PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Departamento de Engenharia Elétrica

FUZZYCOM – COMPONENTE DE LÓGICA FUZZY

Aluno: Cláudio Magno Martins Moraes
Orientador: Marley Vellasco

Sumário

1. Introdução

Esse trabalho visa o estudo aprofundado da lógica fuzzy para o desenvolvimento de um componente de Lógica Fuzzy que implementa seus conceitos de técnicas, como por exemplo Conjunto Fuzzy, Conjunto Singleton, α-cut, Sistema de Inferência Fuzzy, Aritmética Intervalar e Operações com Números Fuzzy. Para tal, foram necessários estudos preliminares ao desenvolvimento do projeto em si.

Na seção 2 será feito um detalhamento sobre a lógica fuzzy mostrando todas as suas particularidades que devem servir como base para a implementação do componente. Na seção 3 segue a metodologia do uso de componentes e na seção 4 é feito um estudo de outro sistema Fuzzy e enfim, é feita a proposta de componente a ser desenvolvido em trabalhos futuros.

2. Uma introdução à Lógica Fuzzy

A lógica tradicional lida com problemas assumindo sempre a possibilidade de apenas dois estados: verdadeiro ou falso. Geralmente, esta forma de modelagem de problemas é suficiente, entretanto, existem situações em que é desejável a solução com valores intermediários.

De fato, problemas desse tipo podem ser resolvidos utilizando a lógica clássica, mas para isso seria necessário a utilização de equações matemáticas complexas que nem sempre resultariam em uma boa solução.

A fim de ajudar na solução de desafios como esses é que surgiu a Lógica Fuzzy (LF). Desenvolvida por Lofti Zadeh (1965 – Fuzzy Sets), a LF considera os elementos pertencentes a um determinado conjunto com um certo *grau de pertinência*. Portanto, um dado elemento que na lógica tradicional simplesmente não pertencia a um certo conjunto, pode pertencer a esse mesmo conjunto com um grau de pertinência baixo, se a modelagem for feita utilizando lógica fuzzy.

Uma boa definição para a LF é: *"Sistema não-linear de mapeamento de um vetor de entrada em uma saída escalar, capaz de incorporar tanto o conhecimento objetivo quanto o conhecimento subjetivo"*. Nesse caso, conhecimento objetivo, são os dados históricos, enquanto o conhecimento subjetivo representa a informação linguística que, em geral, é muito difícil de quantificar utilizando a matemática tradicional.

2.1. Sistemas Fuzzy

"Conforme a complexidade de um sistema aumente, a nossa habilidade de fazer declarações precisas e significativas sobre o seu comportamento diminui, até alcançar um limite além do qual precisão e relevância tornam-se características mutuamente exclusivas" (Lofti Zadeh). Conforme estudado por Zadeh, a transcrição acima, é definida como o Princípio da Incompatibilidade, e mostra a relevância em utilizar a Lógica Fuzzy para auxiliar na resolução de problemas que tradicionalmente são difíceis de resolver.

A idéia básica de um sistema fuzzy, é a existência do que é chamado de conjuntos fuzzy. Esses conjuntos são funções que mapeiam um valor escalar em um número limitado entre 0 e 1, que indica o grau de pertinência desse valor ao conjunto.

A visão geral de um sistema fuzzy pode ser percebida na figura a seguir:

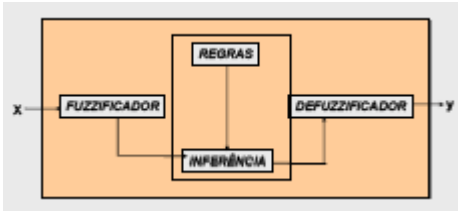

Figura 1: Visão geral de um sistema fuzzy

Como pode ser observado, o processo de inferência passa por diferentes etapas, que serão detalhadas a seguir:

Fuzzificador

Nesse passo, é feita a transformação das variáveis de entrada do problema em valores fuzzy. Para cada valor de entrada é aplicada uma função de pertinência, a qual retornará o *grau de pertinência* da proposição. Esse valor deve estar necessariamente limitado entre 0 a 1. O grau de pertinência 0 significa que o valor não pertence ao conjunto, enquanto o grau de pertinência 1 indica que o valor é uma representação completa do conjunto.

Regras

As regras são fornecidas por especialistas ou são extraídas de dados numéricos, utilizando, por exemplo, a capacidade de aprendizado das redes neurais. Para a elaboração dessas regras é importante ter em mente alguns conceitor importantes. São eles:

- Variáveis linguísticas: elas são o centro da técnica de modelagem de sistemas fuzzy. Com elas é possível nomear os conjuntos, e ainda qualificá-los utilizando os qualificadores (muito, pouco). Dessa forma, a modelagem do sistema se torna muito mais próxima do mundo real.
- Conexões lógicas: do tipo E/OU, para criar a relação entre as variáveis.
- Implicações: do tipo Se A então B.

