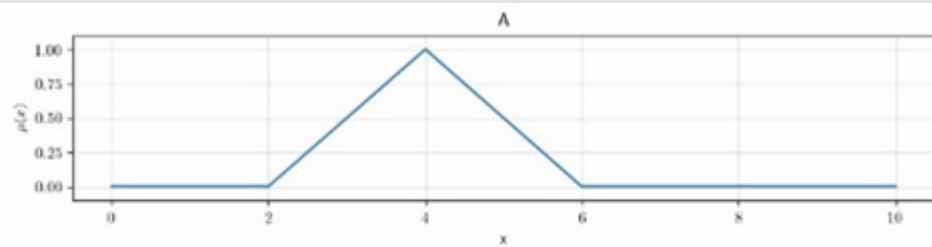
- 隶属函数值 μ 是集合中的隶属量。
- 例如：在温度为 80 度时，发动机可以被描述为热的程度为 0.8，非常热的程度为 0.2。
- 计算两个模糊集**并集**的最常见**方法**是：两个集合上的**最大算子**、**最小算子**、**乘积算子**

隶属函数

- **模糊集的补集**是通过用 1 减去集合隶属函数来计算的。
- 一个关键的观察是，一个元素可以在一个集合中及其补集中同时具有隶属度。
- 例如，元素 x 可以在集合 A 中，也可以在 '非 A ' 中，具有特定的隶属度。
- 定义一个图，说明如何将输入空间中的每个点映射到 0 和 1 之间的隶属值。

隶属函数的三种模糊化器类型：

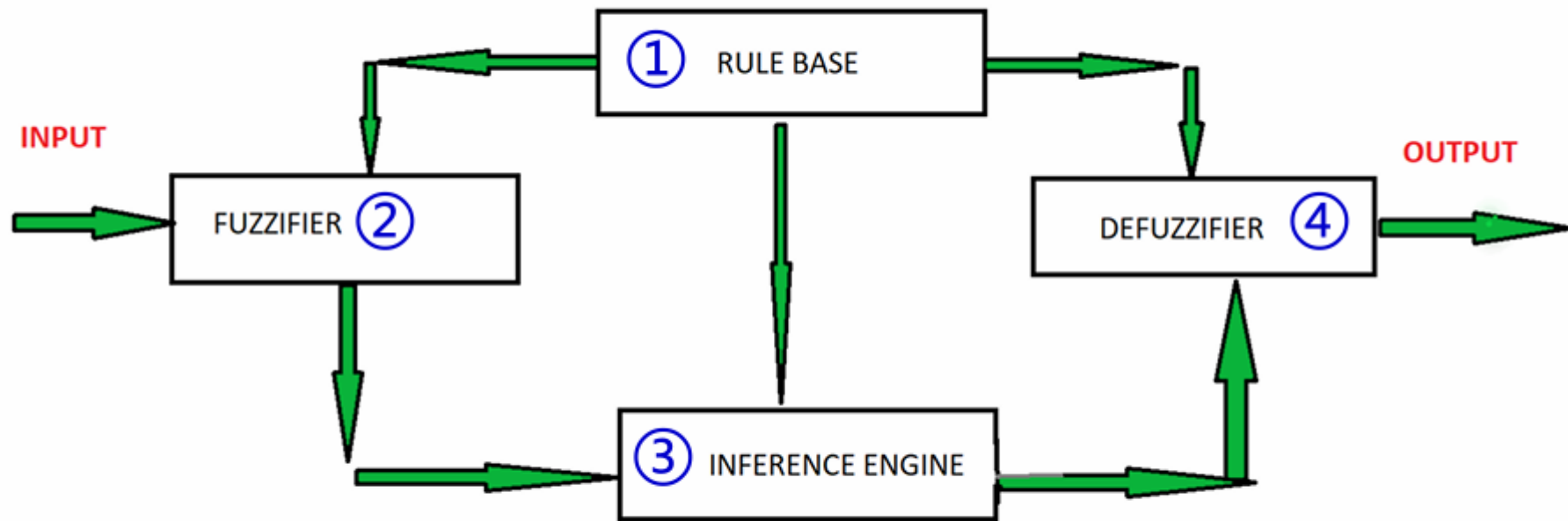
- 单值模糊化器
- 高斯模糊化器
- 梯形或三角形模糊化器



架构（配置）

- '模糊'系统的架构包含四个部分：
- 规则库
- 模糊化（模糊化器）
- 推理引擎
- 去模糊化（去模糊化器）

架构



FUZZY LOGIC ARCHITECTURE

1) 规则库

- 它包含专家提供的规则集和 **IF-THEN** 条件，用于基于**语言信息**来指导决策系统。
- 模糊理论的最新发展提供了几种设计和调整模糊控制器的有效方法。
- 这些发展大多减少了模糊规则的数量。

②模糊化（模糊化器）

- 它用于通过隶属函数将输入，即清晰数，转换为模糊集。
- 清晰输入基本上是由传感器测量并传递到控制系统进行处理的精确输入，例如温度、压力、转速等。

3) 推理引擎

- 它确定当前**模糊输入**相对于每条规则的**匹配度**，并根据输入字段决定要触发哪些规则。
- 接下来，使用计算两个模糊集并集的方法（最大算子、最小算子、乘积算子等）将触发的规则**组合**起来以形成控制动作。

去模糊化（去模糊化器）

- 它用于将推理引擎获得的模糊集转换为清晰值。
- 有几种去模糊化方法 [7]，使用最适合特定专家系统的方法来减少误差。
- 最大隶属度法
- 质心法
- 加权平均法
- 平均最大隶属度法
- 和中心法
- 最大面积中心法

模糊推理系统

- 模糊系统是模糊专家知识的存储库，可以对模糊数据进行推理。
- 专家知识是模糊隶属函数和一组模糊规则的集合，称为规则库，形式如下：
- IF (条件满足)
- THEN (推断出结果)
- Mamdani 推理模型（最大-最小推理方法）
- Takagi-Sugeno 推理模型 (*IF x IS X and y IS Y THEN $z=f(x,y)$*)

优点

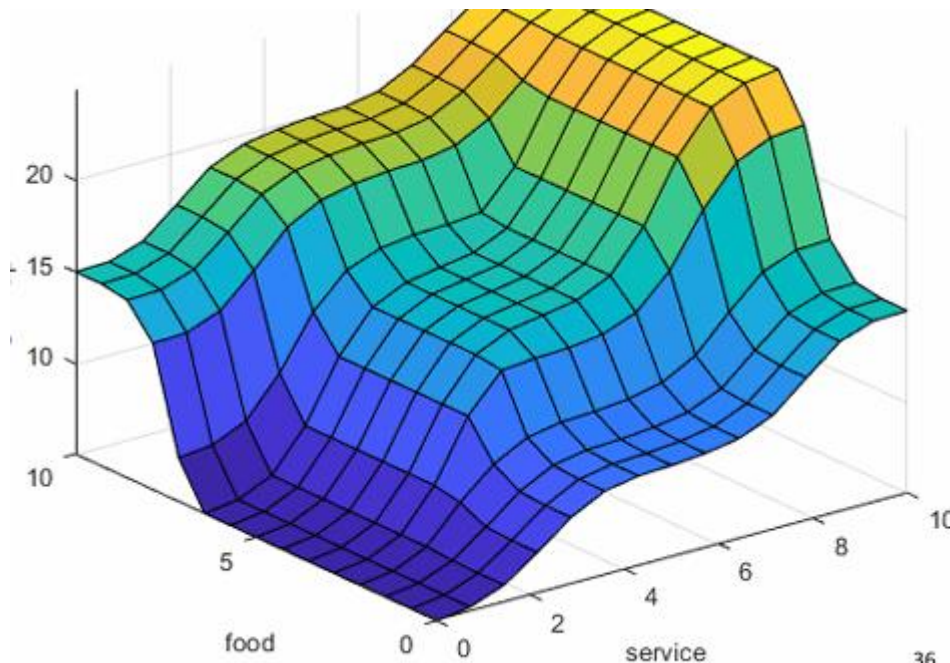
- 该系统可以处理任何类型的输入，无论是**不精确的、失真的还是有噪声的**输入信息。
- 模糊逻辑系统的构建**容易且易于理解**。
- 模糊逻辑带有**集合论的数学概念**，其**推理**相当简单。
- 它为生活所有领域的复杂问题提供了非常**有效的解决方案**，因为它类似于人类的推理和决策。
- 算法可以用**少量数据**描述，因此所需内存很少。

缺点

- 许多研究人员提出了不同的方法通过模糊逻辑来解决给定问题，这导致了模糊性。没有系统的方法通过模糊逻辑来解决给定问题。
- 在大多数情况下，其特性的证明是困难或不可能的，因为我们并非每次都能得到我们方法的数学描述。
- 由于模糊逻辑既处理精确数据也处理不精确数据，因此大多数时候准确性会受到影响。

模糊系统 – 基本小费问题[9]

- 给定一个从 0 到 10 的数字，代表餐厅的**服务质量**（其中 10 是优秀），小费应该是多少？
- 如果服务差或食物难吃，小费便宜
- 如果服务好，小费中等
- 如果服务优秀或食物美味，小费慷慨



应用

- 航空航天：航天器和卫星的控制
- 汽车：速度控制、交通控制
- 决策支持系统
- 化学工业用于控制 pH、干燥、化学蒸馏过程。
- 自然语言处理
- 专家系统。
- 与神经网络协同工作，因为它模仿人的决策方式，速度更快。
- 数据聚合并通过形成部分真值作为模糊集将其转换为更有意义的。

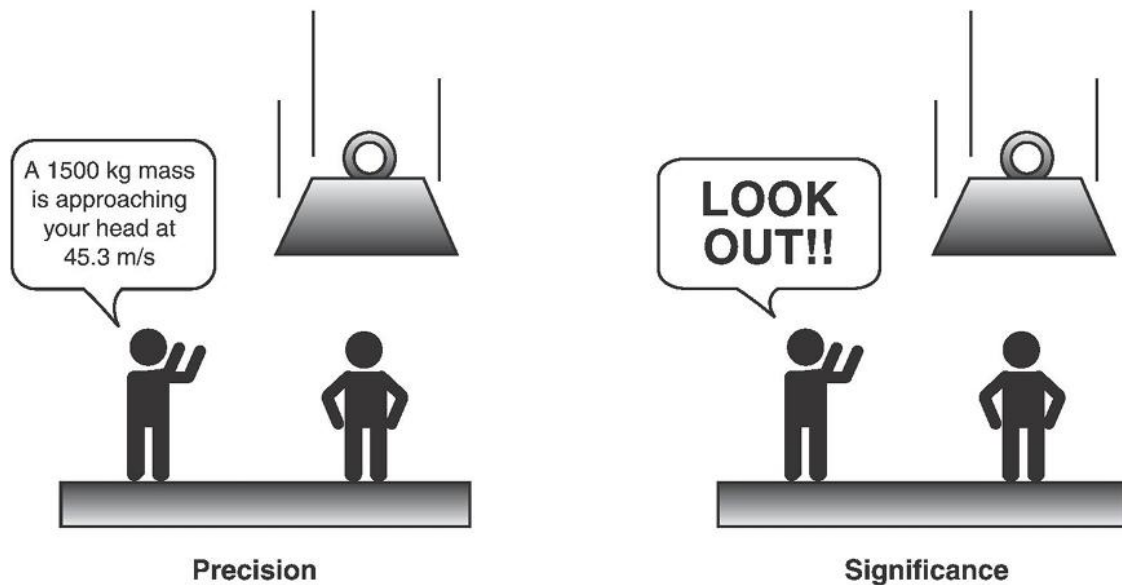
章节内容

- 什么是模糊逻辑
- 模糊逻辑基础
- 模糊逻辑控制
- MATLAB/SIMULINK 中的模糊逻辑
- 教程与课程作业
- 实验
- 参考文献

什么是模糊逻辑 (FL)

