What is a fuzzy rule?

A fuzzy rule can be defined as a conditional statement in the form:

IF ***x*** **is** ***A***
THEN ***y*** **is** ***B***

where *x* and *y* are linguistic variables; and *A* and *B* are linguistic values determined by fuzzy sets on the universe of discourses *X* and *Y*, respectively.

1

Linguistic variables

- At the root of fuzzy set theory lies the idea of linguistic variables.
- **A** linguistic variable is a fuzzy variable. For example, the statement "John is tall" implies that the linguistic variable *John* takes the linguistic value *tall*.
- The range of possible values of a linguistic variable represents the universe of discourse of that variable. For example, the universe of discourse of the linguistic variable *speed* might have the range between 0 and 220 km/h and may include such fuzzy subsets as *very slow*, *slow*, *medium*, *fast*, and *very fast*.

2

What is the difference between classical and fuzzy rules?

A classical IF-THEN rule uses binary logic, for

Rule: 1	Rule: 2
IF speed is > 100	IF speed is < 40
THEN Min_stopping_Dist = 300	THEN Max_stopping_distance = 40

The variable *speed* can have any numerical value between 0 and 220 km/h, and *stopping_distance* can take either value *300* or *40*. In other words, classical rules are expressed in the black-and-white language of Boolean logic.

3

We can also represent the stopping distance rules in a fuzzy form:

Rule: 1	Rule: 2
IF speed is fast	IF speed is slow
THEN stopping_distance is long	THEN stopping_distance is short

In fuzzy rules, the linguistic variable *speed* also has the range (the universe of discourse) between 0 and 220 km/h, but this range includes fuzzy sets, such as *slow*, *medium* and *fast*. The universe of discourse of the linguistic variable *stopping_distance* can be between 0 and 300 m and may include such fuzzy sets as *short*, *medium* and *long*.

4

- **Hedges are terms that modify the shape of fuzzy sets. They include adverbs such as *very*, *somewhat*, *quite*, *more or less* and *slightly*.**

IF height is *very tall*
THEN weight is *very heavy*

IF speed is very slow
THEN stopping_distance is very short

5

Fuzzy Rules fire *partially* And relate fuzzy sets

- Fuzzy rules relate fuzzy sets.
- In a fuzzy system, all rules can fire to some extent, or in other words fire *partially*. If the antecedent is true to some degree of membership, then the consequent is also true to some degree.

6

A fuzzy rule can have multiple antecedents, for example:

IF project_duration is long
AND project_staffing is large
AND project_funding is inadequate
THEN risk is high

IF service is excellent
OR food is delicious
THEN tip is generous

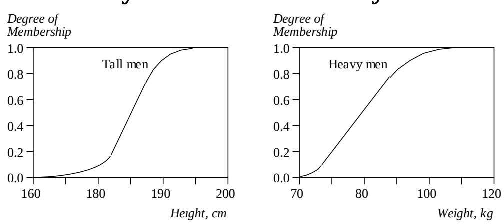
7

OR Multiple Consequents

IF temperature is hot
THEN hot_water is reduced;
cold_water is increased

8

Fuzzy sets of *tall* and *heavy* men



These fuzzy sets provide the basis for a weight estimation model. The model is based on a relationship between a man's height and his weight:

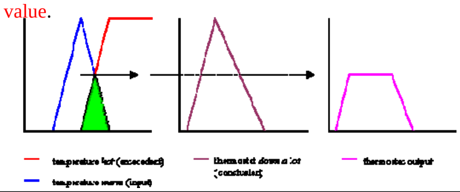
IF height is *tall*
THEN weight is *heavy*

9

The diagrams depict the rule:

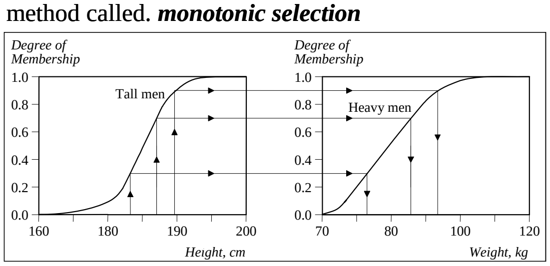
If *temperature is hot* **then** *turn thermostat down a lot*,

with the input *temperature is warm*.
Fist we'll look at the rule executor. The effect of this type of inference is to generate an output fuzzy value that is the conclusion fuzzy value *clipped* at the maximum value of the intersection of the antecedent fuzzy value and the input fuzzy value.



