V Escola de Redes Neurais, Promoção: Conselho Nacional de Redes Neurais
pp. c073-c090, 19 de julho, 1999 - ITA, São José dos Campos - SP

Lógica Nebulosa

Sandra Sandri, Cláudio Correa
INPE
São José dos Campos, SP 12201-970, Brazil
E-mails: sandri@lac.inpe.br, correa@pgrad.inpe.br

Abstract

Este trabalho é um tutorial sobre lógica nebulosa, trazendo conceitos básicos desta área de pesquisa bem como sua ligação com algumas outras áreas importantes como controle, algoritmos genéticos e redes neurais.

1. Introdução

Dois dos principais aspectos da imperfeição da informação são a imprecisão e a incerteza. Estas duas características são intrinsecamente ligadas e opostas entre si: quanto mais se aumenta a incerteza mais se diminui a imprecisão e vice-versa. Por exemplo, suponhamos que estejamos seguros em dizer que um dado filme começa entre 3h e 4h. Se formos obrigados a dar uma opinião mais precisa, tenderemos a aumentar a incerteza, dizendo, por exemplo, que o filme comecará as 3h30' com uma probabilidade diferente de 1. Uma informação imprecisa também pode ser vaga, como por exemplo, quando dizemos que o filme começa "por volta das 3h30' ".

As teorias mais conhecidas para tratar da imprecisão e da incerteza são respectivamente a teoria dos conjuntos e a teoria de probabilidades. Estas teorias, embora muito úteis, nem sempre conseguem captar a riqueza da informação fornecida por seres humanos. A teoria dos conjuntos não é capaz de tratar o aspecto vago da informação e a teoria de probabilidades, na qual a probabilidade de um evento determina completamente a probabilidade do evento contrário, é mais adaptada para tratar de informações frequentistas do que aquelas fornecidas por seres humanos.

A teoria dos conjuntos nebulosos foi desenvolvida a partir de 1965 por Lotfi Zadeh, para tratar do aspecto vago da informação [1]. A teoria dos conjuntos, que será chamada neste trabalho de "clássica", pode ser vista então como um caso particular desta teoria mais geral. A partir de 1978, Lotfi Zadeh desenvolveu a teoria de possibilidades [2], que trata a incerteza da informação, podendo pois ser comparada com a teoria de probabilidades. Esta teoria, por ser menos restritiva, pode ser considerada mais adequada para o tratamento de informações fornecidas por seres humanos que a de probabilidades. Efetivamente, mesmo no discurso usual percebemos que a noção de possibilidade é menos restritiva que aquela de probabilidade: é mais fácil dizer que algum evento é possível do que provável.

A teoria dos conjuntos nebulosos e a teoria de possibilidades são intimamente ligadas. Por exemplo, o conjunto nebuloso que modela a informação "idade avançada" pode ser usada para modelar a distribuição de possibilidade da idade de uma dada pessoa, da qual só sabemos que ela é idosa. O fato destas teorias serem ligadas é muito importante no sentido de que é possível se tratar tanto a imprecisão quanto a incerteza de um conjunto de informações em um único ambiente formal. De fato, a maior parte do tempo não é necessário fazer a distinção entre um conjunto nebuloso e uma distribuição de possibilidades.

Estas teorias têm sido cada vez mais usadas em sistemas que utilizam informações fornecidas por seres humanos para automatizar procedimentos quaisquer, como por exemplo no controle de processos, no auxílio à decisão, etc. Estas teorias têm sido utilizadas em aplicações que vão do controle de eletrodomésticos ao controle de satélites, do mercado financeiro à medicina, e tendem a crescer cada vez mais, sobretudo em sistemas híbridos, que incorporam abordagens conexionistas e evolutivas, no que é chamado hoje em dia, de "*soft computing*".

A teoria dos conjuntos nebulosos, quando utilizada em um contexto lógico, como o de sistemas baseados em conhecimento, é conhecida como lógica nebulosa, lógica difusa ou lógica "fuzzy". Na próxima seção deste trabalho, trazemos as definições básicas da teoria dos conjuntos nebulosos. São abordados conceitos como o complemento, a intersecção, a união, e a implicação nesta teoria, além de um estudo sobre algumas das propriedades destas operações.

A lógica nebulosa é uma das tecnologias atuais bem sucedidas para o desenvolvimento de sistemas para controlar processos sofisticados [3] [4] [5] [6] [7]. Com sua utilização, requerimentos complexos podem ser implementados em controladores simples, de fácil manutenção e baixo custo. O uso de sistemas construídos desta maneira, chamados de *controladores nebulosos*, é especialmente interessante quando o modelo matemático está sujeito a incertezas [8] [9] [10] [11] [12] [13] [14].

Um controlador nebuloso é um sistema nebuloso a base de regras, composto de um conjunto de regras de produção do tipo **Se** <*premissa*> **Então**<*conclusão*>, que definem ações de controle em função das diversas faixas de valores que as variáveis de estado do problema podem assumir [15] [16]. Estas faixas (usualmente mal

definidas) de valores são modeladas por conjuntos nebulosos e denominados de termos linguísticos. Na Seção 3, são descritos a estrutura e as componentes do controle nebuloso e, ao final desta seção apresentam-se diversos modelos encontrados na literatura.

A maior dificuldade na criação de sistemas nebulosos em geral, e de controladores nebulosos em particular, encontra-se na definição dos termos linguísticos e das regras. Uma das maneiras de sanar este problema consiste em lançar mão dos chamados modelos "neuro-fuzzy" [17] [18] [19] [20], em que estes parâmetros são aprendidos com a apresentação de pares (entrada, saída desejada) a uma rede neural cujos nós computam basicamente operadores de intersecção e união. A Seção 4 apresenta os conceitos principais envolvidos nesta abordagem.

Uma outra maneira de se aprender parâmetros de sistemas nebulosos consiste no uso de algoritmos genéticos (AG). Algoritmos genéticos são estratégias de busca adaptativa, baseados em um modelo altamente abstrato da evolução biológica [21]. São utilizados em problemas de otimização, em que se busca, se não uma solução ótima, ao menos uma solução suficientemente adequada para um dado problema [22] [23]. Nestes algoritmos, uma população de indivíduos (soluções potenciais) sofrem uma série de transformações unárias (mutação) e de ordem mais alta (cruzamento). Estes indivíduos competem entre si pela sobrevivência: um esquema de seleção, que favorece os indivíduos mais aptos seleciona os que sofrerão transformações, dando origem à próxima geração. Depois de algumas gerações, o algoritmo usualmente converge e o melhor indivíduo representa uma solução próxima da ótima.

Algoritmos genéticos têm sido usados em muitas aplicações ligadas a controle, em particular, envolvendo o desenvolvimento de controladores nebulosos, tanto no exterior [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] (ver também [34] [35] [36]) quanto no Brasil [37] [38] [39] [40]. A Seção 5 descreve algumas das características do uso de algoritmos genéticos no aprendizado de parâmetros de sistemas nebulosos.

O material apresentado nas próximas 3 seções deste trabalho é baseado em notas de aula de cursos apresentados no Curso de Pós-Graduação em Computação Aplicada do INPE, que, por sua vez, foram baseadas principalmente nos livros de Didier Dubois e Henri Prade [41] e de George Klir e Tina Folger [15], que trazem um material teórico denso sobre o assunto, no livro de Bernadette Bouchon-Meunier [Bouchon- Meunier, 1993] que, ao se propor a apresentar os conceitos fundamentais da teoria para um público mais leigo, desenvolveu uma obra mais sucinta e leve, e no livro de Dimiter Driankov, Hans Hellendoorn e Michael Reinfrank [3], que aborda principalmente aspectos ligados ao controle nebuloso. Estas 3 seções e também a seção relativa ao aprendizado de parâmetros nebulosos através de algoritmos genéticos foi extraída da dissertação [37], com pequenas alterações. O material relativo ao aprendizado de parâmetros nebulosos através de redes neurais, os sistemas conhecidos como "neuro-fuzzy", foi baseado principalmente nos artigos [18] e [17] e no tutorial [42].

2. Conceitos Básicos da Teoria dos Conjuntos Nebulosos

A teoria dos conjuntos nebulosos foi desenvolvida a partir de 1965 com os trabalhos de Lotfi Zadeh, professor na Universidade da Califórnia em Berkeley [1] (ver também [43] e [41]).

Formalmente, um conjunto nebuloso A do universo de discurso Ω é definido por uma função de pertinência $\mu_A : \Omega \rightarrow [0,1]$. Essa função associa a cada elemento x de Ω o grau $\mu_A(x)$, com o qual x pertence a A [1]. A função de pertinência $\mu_A(x)$ indica o grau de compatibilidade entre x e o conceito expresso por A:

- $\mu_A(x) = 1$ indica que x é completamente compatível com A;
- $\mu_A(x) = 0$ indica que x é completamente incompatível com A;
- $0 < \mu_A(x) < 1$ indica que x é parcialmente compatível com A, com grau $\mu_A(x)$.

Um conjunto A da teoria dos conjuntos clássica pode ser visto como um conjunto nebuloso específico, denominado usualmente de "crisp", para o qual $\mu_A : \Omega \rightarrow \{0,1\}$, ou seja, a pertinência é do tipo "tudo ou nada", "sim ou não", e não gradual como para os conjuntos nebulosos.

A diferença entre estes conceitos em relação à variável idade é ilustrada na Figura 1a e na Figura 1b, que descrevem respectivamente a representação do conceito "adolescente" através de um conjunto "crisp" e de um conjunto nebuloso.

O conjunto "crisp" A não exprime completamente o conceito de "adolescente", pois uma pessoa com 12 anos e 11 meses seria considerada completamente incompatível com este conceito. Na verdade, qualquer intervalo "crisp" que se tome para representar este conceito é arbitrário.

Já o conjunto nebuloso B permite exprimir que qualquer pessoa com idade entre 13 e 17 anos é um adolescente, acima de 19 ou abaixo de 11 não é considerado um adolescente, e no intervalo $[11,13]$ (respectivamente $[17,19]$) é considerado tanto mais adolescente quanto mais próxima de 13 (respectivamente de 17) é sua idade.

A cardinalidade de um conjunto nebuloso A é expressa como:

- Para Ω discreto

$$|A| = \sum_{x \in \Omega} \mu_A(x) \tag{1}$$

- Para Ω contínuo

$$|A| = \int_{\Omega} \mu_A(x) \tag{2}$$

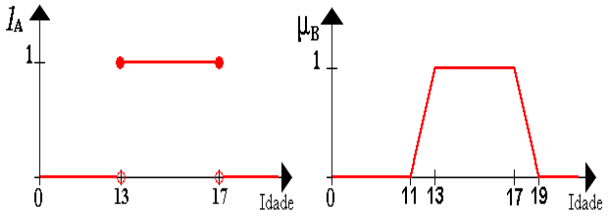

Figura 1: a) Função característica do conjunto "crisp" adolescente. b) Função trapezoidal característica do conjunto nebuloso adolescente.

Pode-se obter a representação aproximada de um conjunto nebuloso A em Ω através de conjuntos "crisp" em Ω. Estes subconjuntos, denotados por A_α e denominados de cortes de nível ou cortes-α, são definidos como:

$$A_\alpha = \{x \in \Omega \ / \ \mu_A(x) \geq \alpha\} \tag{3}$$

Os casos extremos destes conjuntos são o suporte de A, denotado como $Su(A)$, que agrupa elementos de Ω que são de alguma forma compatíveis com o conceito expresso por A, e o núcleo de A, denotado por $Nu(A)$, que agrupa elementos de A que são completamente compatíveis com o conceito expresso por A.

$$Su(A) = \{x \in \Omega \ / \ \mu_A(x) > 0\} = \lim_{\alpha \to 0} A_\alpha \tag{4}$$

$$Nu(A) = \{x \in \Omega \ / \ \mu_A(x) = 1\} = A_1 \tag{5}$$

A altura de A representa o maior grau de compatibilidade dos elementos de Ω em relação ao conceito expresso por A:

$$Al(A) = \sup_{x \in \Omega} \mu_A(x) \tag{6}$$

Um conjunto nebuloso A é dito normalizado se e somente se $Al(A) = 1$. A Figura 2 ilustra a cardinalidade, a altura, o suporte, o núcleo, e o corte de nível 0.5 de um conjunto nebuloso A [44] [45] [46].