- 模糊逻辑有两种不同的含义：
- 在**狭义**上，模糊逻辑是一种**逻辑**系统，是多值逻辑或**布尔**逻辑的扩展。
- 在**广义**上，模糊逻辑 (FL) 几乎与模糊集理论同义，该理论涉及具有不清晰边界的类，其中隶属度是**程度问题**。
- **FL 的基本概念**是语言变量的概念，即其值是**词语**而非**数字**的变量。

现实世界中的精确性与意义



- 模糊逻辑完全是关于**精确性**的相对**重要性**：当一个粗略的答案就足够时，完全正确有多重要？ [1]

为什么使用模糊逻辑？ (1/4)

- 关于模糊逻辑的一般观察列表：
 - **1 模糊逻辑在概念上易于理解。**
 - *模糊推理背后的数学概念非常简单。模糊逻辑是一种更直观的方法，没有深远的复杂性。*
 - **2 模糊逻辑是灵活的。**
 - *对于任何给定的系统，可以轻松地添加更多功能，而无需从头开始。*

为什么使用模糊逻辑？ (2/4)

- **3 模糊逻辑对不精确数据具有容错性。**
 - 如果你仔细观察，一切都不精确，但更重要的是，即使仔细检查，大多数事情也是不精确的。模糊推理将这种理解构建到过程中，而不是在最后附加它。
- **4 模糊逻辑可以对任意复杂度的非线性函数进行建模。**
 - 你可以创建一个模糊系统来匹配任何一组输入-输出数据。自适应技术（如自适应神经模糊推理系统(ANFIS)）使这个过程特别容易，这些技术在模糊逻辑工具箱软件中可用。

为什么使用模糊逻辑？ (3/4)

- **5 模糊逻辑可以建立在专家的经验之上。**
- 与神经网络直接相反，神经网络获取训练数据并生成不透明、难以理解的模型，而模糊逻辑让你可以依赖已经理解你系统的人的经验。
- **6 模糊逻辑可以与传统控制技术结合。**
- 模糊系统不一定取代传统的控制方法。在许多情况下，模糊系统增强了它们并简化了它们的实现。

为什么使用模糊逻辑？ (4/4)

- 7 模糊逻辑基于自然语言。
- 模糊逻辑的基础是人类交流的基础。这一观察支撑了许多关于模糊逻辑的其他陈述。因为模糊逻辑建立在日常语言中使用的定性描述结构之上，所以模糊逻辑易于使用。
- 最后一个陈述也许是最重要的一个，值得更多讨论。普通人日常使用的**自然语言**经过数千年人类历史的塑造，变得方便而高效。用普通语言写成的句子代表了高效沟通的胜利。

何时不使用模糊逻辑

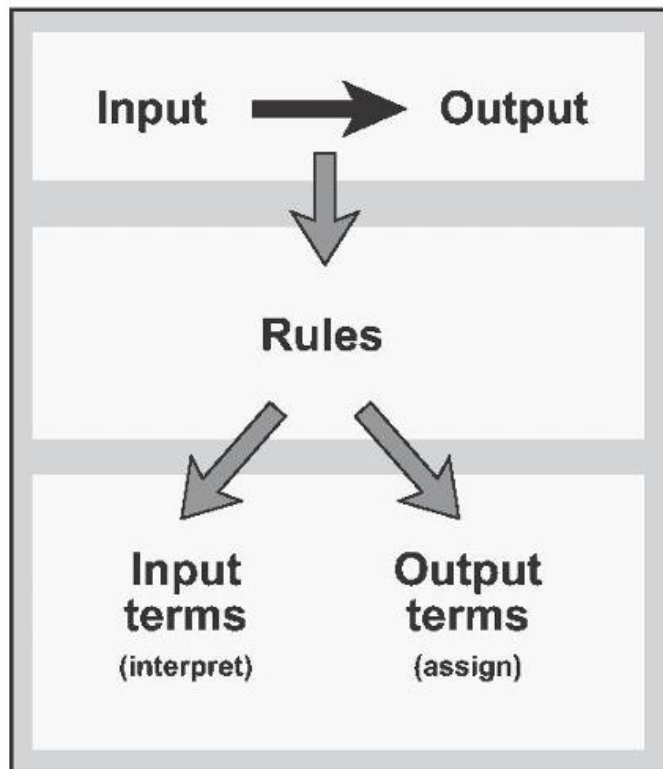
- 模糊逻辑**不是万灵药**。你什么时候不应该使用模糊逻辑？
- 模糊逻辑是将输入空间映射到输出空间的一种便捷方法。
- **如果你发现它不方便，试试别的。** 如果存在更简单的解决方案，就使用它。模糊逻辑是常识的编撰——当你实现它时，运用非凡的常识，你可能会做出正确的决定。

模糊逻辑基础 - 概述

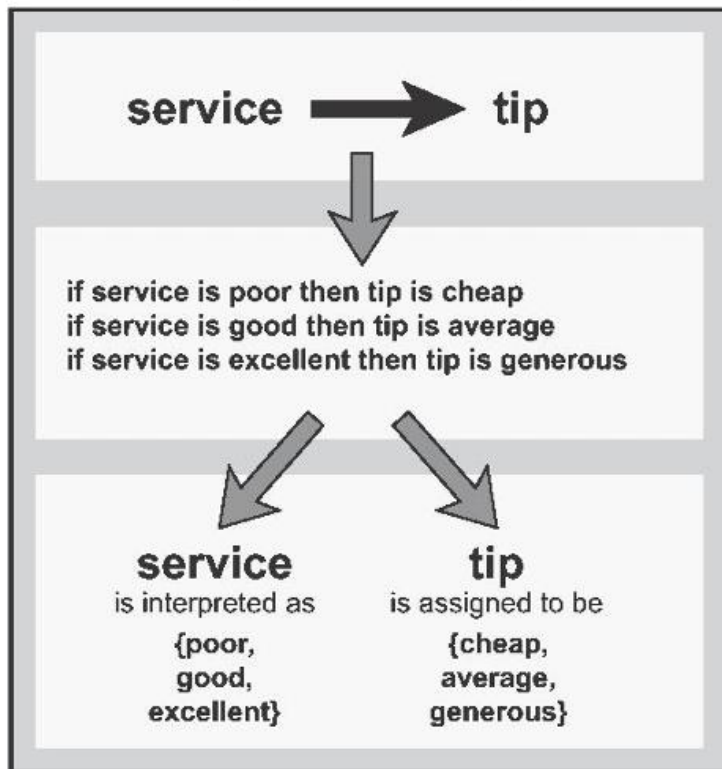
- 模糊逻辑的重点是**将**一个输入空间映射到一个输出空间。
- 实现这一目标的主要机制是一组称为**规则**的 if-then 语句。
- 所有规则都是并行评估的，规则的顺序不重要。
- 在构建解释规则的系统之前，你必须**定义**你计划使用的所有术语以及描述它们的形容词。