10

The value of the output or a truth membership grade of the rule consequent can be estimated directly from a corresponding truth membership grade in the antecedent. This form of fuzzy inference uses a method called. ***monotonic selection***



11

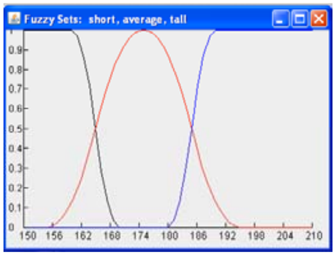
Fuzzy Rules in Jess

If you are tall you are heavy

```
FuzzyValue condition = new FuzzyValue(height,"tall");
FuzzyValue conclusion = new FuzzyValue(weight, "heavy");
FuzzyRule rule1 = new FuzzyRule();
FuzzyValue input = new FuzzyValue(height,“tall");
 rule1.addAntecedent(condition);
 rule1.addConclusion(conclusion);
 rule1.addInput(input);
 FuzzyValueVector fvv = rule1.execute();
```

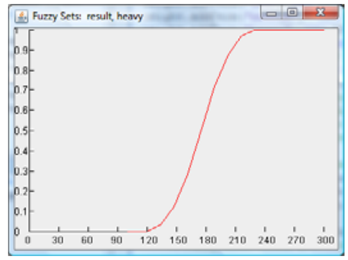
FuzzyExample4.java

12



13

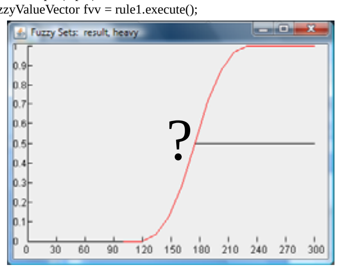
If you are tall you are heavy



14

If you are average what are you?

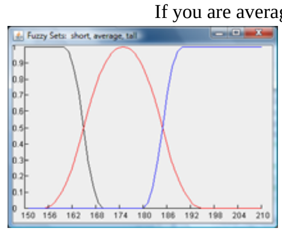
```
FuzzyValue condition = new FuzzyValue(height,"tall");
FuzzyValue conclusion = new FuzzyValue(weight, "heavy");
FuzzyValue input = new FuzzyValue(height,"average");
FuzzyRule rule1 = new FuzzyRule();
rule1.addAntecedent(condition);
rule1.addConclusion(conclusion);
rule1.addInput(input);
FuzzyValueVector fvv = rule1.execute();
```



15

FuzzyExample4.java

If you are tall then you are heavy
If you are average then you are ??



The output fuzzy value that is the conclusion fuzzy value *clipped* at the maximum value of the intersection of the antecedent fuzzy value and the input fuzzy value

16

More on Fuzzy inference

The most commonly used fuzzy inference technique is the so-called Mamdani method. In 1975, Professor **Ebrahim Mamdani** of London University built one of the first fuzzy systems to control a steam engine and boiler combination. He applied a set of fuzzy rules supplied by experienced human operators.

17

Mamdani fuzzy inference Algorithm

- The Mamdani-style fuzzy inference process is performed in four steps:
  - fuzzification of the input variables,
  - rule evaluation;
  - aggregation of the rule outputs, and finally
  - defuzzification.