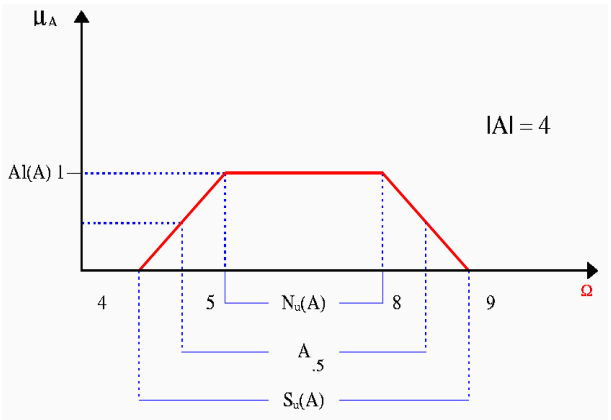

Figura 2: Cardinalidade, altura, suporte, núcleo e o corte de nível 0.5 do conjunto nebuloso A.

Um conjunto nebuloso A é dito ser *convexo* em $\Omega \subseteq \Re$ se e somente se, seus cortes-α são convexos, i.e. sse

$$\forall x, y \in \Omega, \forall \lambda \in [0,1], \\ \mu_A(\lambda x + (1-\lambda)y) \geq \min(\mu_A(x), \mu_A(y)) \tag{7}$$

Seja A um conjunto nebuloso convexo, com $Su(A) = [s_{\inf},\ s_{\sup}]$ e $Nu(A) = [n_{\inf},\ n_{\sup}]$. Um intervalo nebuloso é um conjunto nebuloso normalizado e convexo em $\Re$ tal que a função que descreve $\mu_A(x)$ entre $s_{\inf}$ e $n_{\inf}$ e aquela entre $n_{\sup}$ e $s_{\sup}$ são estritamente monotônicas (respec. crescente e decrescente). O conjunto nebuloso A da Figura 2 é um intervalo nebuloso. Um número nebuloso é um intervalo nebuloso unimodal.

2.1. Operações em Conjuntos Nebulosos

Similarmente às operações nos conjuntos "crisp", existe a necessidade de proceder às operações de *intersecção*, *união* e *negação*, entre outras, nos conjuntos nebulosos.

2.2. Operadores de Intersecção, União e Complemento

Sejam A e B conjuntos nebulosos definidos em Ω. Pode-se expressar a intersecção destes conjuntos, como um outro conjunto $E = A \cap B$. Da mesma forma, pode-se expressar a união como um conjunto $F = A \cup B$.

Na teoria dos conjuntos nebulosos, a intersecção é implementada por uma família de operadores denominados de $t-normas$, e a união é implementada por uma família de operadores denominados de $t-conormas$ ou $S-normas$ [41].

Uma função $\nabla : [0,1]^2 \to [0,1]$, é dita ser comutativa, associativa e monotônica se ∇ satisfaz as seguintes propriedades, respectivamente, para $\forall a, b \in [0,1]$:

Comutatividade: $\nabla(a,b) = \nabla(b,a)$;

Associatividade: $\nabla(a, \nabla(b,c)) = \nabla(\nabla(a,b), c)$;

Monotonicidade: $\nabla(a,b) \leq \nabla(c,d) \ \ se\ a \leq c\ e\ b \leq d$.

Um operador $\top : [0,1]^2 \to [0,1]$ é denominado de $t-norma$ se $\top$ é comutativo, associativo e monotônico e verifica a seguinte propriedade, para $\forall a \in [0,1]$:

Elemento neutro = 1: $\top(a,1) = a$.

Da mesma maneira, uma $t-conorma$ $\perp$ é uma função $\perp: [0,1]^2 \to [0,1]$ que é comutativa, associativa e monotônica e verifica a propriedade, para $\forall a \in [0,1]$:

Elemento neutro = 0: $\perp(a,0) = a$.

Uma t-norma $\top$ e uma t-conorma $\perp$ são duais em relação a uma operação de negação $\neg : [0,1] \to [0,1]$ se elas satisfazem as relações de De Morgan, dadas por, para $\forall a, b \in [0,1]$:

Leis de De Morgan:

$$\neg(\top(a,b)) = \bot(\neg a, \neg b);$$
$$\neg(\bot(a,b)) = \top(\neg a, \neg b).$$

O principal operador de negação é dado por $\neg a = 1 - a$, mas outros operadores podem ser encontrados na literatura [15].

É importante notar que as t-normas e t-conormas se reduzem aos operadores clássicos de união e intersecção quando os conjuntos são "crisp".

A Tabela 1 indica as t-normas e t-conormas mais utilizadas e as Figuras 3 e 4 ilustram alguns destes operadores, em relação a dois conjuntos nebulosos A e B. Estes operadores satisfazem as leis de De Morgan em relação ao operador de negação $\neg a = 1 - a$.

Tabela 1: Principais $\top$-normas e $\top$-conormas duais.

$t-norma$	$t-conorma$	nome
$min(a,b)$	$max(a,b)$	Zadeh
$a \,.\, b$	$a+b-ab$	probabilista
$max(a+b-1,0)$	$min(a+b,1)$	Lukasiewicz
$\begin{cases} a, & \text{se } b=1 \\ b, & \text{se } a=1 \\ 0, & \text{senão} \end{cases}$	$\begin{cases} a, & \text{se } b=0 \\ b, & \text{se } a=0 \\ 1, & \text{senão} \end{cases}$	Weber

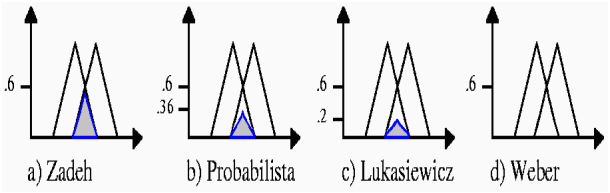

Figura 3: Ilustração das principais $t-normas$.

A maior (respec. a menor) t-norma é o min (respec. a t-norma de Weber); a maior (respec. a menor) t-conorma é a t-conorma de Weber (respec. o max). Os operadores mais usualmente utilizados são os operadores de Zadeh e os probabilistas.

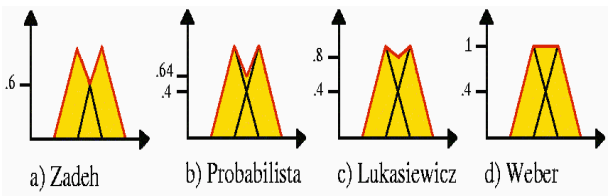

Figura 4: Ilustração das principais $t-conormas$.

2.2.1 Operadores de Implicação

Os operadores de implicação $I : [0,1]^2 \to [0,1]$ são usados para modelar regras de inferência do tipo **Se** <*premissa*> **Então** <*conclusão*>. Considerando A e B dados por $\mu_A : X \to [0,1]$, $\mu_B : Y \to [0,1]$, a relação $A \to B$ é expressa como:

$$\mu_{A\to B}(x,y) = I(\mu_A(x), \mu_B(y)) \qquad (8)$$

Existem três grandes classes de implicações propriamente ditas: As implicações S, que são da forma $I_S(a,b) = \bot(\neg a, b)$, as implicações R, que são tais que $I_R(a,b) = \sup\{c \in [0,1] \,/\, \top(a,c) \leq b\}$, e as implicações QM, que são da forma $I_{QM}(a,b) = \bot(\neg a, \top(a,b))$, onde $\top$ é uma t-norma, $\bot$ é uma t-conorma e $\neg$ é uma negação.

As t-normas não são implicações propriamente ditas, mas são muito empregadas na prática como implicações, notadamente em aplicações de controle nebuloso. Um estudo mais detalhado pode ser encontrado em [44] [47].

A Tabela 2 mostra os principais operadores de implicação e a Figura 5 ilustra o uso de alguns destes operadores.

Tabela 2: Principais operadores de implicação.

Implicação	*nome*
$max(1-a,b)$	Kleene−Diemes
$min(1-a+b,1)$	Lukasiewicz
$\begin{cases} 1, & \text{se } a \leq b \\ 0, & \text{senão} \end{cases}$	Rescher−Gaines *"Sharp"*
$\begin{cases} 1, & \text{se } a \leq b \\ b, & \text{senão} \end{cases}$	Brower−Gödel
$\begin{cases} min(b/a), & \text{se } a \neq b \\ 1, & \text{senão} \end{cases}$	Goguen
$1-a+ab$	Reichenbach *"Estocástica"*
$max(1-a, min(a,b))$	Zadeh−Wilmott
$min(a,b)$	Mamdani
$a \,.\, b$	Larsen

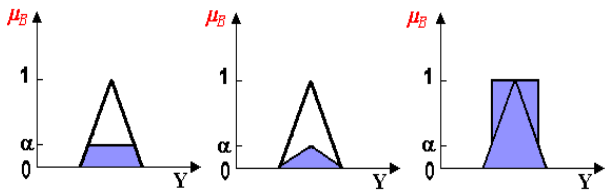

Figura 5: Ilustração das implicações $I(\alpha, \mu_B(y))$: *i*) Mamdani, *ii*) Larsen, *iii*) Göedel.

3. Controladores Nebulosos

As técnicas de *controle nebuloso* originaram-se com as pesquisas e projetos de E. H. Mamdani [48] [49] e ganharam espaço como área de estudo em diversas instituições de ensino, pesquisa e desenvolvimento do mundo, sendo até hoje uma importante aplicação da teoria dos conjuntos nebulosos.

Ao contrário dos controladores convencionais em que o algoritmo de controle é descrito analiticamente por equações algébricas ou diferenciais, através de um modelo matemático, em controle nebuloso utilizam-se de regras lógicas no algoritmo de controle, com a intenção de descrever numa rotina a experiência humana, intuição e heurística para controlar um processo [1].

Uma *variável linguística* [45] pode ser definida por uma quádrupla $(X, \Omega, T(X), M)$, onde X é o nome da variável, Ω é o universo de discurso de X, $T(X)$ é um conjunto de nomes para valores de X, e M é uma função que associa uma função de pertinência a cada elemento de $T(X)$. Chamamos aqui de *termos linguísticos*, indistintamente, tanto os elementos de $T(X)$ quanto suas funções de pertinência.

A Figura 6 ilustra a variável linguística velocidade com os termos nebulosos dados por $\{Negativa\ Alta, Negativa\ Baixa, Zero, Positiva\ Baixa, Positiva\ Alta\}$.

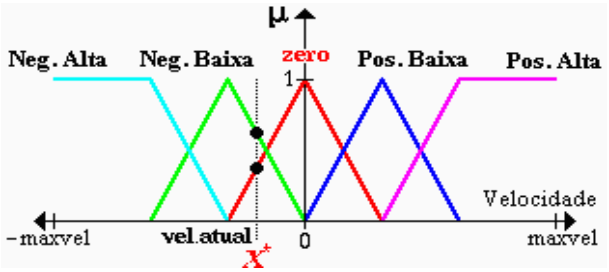

Figura 6: Termos linguísticos mapeiam a variável **Velocidade**.
Adaptada de Bauer (1998).

O grau com que um valor x^* em Ω satisfaz o termo linguístico A é a pertinência de x^* em A, dada por $\mu_A(x^*)$.

Os controladores nebulosos são robustos e de grande adaptabilidade, incorporando conhecimento que outros sistemas nem sempre conseguem acomodar [46]. Também são versáteis, principalmente quando o modelo físico é complexo e de difícil representação matemática.

Em geral, os controladores nebulosos encontram maior utilidade em sistemas não-lineares, sendo capazes de superar perturbações e plantas com níveis de ruídos. Além disso, mesmo em sistemas onde a incerteza se faz presente de maneira intrínseca, agregam uma robustez característica. No entanto, provar determinadas propriedades de robustez é uma tarefa difícil neste tipo de abordagem.