模糊推理过程路线图。

一般情况

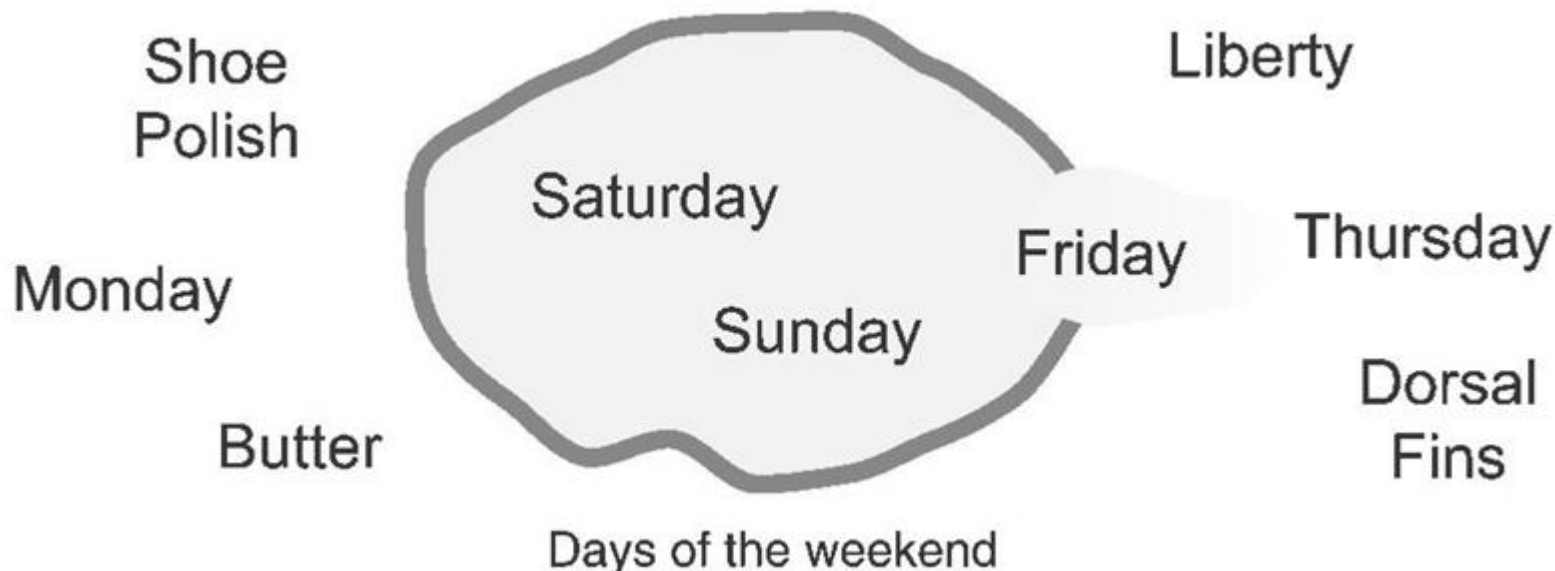


具体示例



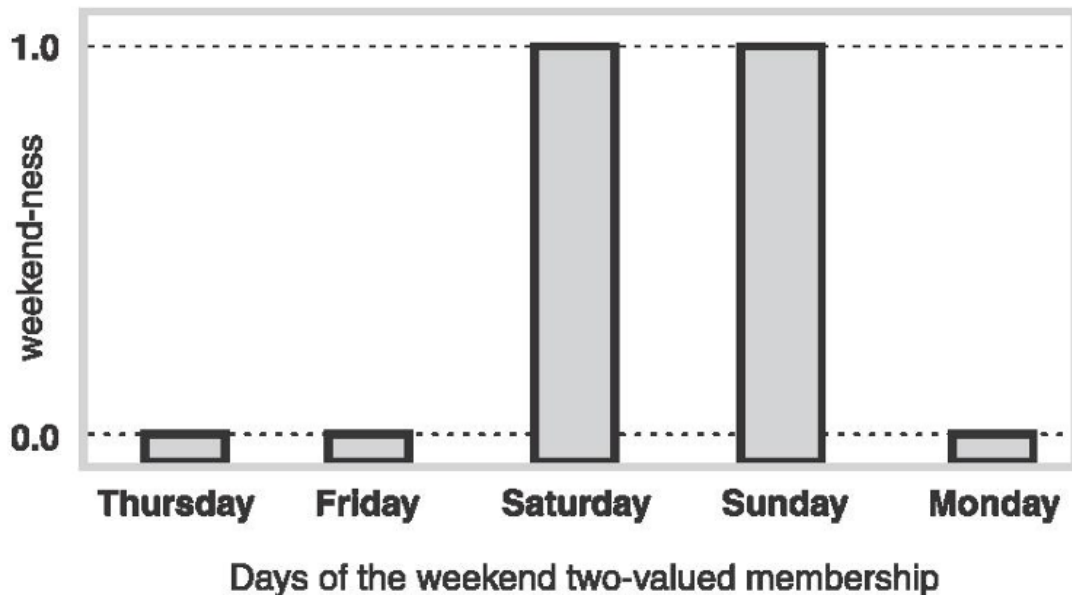
模糊集

- 模糊集是一个没有清晰、明确定义边界的集合。它可以包含仅具有部分隶属度的元素。
- **经典集合**是一个完全包含或完全排除任何给定**元素**的容器。在模糊逻辑中，任何陈述的真值都成为**程度问题**。



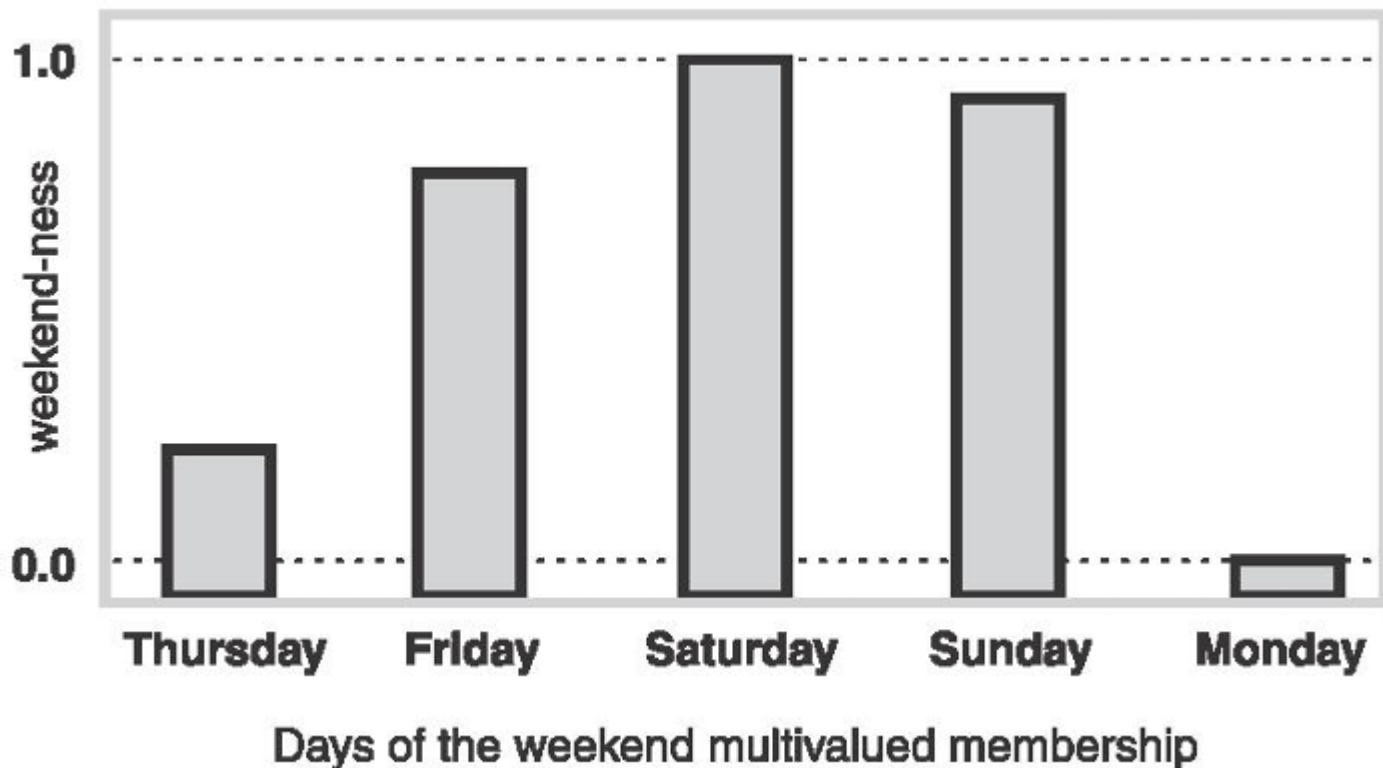
它是如何工作的？

- 模糊逻辑中的推理只是将熟悉的是-否（布尔）逻辑推广。如果你给真赋予数值 1，给假赋予数值 0。



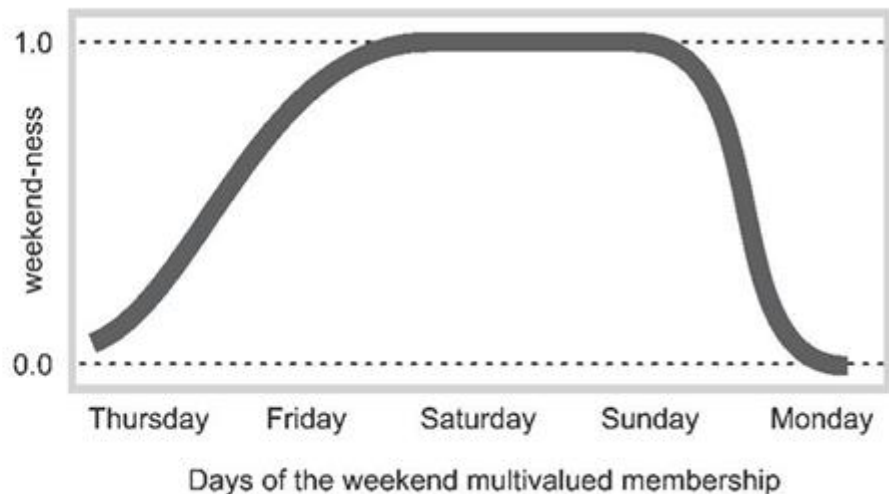
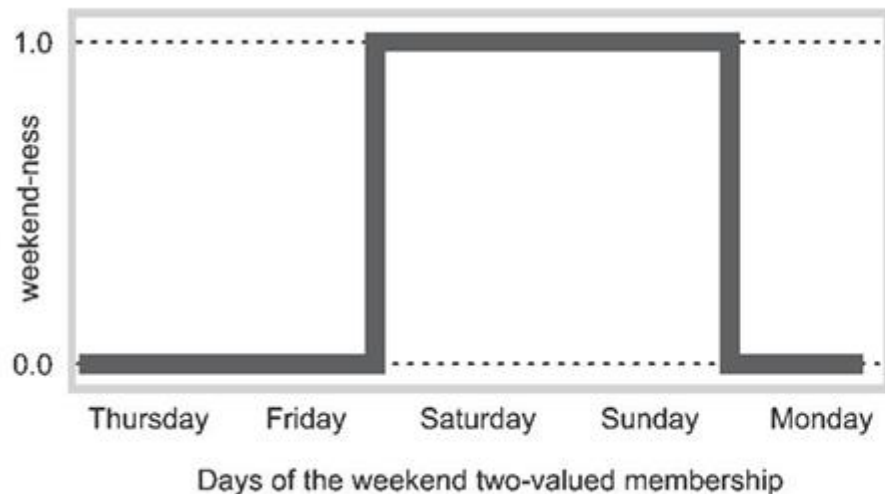
它是如何工作的？

