

# Funky Systems and Neural Networks

Jéssica Consciência e Tiago Leite

October 5, 2024

Part I

**Fuzzy System**

Firstly we started by deciding between which type of fuzzy system we should implement: Mamdani, Takagi-Sugeno or Tsukamoto. From the project statement we observe that the output *CLP-Variation* is not any clear function of the input, rulling out Takagi-Sugeno, also meaning that our output is a Fuzzy Set. If we wish for our output to be monotonic then the choice would be Tsukamoto, since we did not want this restriction and decided for starting with a simple approach then later on adding difficulty when needed. (Early on we decided to try to make data-driven decisions with an iterative improving process)

## 0.1 First Iterations

In the initial iteration, we selected the variables *ProcessorLoad*, *MemoryUsage*, and *Latency* based on common sense. These variables were chosen as inputs, while *CLP* was designated as the output. We opted for triangular membership functions, defining four levels for each input variable: (low, medium, high, critical) for *ProcessorLoad* and *MemoryUsage*, and (poor, fair, good, great) for *Latency*.

To start building the system, we decided to focus on just two variables: *MemoryUsage* and *ProcessorLoad*. We then defined the range of the membership functions associated with each term of the two linguistic variables. Considering that a device with more than 85% processor load or memory usage is typically unable to perform its basic tasks, it became clear that this threshold would correspond to a specific term, which we labeled as "critical". The ranges for the other membership function terms were distributed between 0 and 1 based on what we deemed appropriate. We also decided to keep the terms associated with *CLP* straightforward, using only three terms: "decrease," "increase," and "maintain." The values for the membership functions of these terms were distributed between -1 and 1.

The figures below illustrate the membership function graphs for these variables.

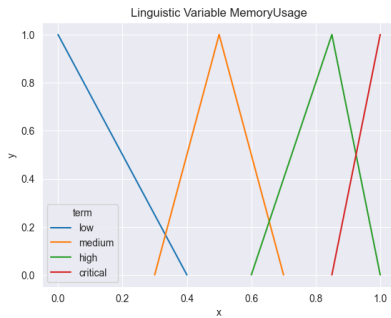


Figure 1: Memory Usage

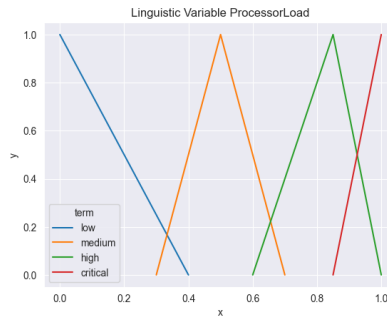


Figure 2: Processor Load

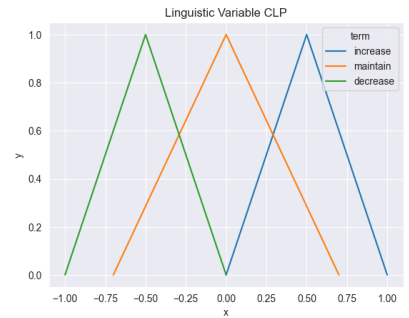


Figure 3: CLP Variation

To design the system's rules, we created a truth table, which can be found below in Table ???. The logic behind the table was as follows: when both *MemoryUsage* and *ProcessorLoad* were either "low" or "medium," the *CLP* would increase. When one of them reached "high," the *CLP* remained unchanged (this decision was made to ensure that the node's processing capacity stayed above average). Finally, if any of these variables entered a "critical" state, the *CLP* had to decrease.

To visualize the system's output, we generated 50 data points for *MemoryUsage* and *ProcessorLoad* ranging between 0 and 1. We then created an interactive 3D graph that showed the evolution of *CLP* based on these two values.

Subsequently, we explored the effect of switching the membership functions to a Gaussian distribution.