Inferência

Nessa etapa, as regras são aplicadas sobre os valores de entrada já "fuzzificados", fazendo portanto o mapeamento de conjutos fuzzy em conjuntos fuzzy. Aqui também é determinado como as regras são ativadas e combinadas, e é criada a região resultante da aplicação das regras.

Defuzzificador

Aqui, as regiões resultantes do processo de inferência são convertidas em valores precisos para a variável de saída do sistema. Esta etapa corresponde à ligação fucional entre as regiões Fuzzy e o valor esperado.

Existem diversas técnicas de defuzzificação. As mais comuns são:

- Centróide
- Máximo
- Média dos máximos

2.2. Características da Lógica Fuzzy

As características da lógica fuzzy que merecem destaque são:

- Modelagem de problemas complexos: capazes de lidar com proclemas complexos, com propriedades não-lineares;
- Modelagem cognitiva: Têm habilidate de codificar o conhecimento de forma similar ao modo como os especialistas expressam o processo de decisão. Dessa maneira a aquisição de conhecimento se torna mais fácil, mais confiável e menos sujeita a erros;
- Complexidade reduzida: Sistemas Fuzzy possuem menos regras, similares às expressas por especialistas;
- Modelagem de sistemas envolvendo múltiplos especialistas: capazes de conciliar informações de especialistas consistentes ou conflitantes;
- Manipulação de incertezas: tratam as incertezas de forma consistente e matemática.

2.3. Conjuntos Fuzzy

A teoria de conjuntos fuzzy é uma extensão da teoria dos conjuntos clássicos. Na última, os conjuntos são denominados "crisp" e um determinado elemento do universo de discurso, ou seja, o domínio, pertence ou não pertence ao referido conjunto. Já na primeira, para cada elemento existe um **grau de pertinência** a um dado conjunto.

Em alguns casos a teoria clássica é satisfatoriamente utilizada, entretanto para outros ela pode não ser suficiente, como é o caso dos conjuntos cujo limite entre pertinência e não-pertinência não é claro, e existe uma transição gradual entre esses dois grupos. Por exemplo:

- O conjunto de pessoas altas
- O conjunto de carros velozes
- O conjunto de pessoas com alta renda

Utilizando os conjuntos fuzzy podemos definir critérios e graus de pertinência para essas situações. A função característica é generalizada de forma a assumir um número infinito de valores dentro do intervalo [0,1].

Um conjunto fuzzy A em um universo U é definido por uma fuunção de pertinência:

$$\mu_A(x): X \rightarrow [0,1]$$

Exemplos:

- Pessoas altas:

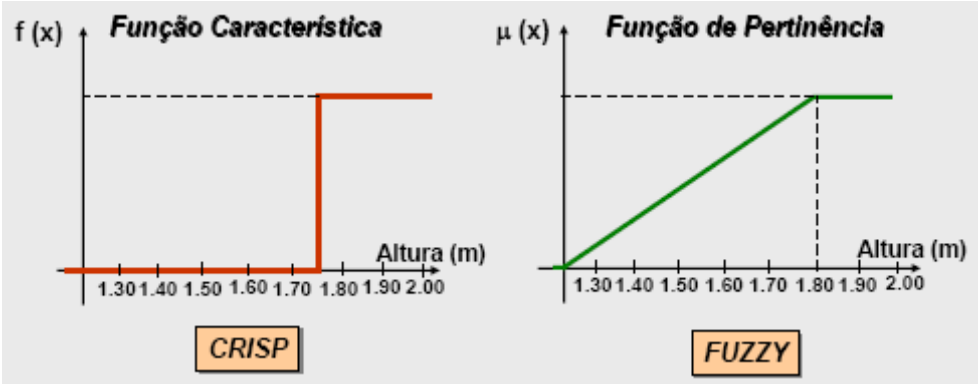

Figura 2: Comparação entre função característica e função de pertinência para o conjuntos de pessoas altas

- Carros caros:

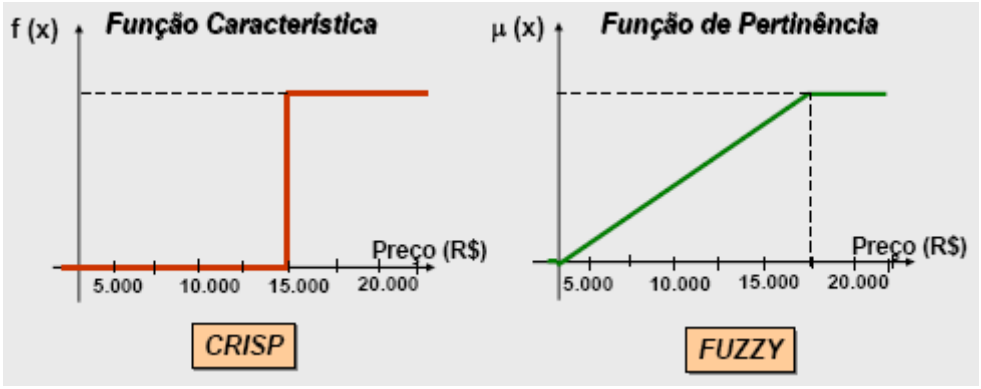

Figura 3: Comparação entre função característica e função de pertinência para o conjuntos de carros caros