- 这个值表明**模糊逻辑**也允许中间值，如 0.2 和 0.7453。



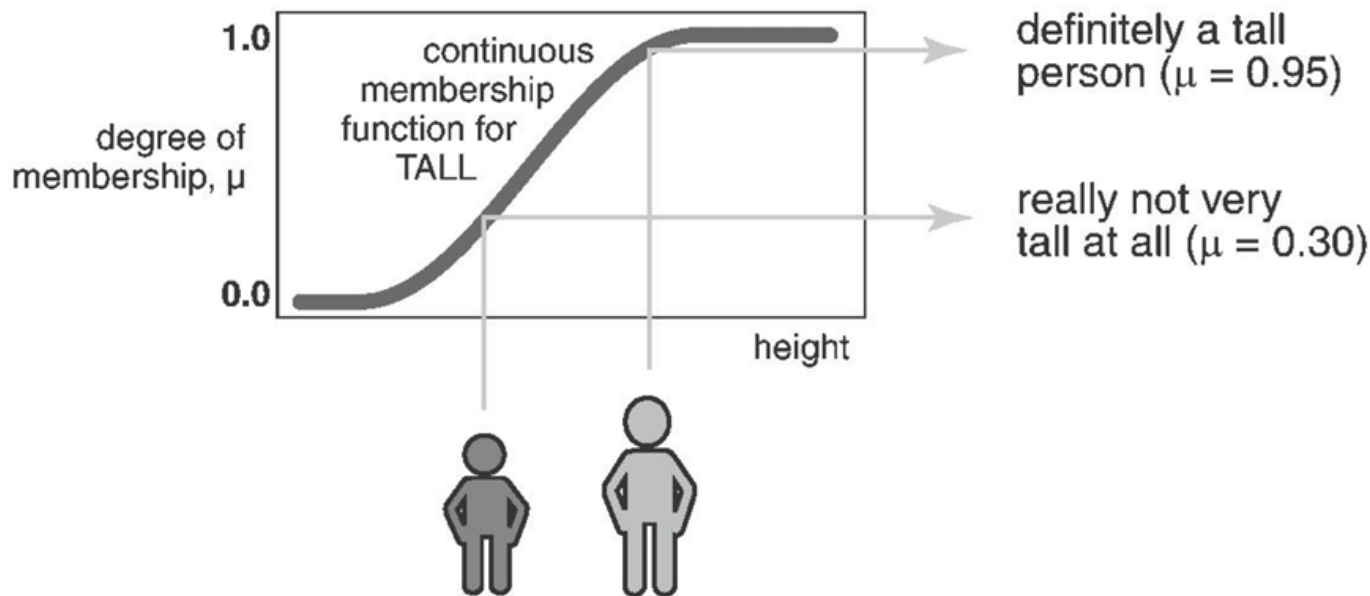
多值逻辑 vs. 二值（或二值是-否）逻辑

- 考虑以下图中显示的周末属性的连续时间比例图。



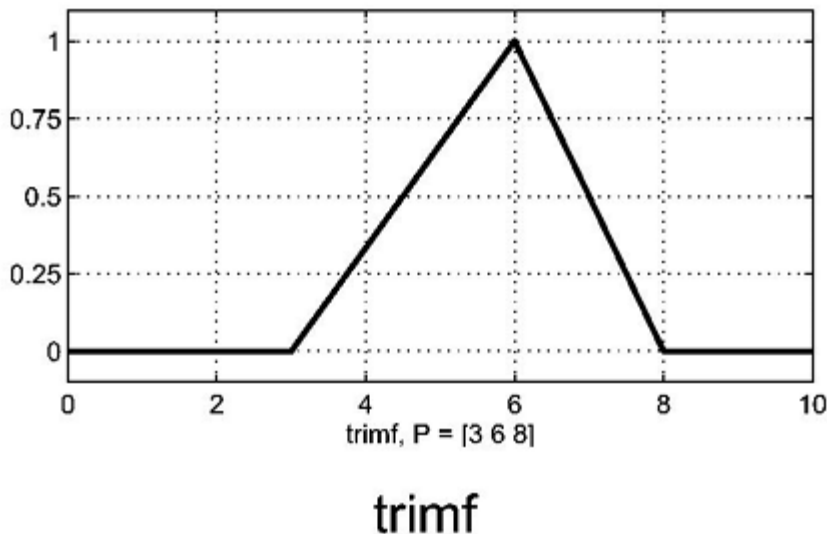
隶属函数

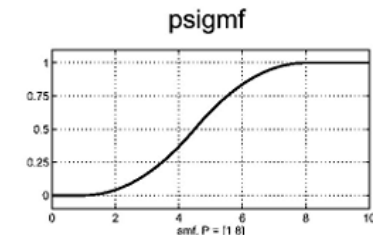
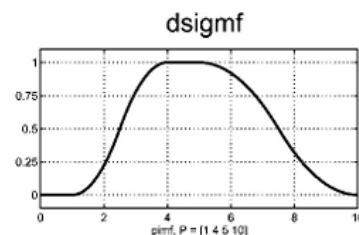
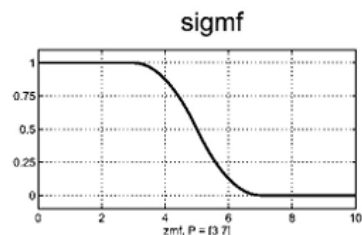
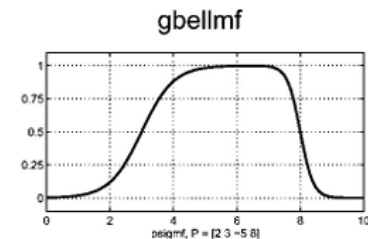
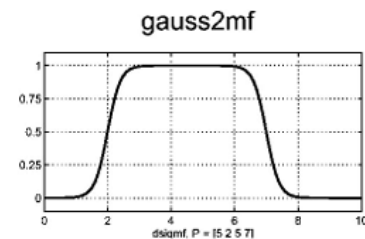
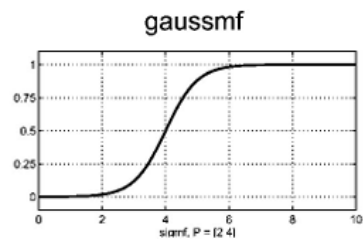
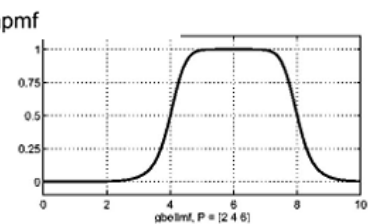
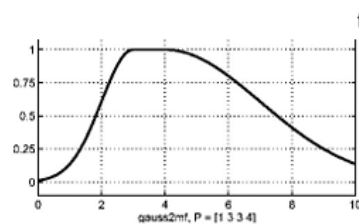
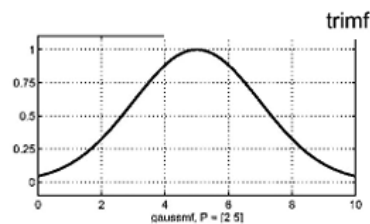
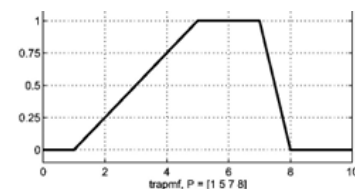
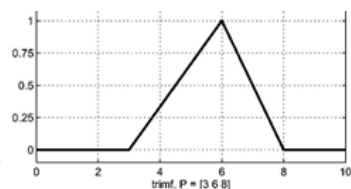
- 隶属函数 (MF) 是一条曲线, 它定义了如何将输入空间中的每个点映射到 0 和 1 之间的隶属值 (或隶属度)。



模糊逻辑工具箱中的隶属函数

- 该工具箱包含 **11 种内置的隶属函数类型**。
这 11 个函数又由几个基本函数构建：
- 分段线性函数
- 高斯分布函数
- S 型曲线
- 二次和三次多项式曲线



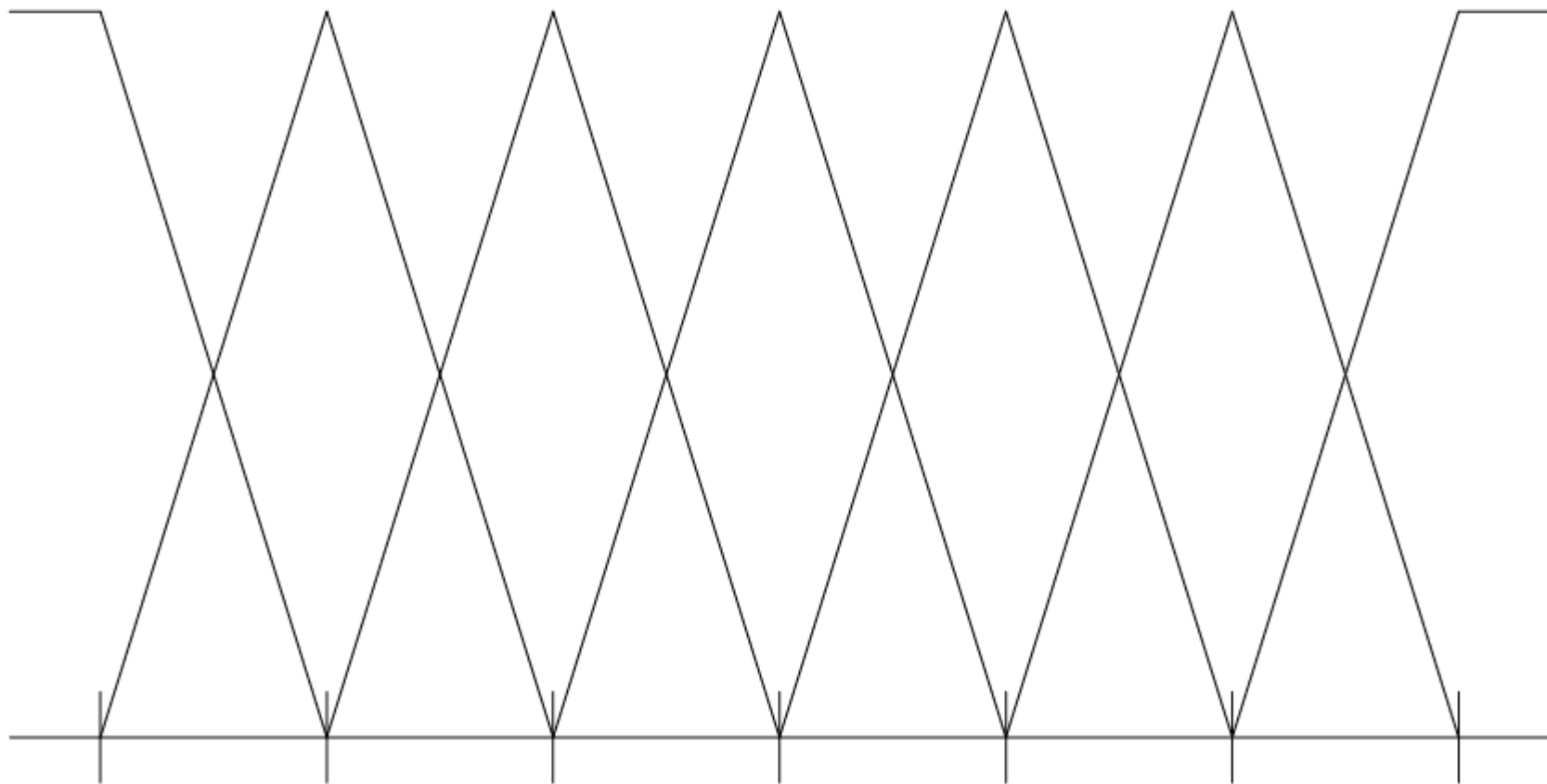


zmf

pimf

smf

七个元素的对称隶属函数



隶属函数总结

- 模糊集**描述**模糊概念（例如，跑得快的人、热天气、周末日子）。
- 模糊集**允许**部分隶属于它。（例如，星期五算是周末日子，天气相当热）。
- 对象属于模糊集的**程度**由 0 到 1 之间的隶属值表示。（例如，星期五是周末日子的程度为 0.8）。
- 与给定模糊集**相关联**的隶属函数将输入值映射到其相应的隶属值。