A Figura 7 é a representação da estrutura básica de um controlador nebuloso, como descrito em [4]. Muitas variações são propostas na literatura de acordo com o objetivo do projeto, mas esse é um modelo geral o suficiente para a identificação dos módulos que o compõem, fornecendo uma idéia do fluxo da informação.

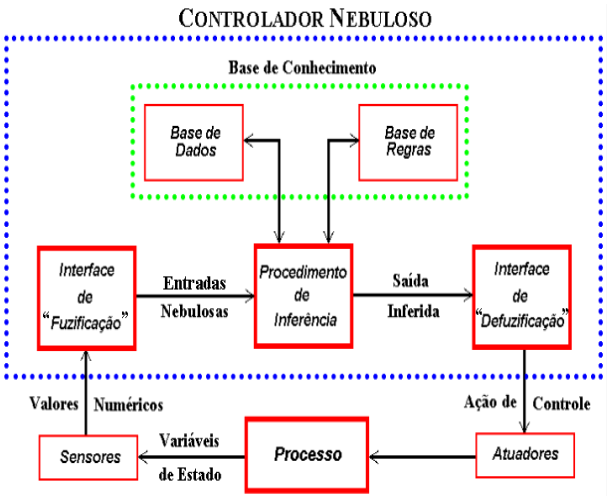

Figura 7: Estrutura de um Controlador Nebuloso.

3.1. Interface de "Fuzificação"

A *interface de "fuzificação"* faz a identificação dos valores das variáveis de entrada, as quais caracterizam o estado do sistema (variáveis de estado), e as normaliza em um universo de discurso padronizado. Estes valores são então "fuzificados", com a transformação da entrada "crisp" em conjuntos nebulosos para que possam se tornar instâncias de variáveis linguísticas.

3.2. Base de Conhecimento

A *base de conhecimento* consiste de uma base de dados e uma base de regras, de maneira a caracterizar a estratégia de controle e as suas metas.

Na *base de dados* ficam armazenadas as definições sobre discretização e normalização dos universos de discurso, e as definições das funções de pertinência dos termos nebulosos.

A *base de regras* é formada por estruturas do tipo

Se <*premissa*> **Então** <*conclusão*>,

como por exemplo:

Se *Erro é Negativo Grande* **e** Δ*Erro é Positivo Pequeno*
Então *Velocidade é Positiva Pequena*.

Estas regras, juntamente com os dados de entrada, são processados pelo *procedimento de inferência*, o qual infere as ações de controle de acordo com o estado do sistema, aplicando o operador de implicação, conforme o procedimento de inferência que será descrito na Seção 3.3.

Em um dado controlador nebuloso, é importante que existam tantas regras quantas forem necessárias para mapear totalmente as combinações dos termos das variáveis, isto é, que a base seja completa, garantindo que exista

sempre ao menos uma regra a ser disparada para qualquer entrada. Também são essenciais a consistência, onde procura-se evitar a possibilidade de contradições e a interação entre as regras, gerenciada pela função de implicação de modo a contornar as situações de ciclo.

As premissas são relacionadas pelos conectivos lógicos, dados pelo *operador de conjunção* (**e**) e o *operador de disjunção* (**ou**). Em geral as regras tem a forma de um sistema de múltiplas entradas e múltiplas saídas (MIMO), mas que pode ser transformado em vários sistemas com múltiplas entradas e uma saída (MISO). Por exemplo, a regra MIMO

$$\begin{array}{c}\textbf{Se } x_1 \textit{ é } A_1 \textbf{ e} \ldots \textbf{e } x_n \textit{ é } A_n \\ \textbf{Então } y_1 \textit{ é } C_1 \textbf{ e} \ldots \textbf{e } y_m \textit{ é } C_m\end{array}$$

é equivalente a m regras MISO:

$$\textbf{Se } x_1 \textit{ é } A_1 \textbf{ e} \ldots \textbf{e } x_n \textit{ é } A_n \textbf{ Então } y_j \textit{ é } C_j$$

Em geral não se aceitam conectivos "**ou**" na conclusão [44].

3.3. Procedimento de Inferência

Um controlador nebuloso é um sistema especialista simplificado onde a consequência de uma regra não é aplicada como antecedente de outra [3]. Assim, o processo de inferência consiste em:

1. Verificação do grau de compatibilidade entre os fatos e as cláusulas nas premissas das regras;
2. Determinação do grau de compatibilidade global da premissa de cada regra;
3. Determinação do valor da conclusão, em função do grau de compatibilidade da regra com os dados e a ação de controle constante na conclusão (precisa ou não);
4. Agregação dos valores obtidos como conclusão nas várias regras, obtendo-se uma ação de controle global.

Os tipos de controladores nebulosos encontrados na literatura são os modelos clássicos, compreendendo o modelo de Mamdani e o de Larsen, e os modelos de interpolação, compreendendo o modelo de Takagi-Sugeno e o de Tsukamoto (vide [3] [4] [50]).

Os modelos diferem quanto à forma de representação dos termos na premissa, quanto à representação das ações de controle e quanto aos operadores utilizados para implementação do controlador.

3.3.1 Controle nebuloso clássico

Nos modelos clássicos, a conclusão de cada regra especifica um termo nebuloso dentre um conjunto fixo de termos (geralmente em número menor que o número de regras). Estes termos são usualmente conjuntos nebulosos convexos como triângulos, funções em forma de sino ("bell-shaped") e trapézios.

Dado um conjunto de valores para as variáveis de estado, o sistema obtém um conjunto nebuloso (muitas vezes sub-normalizado), como o valor da variável de controle. Este conjunto nebuloso representa uma ordenação no conjunto de ações de controle aceitáveis naquele momento. Finalmente, uma ação de controle global é selecionada dentre aquelas aceitáveis em um processo conhecido como desfuzificação.

Sejam as regras R_j codificadas como:

$$R_j : \textbf{Se } x_1 \textit{ é } A_{1,j} \textbf{ e} \ldots \textbf{ e } x_n \textit{ é } A_{n,j} \textbf{ Então } y_j \textit{ é } C_j$$

No modelo clássico, o processamento de inferência é feito da seguinte maneira:

- ▷ **Passo 1**: Seja x_i uma variável de estado, definida no universo X_i, a realização de x_i é definida como o valor $x_i^* \in X_i$ que esta assume em X_i em um dado momento;
- ▷ **Passo 2**: A *compatibilidade* da $i-ésima$ premissa da $j-ésima$ regra com x_i^*, ou seja, a compatibilidade de x_i^*, $1 \leq i \leq n$, com $A_{i,j}$ da regra R_j, $1 \leq j \leq m$, é definida por:

$$\alpha_{i,j} = \mu_{A_{i,j}}(x_i^*), \quad 1 \leq i \leq n, \quad 1 \leq j \leq m \quad (9)$$

- ▷ **Passo 3**: Com as premissas de uma dada regra avaliadas, a *compatibilidade global* α_j da regra R_j, $1 \leq j \leq m$, com os x_i^* é determinada com uma t-norma $\top$:

$$\alpha_j = \top(\alpha_{1,j}, \ \ldots, \alpha_{n,j}), \quad 1 \leq j \leq m \quad (10)$$

- ▷ **Passo 4**: O α_j assim obtido é relacionado com o respectivo conjunto nebuloso C_j do consequente da regra R_j, dando origem a um conjunto C_j', $1 \leq j \leq m$, através de um operador de implicação I:

$$\mu_{C_j'}(y) = I(\alpha_j, \mu_{C_j}(y)), \quad \forall y \in Y \quad (11)$$

- ▷ **Passo 5**: Um operador ∇ faz a *agregação* das contribuições das várias regras acionadas C_j' num único conjunto nebuloso C':

$$\mu_{C'}(y) = \nabla(\mu_{C_1'}(y), \ \ldots, \mu_{C_m'}(y)), \ \ \forall y \in Y \quad (12)$$

O operador ∇ é usualmente uma t-conorma, quando o operador de implicação I é uma t-norma, e uma t-norma em caso contrário.

Modelos de Mamdani e Larsen Os modelos clássicos seguem estritamente os passos mostrados acima, sendo que no modelo de Mamdani temos $\top(a,b) = \min(a,b)$, $I = \min(a,b)$ e $\nabla(a,b) = \max(a,b)$ e no modelo de Larsen temos $\top(a,b) = a * b$, $I = a * b$ e $\nabla(a,b) = \max(a,b)$.

As Figuras 8 e 9 ilustram o processo de raciocínio do modelo de Mamdani e do modelo de Larsen, respectivamente.

Os controladores de Mamdani e Larsen necessitam da utilização de uma interface de desfuzificação para gerar a ação de controle, isto é, escolher um único valor no suporte de C', o que será visto na Seção 3.4.

3.3.2 Controle nebuloso por interpolação

Nos modelos de interpolação, cada conclusão é dada através de uma função estritamente monotônica, usualmente diferente para cada regra. No modelo de Takagi-Sugeno, a função é uma combinação linear das entradas, tendo como parâmetros um conjunto de constantes. No esquema de Tsukamoto, a função é geralmente não linear, tendo como domínio os possíveis graus de compatibilidade entre cada premissa e as entradas.

Em ambos os esquemas, obtém-se, para cada regra, um único valor para a variável de controle. Finalmente, uma ação de controle global é obtida fazendo-se uma média ponderada dos valores individuais obtidos, onde cada peso é o próprio grau de compatibilidade entre a premissa da regra e as entradas, normalizado.

Para os modelos de interpolação, também são válidos os passos 1, 2 e 3 descritos nos modelos clássicos. No entanto, a operação de implicação do passo 4 determina uma ação de controle precisa para cada regra. Essas ações individuais são interpoladas no passo 5, gerando uma ação de controle única e precisa.

Modelo de Tsukamoto O modelo de Tsukamoto [3] [44] exige que pelo menos os conjuntos nebulosos C_j, que estão associados com os consequentes das regras, sejam funções monotônicas. No passo 4, o método de interpolação obtém um valor preciso y'_j relativo à ação de controle da regra R_j, que é dado por:

$$y'_j = \mu_{C_j}(\alpha_j) \tag{13}$$

Por sua vez, no passo 5, os valores obtidos como conclusão nas várias regras são agregados em uma única ação de controle precisa y', através de uma média ponderada, dada por:

$$y' = \frac{\sum_{j=1}^{n} \left(\alpha_j \cdot y'_j\right)}{\sum_{j=1}^{n} \alpha_j} \tag{14}$$

Neste caso, a interface de "desfuzificação" não é utilizada. A Figura 10 representa uma interpretação gráfica do modelo de interpolação de Tsukamoto.

Modelo de Takagi-Sugeno O modelo de Takagi e Sugeno [3] [44] exige que todos os termos nebulosos $A_{i,j}$ na premissa sejam funções monotônicas e que as conclusões das regras sejam dadas por funções:

$$f_j(x_1, \ \ldots \ , \ x_m) = d_{0,j} + d_{1,j} \cdot x_1 + \ \ldots \ + d_{m,j} \cdot x_m \tag{15}$$

onde cada d_k é uma constante.

A ação de controle obtida por cada regra R_j é dada por:

$$y'_j = f_j(x^*_1, \ x^*_2, \ \ldots \ , \ x^*_m) \tag{16}$$

A ação de controle y' é então obtida pela Equação 14, como no modelo de Tsukamoto. Na Figura 11, ilustra-se a inferência através do método de Takagi e Sugeno de duas regras MISO.

É interessante notar que um controlador nebuloso de Takagi-Sugeno se comporta como um sistema do tipo PD quando existe somente uma única regra na base, dada por **Se** $x_1 = A_1$ **e** $x_2 = A_2$ **Então** $c = d_0 + d_1 \cdot x_1 + d_2 \cdot x_2$, onde:

- $x_1 = erro$, $x_2 = \Delta erro$;
- $d_0 = 0$;
- os termos A_i são tais que $\mu_{A_i}(x) = 1, \forall x \in X_i$;

Quando $x_1 = erro$, $x_2 = \Delta erro$ e $d_{0,j} = 0, 1 \leq j \leq m$, um controlador do tipo Takagi-Sugeno se comporta pois como se interpolássemos os resultados de um conjunto de controladores PD, cada um definido para uma região do espaço de estados.

