# CONTROLADOR "NEURO-FUZZY" PARA PROCESSOS CONTÍNUOS EM SISTEMAS LINEARES E NÃO-LINEARES[1]

Alexandre Carvalho Leite[2]
William da Silva Vianna[3]

## Resumo

*Neste trabalho foi desenvolvido um algoritmo de controle que age baseado em uma lógica de controle similar à interpretação de controladores humanos tendo como fundamento princípios de lógica fuzzy e redes neurais artificiais. Esse algoritmo de controle se vale do ferramental de lógica fuzzy para trabalhar com regras lingüísticas e de redes neurais artificiais para ter a autonomia da criação das regras condizentes à situação atual. Em sua fase de desenvolvimento, o algoritmo foi treinado e testado com o suporte de um simulador desenvolvido especificamente para esse fim.*

Palavras-Chave: Controle e Modelagem Fuzzy, Lógica Fuzzy, RNA.

[1] 8º seminário de automação de processos. Belo Horizonte – MG/2004
[2] Tecnólogo em Automação e Controle, professor do CEFET-Campos
[3] Mestre em Engenharia de Materiais, professor do CEFET-Campos

## 1 INTRODUÇÃO

Em alguns processos as variáveis se comportam de tal forma que exigem uma modelagem matemática complexa para descrevê-los. Em certos casos a complexidade é tanta que se torna inviável. Nesse último, o controle dessas variáveis passa a ser feito manualmente, deixando de lado toda a praticidade e precisão oferecidas pelas tecnologias de controle. Outro fator que leva à adoção do controle manual é a dificuldade ou não-realização da sintonia de controladores clássicos, como o PID.

O algoritmo “Neuro-Fuzzy” funciona de forma similar ao controle manual, porém com precisão mais elevada, e isso o torna funcional diante de sistemas lineares e não-lineares. Utiliza-se lógica fuzzy para a interpretação de valores numéricos como variáveis lingüísticas, e Redes Neurais Artificiais (RNA’s) para gerar as decisões relativas ao controle específico.

Caso a lógica de controle seja alterada objetivando atender a um determinado processo, o código-fonte não necessitará ser modificado. Nesse caso a RNA é treinada para atender à lógica específica de controle da variável de processo.

## 2 LÓGICA DE CONTROLE FUZZY

Sabe-se que em teoria clássica dos conjuntos, os elementos têm sua pertinência a um determinado conjunto bem definida de forma bivalente, esse elemento pertence (1) ou não (0) ao conjunto. Em conjuntos "fuzzy" a pertinência de um elemento a um conjunto varia no intervalo [0,1], ao invés de apenas dois valores de pertinência são assumidos infinitos valores (TANSCHEIT, 2001).

Controladores operando em modo manual são capazes de controlar processos muito complexos, os operadores humanos baseiam-se em informações imprecisas para fazer esse controle. Essas informações podem ser expressas em forma de variáveis lingüísticas, uma vez que a estratégia de controle é traduzida em uma coleção de regras lingüísticas do tipo SE ENTÃO (TANSCHEIT 2, 2001), pode-se abstrair esta coleção de regras para um algoritmo fundamentado em lógica fuzzy capaz de executá-las, e conseqüentemente controlar o processo de forma automática.

Para oferecer esse controle o algoritmo deve traduzir valores numéricos em termos de variáveis lingüísticas, que são nomes de conjuntos "fuzzy", aplicar as regras lingüísticas em um mecanismo de inferência e por fim fornecer uma saída numérica precisa para o processo.

Nesse tipo de sistema não é necessário que se conheça o modelo matemático do processo, algo que é vantajoso sobre outras metodologias em que é fundamental a descrição matemática do sistema que se deseja controlar (TANSCHEIT 2, 2001).

### 2.1 Variáveis linguísticas

Conjuntos “fuzzy” permitem uma forma de manipular dados vagos e imprecisos, esses conjuntos podem ser empregados para representar variáveis lingüísticas. Uma variável lingüística pode ser considerada tanto como uma variável cujo valor é um número “fuzzy” quanto como uma variável definida por termos lingüísticos (TANSCHEIT, 2001). Um número “fuzzy” é uma determinação imprecisa como por exemplo: em torno de 200, maior do que 1000 e considerações afins. Se nível, por exemplo, for interpretada como uma variável lingüística, o conjunto de seus termos T(nível) seria T = {Muito baixo, Baixo, Médio, Alto, Muito Alto}. Onde cada termo é caracterizado por um conjunto “fuzzy” em um universo de discurso

definido entre o intervalo [0,100]. Pode-se interpretar como Muito Baixo um nível abaixo de 20; Médio como um nível próximo de 50; Alto como um nível próximo de 70.