逻辑运算

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

AND

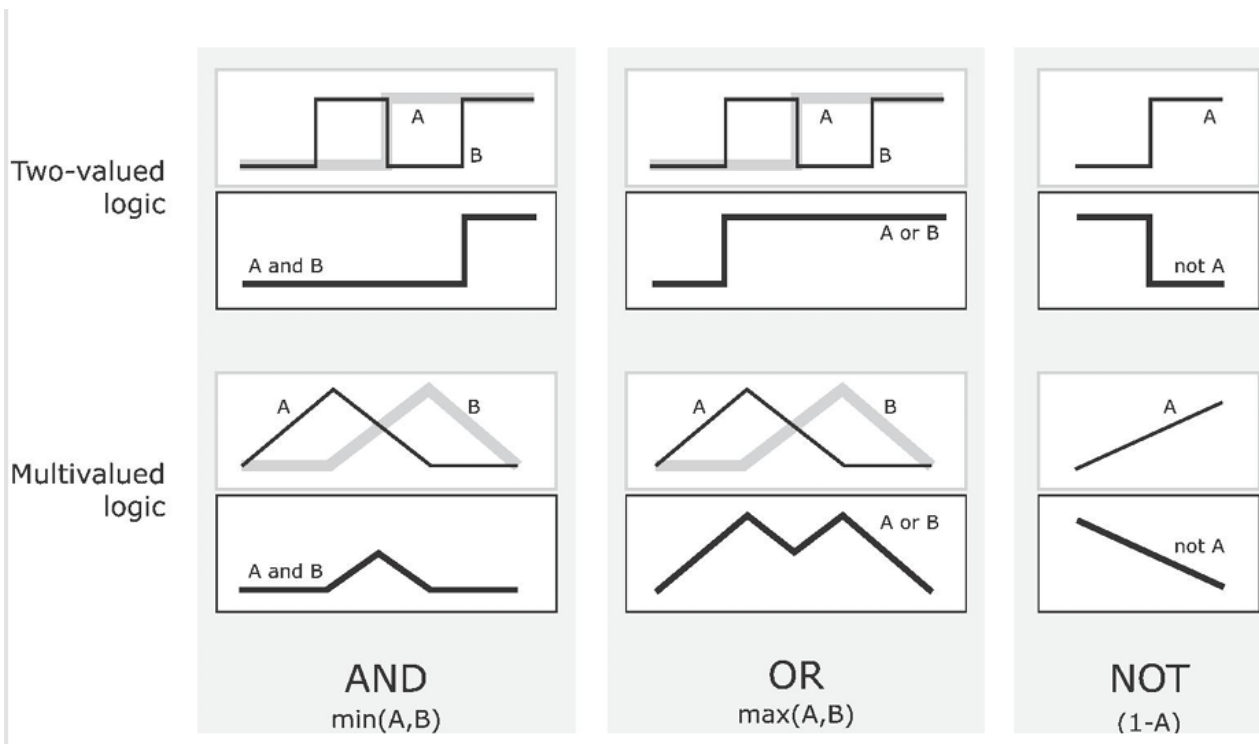
A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

OR

A	not A
0	1
1	0

NOT

逻辑运算



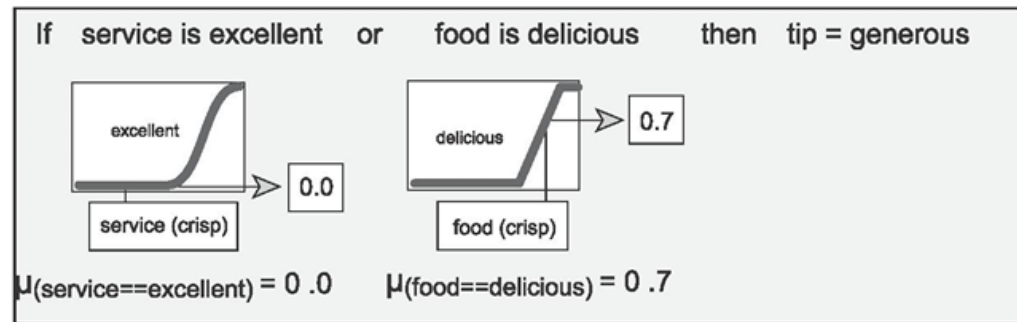
附加模糊运算符

- 模糊交或合取 (AND)
- 模糊并或析取 (OR)
- 模糊补 (NOT)
- 这些函数的经典算子是： $\text{AND} = \min$, $\text{OR} = \max$, $\text{NOT} = \text{加法补}$ 。
- T-范数（或 S-范数）

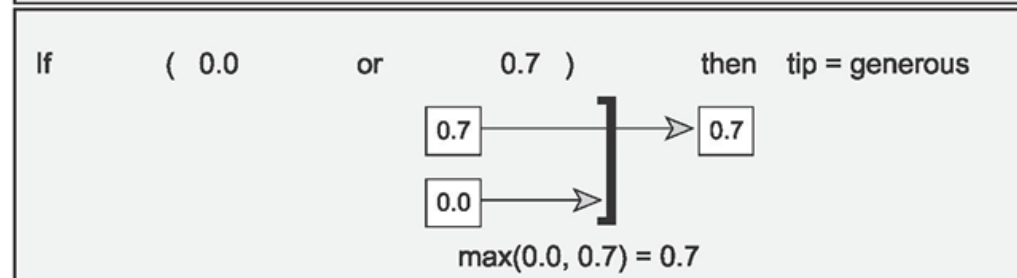
If-Then 规则: If x is A, then y is B

- **1 模糊化输入:**
 - 将前件中的所有模糊语句解析为 **0 到 1 之间的隶属度**。如果前件只有一个部分, 那么这就是该规则的支持度。
- **2 对多部分前件应用模糊算子:** 如果前件有多个部分, 应用模糊逻辑算子并将前件解析为 **0 到 1 之间的单个数字**。这就是该规则的支持度。
- **3 应用蕴涵方法:**
 - 使用整个规则的支持度来塑造输出模糊集。模糊规则的后件将整个模糊集分配给输出。这个模糊集由一个隶属函数表示, 该函数被选择用来指示后件的**品质**。如果前件只是部分为真 (即, 被赋予的值小于 1), 则根据蕴涵方法截断输出模糊集。

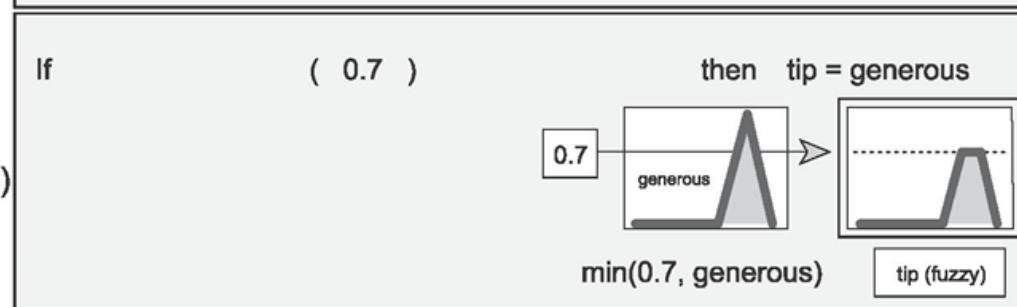
1. Fuzzify inputs



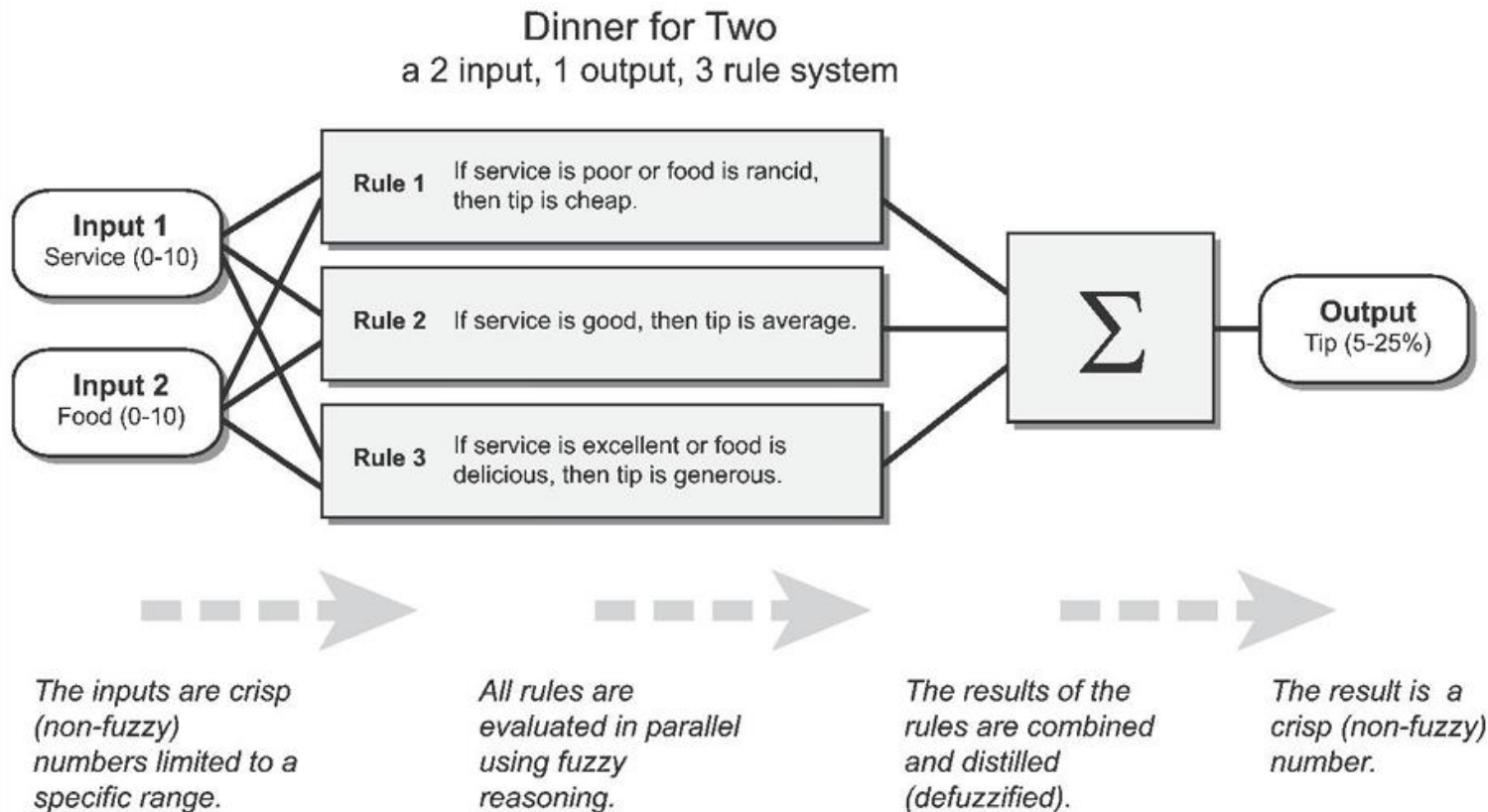
2. Apply OR operator (max)