2.3.1. Propriedades dos conjuntos fuzzy

Os conjuntos fuzzy carregam algumas propriedades e conceitos que serão descritos a seguir:

- Altura: é o maior grau de pertinência permitido pela função de pertinência

- Normalização: um certo conjunto fuzzy é normal se sua altura for igual a 1. É sabido que para obter um bom desempenho, os conjuntos fuzzy devem ser normalizados
- Domínio do conjunto fuzzy: é o universo total de valores possível para os elementos de um conjunto. Essa gama de valores é dependente do contexto
- Suporte do conjunto: é a área efetiva do domínio de um conjunto fuzzy que apresenta valores $\mu(x) > 0$. Os conjuntos que apresentam um único ponto em X com valor $\mu(x) = 1$, é chamado de Conjunto Singleton
- Conjunto α-cut: é uma restrição imposta ao domínio, baseada no valor de α. Os elementos do domínio que possuem $\mu(x)$ acima de um certo valor de α estão contidos nesse conjunto. A utilização dos conjuntos α-cut é útil para as funções com longos "tails", que tendem a possuir valores muito baixos de $\mu(x)$ por um domínio extenso. Isso ajuda a reduzir o ruído.
- Universo de discurso: é o espaço completo de variação de uma variável do modelo.

2.3.2. Variáveis Linguísticas

Como dito anterirmente, as variáveis linguísticas têm a função de fornecer uma maneira sistemática para uma caracterização aproximada de fenômenos complexos ou mal definidos. Através da utilização de variáveis cujos valores são nomes de conjuntos fuzzy, a simplificação e o melhor entendimento do problema é conseguido com maior sucesso que na lógica tradicional.

2.3.3. Funções de pertinência

As funções de pertinência podem ter forma padrão ou podem ser definidas pelo usuário. Seu objetivo é fazer a correspondência de um valor ou uma variável linguística em conjuntos fuzzy.

São diversos os tipos de funções geralmente utilizados para a modelagem de uma problema Fuzzy. O formato do conjunto fuzzy é definido pela função de pertinência utilizada. São elas:

- *Linear*: é o conjunto mais simples, e em geral é uma boa escolha para aproximação de conceitos que não são bem compreendidos

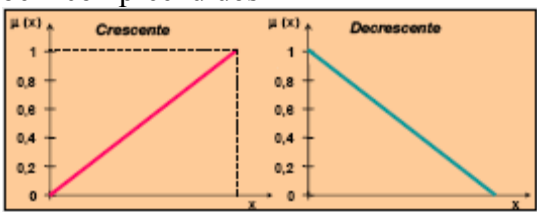

Figura 4: Formato de conjunto linear

- *Trapezoidal*: nesse formato são reparadas duas características: rápido processamento e presença de descontinuidades. Sua função e formato é mostrado na figura 5

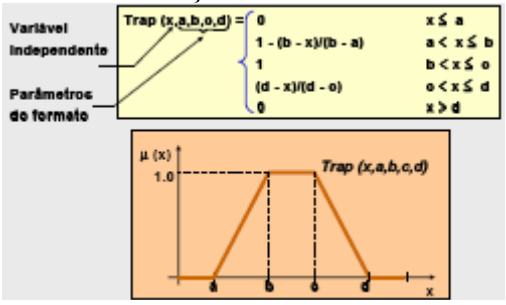

Figura 5: Função trapezoidal e o formato do conjunto

- *Triangular*: muito parecida com a trapzoidal, não apresentando descontinuidade entre a mudança do crescimento da função

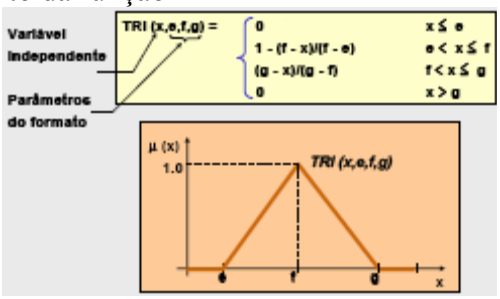

Figura 6: Função triangular e o formato do conjunto

- *Formato S*: caracterizado pela equação quadrática

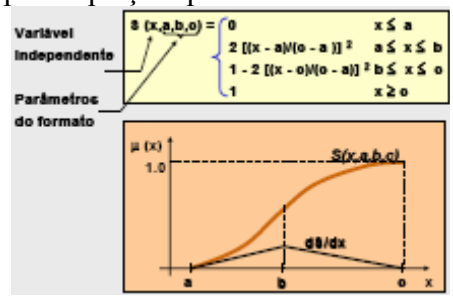

Figura 7: Função S e o formato do conjunto

- *Formato Z*: esse formato é complemento do formato S de dois parâmetros, ou seja, Z(x, a, b) = 1 – S(x, a, b).