3.4. Interface de "Desfuzificação"

Nos controladores nebulosos do tipo clássico, a interface de "desfuzificação" é utilizada para obter uma única ação de controle precisa, a partir do conjunto nebuloso C' obtido no passo 4.

O procedimento compreende a identificação do domínio das variáveis de saída num correspondente universo de discurso e com a ação de controle nebulosa inferida evolui-se uma ação de controle não nebulosa.

Os métodos de "desfuzificação" mais utilizados são:

1. Primeiro Máximo (SOM): Encontra o valor de saída através do ponto em que o grau de pertinência da distribuição da ação de controle atinge o primeiro valor máximo;

2. Método da Média dos Máximos (MOM): Encontra o ponto médio entre os valores que têm o maior grau de pertinência inferido pelas regras;

3. Método do Centro da Área (COA): O valor de saída é o centro de gravidade da função de distribuição de possibilidade da ação de controle.

A Figura 12 é um exemplo para o método do centro da área.

A seleção do método está relacionada diretamente com as características do processo controlado e o comportamento de controle necessário. O método do último máximo ou a média dos máximos, por exemplo, que assemelham o efeito de um controlador "bang-bang", podem conduzir a ações de controle inadequadas ao modo

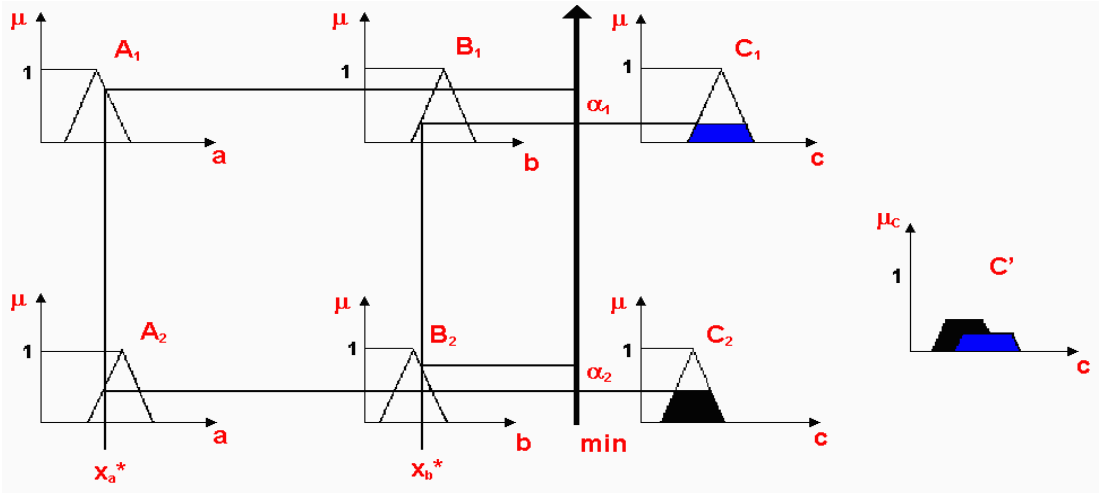

Figura 8: Modelo clássico de Mamdani.

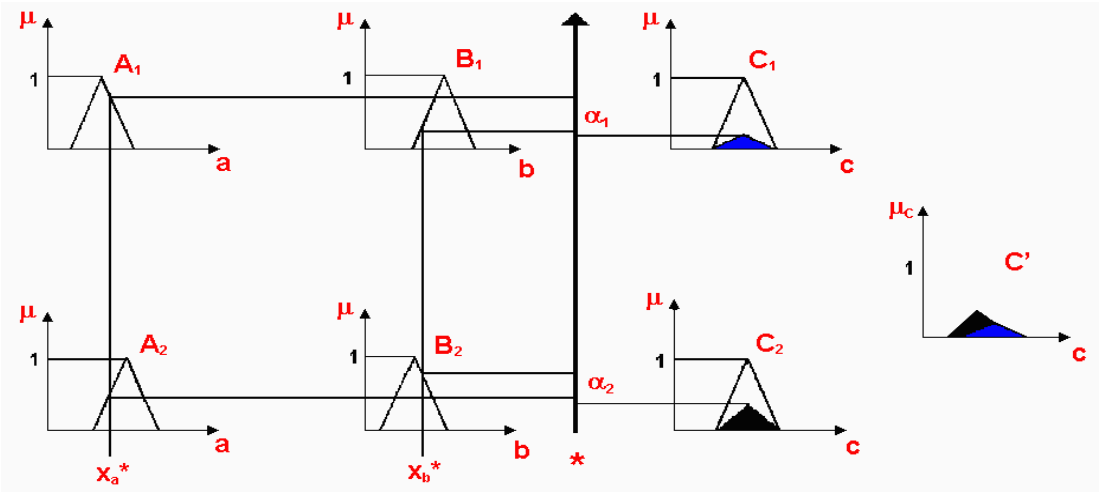

Figura 9: Modelo clássico de Larsen.

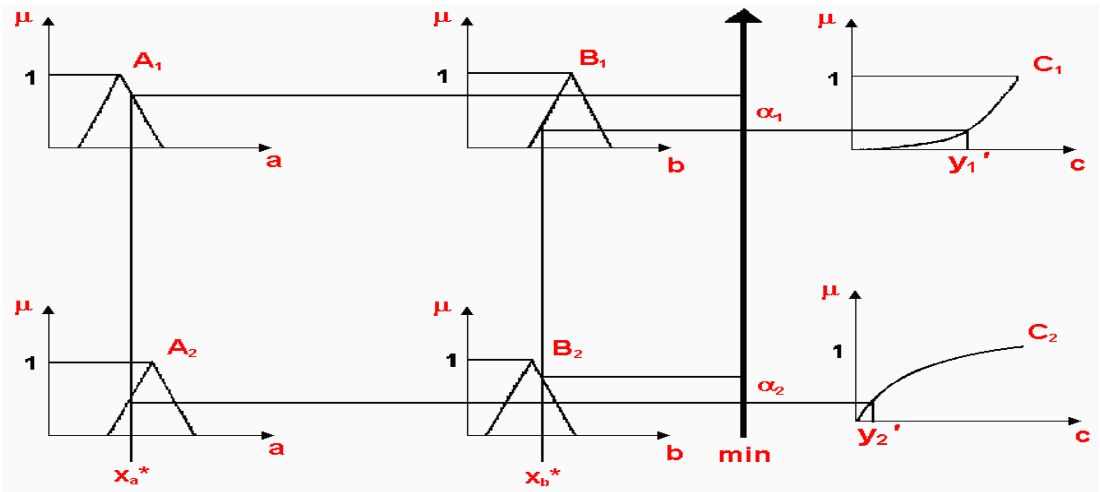

Figura 10: Modelo de interpolação de Tsukamoto.

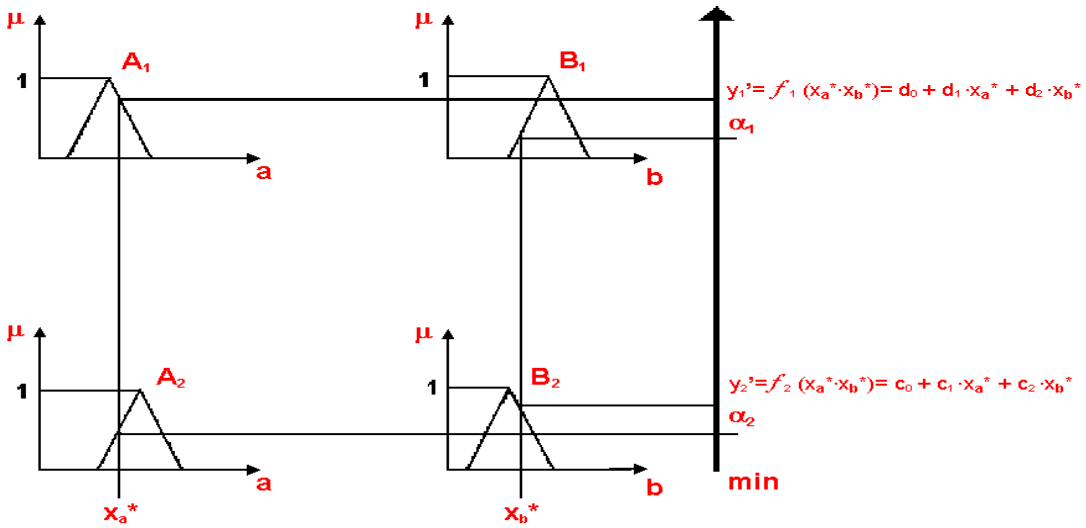

Figura 11: Método de interpolação de Takagi-Sugeno.

de operação (p.ex. produzindo solavancos) e, assim, causar danos de ordem prática em equipamentos como os atuadores.

Existem ainda outros métodos de "desfuzificação", apresentando diferenças em termos de velocidade e eficiência, características que devem ser analisadas em conjunto com os requisitos do projeto [44] [51] [46] [3].

É importante lembrar que os controladores nebulosos do tipo interpolação não necessitam de uma interface de "desfuzificação", pois já obtém-se diretamente os valores precisos para as entradas do processo controlado [44] [51] [3].

3.5. Projeto de Controladores Nebulosos

De uma maneira geral, pode-se descrever as tarefas de construção de um controlador nebuloso brevemente como [46]:

1. Definição do modelo e das características operacionais, para estabelecer as particularidades da arquitetura do sistema (como sensores e atuadores) e definição das propriedades operacionais do controlador nebuloso do projeto, como o tipo de controlador, operadores a serem utilizados, desfuzificador, etc;

2. Definição dos termos nebulosos de cada variável. Para garantir suavidade e estabilidade deve-se permitir que haja uma sobreposição parcial entre conjuntos nebulosos vizinhos;

3. Definição do comportamento do controle, que envolve a descrição das regras que atrelam as variáveis de entrada às propriedades de saída do modelo.

No projeto de controladores nebulosos é necessária portanto a definição de alguns parâmetros, obtidos a partir da experiência do projetista ou através de experimentos. Tendo em vista um determinado processo, alguns destes parâmetros são fixos, os denominados parâmetros estruturais, e outros, os parâmetros de sintonização, são aqueles que variam com o tempo:

1. Parâmetros estruturais:

   - Número de variáveis de entrada e saída;
   - Variáveis linguísticas;
   - Funções de pertinência parametrizadas;
   - Intervalos de discretização e normalização;
   - Estrutura da base de regras;
   - Conjunto básico de regras;
   - Recursos de operação sobre os dados de entrada.

2. Parâmetros de sintonização:

   - Universo de discurso das variáveis;
   - Parâmetros das funções de pertinência (p.ex. núcleo e suporte);

Propriedades da base de regras como a completude, consistência, interação e robustez precisam ser testadas. A robustez relaciona-se com a sensibilidade do controle frente a ruídos ou algum comportamento incomum não-modelado. Para ser medida, introduz-se um ruído aleatório de média e variância conhecidas e observa-se então a alteração dos valores das variáveis de saída.

A sintonização é uma tarefa complexa devido à flexibilidade que decorre da existência de muitos parâmetros, exigindo esforço do projetista na obtenção do melhor desempenho do controlador. Alguns dos parâmetros podem ser alterados por mecanismos automáticos de adaptação e aprendizado, contudo, normalmente é tarefa do projetista o treinamento e a sintonia da maioria dos parâmetros.

