Josue Diaz

Making Fuzzy Sets and Elements

Fuzzy Element: Having a value between 0 and 1 is represented as a fuzzy element.

```
val element = FuzzyElement("elementName", membershipValue)
```

For instance:

```
val x1 = FuzzyElement("x1", 0.2)
```

Fuzzy Set: an assembly of fuzzy components.

```
val set = FuzzySet("setName", List(element1, element2, ...))
```

For instance:

```
val A = FuzzySet("A", List(x1, x2, x3))
```

Carrying Out Fuzzy Operations

The `FuzzyOperations` object to perform operations:

Union:

```
val unionSet = FuzzyOperations.union(setA, setB)
```

Intersection:

```
val intersectionSet = FuzzyOperations.intersection(setA, setB)
```

Complement:

```
val complementSet = FuzzyOperations.complement(set)
```

Addition:

```
val additionSet = FuzzyOperations.addition(setA, setB)
```

Multiplication:

```
val multiplicationSet = FuzzyOperations.multiplication(setA, setB)
```

XOR:

```
val xorSet = FuzzyOperations.xor(setA, setB)
```

Alpha-Cut:

```
val alphaCutElements = FuzzyOperations.alphaCut(set, alphaValue)
```

Assigning Variables and Managing Scopes

Assign a Variable:

```
Environment.assign("variableName", value)
```

Retrieve a Variable:

```
val value = Environment.get[ValueType]("variableName")
```

Enter a New Scope:

```
Environment.enterScope()
```

Exit the Current Scope:

```
Environment.exitScope()
```

Defining and Evaluating Logic Gates

Define a Logic Gate:

```
val gate = LogicGate("gateName", inputs => {
   val a = inputs("A")
  val b = inputs("B")
 })
```

For instance:

```
val logicGate1 = LogicGate("logicGate1", inputs => {
  val a = inputs.getOrElse("A", 0.0)
  val b = inputs.getOrElse("B", 0.0)
  Math.min(1.0, a + b)
})
```

Assign Logic Gate to a Variable:

```
Environment.assign("gateName", gate)
```

Test a Logic Gate:

```
val result = TestGate.testGate("gateName", Map("input1" -> value1,
"input2" -> value2))
```

For instance:

```
val testResult = TestGate.testGate("logicGate1", Map("A" -> 0.5, "B"
-> 0.7) which is 1
```