NATIONAL TECHNICAL UNIVERSITY OF UKRAINE
“IGOR SIKORSKY KYIV POLYTECHNIC INSTITUTE”

Faculty of Informatics and Computer Engineering

Department of Computer Engineering

Lab 1 Report

Implementation and Visualization of Fuzzy Logic Membership Functions and Operators

Variant 12

Student, group IM-14
(group code)

in the educational and professional program

“**Software Engineering For Computer System”**

Specialty 121 "Computer Engineering"

Mehmet KULUBECİOGLU

Reviewer Volodymyr Shymkovych
(position, academic degree, academic status, surname and initials)

Kyiv – 2023

Abstract

This laboratory work focuses on the implementation and visualization of fuzzy logic concepts using Python. The project explores various membership functions, logical operators, and the complement of fuzzy sets. By leveraging the NumPy and Matplotlib libraries, the study defines and visualizes triangular, trapezoidal, Gaussian, generalized bell, sigmoid, and polynomial-based membership functions. Additionally, it implements logical operators using minimax and plausible interpretations and demonstrates the complement of a fuzzy set. The results provide a clear understanding of fuzzy logic principles and their applications in computational systems.

Introduction

Fuzzy logic is a mathematical approach to handle uncertainty and imprecision, widely used in artificial intelligence, control systems, and decision-making processes. Unlike classical logic, fuzzy logic allows for partial membership, where elements can belong to a set with a degree between 0 and 1. This project aims to:

1. Define and visualize various fuzzy membership functions.
2. Implement logical operators using minimax and plausible interpretations.
3. Compute and visualize the complement of a fuzzy set.

The implementation is carried out in Python, utilizing NumPy for numerical computations and Matplotlib for visualization.

Methodology

The project is divided into eight tasks, each addressing a specific aspect of fuzzy logic:

1. **Triangular and Trapezoidal Membership Functions:** These functions were defined to model linear transitions in membership degrees. The triangular function uses three parameters (a, b, c), while the trapezoidal function uses four (a, b, c, d).
2. **Gaussian Membership Function:** A two-sided Gaussian function was implemented to model smooth, bell-shaped membership degrees, defined by a mean and standard deviation.
3. **Generalized Bell Membership Function:** This function was defined using parameters (a, b, c) to create a flexible bell-shaped curve.

4. **Sigmoid Functions:** Three types of sigmoid functions (one-sided, two-sided, and asymmetric) were implemented to model S-shaped transitions.
5. **Polynomial Membership Functions:** Z-, Pi-, and S-shaped polynomial functions were defined using piecewise functions to model non-linear membership transitions.
6. **Minimax Interpretation of Logical Operators:** Logical AND and OR operators were implemented using the minimax approach, where AND returns 1 only if all inputs are 1, and OR returns 0 only if all inputs are 0.
7. **Plausible Interpretation of Operators:** Conjunctive (minimum) and disjunctive (maximum) operators were implemented to model fuzzy AND and OR operations.
8. **Complement of a Fuzzy Set:** The complement of a fuzzy set was computed as 1 minus the membership degree of each element and visualized.

The universe of discourse was defined as a range from 0 to 10 with 100 evenly spaced points using np.linspace. Each membership function was plotted using Matplotlib to visualize the membership degrees.

Results and Discussion

The implementation successfully achieved the following outcomes:

1. **Membership Functions Visualization:**
    - The triangular and trapezoidal functions produced linear membership transitions, as expected. The triangular function peaked at the midpoint, while the trapezoidal function maintained a plateau between two points.
    - The Gaussian function exhibited a smooth, bell-shaped curve centered at the mean value, with the spread controlled by the standard deviation.
    - The generalized bell function provided a more flexible bell shape, adjustable through its parameters.
    - Sigmoid functions showed S-shaped transitions, with the one-sided sigmoid increasing monotonically, the two-sided sigmoid forming a peak, and the asymmetric sigmoid allowing for different slopes on either side.
    - Polynomial functions (Z-, Pi-, and S-shaped) demonstrated non-linear transitions, with the Z-shaped function decreasing, the Pi-shaped function forming a peak, and the S-shaped function increasing.

2. **Logical Operators:**
    - The minimax interpretation of AND and OR operators produced binary outputs (0 or 1) based on the minimum and maximum values of the input sets, respectively.
    - The conjunctive and disjunctive operators returned the minimum and maximum of two membership degrees, respectively, aligning with fuzzy logic principles.
3. **Fuzzy Set Complement:**
    - The complement of a sample fuzzy set was computed and visualized, showing that membership degrees were inverted (e.g., 0.2 became 0.8), as expected in fuzzy logic.

The visualizations provided clear insights into the behavior of each membership function and operator, making it easier to understand their roles in fuzzy systems.

Conclusion

This project successfully implemented and visualized key concepts of fuzzy logic, including various membership functions, logical operators, and the complement of a fuzzy set. The use of Python, NumPy, and Matplotlib enabled efficient computation and clear visualization of the results. The study enhanced my understanding of fuzzy logic principles and their applications in computational systems. Future work could extend this project by applying these concepts to real-world problems, such as fuzzy control systems or decision-making models.

References

1. Zadeh, L. A. (1965). "Fuzzy Sets." *Information and Control*, 8(3), 338-353.
2. Ross, T. J. (2010). *Fuzzy Logic with Engineering Applications*. Wiley.
3. NumPy Documentation: https://numpy.org/doc/
4. Matplotlib Documentation: https://matplotlib.org/stable/contents.html