18

We examine a simple two-input one-output problem that includes three rules:

Rule: 1
IF *x* **is** *A*3
OR *y* **is** *B*1
THEN *z* **is** *C*1

Rule: 1
IF ***project_funding*** **is** ***adequate***
OR ***project_staffing*** **is** ***small***
THEN ***risk*** **is** ***low***

Rule: 2
IF *x* **is** *A*2
AND *y* **is** *B*2
THEN *z* **is** *C*2

Rule: 2
IF ***project_funding*** **is** ***marginal***
AND ***project_staffing*** **is** ***large***
THEN ***risk*** **is** ***normal***

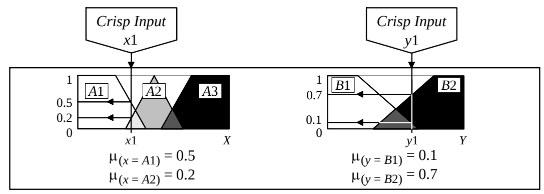
Rule: 3
IF *x* **is** *A*1
THEN *z* **is** *C*3

IF ***project_funding*** **is** ***inadequate***
THEN ***risk*** **is** ***high***

19

Step 1: Fuzzification

The first step is to take the crisp inputs, *x*1 and *y*1 (*project funding* and *project staffing*), and determine the degree to which these inputs belong to each of the appropriate fuzzy sets.



20

Step 2: Rule Evaluation

The second step is to take the fuzzified inputs, $\mu_{(x=A1)} = 0.5$, $\mu_{(x=A2)} = 0.2$, $\mu_{(y=B1)} = 0.1$ and $\mu_{(y=B2)} = 0.7$, and apply them to the antecedents of the fuzzy rules. If a given fuzzy rule has multiple antecedents, the fuzzy operator (AND or OR) is used to obtain a single number that represents the result of the antecedent evaluation. This number (the truth value) is then applied to the consequent membership function.

21

To evaluate the disjunction of the rule antecedents, we use the **OR fuzzy operation**. Typically, fuzzy expert systems make use of the classical fuzzy operation **union**:

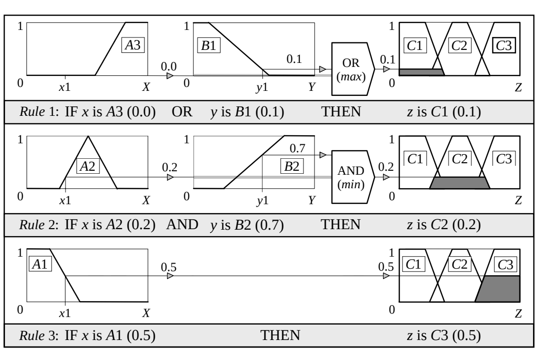
$$\mu_{A \cup B}(x) = max\,[\mu_A(x),\, \mu_B(x)]$$

Similarly, in order to evaluate the conjunction of the rule antecedents, we apply the **AND fuzzy operation intersection**:

$$\mu_{A \cap B}(x) = min\,[\mu_A(x),\, \mu_B(x)]$$

22

Mamdani-style rule evaluation



23

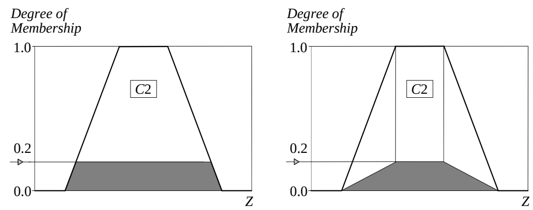
Now the result of the antecedent evaluation can be applied to the membership function of the consequent.

- The most common method of correlating the rule consequent with the truth value of the rule antecedent is to cut the consequent membership function at the level of the antecedent truth. This method is called **clipping**. Since the top of the membership function is sliced, the clipped fuzzy set loses some information. However, clipping is still often preferred because it involves less complex and faster mathematics, and generates an aggregated output surface that is easier to defuzzify.

24

- While clipping is a frequently used method, **scaling** offers a better approach for preserving the original shape of the fuzzy set. The original membership function of the rule consequent is adjusted by multiplying all its membership degrees by the truth value of the rule antecedent. This method, which generally loses less information, can be very useful in fuzzy expert systems.

Clipped and scaled membership functions

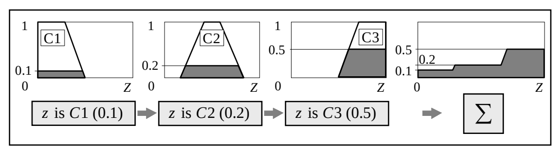


Step 3: Aggregation of the rule outputs

Aggregation is the process of unification of the outputs of all rules. We take the membership functions of all rule consequents previously clipped or scaled and combine them into a single fuzzy set.