<i>CLP</i>		<i>ProcessorLoad</i>			
		low	medium	high	critical
<i>MemoryUsage</i>	low	increase	increase	maintain	decrease
	medium	increase	increase	maintain	decrease
	high	maintain	maintain	maintain	decrease
	critical	decrease	decrease	decrease	decrease

Table 1: Truth table

During the experimentation phase with different membership functions, the need for visualization became apparent. To facilitate this, we developed a helper script [fuzzy/visualization/fuzzy\_system\_to\_dataframe] that converts the FuzzySystem Python object into a dynamic dataframe, enabling easy plotting and analysis of the membership functions.

## 0.2 Generalized Bell

We decided to experiment with a more generic Membership function, so we extended simpful's Base Membership Function class and created Bell\_MF [in fuzzy/models/bell\_mf.py]. The first results are shown in the figure bellow.

## 0.3 Architecture

This should contain choice of architecture and why.

## 0.4 Membership Functions

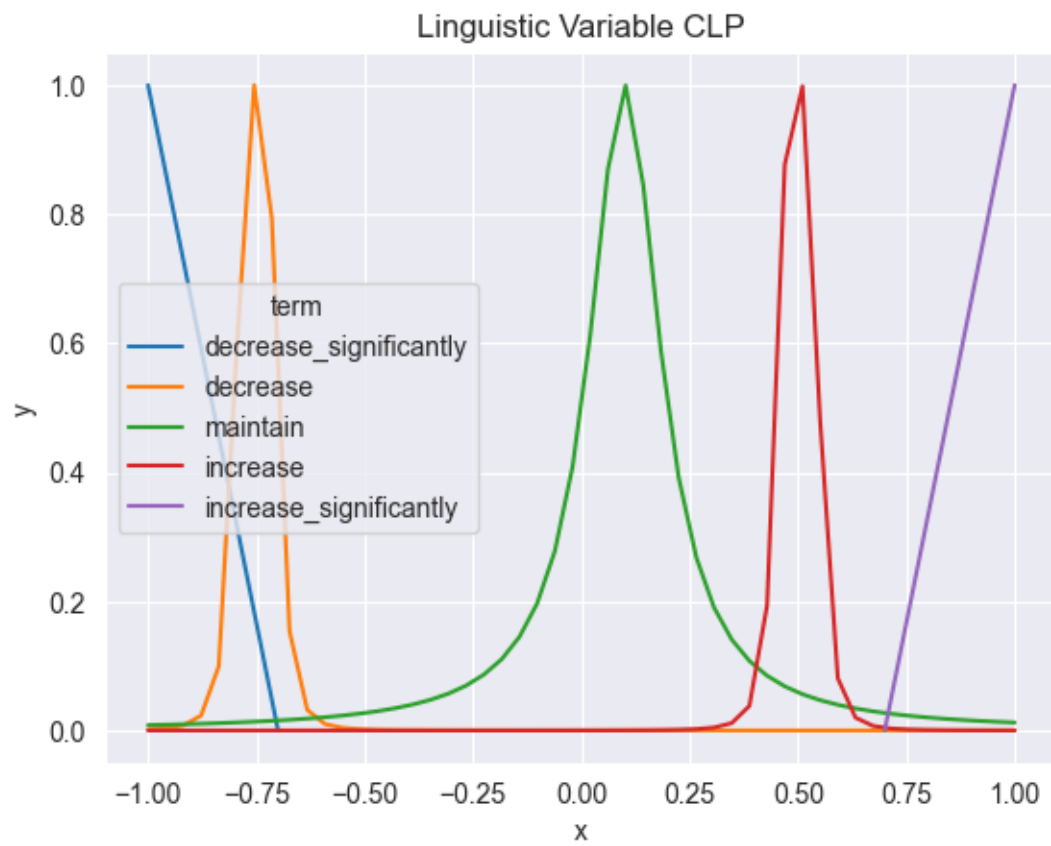
all the membership functions and linguistic terms

## 0.5 Rules

rules

## 0.6 Results

CLP Variation		Latency			
		low	moderate	high	very high
System Load	low	IS	IS	I	I
	moderate	I	I	I	I
	high	M	M	D	D
	critical	DS	DS	DS	DS



**Part II**

**Neural Networks**