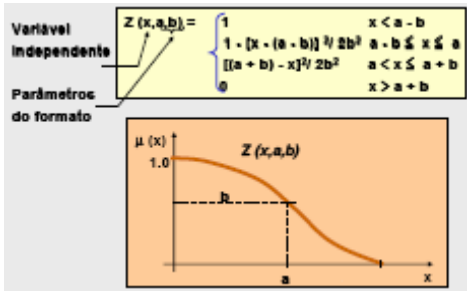

Figura 8: Função Z e o formato do conjunto

- *Formato PI*: é a junção das curvas S e Z:

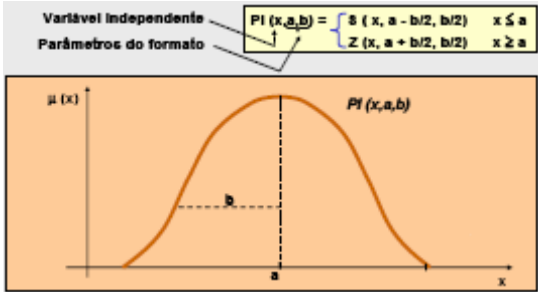

Figura 9: Função PI e o formato do conjunto

- *Gaussiana*: tem uma distribuição normal, ou seja, tende a zero para valores muito maiores ou muito menores do que a média:

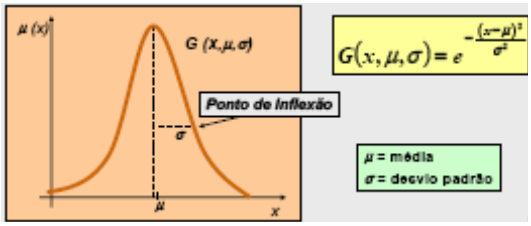

Figura 10: Função gaussiana e o formato do conjunto

- *Singleton*: essa função não representa, de fato, um conjunto fuzzy, porém simplifica os cálculos para produzir as saídas fuzzy:

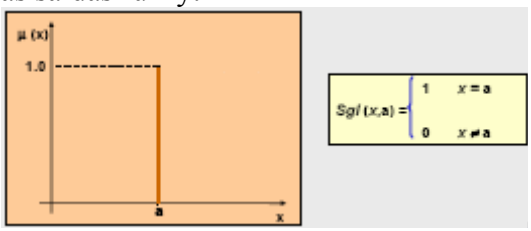

Figura 11: Função singleton e o formato do conjunto

- *Irregulares*: em algumas situações, as formas padrões (citadas acima) não conseguem capturar de forma consistente a semêntica de uma variável. Para esses casos é necessário uma representação arbitrária:

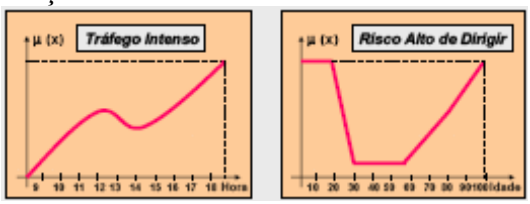

Figura 12: Funções irregulares e formato do conjunto

2.3.4. Operações Básicas

Da mesma forma que nos conjuntos clássicos (ou, crisps), nos conjuntos fuzzy também existem as operações básicas para combinar e modificar conjuntos. Porém, na lógica fuzzy, essas funções são aplicadas às funções de pertinência.

É necessário entender que um certo elemento é membro de um conjunto fuzzy se:

- Está dentro do domínio do conjunto;
- O seu grau de pertinência é maior que 0 (zero);
- Está acima do limite α-cut.