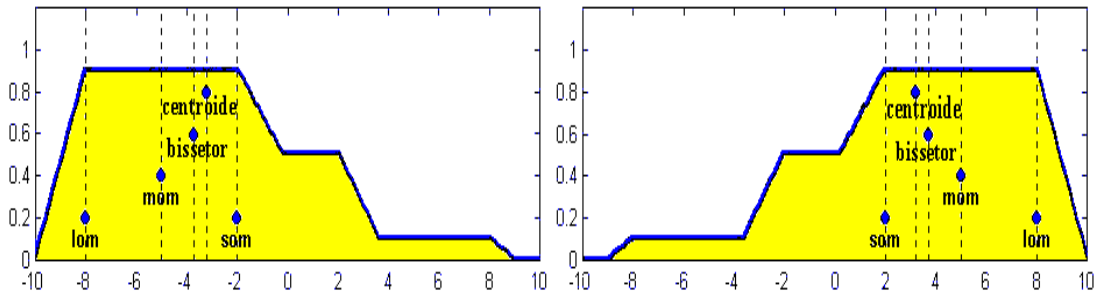

Figura 12: Desfuzificação pelo Método do Centro de Gravidade. Adaptada de Bauer (1998).

Esta sintonização é feita através de busca, uma atividade típica em Inteligência Artificial.

A sintonização pode ser feita da seguinte maneira:

1. Desenvolve-se um controlador simples, que simule um controlador proporcional com:
   - Conjunto de variáveis mais relevantes;
   - Baixo número de variáveis linguísticas;
2. Incrementa-se o conhecimento conforme a experiência resultante do processo:
   - Buscando-se novas variáveis linguísticas ou físicas para contornar as dificuldades;
   - Ajustando-se as funções de pertinência e os parâmetros do controlador;
   - Adicionando-se regras ou modificando a estrutura de controle.
3. Valida-se a coerência do conhecimento incorporado com novas condições de operação para o sistema.

Essas tarefas necessitam de plataformas sofisticadas, com interface poderosa e que permitam uma rápida inferência. Isto é proporcionado pelos pacotes integrados, dedicados a análise de modelos nebulosos [47].

4. Aprendizado Usando Redes Neurais

Nos últimos anos, tem havido um crescente interesse sobre os chamados sistemas "neuro-fuzzy" [17] [18] [19] [20] [42]. A rigor, qualquer sistema que misture os paradigmas de sistemas nebulosos e sistemas conexionistas poderia ser chamado de "neuro-fuzzy", como, por exemplo, a utilização de um controlador nebuloso para alterar dinamicamente a taxa de aprendizado de uma rede neural. No entanto, o termo é utilizado para um tipo específico de sistema que de certa forma engloba os dois paradigmas. Nestes sistemas, os termos e regras de um sistema nebuloso são aprendidos mediante a apresentação de pares (entrada, saída desejada). Eles apresentam dois comportamentos distintos, dependendo de estar numa fase de aprendizado ou numa fase de processamento da informação: na fase de aprendizado, eles têm um comportamento de redes neurais, e na fase de processamento, eles se comportam como um sistema nebuloso.

Estes sistemas são capazes de solucionar problemas apresentados pelos paradigmas em que se baseiam. Por exemplo, duas questões em aberto a respeito de redes neurais são:

- Como determinar quantas camadas e quantos nós devem ser usados em uma rede neural para que ela resolva efetivamente um dado problema, de maneira eficiente ?
- Que tipo de conhecimento pode ser extraído de uma rede treinada e como pode ser feita esta extração ? Por exemplo, se tomarmos um controlador neural para um pêndulo sobre um carrinho (problema do pêndulo invertido), é possivel extrair conhecimento do tipo "se o pêndulo esta caindo para a direita, mova o carrinho para a direita" ?

Em relação a sistemas nebulosos, uma problema importante e aparentemente insolúvel é:

- Como construir um sistema nebuloso, se ao invés de um especialista capaz de fornecer regras, temos somente um conjunto de exemplos do funcionamento ideal do sistema ?

Como um sistema nebuloso, os sistemas "neuro-fuzzy" podem "explicar" o seu comportamento, através de regras que modelam o conhecimento do sistema. Como uma rede neural (do tipo supervisionada), o aprendizado do sistema é feito a partir da apresentação de exemplos. Dito de outra maneira, os sistemas "neuro-fuzzy" não apresentam as desvantagens dos paradigmas em que se baseiam; ao contrário das redes neurais, os sistemas neuro-fuzzy não são uma "caixa preta", e, ao contrário dos sistemas nebulosos, eles prescindem da necessidade de um especialista que lhes forneça as regras.

Na próxima subseção, apresentamos um modelo abstrato que serve para modelar tanto redes neurais, quanto controladores nebulosos e sistemas "neuro-fuzzy". Em seguida, apresentamos brevemente um dos sistemas "neuro-fuzzy" mais conhecidos na literatura.

4.1. Modelo abstrato

Para ilustrar a ligação entre redes neurais e sistemas nebulosos, vamos utilizar um modelo abstrato que envolve 2 tipos de nós:

1. Nós que computam funções, como por exemplo a "desfuzificação", nos sistemas nebulosos, e a função de ativação, nas redes neurais. Nas figuras, eles serão representados por quadrados e retângulos.

2. Nós que buscam dados, como por exemplo os termos nebulosos, nos sistemas nebulosos, e os pesos, nas redes neurais. Nas figuras, eles serão representados por círculos e elipses.

Por exemplo, cada tipo de controlador nebuloso visto anteriormente utiliza uma instância particular deste modelo. As Figuras 13, 14 e 15 trazem respectivamente a abstração dos controladores nebulosos de Mamdani, Sugeno e Tsukamoto respectivamente. Nestas figuras, os nós de busca de dados são ilustrados pelos termos A, B e C, que representam conjuntos nebulosos (vistos como dados), e por C^{-1}, que representa uma função tendo graus de pertinência em $[0, 1]$ como domínio. Os nós de funções são ilustrados pela função *I* (identidade), $\top$ (uma t-norma qualquer), $\perp$ (uma t-conorma qualquer), D (um método de desfuzificação qualquer), Σ (média ponderada) e pela função $apply$ que obtém o grau de pertinência de um valor em um dado domínio em relação a um conjunto nebuloso definido neste mesmo domínio.

A Figura 16 ilustra uma rede neural, utilizando as convenções do modelo abstrato. Os nós de busca de dados rotulados como w representam pesos. Os nós de função são rotulados como I (identidade), $*$ (produto), Σ (média ponderada). O nó da função de ativação está rotulada por uma ilustração da função sigmóide. Os quadrados pontilhados representam os nós da rede neural, tal como são usualmente conhecidos.

Podemos ver nesta figura que o modelo abstrato representa uma rede neural já treinada, independentemente de fatores tais como a relação deste modelo abstrato com o modelo biológico e o método de treinamento das redes. Os pesos podem ter sido obtidos por algoritmos de aprendizado (p.ex.: "backpropagation"), ou por outros meios (p.ex.: algoritmos genéticos).

Podemos notar que neste modelo abstrato, o conceito de camadas é mais geral do que aquele utilizado em redes neurais.

4.2. Modelo de Lin e Lee

O modelo de Lin e Lee [18] tem como principal aplicação a criação de controladores nebulosos do tipo clássico (p.ex. controladores de Larsen ou Mamdani).

Os dados de entrada para este sistema são o número de termos nebulosos para cada variável (tanto de entrada quanto de saída). Assume-se que os termos nebulosos são modelados por funções em formato de sino, com parâmetros m (centro do termo) e σ^2 (largura aproximada do termo ou "variância").

A rede aprende os parâmetros para cada um dos termos nebulosos e também a estrutura da rede, i.e., a que conclusão ligar cada uma das premissas.

A criação do sistema pode ser dividida em 4 fases:

1. Aprendizado dos termos nebulosos iniciais: Na primeira fase, o conjunto de treinamento, formado por pares ordenados (vetor_de_entrada, vetor_de_saída_desejada), são utilizados para se criar os termos nebulosos iniciais. Cada termo A_i é representado por um par (m_i, σ_i^2). Por exemplo, um termo pode ser definido como [1]

$$\mu_{A_i}(\omega) = e^{-\frac{\omega - m_i}{2{\sigma_i}^2}}$$

Para cada variável linguística, os exemplos (elementos dos vetores presentes no conjunto de treinamento) sao"clusterizados", no espírito da rede de Kohonen, no número de termos que a variável pode assumir.

2. Criação de uma estrutura inicial da rede: Na segunda fase, a estrutura inicial da rede é determinada com um conjunto de 5 camadas de nós de função, como pode ser visto na Figura 17. A camada 1 contém nós que distribuem a entrada (função identidade) e é chamada de *nós de entrada linguística*. A camada 2 contém nós que computam o valor de pertinência da entrada em relação a um termo nebuloso (função "apply") e é chamada de camada de *nós de termos de entrada*. A camada 3 contém nós que computam a compatibilidade de uma premissa em relação às entradas (t-norma $\top$) e é chamada de camada de *nós de regras*. A camada 4 contém nós que implementam uma dada t-conorma $\perp$. Nesta fase, para cada nó da camada 3, são criadas conexões para todos os nós da camada 4 (na fase seguinte, algumas conexões são eliminadas). Isto é equivalente a se ligar cada premissa com todas as conclusões possíveis (termos) da cada variável de saída. A camada 5 contém nós que implementam a "desfuzzificação", tendo como entrada pares formados pelo valor computado por cada nó da camada 3 com um termo nebuloso de saída associado. A Figura 17 ilustra a estrutura inicial em um sistema com 2 variáveis de entrada e 2 variáveis de saída, com 3 termos para cada variável (p.ex. o conjunto {N,Z,P}).

3. Aprendizado das regras:

- 3.a) Treinamento para obtenção das regras.

[1] No artigo original, esta formula é dada como $\mu_{A_i}(\omega) = e^{-\frac{\omega - m_i}{{\sigma_i}^2}}$, criando uma superposição muito acentuada dos conjuntos nebulosos.

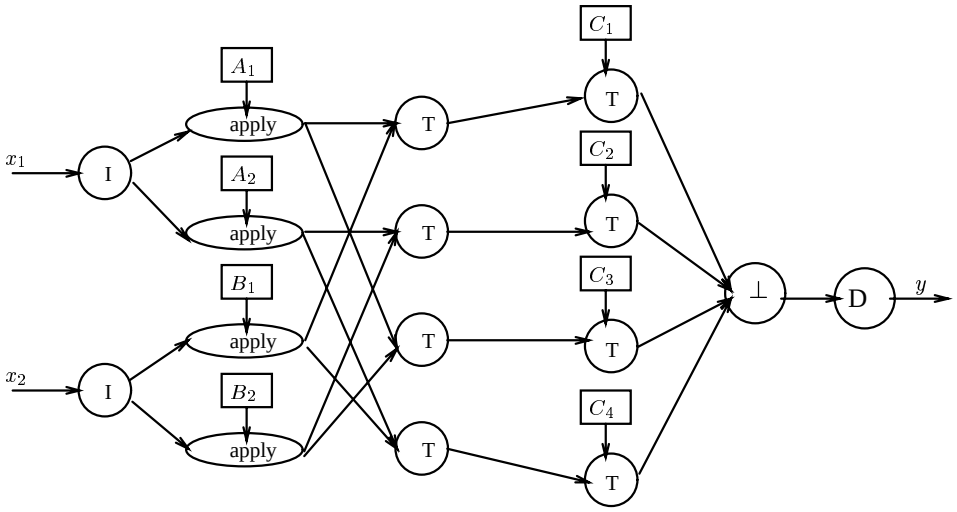

Figura 13: Arquitetura de controlador do tipo Mamdani.