3. Apply implication operator (min)



模糊推理过程

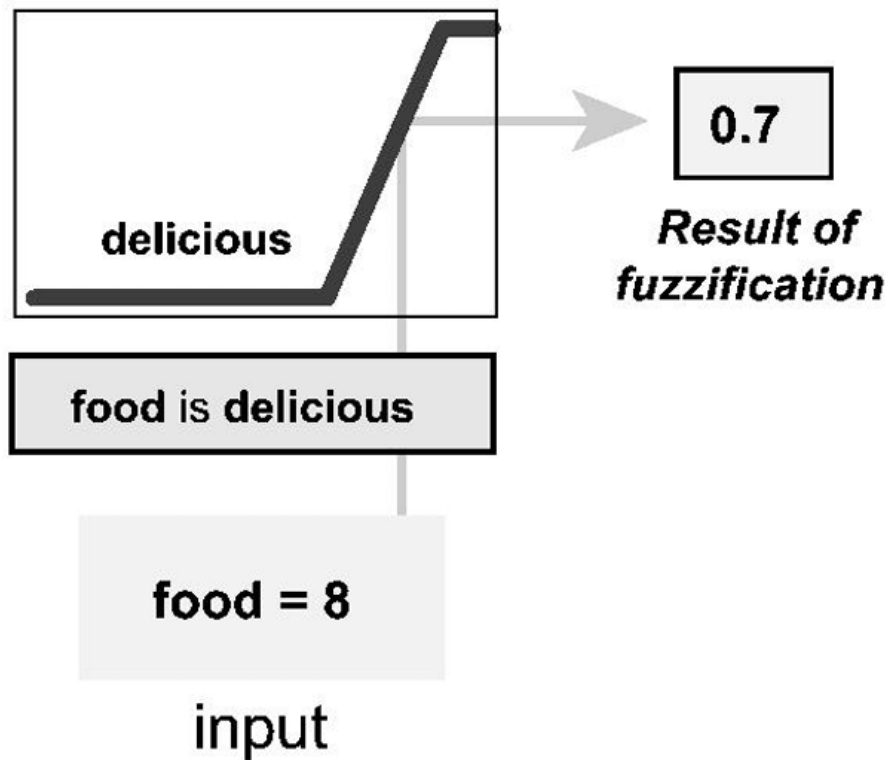


模糊推理过程包括五个部分：

- 输入变量的**模糊化**
- 在前件中应用**模糊算子** (AND 或 OR)
- 从前件到后件的**蕴涵**
- 跨规则的**后件聚合**
- **去模糊化**

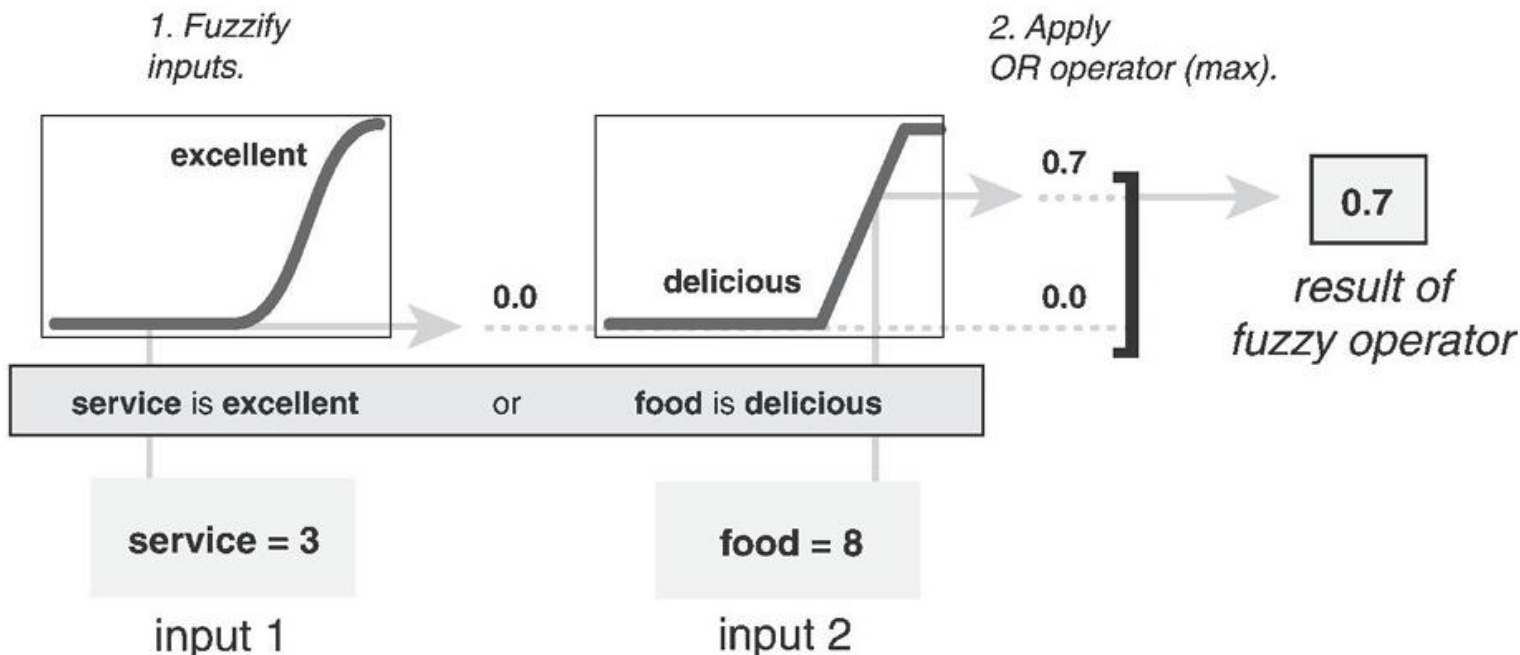
1 模糊化输入

**1. Fuzzify
inputs.**



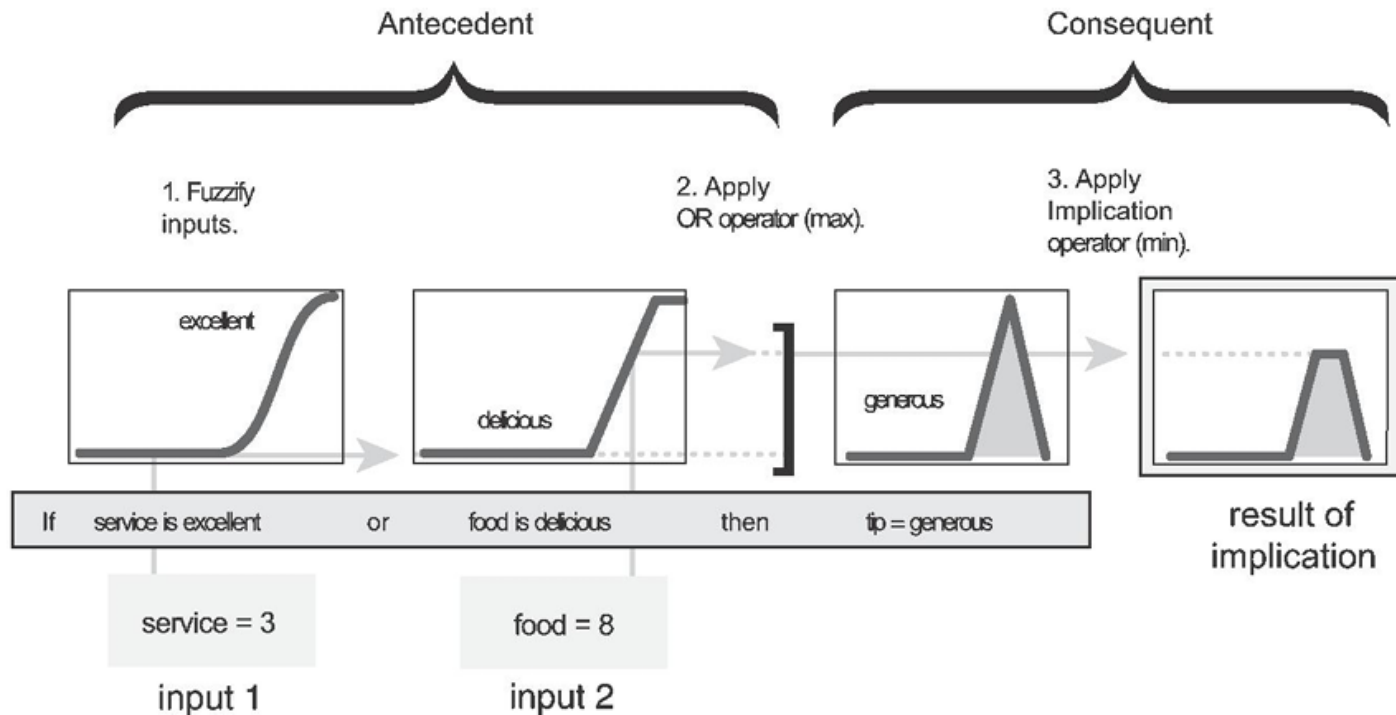
2 应用模糊算子

- **AND:** *min* (最小值) 和 *prod* (乘积)
- **OR:** *max* (最大值), 和概率 OR *probor*.