A precisão do algoritmo é maior para uma maior quantidade de conjuntos. De forma geral, o valor de uma variável lingüística é um termo composto resultado da união de termos primários, modificadores e termos de negação e conectivos.

Os termos primários são os nomes dos conjuntos "fuzzy" (*baixo*, *médio* e *alto* como no exemplo acima), e suas funções de pertinência podem ser contínuas, descontinuas (como as triangulares do exemplo acima) ou discretizadas (TANSCHEIT, 2001). Tendo os termos primários já definidos, para aumentar a quantidade de conjuntos utilizam-se os termos modificadores como *muito* e *pouco*. Os conectivos e a negação são utilizados para operações entre conjuntos pertencentes tanto a universos distintos quanto ao mesmo.

Para aplicação em controle, as variáveis lingüísticas podem ser, por exemplo: variável manipulada, variável de processo, erro e variação do erro. Fazendo essas atribuições obtém-se uma transição simples e eficiente do modelo real para o virtual.

## 2.2 Fuzzificação

Antes que se suceda qualquer operação com variáveis lingüísticas é indispensável a tradução dos números reais do processo em membros de variáveis lingüísticas com seus respectivos valores de pertinência (FULLÉR, 1995). Um só valor real do processo pode ser mais de um membro em graus diferentes de relevância, o que influencia na criação de regras, pois a quantidade passa a variar, ocasionando combinações de diferentes condições e geração de outros resultados condizentes às condições particulares da regra.

Por exemplo, o número -22 é ao mesmo tempo considerado como médio e pequeno, porém em graus diferentes de importância; simultaneamente 10% pequeno e 26,674% médio. O algoritmo "NEURO-FUZZY" é capaz de identificar quaisquer valores numéricos dentro de um universo de discurso como um membro de uma variável lingüística em seu percentual de relevância. Na figura 3, foram fornecidos valores entre o intervalo [-100,100] como entrada, esses valores foram fuzzificados, e posteriormente houve a plotagem dos pontos gerados pelo par ordenado (valor de entrada, pertinência).

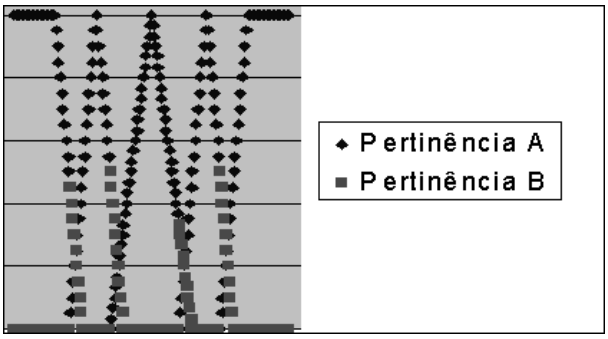


**Figura 1 – Fuzzificação de valores de ERRO**

A conversão de valores reais do processo como mais de um membro de uma variável lingüística gera uma quantidade maior de regras, mas a fase de fuzzificação não é responsável pela interpretação da lógica que efetua o controle, ela apenas trata os valores reais do processo em valores sujeitos a manipulação por lógica fuzzy para dar suporte aos passos seguintes.

## 2.3 Regras lingüísticas

A criação das regras lingüísticas depende de um especialista, e a base de conhecimento deste especialista normalmente pode ser traduzida da seguinte forma: SE (condições que devem ser satisfeitas) ENTÃO (conseqüências que devem ocorrer).

Em uma regra lingüística há utilização de termos primários, modificadores e conectivos; além de marcadores, que são os parênteses. E ainda se nota a presença de variáveis lingüísticas distintas (*ERRO*, *VARIAÇÃO DO ERRO* e *VARIÁVEL MANIPULADA*). Elas são as mesmas empregadas no algoritmo "NEURO-FUZZY". A escolha destas ocorreu porque são comuns a todos os processos que se deseja controlar. Em qualquer caso o desejável é normalizar a variável de processo reduzindo o seu erro com velocidade.

Se as variáveis lingüísticas fossem específicas como *NÍVEL*, *VARIAÇÃO DE NÍVEL* e *VÁLVULA*, o algoritmo só seria relevante para casos em que o objetivo fosse controle de nível, conseqüentemente o algoritmo não serviria para controlar temperatura, vazão, pH, pressão e etc. Idem a respeito da variável manipulada.

Quando são mapeados valores do processo que permitem criação de uma ou mais regras lingüísticas, e por conseqüência uma ou mais decisões a serem tomadas, são utilizados mecanismos de inferência, como o Mamdani exemplificado na figura 5, que permitem uma acepção da saída que será direcionada à variável manipulada (FULLÉR, 1995).