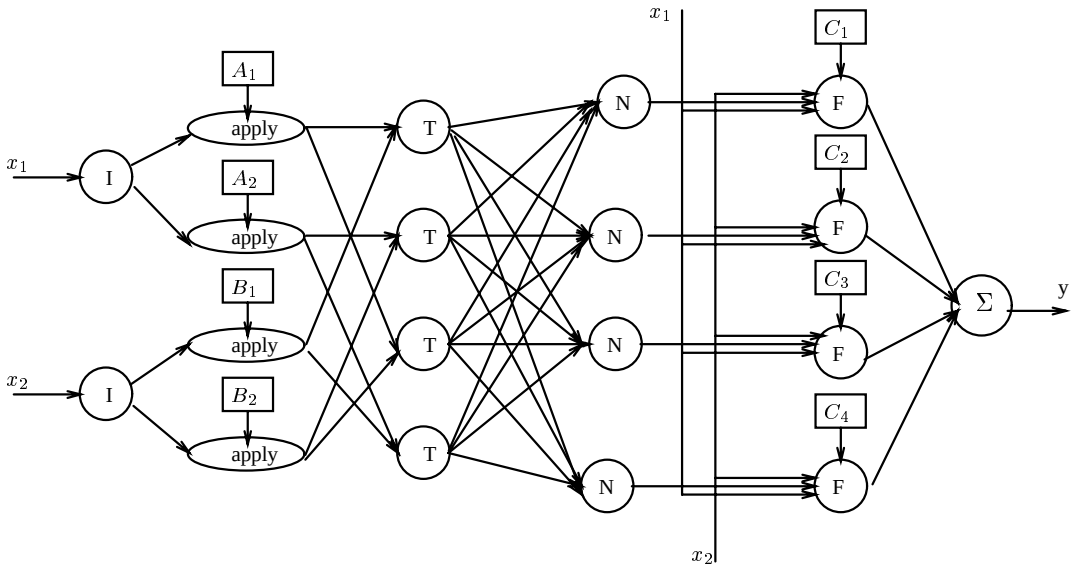

Figura 14: Arquitetura de controlador do tipo Sugeno.

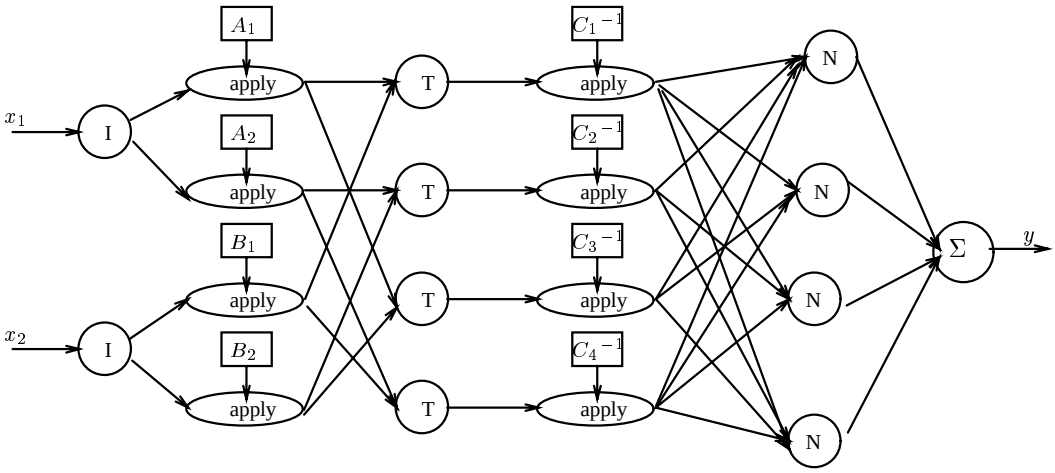

Figura 15: Arquitetura de controlador do tipo Tsukamoto.

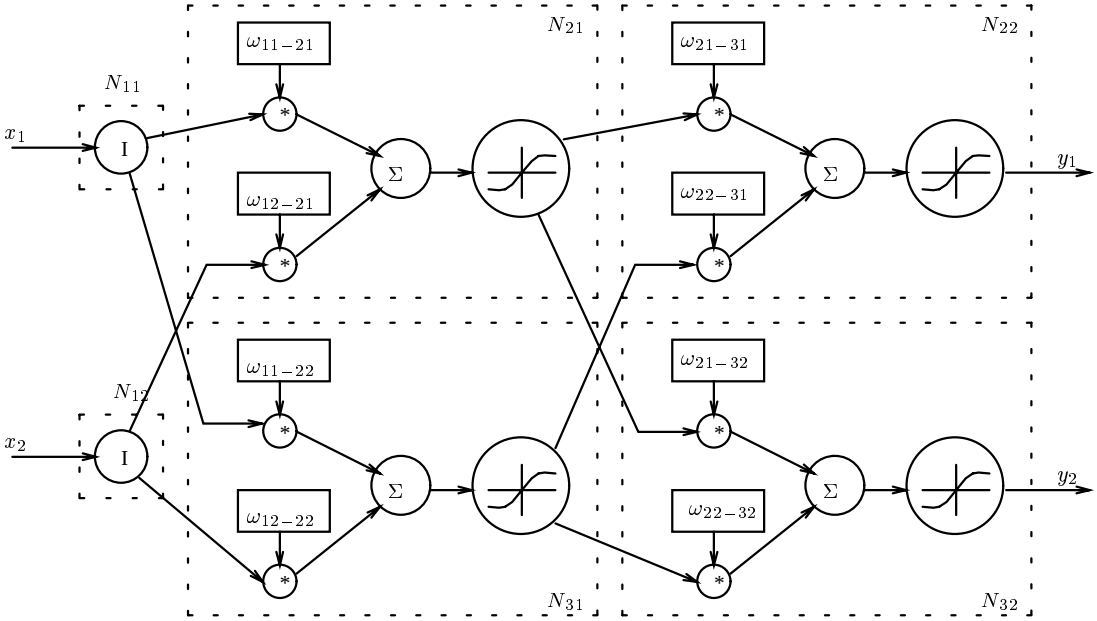

Figura 16: Rede neural.

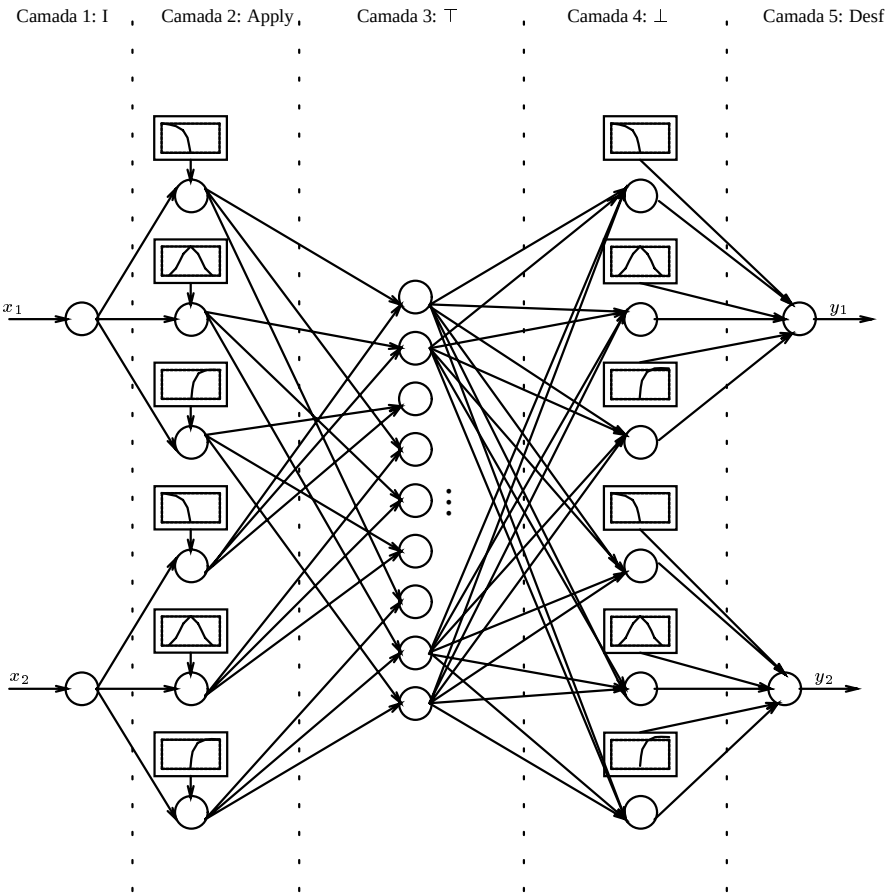

Figura 17: Fase de aprendizado das regras no sistema de Lin e Lee.

Na primeira parte da terceira fase, treina-se a rede no espírito competitivo e são determinados pesos para as conexões que saem de uma dada premissa para todas as conclusões possíveis.

- 3.b) Compilação das regras (determinação da estrutura final).

  Na segunda parte desta fase, para cada variável de saida, a conexão com maior peso que sai de uma premissa torna-se a conclusão da regra, e as demais são eliminadas. A Figura 18 ilustra a compilação de um sistema, equivalente ao uso da base de regras ilustrada na Tabela 3 para ambas as variáveis de saída.

Tabela 3: Controlador nebuloso.

	N	Z	P
N	N	N	Z
Z	N	Z	P
P	Z	P	P

4. Otimização de termos nebulosos: Na última fase, um algoritmo de "backpropagation" é utilizado para ajustar o centro e a largura de cada termo.

5. Aprendizado Usando Algoritmos Genéticos

Os Algoritmos Genéticos (AG), sobre os quais [52] é uma das principais referências, constituem um método de otimização baseado na analogia entre otimização e a evolução natural das espécies, combinando os conceitos de adaptação seletiva e sobrevivência dos indivíduos mais capazes [40]. Trata-se de um dos paradigmas de uma nova linha de pesquisa alternativa em Ciência da Computação, conhecida como Computação Evolutiva (CE), também considerada uma técnica emergente de Inteligência Artificial (IA) por suas características particulares.

Esses algoritmos usam uma base conceitual comum na evolução de populações de estruturas individuais, que representam soluções possíveis para um dado problema, através da aplicação operações de seleção, mutação e cruzamento nos indivíduos de uma dada população.

O indivíduo na população recebe uma medida conforme sua adequação ao ambiente e maior facilidade de reprodução é dada aos indivíduos mais aptos. Além disso, um processo de mutação introduz uma leve perturbação em alguns dos indivíduos de uma geração, o que tende a levar a um aprimoramento das gerações subsequentes [53].

O modelo genético computacional pode ser implementado através de vetores de bits ou caracteres que representam os cromossomas e as operações genéticas são implementadas por operadores simples de manipulação de bits. Em geral, os algoritmos genéticos utilizam cadeias de caracteres ("strings") de tamanho fixo, ao contrário do que ocorre na área de Programação Genética, que não fixa a representação e tipicamente não há a codificação dos problemas [54] [53].

5.1. Estágios de um Algoritmo Genético

Descreve-se em seguida um modelo básico de algoritmo genético. Maiores detalhes podem ser encontrados em [54] [34] [53] [40].

Um ciclo de um algoritmo genético consiste dos estágios de criação de uma população de soluções potenciais codificadas (indivíduos), avaliação dos indivíduos, seleção dos indivíduos mais capazes e geração de uma nova população através de manipulações genéticas, como cruzamento e mutação.

Em geral, um AG modela uma solução possível para o problema na forma de um vetor com um número fixo de posições (cromossoma), onde cada posição assume um valor em $\{0,1\}$.

Inicialmente, o algoritmo genético utiliza uma população inicial de cromossomas, oriunda de um processo aleatório. Uma função, a denominada função objetivo ("fitness"), verifica o quanto a solução representada por cada um dos cromossomas da população em análise se aproxima daquela ideal, medindo os efeitos que ela causa ao ser aplicada como solução do problema. Uma nova população é então gerada, através dos operadores genéticos. Normalmente, substitui-se a população progenitora pela gerada, o que mantém o tamanho da população fixo.

Os principais operadores são a *seleção*, que escolhe alguns cromossomas como aptos a passarem suas informações para a próxima geração, o *cruzamento* que toma dois cromossomas selecionados e casando-os aleatoriamente gera filhos que tem características de ambos os pais, e a *mutação*, que introduz uma pequena modificação em um cromossoma selecionado. As chances de um cromossoma ser selecionado são maiores quanto melhor é seu desempenho em relação a uma função objetivo. Usando-se esse conceito de adequação, as operações genéticas são executadas no sentido do aprimoramento da população.

Repete-se o ciclo usando a nova população e cada nova iteração neste ciclo dá origem a uma geração. Essa evolução ocorre até que alguma condição de parada seja satisfeita, como por exemplo quando um número máximo de gerações ou tempo limite é atingido, ou quando é encontrada uma solução estável.

A Figura 19 ilustra o esquema do ciclo típico de um algoritmo genético com a sequência das operações realizadas sobre a composição de uma geração, como proposto por [55].