As operações básicas a que nos referimos são:

- *Interseção*: de forma análoga com os conjuntos ordinários, que fazem uso do operador AND, no mundo fuzzy geralmente se utiliza o Mínimo das funções de pertinência. Esse é um dos operadores de Zadeh. Portanto, uma operação de interseção de conjuntos fuzzy é feita com base na seguinte fórmula:

$$\mu_{A\cap B}(x) = \mu_A(x) \wedge \mu_B(x) \qquad \forall x \in X$$

- *União:* assim como nos conjuntos crisp, que utilizam o operador OR, em conjuntos fuzzy o Máximo das funções de pertinência é o operador (de Zadeh) geralmente utilizado:

$$\mu_{A\cup B}(x) = \mu_A(x) \vee \mu_B(x) \qquad \forall x \in X$$

- *Complemento:* da mesma maneira como nos conjuntos crisp, o complemento de um conjunto fuzzy A (~A) contém todos os elementos que não estão em A. Geralmente é utilizada a seguinte expressão para simbolizar a operação de complemento:

$$\mu_{\sim A}(x) = 1 - \mu_A(x) \qquad \forall x \in X, \text{ supondo conjuntos normalizados}$$

2.3.5. Modificadores – Operadores Semânticos

Como sabemos, em lógica fuzzy, é utilizado a todo momento variáveis linguísticas para a criação de regras. Essas variáveis geralmente necessitam do uso de modificadores a fim de alterar a intensidade com que um certo elemento faça parte de um determinado conjunto. Analogicamente, esses modificadores atuam na modelagem de sistemas fuzzy da mesma forma que advérbios e adjetivos atuam em um sentença, com isso, modificam a natureza de um conjunto fuzzy.

Os modificadores, também referidos como Hedges, são de diversas classes:

- *Intensificadores*: muito, extrememente;
- *Diluidores*: pouco, mais ou menos;
- *Complemento*: não;
- *Aproximadores*: em torno, aproximadamente, quase;
- *Etc*.

2.3.5.1. Quanto a aplicação dos modificadores

- Um ponto importante de se considerar é que da mesma forma que a ordem dos advérbios e adjetivos são importantes, a ordem dos hedges também é. Por exemplo:

  $$NÃO\ MUITO\ CHEIO \neq MUITO\ NÃO\ CHEIO$$

- Outro ponto importante é a utilização de vários hedges para modificar um único conjunto fuzzy. Por exemplo:

  $$POSITIVAMENTE\ NÃO\ MUITO\ CARO$$

  No exemplo acima, CARO é o conjunto fuzzy, e POSITIVAMENTE NÃO MUITO são os múltiplos modificadores.
- O processamento dos hedges também é feito de forma parecida com a linguagem:

  $$(POSITIVALEMENTE\ (NÃO\ (MUITO\ CARO)))$$

- Os hedges podem ser aplicados tanto no antecedente quanto no consequente da regra:
  - SE custo é muito ALTO ENTÃO margem de lucro é BAIXA
  - SE inflação(t-1) é muito GRANDE ENTÃO vendas são positivamente pequenas

2.3.5.2. Tipos de modificadores

Iremos detalhar agora como é feito o processamento em termos numéricos do uso de modificadores:

- *Intensificadores (muito, extremamente):* reduzem o grau de pertinência dos elementos que pertencem ao conjunto fuzzy, ou seja:

$$\mu_A(x) \geq \mu_{MUITO\ A}(x)$$

$$\mu_{MUITO\ A}(x) = [\mu_A(x)]^2$$

$$\mu_{CONC\ A}(x) = [\mu_A(x)]^n\ ,\ n \in [1,4]$$

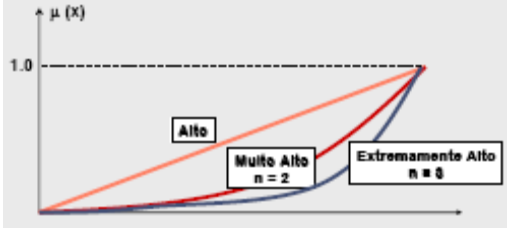

Figura 13: diferença do grau de pertinência devido a aplicação do hedge intensificador

- *Diluidores (um pouco, levemente):* têm a função de diluir a função de pertinência de uma certa região fuzzy:

$$\mu_A(x) \leq \mu_{\text{UM POUCO A}}(x)$$
$$\mu_{\text{UM POUCO A}}(x) = [\mu_A(x)]^{1/2}$$
$$\mu_{\text{DILUIDOR A}}(x) = [\mu_A(x)]^{1/n} \quad , n \in [1,4]$$

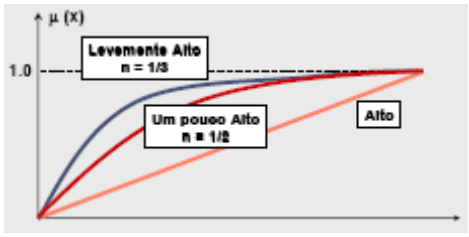

Figura 14: diferenção do grau de pertinência devido à aplicação do hedge diluidor

- *Aproximadores (em torno, perto de):* têm a função de alargar ou estreitar uma região fuzzy (tipo "sino"). As 2 figuras a seguir exemplificam a utilização dos aproximadores:

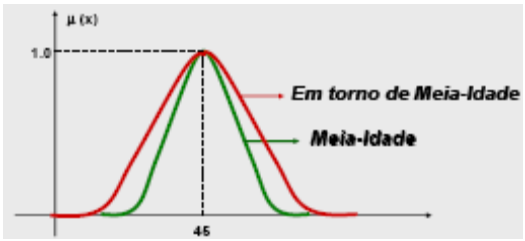

Figura 15: exemplificação de alargamento da região fuzzy com a utilização do aproximador "EM TORNO DE"

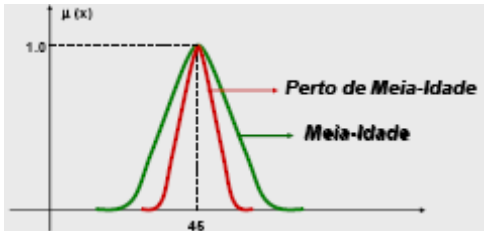

Figura 16: exemplificação do estreitamento da região fuzzy com a utilização do aproximador "PERTO DE"

2.4. Defuzificação

Nesta seção será melhor detalhada as formas de interpretação dos conjuntos fuzzy de saída. Como já citado anteriormente, existem vários métodos diferentes. São eles:

- *Máximo:* a utilização desse método leva a uma análise do conjunto fuzzy (B) de saída e a escolha, como valor preciso, o valor no universo de discurso da variável de saída *y* para o qual o grau de pertinência ($\mu_B(y)$) é máximo.

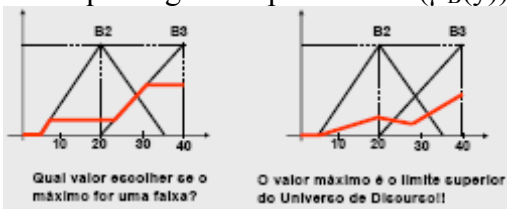

Figura 17: com o método Máximo, é escolhido o valor cujo grau de pertinência seja o maior

- *Média dos Máximos:* a saída precisa é obtida calculando-se a média entre os dois elementos extremos no universo de discurso que tiverem os maiores valores da função de pertinência do conjunto fuzzy de saída (B)

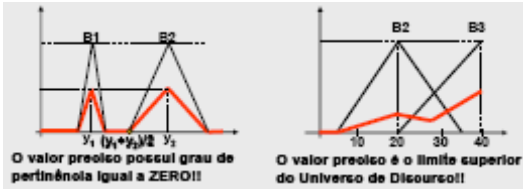

Figura 18: com o método Média dos Máximo, a saída precisa é dada através da média dos maiores valores da função de pertinência do conjunto de saída

- *Centróide:* a saída precisa (y_C) é o valor no universo de discurso que corresponde ao centro de gravidade do conjunto fuzzy de saída (B)

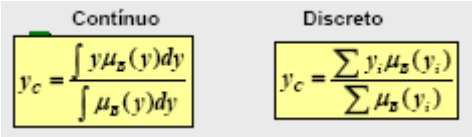

Figura 19: fórmula para obtenção da saída precisa utilizando o método Centróide de defuzificação

3. Metodologia

Componentes de software é um elemento de software que encapsula uma séria de funcionalidades, e funciona como uma unidade independente que pode ser utilizado juntamente com outros componentes a fim de formar um sistema complexo. Segundo Brown e Wallnau[1], um componente é "uma não trivial, quase independente, e substituível parte de um sistema que cumpre uma função clara no contexto de uma arquitetura bem definida". Todos os componentes que formam um sistema se comunicam através de interfaces, que devem ser claramente definidas e documentadas.

A técnica do desenvolvimento de softwares com o uso de componentes é uma tendência mundial, uma vez que propicia maior velocidade de implementação do produto final. Esses componentes, em geral, são desenvolvidos com a preocupação de reuso, e por isso a confiabilidade dos sistemas por eles formados é maior, já que com o reuso é aumentado o número de testes realizados sobre o componente.

De acordo com a programação orientada a objetos, componentes são compostos por objetos que colaboram entre si. Segundo Sametinger[2]: "componentes de *software* reutilizáveis são artefatos autocontidos, facilmente identificáveis, que descrevem e/ou executam funções específicas e têm interfaces claras, documentação apropriada e uma condição de reuso definida". A partir dessa definição podemos identificar que as classes e as interfaces da orientação a objetos são as responsáveis por descrever as funções específicas citadas. A primeira, é uma modelagem descritiva através da qual são definidos métodos e atributos de um objeto, enquanto a segunda se restringe a descrever uma lista de serviços, não descrevendo portanto como são implementados, mas apenas quais são oferecidos.

A criação de uma biblioteca básica de sistemas fuzzy é importante pois ela provê, de forma modular e estruturada, alguns recursos mínimos para componente de neuro-fuzzy (NFCOM), que é o componente base para diversas aplicações de apoio a decisão em desenvolvimento pelo laboratório de inteligência computacional aplicada (ICA).