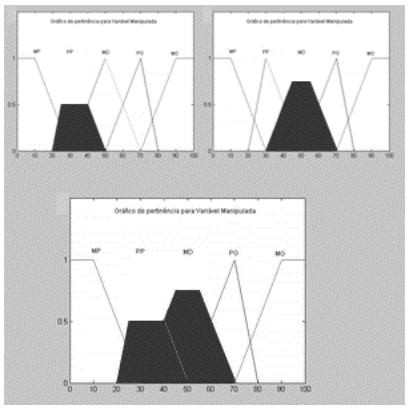


**Figura 2 – Exemplo do mecanismo de inferência Mamdani**

## 2.4 Defuzzificação

Os valores adquiridos através das regras lingüísticas são humanamente inteligíveis, mas da forma em que estão não são de grande utilidade ao processo. Então é utilizada a interface de defuzzificação, onde os valores lingüísticos são mapeados em valores numéricos de saída (BAUCHSPIESS, 2002). Obtendo-se um valor discreto que possa ser usado numa ação de controle no mundo real.

Existem vários métodos de defuzzificação, dentre eles pode-se citar: centro de gravidade (centróide), média balanceada, média dos máximos, centro da maior área e valor máximo (FULLÉR, 1995). O método utilizado no algoritmo "NEURO-FUZZY" é o centróide, que exige um poder de processamento mais alto, mas apresenta resultados bastante satisfatórios diante dos testes inferidos.

## 3 REDES NEURAIS ARTIFICIAIS (RNA's)

Redes Neurais são modelos computacionais não-lineares inspirados na estrutura de operação do cérebro humano, que procuram reproduzir características humanas, tais como: aprendizado, associação, generalização e abstração (ICA, 2001).

RNA's apresentam um comportamento bastante aceitável diante de padrões não-lineares, incompletos, com ruído e até compostos de eventos contraditórios (CASTRO e CASTRO, 2001). A exemplo das "Redes Neurais Naturais", as redes artificiais consistem da interconexão de um grande número de unidades de processamento chamadas neurônios (KOVÁCS, 1996). As comunicações entre as unidades computacionais (os neurônios) são chamadas sinapses ou pesos sinápticos.

Os neurônios de uma RNA são dispostos em camadas, os neurônios que recebem as entradas da rede compõem a camada de entrada, e os neurônios que têm como entradas as saídas dos neurônios da camada de entrada formam a segunda camada e assim por diante até a camada final, denominada camada de saída. As camadas intermediárias às camadas de entrada e saída são as camadas ocultas (KOVÁCS, 1996).

O aprendizado da Rede Neural utilizada no algoritmo "NEURO-FUZZY" se dá em uma etapa de treinamento, onde são apresentados pares de matrizes de entradas e saídas conhecidas; de acordo com esses padrões o algoritmo de treinamento ajusta os valores dos pesos sinápticos. Dependendo dos valores dos pesos sinápticos as saídas serão excitadas ou inibidas, nunca excitadas e inibidas ao mesmo tempo. Um modelo neural muito simples é o ANDALINE (ADAptative LInear Element) que tem uma generalização multidimensional, o MADALINE (Múltipla ADALINE) (KOVÁCS, 1996).

No algoritmo "NEURO-FUZZY" a RNA foi aplicada na criação de regras lingüísticas a partir de valores já traduzidos em variáveis lingüísticas com seus respectivos valores de pertinência. A RNA cria a quantidade de regras necessárias ao controle ótimo da variável de processo. Os valores de entrada da rede são as variáveis *ERRO* e *VARIAÇÃO DE ERRO* e a saída é *VARIÁVEL MANIPULADA*, ou seja, a formulação das conseqüências fica a cargo da RNA.

### 3.1 Neurônio Artificial

Compondo o corpo de um neurônio artificial temos: soma, função de ativação e função de transferência. A unidade soma se encarrega de somar o valor das entradas multiplicadas pelos pesos correspondentes, já a função de ativação é incumbida de transmitir o sinal para a saída do neurônio. A função de ativação determina o que deve ser feito com o valor da soma dos produtos entre as entradas e os pesos sinápticos, o que for determinado diz respeito somente a este neurônio. Em um próximo momento a saída da função de ativação é repassada à função de transferência, onde o valor é comparado com um valor estipulado, caso esse valor seja atingido ele será passado adiante através da saída. Mas se esse valor estipulado não for atingido, o sinal não prosseguirá (ALMEIDA, 2000). Na figura 7 tem-se um modelo não-linear de um neurônio segundo (CASTRO e CASTRO, 2001).

De acordo com o sinal de entrada, o sinal repassado ao neurônio seguinte será estimulante ou não (ALMEIDA, 2000).