No esquema ilustrado na figura, o mecanismo de seleção é implementado pelo método da roleta, no qual cada indivíduo corresponde a um setor circular de ângulo $2\pi \cdot (f_i/\overline{f})$, onde f_i e $\overline{f}$ representam a aptidão do in-

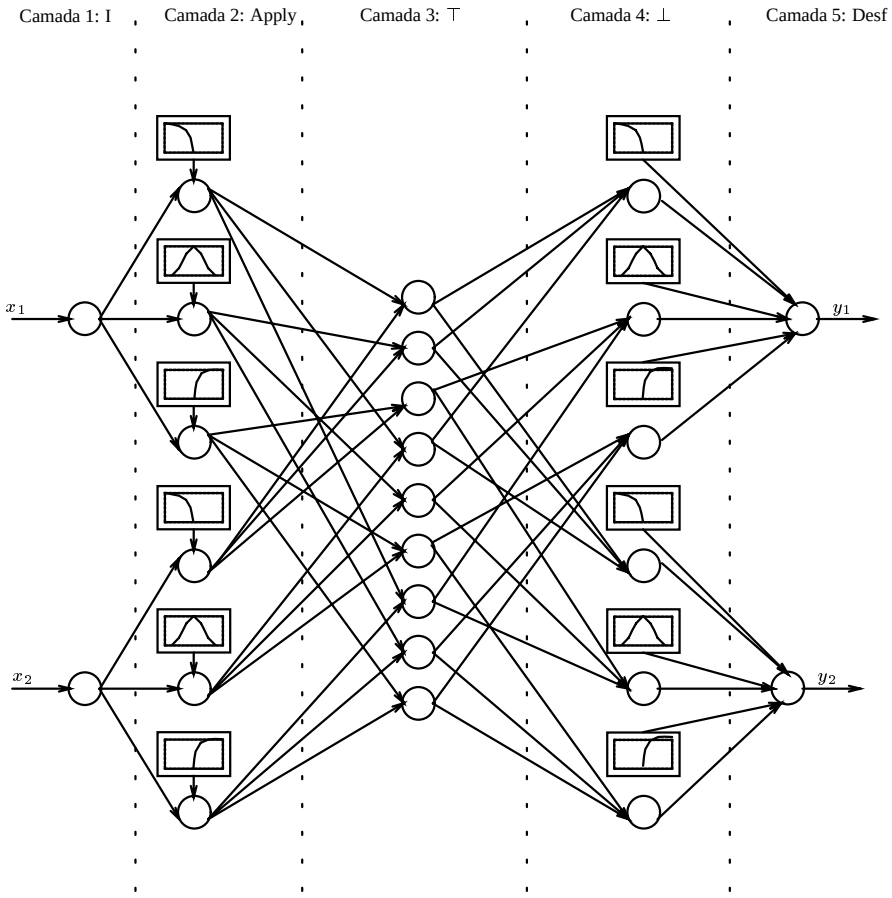

Figura 18: Fase de compilação no sistema de Lin e Lee.

divíduo e a aptidão média da população, respectivamente. Um número aleatório entre 0 e 2π determina qual o setor e o indivíduo correspondente, que pela própria natureza da roleta, tem a tendência de manter o carácter proporcional nessa determinação.

No esquema ilustrado acima, o cruzamento depende de uma probabilidade p_c e a decisão para executá-la é tomada usando-se um gerador de números aleatórios no intervalo $[0, 1]$. Somente se o número gerado for superior a p_c o cruzamento será realizado. Para realizar o cruzamento, outro número aleatório k é utilizado para determinar a posição de corte na cadeia de bits (cromossoma) de tamanho l, com $k \in [1, l-1]$. Então, o material genético à direita desse ponto é permutado entre os cromossomas selecionados.

No esquema acima, a mutação altera bits numa cadeia de caracteres com a probabilidade p_m. Esta operação introduz novas informações, conferindo ao processo de busca da solução uma característica que o cruzamento por si só não consegue. Isto permite vasculhar novos pontos no espaço de busca, aumentando a probabilidade de se encontrar o ótimo global. Esse processo é realizado bit a bit. Para cada bit considerado (bit da posição k), gera-se um número aleatório no intervalo $[0, 1]$. A operação é executada se esse número é superior à probabilidade p_m, e então, o valor do bit é alterado.

5.2. Controladores Nebulosos Projetados por um AG

Existe atualmente um grande interesse no aprendizado de parâmetros de sistemas nebulosos com o uso de algoritmos genéticos, principalmente, de controladores nebulosos. A seguir descrevemos um procedimento geral para o aprendizado de um controlador do tipo clássico, implementado em [37], baseado em requerimentos usualmente adotados na construção de controladores nebulosos [3] [56] [57] [34].

Nesta implementação, as estruturas iniciais dos controladores adotam uma base de regras com um número fixo de regras e de termos nebulosos, construída para cada caso.

A tarefa do algoritmo genético consistiu então em se encontrar os centros dos conjuntos nebulosos das premissas e conclusões de cada regra.

Para conferir maior suavidade à superfície que representa as ações de controle, foi utilizada a sobreposição dos conjuntos nebulosos. O ponto de centro de um conjunto corresponde respectivamente ao ponto final da base do conjunto anterior e ao ponto inicial da base do próximo conjunto (Figura 20).

Todos os conjuntos nebulosos centrais são de forma triangular e nas extremidades utilizam-se conjuntos trapezoidais. Além disso, adotou-se a simetria em relação

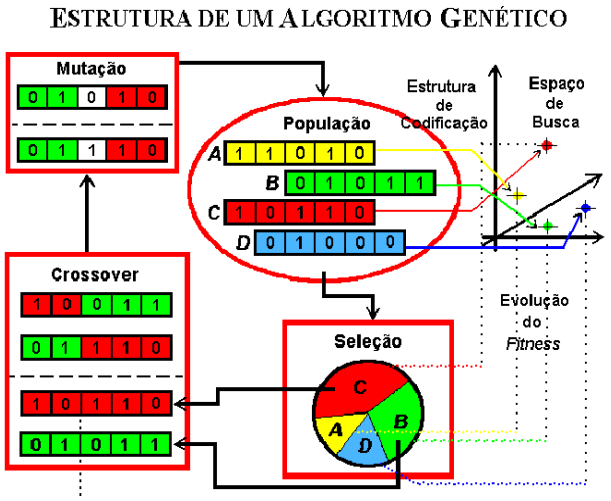

Figura 19: Estrutura básica de um Algoritmo Genético. Adaptada de Hoffmann (1997).

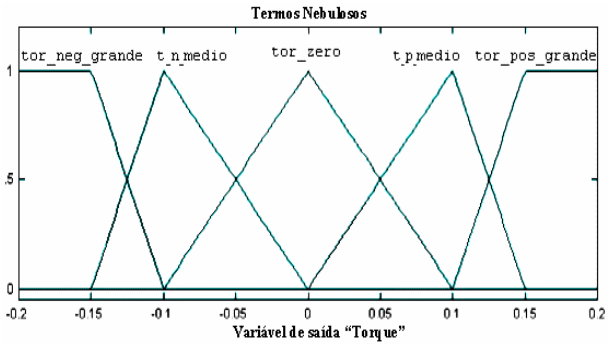

Figura 20: Conjuntos nebulosos para o mapeamento das conclusões das regras de um controlador nebuloso (ilustração).

ao valor zero de uma variável ($erro = 0$). Desta forma, os conjuntos nebulosos são projetados para os valores positivos da variável e então *espelhados* para os valores negativos correspondentes.

Nestes experimentos os cromossomas do AG codificam os valores relacionados aos possíveis centros dos conjuntos nebulosos, mapeados no domínio correspondente conforme as especificações de projeto.

Os parâmetros obtidos na aprendizagem com o AG são tratados como porcentagens do intervalo de busca, tomando-se como referência o centro definido em zero. Assim o primeiro conjunto terá como centro (núcleo) a própria porcentagem encontrada, o segundo será o centro anterior mais a porcentagem obtida com o AG e assim sucessivamente.

Por exemplo, na Figura 20, o intervalo de busca é dado por $[0, .2]$, sendo 0 o núcleo do conjunto nebuloso central tor_zero. O sistema aprendeu a porcentagem de 50% tanto para o núcleo do primeiro termo quanto para o início do núcleo do segundo termo, resultando no valor de 0.1 para t_p_medio, e de 0.15 para tor_pos_grande. Os núcleos de t_n_medio, e e tor_neg_grande foram espelhados a partir de t_p_medio e tor_pos_grande.

Este esquema foi utilizado no aprendizado de um controlador nebuloso para o controle de atitude de um satélite artificial a roda de reação [37], apresentando resultados superiores àqueles relativos ao controlador PD existente para o mesmo satélite.

5.3. Outros tipos de AG

O AG mostrado aqui utiliza operadores genéticos que operam sobre cromossomas nos quais cada parâmetro é representado por um número fixo de bits, codificado segundo especificações do utilizador. Essa representação tem desvantagens quando aplicada em cromossomas multidimensionais e problemas numéricos de alta precisão, pois o tamanho necessário ao cromossoma para a representação dos vários parâmetros com suas respectivas precisões pode ser impraticável devido ao tempo de processamento associado. Eventualmente, pode também ocorrer situações de convergência prematura para um ponto de ótimo não global ou a inabilidade para a sintonização local fina dos parâmetros. Isso porque, dependendo da representação adotada, dois pontos próximos no espaço de representação do AG não necessariamente estão próximos no espaço do problema [2].

Uma forma alternativa de evoluir o processo de busca é realizar as operações em cromossomas cujos genes estão codificados diretamente em ponto flutuante [58]. Desta maneira, as operações de conversão de base são efetuadas diretamente pelo processador. Além disso, garante-se que dois pontos próximos um do outro no espaço de representação do AG também estão próximos no espaço do problema (o inverso também é válido). Essa abordagem é recomendada para problemas de otimização de parâmetros com variação no domínio do contínuo e representa portanto, uma abordagem interessante para desenvolvimentos futuros no contexto deste trabalho.

6. Conclusões

Neste trabalho, apresentamos alguns fundamentos sobre lógica nebulosa. Apresentamos também o uso deste paradigma na construção de sistemas baseados em conhecimento para controle de processos, os chamados controladores nebulosos. Finalmente, apresentamos brevemente o aprendizado dos parâmetros envolvidos nestes sistemas através de redes neurais, os chamados sistemas "neuro-fuzzy", e de algoritmos genéticos.

Para a criação destes sistemas, existem ferramentas disponíveis no pacote integrado MATLAB, como por exemplo o Toolbox Control Systems [59] [60] [61] [62] [63] [64] [65], Toolbox de Fuzzy, também projetado pela MathWorks e um algoritmo genético desenvolvido por [66], utilizando como referência o modelo e a estrutura descrita em [54].

[2] Isto pode eventualmente ser atenuado usando-se a codificação de Gray [58].

Existem outras diversas ferramentas computacionais disponíveis comercialmente e muitas disponibilizadas livremente por instituições educacionais. Alguns exemplos de sisitemas nebulosos são o FuzzyTECH (“Fuzzy Logic Development System by Inform”), FLINT (”Fuzzy Logic Toolkit by Logic Programming Associates”), ANFIS (“Original C code by Roger Jang”) e Xfuzzy (“A Design Environment for Fuzzy Logic Control Systems”). Para simulações de órbita pode-se citar o STK (“Satellite Tool Kit”) e referências para AGs são o GAGS (“Genetic Algorithm C++ class library”), IlliGAL (“Illinois Genetic Algorithms Laboratory”), GAGA (General Architecture for Genetic Algorithms), GAOT (“GA Optimization Toolbox for Matlab”) e o GALOPPS (“Genetic ALgorithm Optimized for Portability and Parallelism System by Michigan State University”), entre muitos outros. Estas ferramentas podem ser em alguns casos utilizadas em conjunto e constituem-se em alternativas para o desenvolvimento de projetos e pesquisas. Uma ferramenta para aprendizado usando algoritmos genéticos pode também ser encontrada em [37].