3 应用蕴涵方法

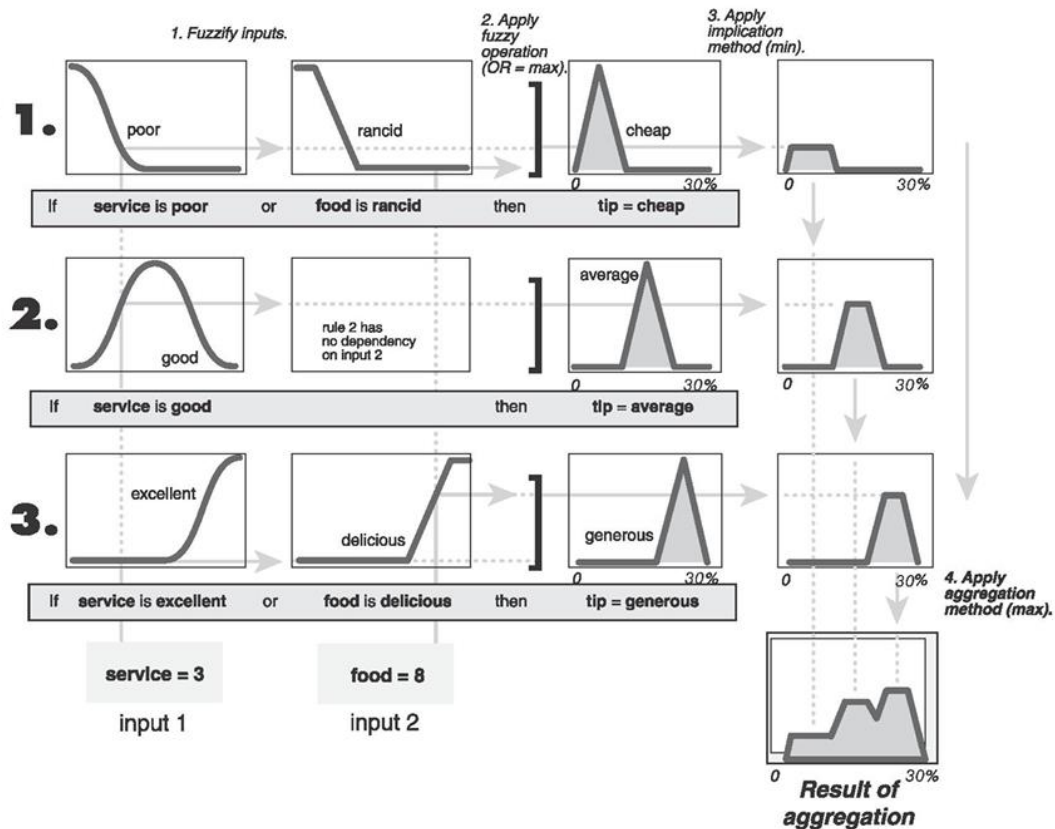
- 蕴涵过程的输入是前件给出的**单个数字**，而**输出是一个模糊集**。
对每条规则实施蕴涵。



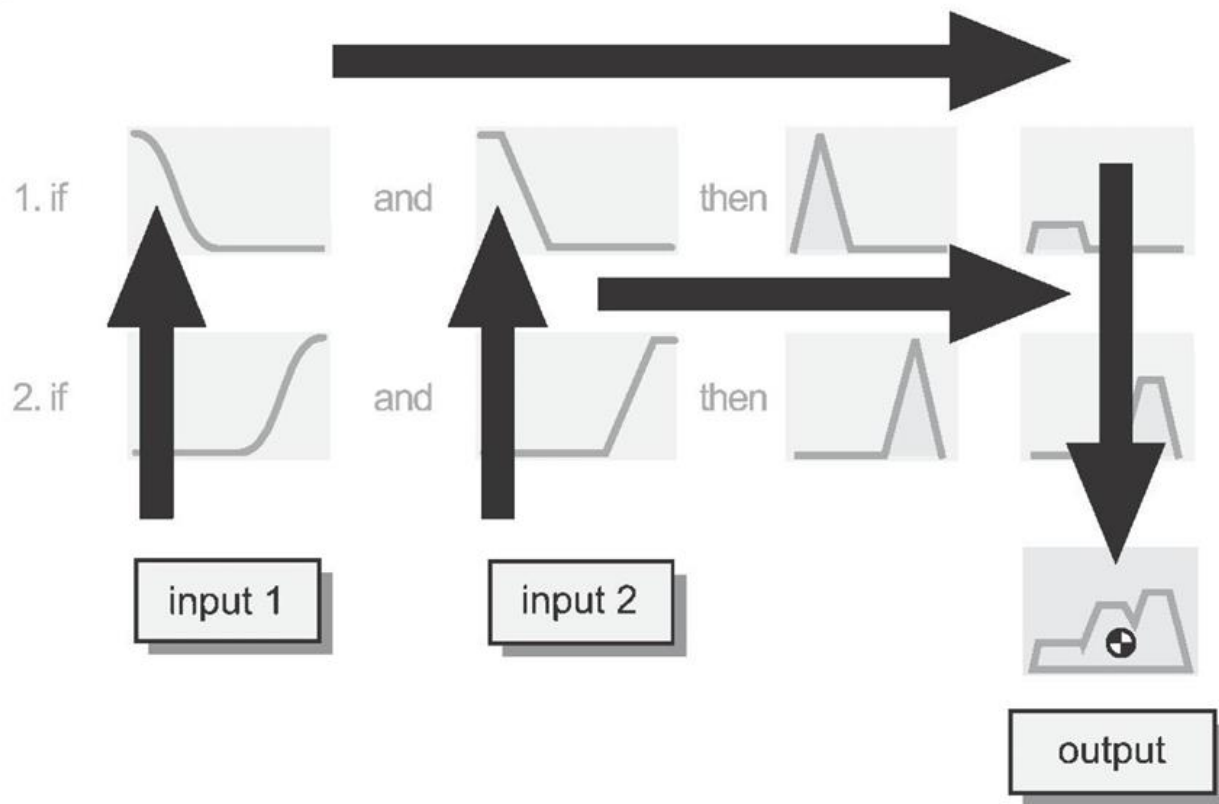
4 聚合所有输出

- 聚合是将代表每条规则输出的模糊组合并为一个模糊集的过程。
- 聚合每个输出变量只发生一次，就在最终去模糊化步骤之前。
- 聚合过程的输入是由每条规则的蕴涵过程返回的截断输出函数列表。
- 聚合过程的输出是每个输出变量一个模糊集。

模糊化输入。

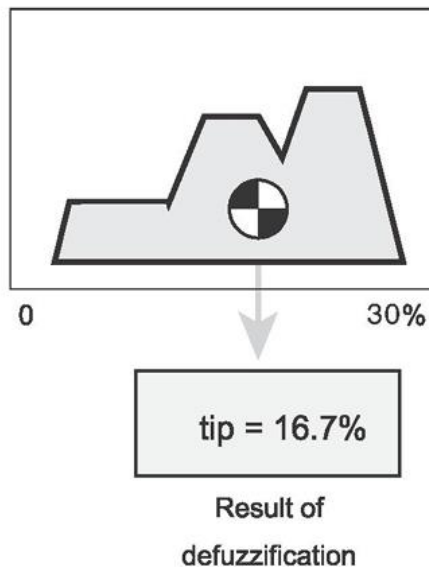


解释模糊推理图

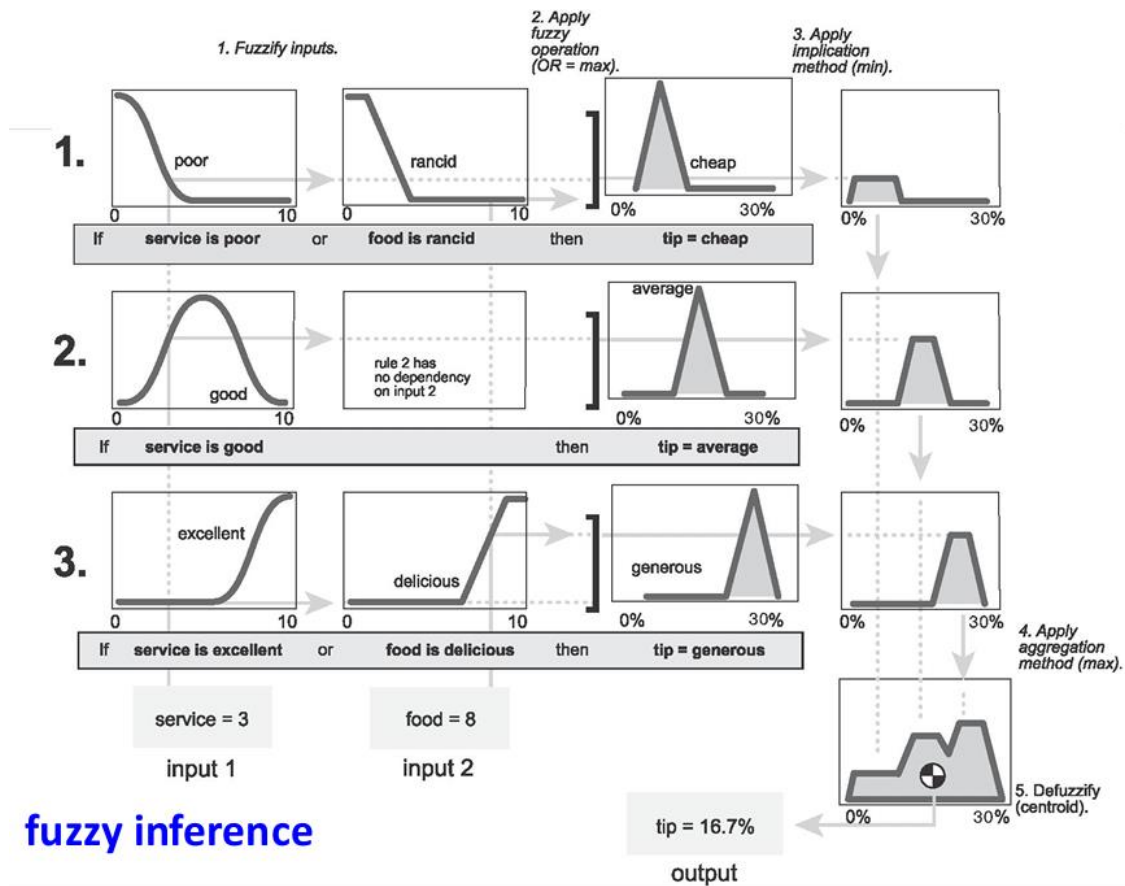


去模糊化

- **去模糊化**过程的**输入**是一个**模糊集**（聚合输出模糊集），输出是一个**单个数字**。
- 也许最流行的**去模糊化**方法是**质心计算**，它返回曲线下面积的中心。



模糊推理

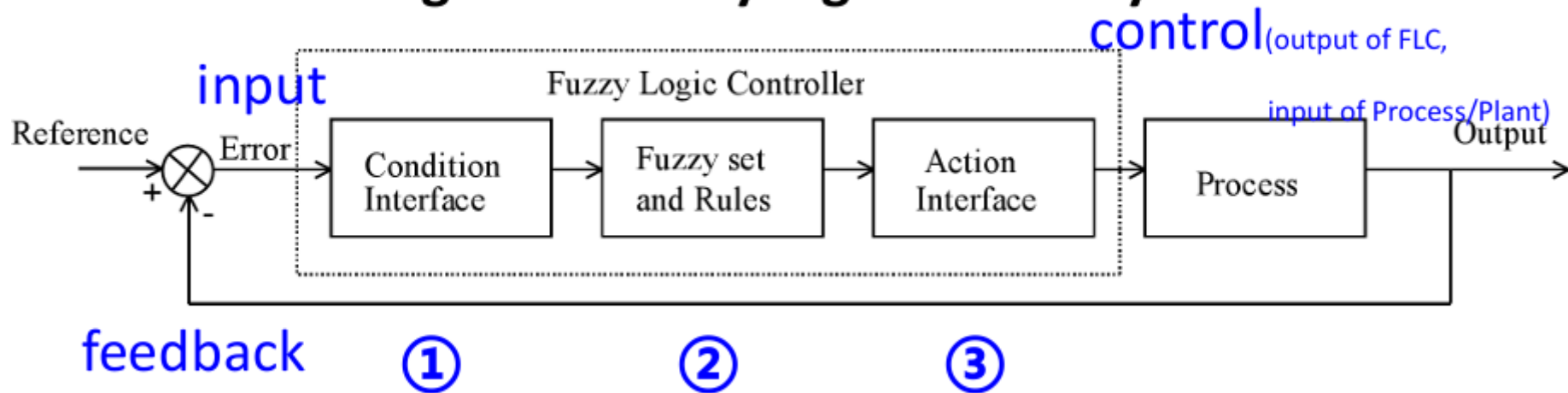


模糊逻辑控制 (FLC)