The input of the aggregation process is the list of clipped or scaled consequent membership functions, and the output is one fuzzy set for each output variable.

Aggregation of the rule outputs



Execute a Fuzzy Rule in Jess

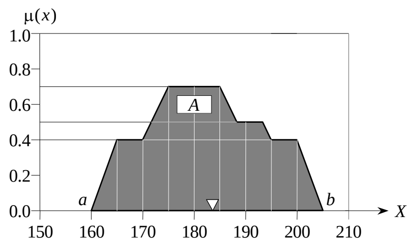
```
rule1.addInput(input);
FuzzyValueVector fvv = rule1.execute();
```

```
FuzzyVariable rhs = new FuzzyVariable("weight",0,300,"pounds");
rhs.addTerm("result",fvv.fuzzyValueAt(0).getFuzzySet());
```

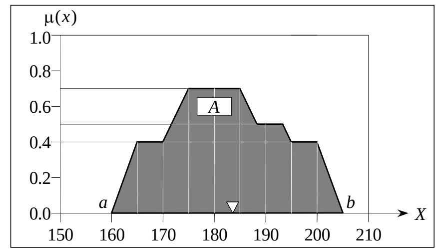

Step 4: Defuzzification

The last step in the fuzzy inference process is defuzzification. Fuzziness helps us to evaluate the rules, but the final output of a fuzzy system has to be a crisp number. The input for the defuzzification process is the aggregate output fuzzy set and the output is a single number.

- There are several defuzzification methods, but probably the most popular one is the **centroid technique**. It finds the point where a vertical line would slice the aggregate set into two equal masses.

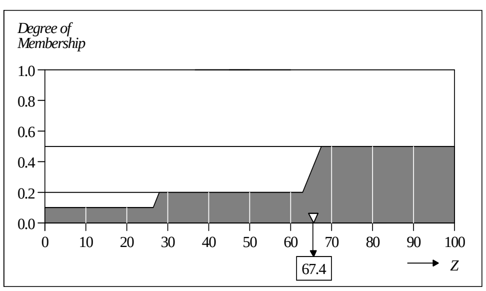


- Centroid defuzzification method finds a point representing the centre of gravity of the fuzzy set, *A*, on the interval, *ab*.
- A reasonable estimate can be obtained by calculating it over a sample of points.



Centre of gravity (COG):

$$COG = \frac{(0+10+20)\times 0.1 + (30+40+50+60)\times 0.2 + (70+80+90+100)\times 0.5}{0.1+0.1+0.1+0.2+0.2+0.2+0.2+0.5+0.5+0.5+0.5} = 67.4$$



Defuzzification in Jess

```
System.out.println("Moment Defuz: " +
   fvv.fuzzyValueAt(0).momentDefuzzify());
```

FuzzyExample4.java

Process of developing a fuzzy expert system

1. Specify the problem and define linguistic variables.
2. Determine fuzzy sets.
3. Elicit and construct fuzzy rules.
4. Encode the fuzzy sets, fuzzy rules and procedures to perform fuzzy inference into the expert system.
5. Evaluate and tune the system.

Sugeno fuzzy inference

- Mamdani-style inference, as we have just seen, requires us to find the centroid of a two-dimensional shape by integrating across a continuously varying function. In general, this process is not computationally efficient.
- **Michio Sugeno** suggested to use a single spike, a *singleton*, as the membership function of the rule consequent. A **singleton**, or more precisely a **fuzzy singleton**, is a fuzzy set with a membership function that is unity at a single particular point on the universe of discourse and zero everywhere else.

Sugeno-style fuzzy inference is very similar to the Mamdani method. Sugeno changed only a rule consequent. Instead of a fuzzy set, he used a mathematical function of the input variable. The format of the **Sugeno-style fuzzy rule** is

IF *x* is *A*
AND *y* is *B*
THEN *z* is *f* (*x*, *y*)

where *x*, *y* and *z* are linguistic variables; *A* and *B* are fuzzy sets on universe of discourses *X* and *Y*, respectively; and *f* (*x*, *y*) is a mathematical function.