Referências

[1] L. A. Zadeh. Fuzzy sets. *Fuzzy Sets, Information and Control*, 8:338 – 353., 1965.

[2] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

[3] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An Introduction to Fuzzy Control*. Springer-Verlag, 1993.

[4] C. C. Lee. Fuzzy logic in control systems: Fuzzy logic controller (part i). *IEEE Transactions on Systems, Man an Cybernetics.*, 20(2):404 – 418., Março/Abril 1990.

[5] *Fuzzy Logic Systems for Engineering: A Tutorial*, volume 83. Proceedings of the IEEE, Março, 1995.

[6] J. L. Castro. Fuzzy logic controllers are universal approximators. *IEEE Transactions on Systems, Man and Cybernetics,*, 25(4):629 – 635., Abril 1995.

[7] S. Dutta. Fuzzy logic applications: Technological an strategic issues. *IEEE Transactions on Engineering Management*, 40(3):237 – 254., Agosto 1993.

[8] *Fuzzy Open-Loop Attitude Control for the FAST Spacecraft*, San Diego - CA, Julho 1996. Proc. of the NASA AIAA, Guidance, Navigation and Control Conference.

[9] P. of the NASA AIAA, editor. *Automated Maneuver Planning Using a Fuzzy Logic Controller*, San Diego - CA, Julho 1996. Guidance: Navigation and Control Conference.

[10] R. Guerra, S. A. Sandri, and M. L. O. S. (1997a). Dynamics and design of autonomous attitude control of a satellite using fuzzy logic. *Anais do COBEM'97*, (COB 1338), Dezembro 1997.

[11] R. Guerra, S. A. Sandri, and M. L. O. S. (1997b). Controle de atitude autônomo de satélites usando légica nebulosa. *Anais do SBAI'97*, pages 337–342., Setembro 1997.

[12] S. Chiu and S. Chand. *Adaptative Traffic Signal Control Using Fuzzy Logic*. The Institute of Eletrical and Eletronics Engineers, Inc., New York, 1994.

[13] C. L. Karr and E. J. Gentry. Fuzzy control of pH using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 1(1):46 – 53., Fevereiro 1993.

[14] H. G. Weil, G. Probst, and F. Graf. *Fuzzy Expert System for Automatic Transmission Control*. The Institute of Eletrical and Eletronics Enginees, Inc., New York, 1994.

[15] G. J. Klir and T. A. Folger. *Fuzzy Sets, Uncertainty, and Informations*. Prentice Hall, Englewood Cliffs - New Jersey, 1988.

[16] R. D'Amore, O. Saotome, and K. H. Kienitz. Controlador nebuloso com detecção de regras ativas. *Anais do 3***?** *Simpósio Brasileiro de Automação Inteligente,*, pages 313–318., Setembro 1997.

[17] J. S. R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Trans. on SMC*, 23(3):665 – 685., 1993.

[18] C.-T. Lin. A neural fuzzy control system with structure and parameter learning. *Fuzzy Sets and Systems,*, (70):183 – 212., 1995.

[19] B. Kosko. *Neural Networks and Fuzzy Systems*. Prentice Hall, Englewood Cliffs - NJ, 1992.

[20] L. Wang and J. M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics.*, 22(6):1414 – 1427., Novembro/Dezembro 1992.

[21] P. Husbands. Genetic algorithms in optimization and adaptation. *Advances in Paralell Algorithms,*, pages 227 – 277., 1992.

[22] J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):122 – 128., Janeiro/Fevereiro 1986.

[23] T. Yoneyama and E. Cardozo. Randomized search méthods for optimization of controller parameters. In *Anais do X Congresso Brasileiro de Automática/ VI Congresso Latino Americano de Controle Automático*, pages 584 – 589., Rio de Janeiro - RJ, Setembro 1994.

[24] C. Karr. Applying genetics to fuzzy logic. *AI Expert,*, pages 38 – 43., Mar. 1991.

[25] C. Karr. *Adaptive Control with Fuzzy Logic and Genetic Algorithms*. Fuzzy Sets, Neural Networks and Soft Computing, (R.R. Yager, L.A. Zadeh, eds), 1994.

[26] D. T. Pham and D. Karaboga. Optimum design of fuzzy logic controllers using genetic algorithms. *J. Syst. Eng.,*, 1,:114 – 118., 1991.

[27] *Modifications of Genetic Algorithms for Designing and Optimizing Fuzzy Controlers*, Orlando - Fl, 1994. Proc. of the 1st IEEE Int. Conf. on Evolutionary Computation.

[28] Proc. of the IEEE/IFAC Joint Symp. on Computer-Aided Control Syst. Design. *Designing Fuzzy Net Controlers Using GA Optimizing*, Tucson - Az, 1994.

[29] *Generating Fuzzy Rules by Genetic Algorithms*, Nagoya - JP, 1994a. Proc of the 3rd IEEE Int. worshop on Robot and Human Communication.

[30] *Tuning and Optimization of Membership of Fuzy Logic Controlers by Genetic Algorithms*, Nagoya - JP, 1994a. Proc of the 3rd IEEE Int. worshop on Robot and Human Communication.

[31] *Genetic Algorithmes for Fuzzy Control, Part I: Offline System Development an Application*, volume 142. IEE Proc.: Control Theory Appl., Maio, 1995.

[32] A. Homaifar and E. McCormick. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetics algoritms. *IEEE Transactions on Fuzzy Systems*, 3(2):46 – 53., Maio 1993.

[33] K. H. Kienitz. A fuzzy control method based on possibility distribuitions. In *II Simpósio Brasileiro de Automação Inteligente*, pages 79 – 84, Curitiba, Setembro 1995.

[34] E. Sanchez, T. Shibata, and L. A. Zadeh. *Genetic Algoritms and Fuzzy Logic Systems: Soft Computing Perspectives*, volume 7 of *Advances in Fuzzy Systems: Applications and Theory*. Singapore: World Scientific Publishing Co. Pte. Ltd., 1997.

[35] M. Stachowitz, D. Yao, and T. Chen. Tuning a PID controller based on a genetic algorithm. Technical report, Tech. Report Lab. for Int. Syst., Dept. Elect. and Comp. Eng., Un. of Minnesotta., 1995.

[36] K. S. Tang, K. F. Man, S. Kwong, and Q. He. Genetic algoritms and their applications. *IEEE Signal Processing Magazine,*, 13(6):22 – 37., Novembro 1996.

[37] C. Correa. Uso de algoritmos genéticos no projeto de controladores nebulosos para o controle de atitude de um satélites artificial durante a fase de apontamento. Master's thesis, Instituto Nacional de Pesquisas Espaciais - INPE, São José dos Campos - SP, Março 1999.

[38] J. C. C. Aya and O. L. V. Costa. Otimização de controladores nebulosos usando algoritmos genéticos. *Anais do SBAI'97*, pages 207–212, Setembro 1997.

[39] J. C. A. Silva and C. K. (1997b). Algoritmo genético aplicado a simulação. *Anais do SBAI'97,*, pages 283 – 288., Setembro 1997.

[40] J. A. D. Vasconcelos, A. M. N. E. N. Cardoso, and F. A. Pinheiro. Algoritmo genético aplicado ao controle de tensão em sistemas elétricos de potência. *Anais do 3***?** *Simpósio Brasileiro de Automação Inteligente,*, pages 38 – 44, 1997.

[41] D. Dubois and H. Prade. *Possibility Theory*. Plenum Press, New York, 1988.

[42] H.-H. Bothe. Fuzzy neural networks. *Seventh International Fuzzy Systems Association World Congress/Tutorials - IFSA97*, pages 01–387, June 1997.

[43] D. Dubois and H. Prade. *Fuzzy Sets and Systems: Theory and Aplications*. Academic Press, New York, 1980.

[44] S. A. Sandri. Introdução a lógica "fuzzy". [online] <http://jupiter.lac.inpe.br/~sandri/files/vitoria/ps>, Agosto 1997.

[45] P. Bauer, S. Nouak, and R. Winkler. A brief course in fuzzy logic and fuzzy control. [online] <http://www.flll.uni-linz.ac.at/fuzzy/introduction.html>, April 1998.

[46] R. Guerra. Projeto e simulação do controle de atitude autônomo de satélites usando lógica nebulosa. Master's thesis, Instituto Nacional de Pesquisas Espaciais - INPE, São José dos Campos - SP, Março 1998.

[47] F. A. C. Gomide, R. R. Gudwin, and RicardoTanscheit. Conceitos fundamentais da teoria de conjuntos fuzzy, lógica fuzzy e aplicações. *Sixth International Fuzzy Systems Association World Congress/ Tutorials - IFSA95,*, pages 01 – 38., July 1995.

[48] E. H. Mamdani. Advances in the linguistic syntesis of fuzzy controllers. *Int. J. Man-Mach. Stud.*, 8:669 – 678., 1976.

[49] E. H. Mamdani, T. Procyk, and N. Baaklini. *Application of Fuzzy Logic to Controller Design Based on Linguistic Protocol*. Queem Mary College.

[50] C. C. Lee. Fuzzy logic in control systems: Fuzzy logic controller (part II). *IEEE Transactions on Systems, Man and Cybernetics.*, 20(2):419 – 430., Março/Abril 1990.

[51] State University of Campinas - UNICAMP Faculty of Eletrical Engineering - FEE Departament of Computer Engineering and Industrial Automation - DCA. *Sixth International Fuzzy Systems Association World Congress - Tutorials -IFSA95*, São Paulo - Brazil, July 1995.

[52] J. H. Holland. *Adaptation in natural and artificial systems*. The Univ. of Michigan Press, 1975.

[53] L. A. M. Palazzo. Algoritmos para computação evolutiva. [online] <http://akira.ucpel.tche.br/bbvirt/ps/comput~1.ps> <http://akira.ucpel.tche.br/bbvirt/art/comput~1.htm> Grupo de Pesquisa em Inteligência Artificial - Universidade Católica de Pelotas - Escola de Informática - lpalazzo@atlas.ucpel.tche.br, Dezembro 1996.

[54] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc., 1989.

[55] H. Hoffmann. Genetic algorithm. [online] <http://www.cs.berkeley.edu/~fhoffman/#gaclass> <http://HTTP.CS.Berkeley.EDU/~fhoffman/ga.htm> fhoffman@cs.berkeley.edu, May 1997.

[56] E. M. Henning Heider, Viktor Tryba. Automatic disign of fuzzy systems by genetic algorithms. *Fuzzy Logic and Soft Computing*, 1995.

[57] P. M. Larsen. *Industrial Aplications of Fuzzy Logic Control*. Academic Press Inc., London, 1981.

[58] Z. Michalewicz. *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, 1996. 3rd edition.

[59] R. H. Bishop. *Modern Control Systems Analysis and Design Using MATLAB©*. Addison-Wesley Publishing Company, 1993.

[60] B. Shahian and M. Hassul. *Control System Design Using Matlab©*. Prentice Hall, Engewood Cliffs - New Jersey 07632, 1993.

[61] D. C. Kuo, B. C.; Hanselman. *MATLAB© tools for control system analysis and design*. Prentice Hall, 1994.

[62] W. S. Leonard, N. E.; Levine. *Using MATLAB© to analyse and design control systems*. Menlo Park-Ca: Addison-Wesley Publishing Company, 1995. 2rd edition.

[63] K. Ogata. *Designing linear control systems with MATLAB©*. Englewood Cliffs: Prentice Hall, 1994.

[64] T. M. Inc. *The student edition of MATLAB^{TM}: Student user guide*. Englewood Cliffs: Prentice Hall, 1992.

[65] N. Roger J., J. S.; Gulley. *Fuzzy logic toolbox: For use with MATLAB©*. Natick: The MathWorks Inc., 1995.

[66] A. Potvin. Genetic algorithm m-files. [online] <ftp://ftp.mathworks.com/pub/contrib/v4/optim/genetic.zip> Modelo original desenvolvido em outubro de 1993. Copyright (c) 1993 by the MathWorks, Inc.Última modifição em 24 de janeiro de 1994., Março 1998.