A motivação para a criação de um componente que une as técnicas de redes neurais e lógica fuzzy, criando assim o NFCOM, é devido ao fato de que ambas as técnicas apresentam

vantagens, mas falham em determinados pontos. Portanto, uma forma de conseguir um bom resultado é unir as vantagens de cada uma a fim de obter o melhor aproveitamento possível dessas técnicas.

As redes neurais, por exemplo, apresentam uma boa capacidade de aprendizado, entretanto, são difíceis de serem interpretadas. São como caixas pretas para o usuário. Por outro lado, sistemas fuzzy são constituídos de regras linguísticas interpretáveis, mas não conseguem aprender como as redes neurais. Portanto, uma forma de criar um componente que utilize a parte boa de cada técnica é utilizando os algoritmos de aprendizado derivados da teoria das redes neurais para gerar os parâmetros de entrada (conjuntos e regras fuzzy) através do processamento de amostra de dados.

3.1. Estudo de caso – FLINT [5]

A ferramenta FLINT *(Fuzzy Logic Interface Toolkit)* é uma ferramenta que aumenta o poder da tomada de decisões. Ela fornece um conjunto de facilidades para programadores que desejam incorporar a incerteza dentro de seus sistemas especialistas e aplicações de apoio a decisão. Diferentemente dos sistemas especialistas tradicionais, Flint fornece suporte onde o domínio do conhecimento não é bem definido.

O componente FUZZYCOM que é proposto nesse trabalho segue a idéia dessa ferramenta.

3.2. Desenvolvimento embrionário do FUZZYCOM

Como início do trabalho de desenvolvimento do componente de lógica fuzzy, foi feito uma implementação básica seguindo o diagrama de classes da figura 20, a seguir. Como dito anteriormente, essa estrutura é baseada na idéia do FLINT.

Faremos agora uma explicação sobre a idéia por trás de cada uma das classes do diagrama:

- *FuzzySystem*: esta é a classe em que é montado todo o sistema fuzzy. Aqui deve ser definido quais variáveis devem existir no sistema, quais os seus conjuntos e limites, e ainda, é nela em que são especificadas as variáveis de entrada com seus respectivos valores ordinários.

  O tratamento com relação ao controle das variáveis do sistema é todo feito com uma tabela hash estática, a qual tem como chave o nome da variável em questão. Essa escolha foi feita devido ao fato de que a utilização de vetores ficaria menos clara e pouco eficiente.
- *FuzzyVariable*: nesta classe se concentra toda a idéia das variáveis fuzzy. A criação de uma determinada variável é feita através da classe FuzzySystem que criará um novo objeto FuzzyVariable. Os únicos atributos necessários para essa criação são os limites superiores e inferiores e o nome da variável.

  A criação dos conjuntos da variável também é feita através da FuzzySystem, mas ela fará uma chamada ao método *addNewSet* da FuzzyVariable, que por sua vez mantém uma tabela hash que contém todos os conjuntos da variável em questão. A chave dessa tabela é o próprio nome do conjunto.
- *InputFuzzyVariable*: no momento de inserir dados no sistema fuzzy é necessário que esses sejam inseridos através da instanciação de um objeto InputFuzzyVariable. Ele será responsável pela transformação de um valor crisp em um valor Fuzzy em cada um dos conjuntos relativos à variável.
- *FuzzySet*: como classe responsável pelos conjuntos de uma determinada variável, FuzzySet necessita saber, de antemão, que tipo de função de pertinência será usada pelo tal conjunto. Essa informação deve ser passada no momento da criação do

conjunto através da criação de uma das classes que estendem MemberFunction, que representam as funções de pertinência.

- *MemberFunction*: esta é uma classe abstrata para representar as funções de pertinência. Dessa forma, toda implementação de tais funções devem estender esta classe imlpementando o método *Fuzzyfy*. Isso é importante pois o processo de fuzzificação é particular de cada função, porém, todas elas retornam um resultado final que é o grau de pertinência de um dado valor escalar a um dado conjunto de determinada variável.
- *Triangle*: uma das classes que estendem a classe anterior. Como dito, ela deve implementar a função *Fuzzyfy* e receber seus parâmetros particulares.
- *Trapezoid*: a mesma explicação dada a Triangle.
- *Linear*: idem Triangle.