The most commonly used **zero-order Sugeno fuzzy model** applies fuzzy rules in the following form:

IF *x* is *A*
AND *y* is *B*
THEN *z* is *k*

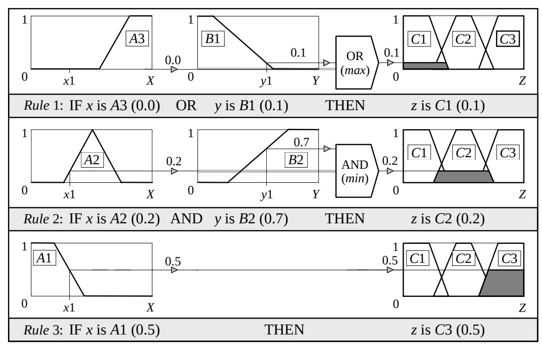
where *k* is a constant.

In this case, the output of each fuzzy rule is constant. ***All consequent membership functions are represented by singleton spikes.***

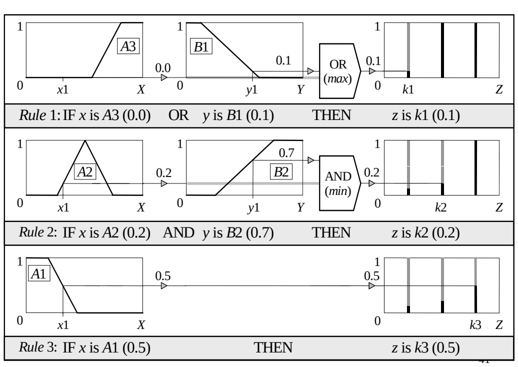
We examine a simple two-input one-output problem that includes three rules:

Rule: 1	**Rule: 1**
IF *x* is *A*3	**IF *project_funding* is *adequate***
OR *y* is *B*1	**OR *project_staffing* is *small***
THEN *z* is *C*1	**THEN *risk* is *low***
Rule: 2	**Rule: 2**
IF *x* is *A*2	**IF *project_funding* is *marginal***
AND *y* is *B*2	**AND *project_staffing* is *large***
THEN *z* is *C*2	**THEN *risk* is *normal***
Rule: 3	**Rule: 3**
IF *x* is *A*1	**IF *project_funding* is *inadequate***
THEN *z* is *C*3	**THEN *risk* is *high***

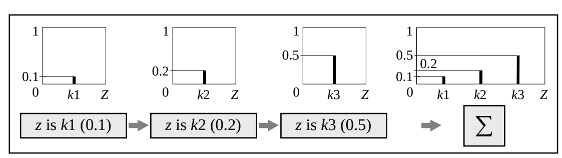
Mamdani-style rule evaluation



Sugeno-style rule evaluation



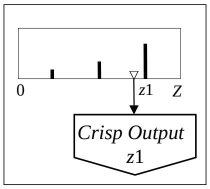
Sugeno-style aggregation of the rule outputs



Weighted average (WA):

$$WA = \frac{\mu(k1)\times k1 + \mu(k2)\times k2 + \mu(k3)\times k3}{\mu(k1)+\mu(k2)+\mu(k3)} = \frac{0.1\times 20 + 0.2\times 50 + 0.5\times 80}{0.1+0.2+0.5} = 65$$

Sugeno-style defuzzification



43

How to make a decision on which method to apply – Mamdani or Sugeno?

- Mamdani method is widely accepted for capturing expert knowledge. It allows us to describe the expertise in more intuitive, more human-like manner. However, Mamdani-type fuzzy inference entails a substantial computational burden.
- On the other hand, Sugeno method is computationally effective and works well with optimisation and adaptive techniques, which makes it very attractive in control problems, particularly for dynamic nonlinear systems.

