Fuzzy Relations

September 27, 2024

[1]:
```python
import numpy as np

A = np.array([0.1, 0.4, 0.7, 1.0])
B = np.array([0.2, 0.5, 0.8])

def fuzzy_cartesian_product(A, B):
    cartesian_product = np.zeros((len(A), len(B)))
    for i in range(len(A)):
        for j in range(len(B)):
            cartesian_product[i][j] = min(A[i], B[j])
    return cartesian_product

def max_min_composition(R1, R2):
    composition = np.zeros((R1.shape[0], R2.shape[1]))
    for i in range(R1.shape[0]):
        for j in range(R2.shape[1]):
            min_values = np.zeros(R1.shape[1])
            for k in range(R1.shape[1]):
                min_values[k] = min(R1[i, k], R2[k, j])
            composition[i, j] = np.max(min_values)
    return composition

def max_product_composition(R1, R2):
    composition = np.zeros((R1.shape[0], R2.shape[1]))
    for i in range(R1.shape[0]):
        for j in range(R2.shape[1]):
            product_values = np.zeros(R1.shape[1])
            for k in range(R1.shape[1]):
                product_values[k] = R1[i, k] * R2[k, j]
            composition[i, j] = np.max(product_values)
    return composition

A = np.array([0.2, 0.6, 0.9])
B = np.array([0.3, 0.5, 0.8])
cartesian_product = fuzzy_cartesian_product(A, B)
print("Fuzzy Cartesian Product (A × B):")
print(cartesian_product)
```

```python
R1 = np.array([[0.2, 0.4, 0.6],
               [0.5, 0.7, 0.9]])

R2 = np.array([[0.1, 0.3],
               [0.6, 0.8],
               [0.5, 0.9]])

max_min_comp = max_min_composition(R1, R2)
print("\nMax-Min Composition (R1 o R2):")
print(max_min_comp)
max_product_comp = max_product_composition(R1, R2)
print("\nMax-Product Composition (R1 o R2):")
print(max_product_comp)
```

```
Fuzzy Cartesian Product (A × B):
[[0.2 0.2 0.2]
 [0.3 0.5 0.6]
 [0.3 0.5 0.8]]

Max-Min Composition (R1 o R2):
[[0.5 0.6]
 [0.6 0.9]]

Max-Product Composition (R1 o R2):
[[0.3  0.54]
 [0.45 0.81]]
```