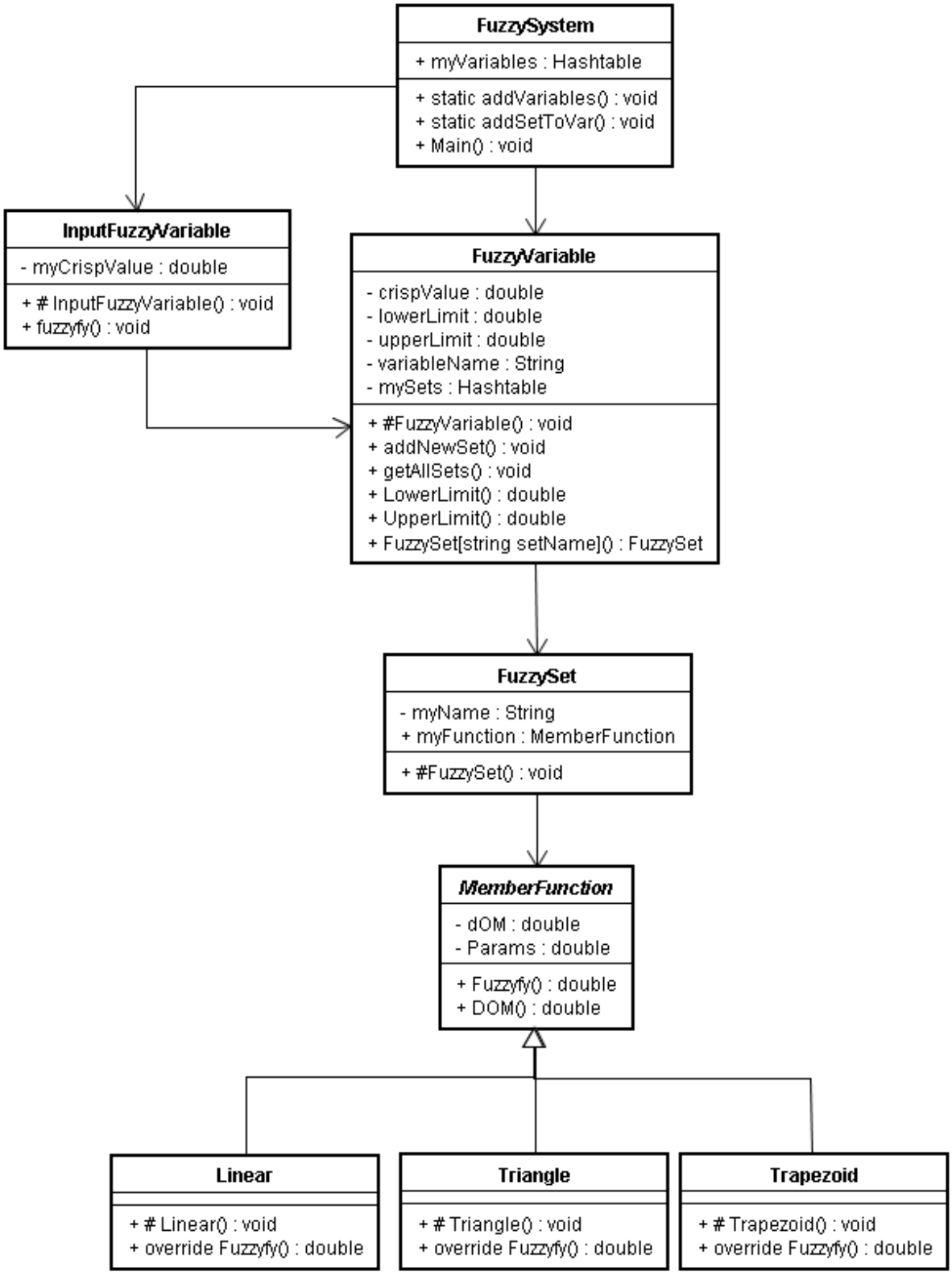

Figura 20: Diagrama de classes do desenvolvimento embrionário do componente de Lógica Fuzzy.

4. Conclusões e Trabalhos Futuros

O estudo teórico permitiu uma maior compreensão da utilização de componentes de software além de ter possibilitado a verificação das técnicas de inteligência computacional sendo aplicadas na realidade, e de forma integrada. Foi possível perceber como a união dessas técnicas aumentam o potencial de um sistema de apoio a decisão através de transformação de simples conjuntos de dados em informações relevantes e muitas vezes cruciais para o futuro de empresas.

Para continuidade do trabalho será necessário a finalização da implementação do componente, que se encontra em fase inicial, além da necessidade de testes massivos para futura integração com o principal interesse, que é a sua integração com o componente NeuroFuzzy.

5. Referências Bibliográficas

[1] - Brown, A.W., K.C Wallnau (1996). *Component-Based Software Engineering*. IEEE Computer Society Press.

[2] - Sametinger, J. (1997). Software Engineering with Reusable Components, New York: Springer.

[3]- J. J. Buckley and Y. Hayashi, Fuzzy neural networks: A survey, Fuzzy Sets and Systems 66 (1994)

[4] – Marley Vellasco, Notas de Aula, www.ica.ele.puc-rio.br

[5] – Fuzzy Logic Interface Toolkit, www.lpa.co.uk/fln.htm