- 模糊控制器由一组控制规则组成，每条规则是关于在给定过程条件下（由以下规则结构给出）要采取的控制动作的语言陈述：
- IF <条件> THEN <控制动作>
- <条件> 称为前件，<控制动作> 是后件。

模糊逻辑控制

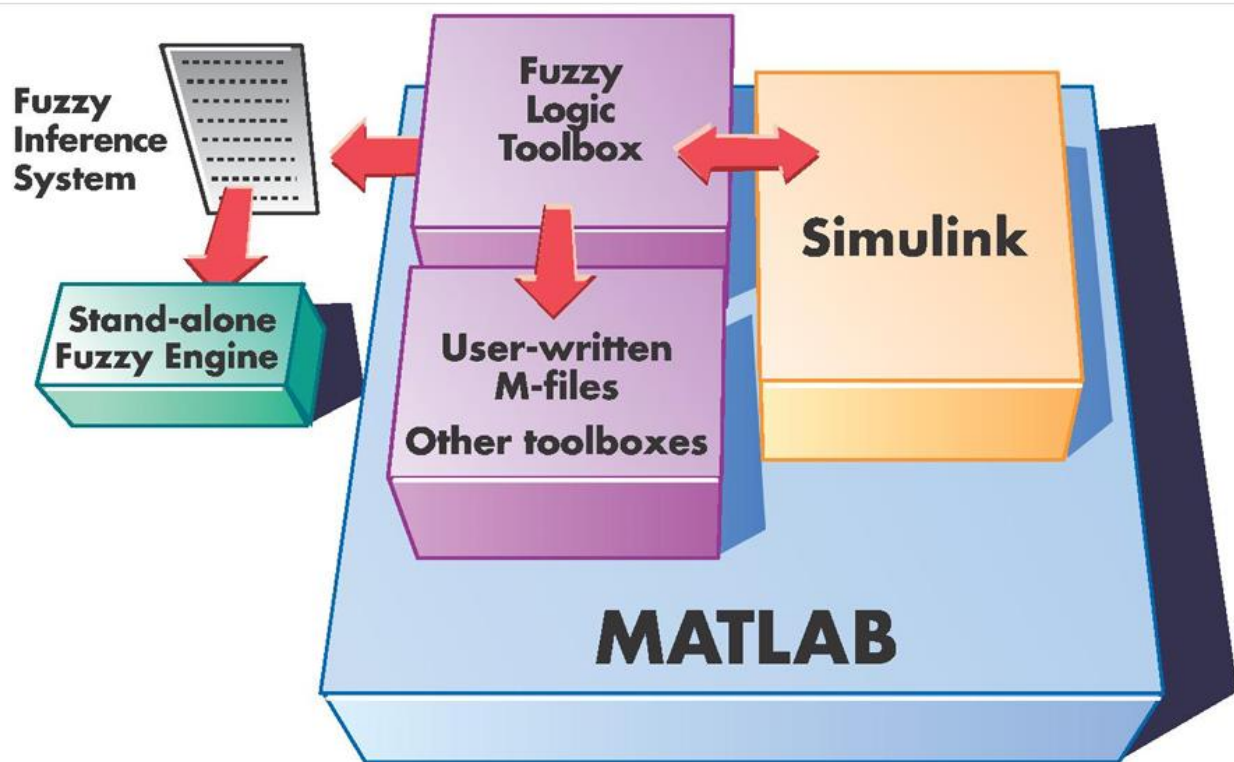
A diagram of fuzzy logic control systems



设计模糊控制器的基本步骤

- 为 FLC **定义输入和输出决策变量**；
- 指定为每个输入和输出**决策变量**定义的所有**模糊集及其隶属函数**；
- **将输入决策变量转换为模糊集（模糊化）**；
- 制定用作**推理引擎**的模糊规则库；
- **设计一种计算单一结果模糊控制动作的方法**；
- **设计一种将模糊控制动作转换为清晰值的转换方法。（去模糊化）**

模糊逻辑工具箱



教程

- 模糊逻辑工具箱入门：
<https://www.mathworks.com/help/fuzzy/index.html>
- 模糊逻辑工具箱入门（视频）：
<https://youtu.be/O348HnWPm7A>

实验

- **实验 1:** 使用 MATLAB/Simulink 进行机器人车辆模糊逻辑控制仿真
- **实验 2:** 模糊 PID 控制器[2,3]

实验 1：使用 MATLAB/Simulink 进行机器人车辆

- 模糊逻辑控制仿真
- A 部分：MATLAB/SIMULINK 中的 2 自由度机器人车辆 (20分钟)
- B 部分：S 函数中的 '2输入1输出' 模糊逻辑控制 (FLC) (20分钟)
- C 部分：使用 MATLAB/SIMULINK 对机器人车辆进行 FLC 仿真 (20分钟)
- 提示：A 部分 + B 部分 = C 部分

A 部分：MATLAB/SIMULINK 中的 2 自由度机器人

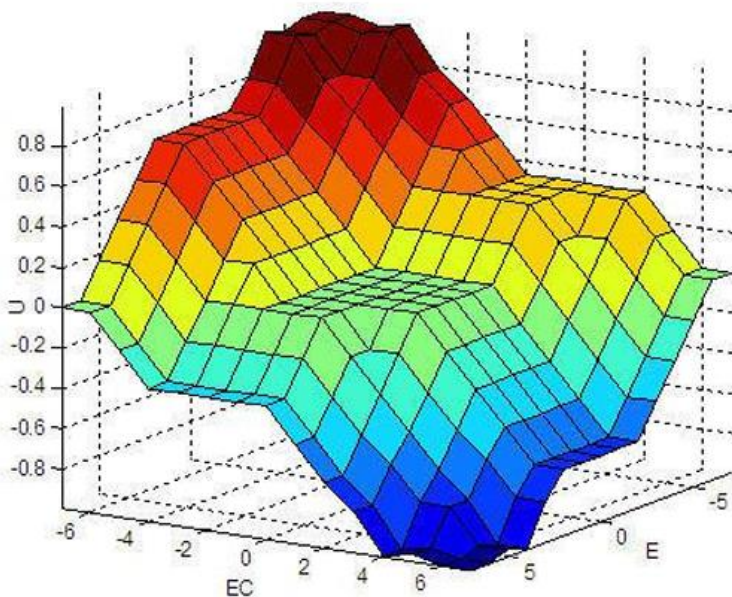
- 车辆 (20分钟)
- 1. 车辆悬架系统建模
- 车辆悬架系统建模.pdf
- 2. FLC&天钩
- FLC&天钩.pdf

B 部分：S 函数中的 '2输入1输出' 模糊逻辑控制 (FLC)

- 什么是 S 函数？
- S 函数（系统函数）为扩展 Simulink® 环境的功能提供了一个强大的机制。
- S 函数是用 MATLAB®、C、C++ 或 Fortran 编写的 Simulink 模块的计算机语言描述。C、C++ 和 Fortran S 函数使用 mex 实用程序编译为 MEX 文件。
- 与其他 MEX 文件一样，S 函数是动态链接的子程序，MATLAB 执行引擎可以自动加载和执行。

模糊规则

- SGA_suspension_flc_std_2in1out_sfunction.m



[rule_base_7rules]=...

[1 1 7 1 1

2 1 7 1 1

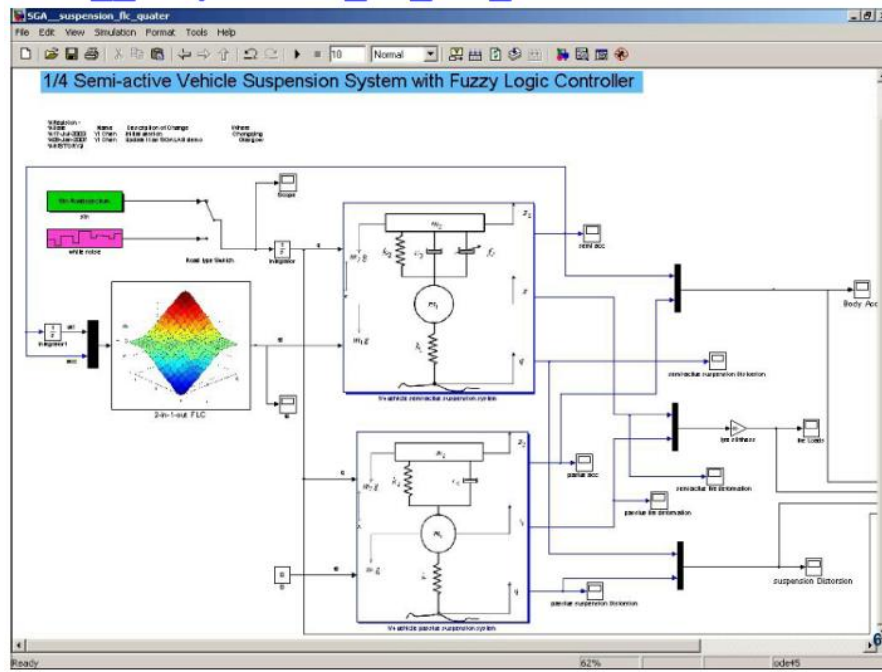
...

3 1 6 1 1

7 7 1 1 1];

C 部分：使用 MATLAB/SIMULINK 对机器人车辆进行 FLC 仿真 (20分钟)

- SGA suspension_flg_std_2in1out.mdl
- 14 带有模糊逻辑控制器的半主动车辆悬架系统



实验 2：模糊 PID 控制器[10,11]

[Products](#)[Solutions](#)[Academia](#)[Support](#)[Community](#)[Events](#)

PID Control with MATLAB and Simulink

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

参考文献

- [1] <https://github.com/LeoYiChen/Fuzzy-Logic-Control--an-introduction>