44

Building a fuzzy expert system: case study

- A service centre keeps spare parts and repairs parts.
- A customer brings a failed item and receives a spare of the same type.
- Failed parts are repaired by **servers**, placed on the shelf, and thus become spares.
- The objective here is to advise a manager of the service centre on certain decision policies to keep the customers satisfied.
- Advise on the initial number of spares to keep **delay** reasonable

45

Step 1: Specify the problem and define linguistic variables

There are four main linguistic variables: average waiting time (mean delay) *m*, repair utilisation factor of the service centre ρ, number of servers *s*, and initial number of spare parts *n*.

$$\rho = \frac{CustomerArrivalRate}{CustemerDepartureRate}$$

The system must advise management on the number of spares to keep as well as the number of servers. Increasing either will increase cost and decrease waiting time in some proportion.

46

Linguistic variables and their ranges

Linguistic Variable: *Mean Delay, m*		
Linguistic Value	Notation	Numerical Range (normalised)
Very Short	VS	[0, 0.3]
Short	S	[0.1, 0.5]
Medium	M	[0.4, 0.7]
Linguistic Variable: *Number of Servers, s*		
Linguistic Value	Notation	Numerical Range (normalised)
Small	S	[0, 0.35]
Medium	M	[0.30, 0.70]
Large	L	[0.60, 1]
Linguistic Variable: *Repair Utilisation Factor*, ρ		
Linguistic Value	Notation	Numerical Range
Low	L	[0, 0.6]
Medium	M	[0.4, 0.8]
High	H	[0.6, 1]
Linguistic Variable: *Number of Spares, n*		
Linguistic Value	Notation	Numerical Range (normalised)
Very Small	VS	[0, 0.30]
Small	S	[0, 0.40]
Rather Small	RS	[0.25, 0.45]
Medium	M	[0.30, 0.70]
Rather Large	RL	[0.55, 0.75]
Large	L	[0.60, 1]
Very Large	VL	[0.70, 1]

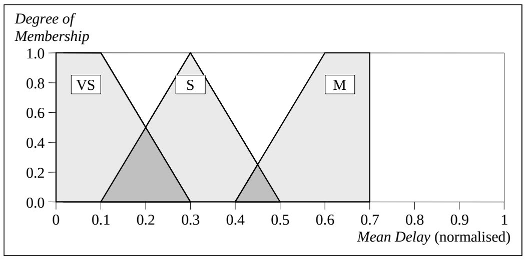
47

Step 2: Determine fuzzy sets

Fuzzy sets can have a variety of shapes. However, a triangle or a trapezoid can often provide an adequate representation of the expert knowledge, and at the same time, significantly simplifies the process of computation.

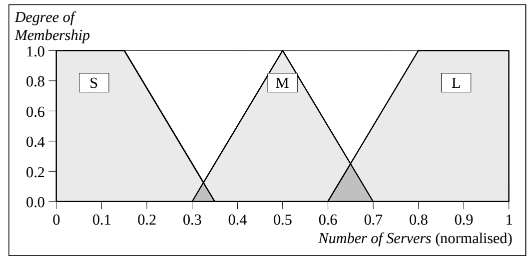
48

Fuzzy sets of *Mean Delay m*



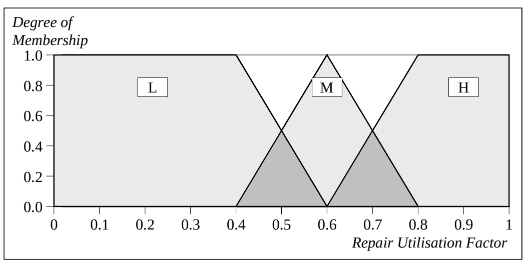
49

Fuzzy sets of *Number of Servers s*



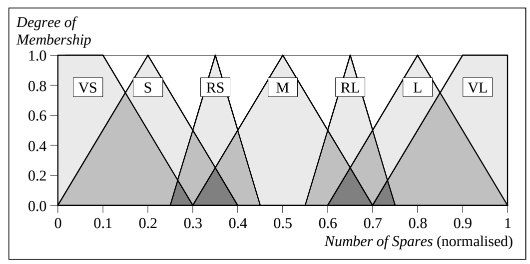
50

Fuzzy sets of *Repair Utilisation Factor ρ*



51

Fuzzy sets of *Number of Spares n*



52

Step **3: Elicit and construct fuzzy rules**

To accomplish this task, we might ask the expert to describe how the problem can be solved using the fuzzy linguistic variables defined previously.

Required knowledge also can be collected from other sources such as books, computer databases, flow diagrams and observed human behaviour.

53

Rules about utilization and spares

1. If (utilisation_factor is L) then (number_of_spares is S)
2. If (utilisation_factor is M) then (number_of_spares is M)
3. If (utilisation_factor is H) then (number_of_spares is L)

54

Rules about delay, servers and spares

4. If (mean_delay is VS) and (number_of_servers is S) then (number_of_spares is VL)
5. If (mean_delay is S) and (number_of_servers is S) then (number_of_spares is L)
6. If (mean_delay is M) and (number_of_servers is S) then (number_of_spares is M)
7. If (mean_delay is VS) and (number_of_servers is M) then (number_of_spares is RL)
8. If (mean_delay is S) and (number_of_servers is M) then (number_of_spares is RS)
9. If (mean_delay is M) and (number_of_servers is M) then (number_of_spares is S)
10. If (mean_delay is VS) and (number_of_servers is L) then (number_of_spares is M)
11. If (mean_delay is S) and (number_of_servers is L) then (number_of_spares is S)
12. If (mean_delay is M) and (number_of_servers is L) then (number_of_spares is VS)

55

The larger rule base for three combinations

Rule	m	s	ρ	n	Rule	m	s	ρ	n	Rule	m	s	ρ	n
1	VS	S	L	VS	10	VS	S	M	S	19	VS	S	H	VL
2	S	S	L	VS	11	S	S	M	VS	20	S	S	H	L
3	M	S	L	VS	12	M	S	M	VS	21	M	S	H	M
4	VS	M	L	VS	13	VS	M	M	RS	22	VS	M	H	M
5	S	M	L	VS	14	S	M	M	S	23	S	M	H	M
6	M	M	L	VS	15	M	M	M	VS	24	M	M	H	S
7	VS	L	L	S	16	VS	L	M	M	25	VS	L	H	RL
8	S	L	L	S	17	S	L	M	RS	26	S	L	H	M
9	M	L	L	VS	18	M	L	M	S	27	M	L	H	RS

if mean_delay is VS
and number_servers is S
and utilization is Low
then spares is VS

56

***Step* 4**: Encode the fuzzy sets, fuzzy rules and procedures to perform fuzzy inference into the expert system

57

***Step* 5**: Evaluate and tune the system

The last, and the most laborious, task is to evaluate and tune the system. We want to see whether our fuzzy system meets the requirements specified at the beginning.

Several test situations depend on the mean delay, number of servers and repair utilisation factor.

The Fuzzy Logic Toolbox can generate surface to help us analyse the system's performance.

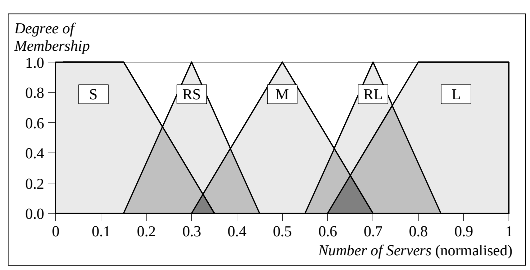
58

However, the expert might not be satisfied with the system performance.

To improve the system performance, we may use additional sets – *Rather Small* and *Rather Large* – on the universe of discourse *Number of Servers,* and then extend the rule base.

59

Modified fuzzy sets of *Number of Servers s*



60

Tuning fuzzy systems

1. Review model input and output variables, and if required redefine their ranges.
2. Review the fuzzy sets, and if required define additional sets on the universe of discourse. The use of wide fuzzy sets may cause the fuzzy system to perform roughly.
3. Provide sufficient overlap between neighbouring sets. It is suggested that triangle-to-triangle and trapezoid-to-triangle fuzzy sets should overlap between 25% to 50% of their bases.

61

4. Review the existing rules, and if required add new rules to the rule base.
5. Adjust the rule execution weights. Most fuzzy logic tools allow control of the importance of rules by changing a weight multiplier.
6. Revise shapes of the fuzzy sets. In most cases, fuzzy systems are highly tolerant of a shape approximation.

62