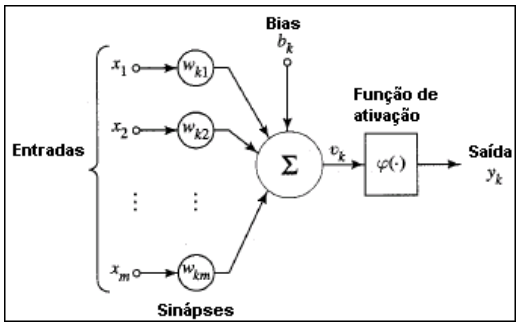


**Figura 3 – Modelo não-linear de um neurônio**

## 4 RNA E LÓGICA FUZZY

Lógica fuzzy e RNA têm propriedades computacionais particulares que fazem com que as mesmas sejam escolhidas para resolver alguns problemas particulares e outros não, RNA's são boas em reconhecimento de caracteres (FULLÉR, 1995), mas não são adequadas para se trabalhar com a manipulação de dados imprecisos, viabilidade notadamente aplicável aos algoritmos fundamentados em lógica fuzzy.

Sistemas baseados em lógica fuzzy podem lidar com dados imprecisos e são bons em manipulação das decisões obtidas a partir das regras lingüísticas. Mas há uma dependência por parte de algoritmos de lógica fuzzy em relação às regras lingüísticas adotadas, a aquisição dessas regras não pode ser feita automaticamente para a posterior aplicação do mecanismo de inferência e defuzzificação. Restrição que pode ser provida por RNA's.

As limitações de cada um podem ser supridas através de combinação entre as técnicas, assim formulando um sistema híbrido. No caso, a incorporação é entre os conceitos de lógica fuzzy e redes neurais artificiais.

Sistemas Híbridos agregam dois ou mais conceitos até então separados, como componentes de natureza discreta (digital) e contínua (analógica); a evolução de um sistema se dá em feições dinâmicas contínuas diferentes. Mas essa mudança é feita de maneira discreta, como nos sistemas inteiramente discretos (máquinas). Esta componente discreta faz com que as técnicas tradicionais de controle sejam de pouco uso, daí surgem técnicas de verificação formal desses sistemas.

Enquanto lógica fuzzy provê um mecanismo de inferência sob incertezas, redes neurais oferecem vantagens complementares como aprendizado, adaptação, tolerância de erro e generalização (FULLÉR, 1995). O aprendizado é uma das propriedades mais importantes dos seres ditos inteligentes, é a capacidade de se adaptar, de modificar e melhorar seu comportamento e suas respostas de acordo com as situações apresentadas ao longo do tempo (BITTENCOURT e OSÓRIO, 2000).

Para permitir que o sistema lide com incertezas de uma forma mais parecida com a interpretação humana foi incorporado os conceitos de lógica fuzzy e redes neurais artificiais, compondo um sistema híbrido, que pode ser definido como um sistema que se vale de mais de uma técnica de identificação de sistemas para a solução de um problema de modelagem (SOUZA, 1999).

Um modelo híbrido faz com que o algoritmo seja mais poderoso e diminua suas deficiências, uma vantagem sobre as técnicas de identificação individuais.

## 5 RESULTADO E DISCUSSÃO

O algoritmo de controle foi desenvolvido visando um funcionamento semelhante ao de um controlador em modo manual (operado por um humano), levando em consideração fatores como classificação de dados imprecisos, criação

de regras, adaptação a diferentes situações, tomada de decisões e interpretação com tolerância de erro. Tendo o algoritmo esse comportamento, procura-se atingir a normalização do estado de uma variável de processo com a maior rapidez possível.

Neste algoritmo de controle não foram descritos os comportamentos físicos de cada variável de processo que se deseja controlar, foi feita uma descrição passo a passo de como o problema (no caso controle de variáveis) é solucionável. Para fornecer uma saída ao processo o programa necessita apenas do erro e variação do erro, onde o erro é a diferença entre o valor desejado da variável de processo ("SetPoint") e o valor atual. A variação do erro é a diferença entre o erro atual e o erro em um instante imediatamente anterior.

A rede neural é utilizada em uma etapa intermediária do algoritmo, após ela e antes são utilizados ferramentais de lógica fuzzy para reconhecimento, decisão e posterior tradução para a saída real do processo.

Inicialmente são mapeados os dados do processo em membros de variáveis lingüísticas com seus valores de pertinência competentes, ou seja, valores de erro e variação de erro são interpretados lingüisticamente por uma interface de fuzzificação. Após definidos os membros com seus valores de pertinência, define-se as combinações possíveis entre estes como sendo antecedentes em uma regra lingüística. Por exemplo, se determinados valores de erro foram considerados como *Positivo Grande* e *Positivo Pequeno*, e, ao mesmo tempo, a variação do erro foi considerada como *Média*, poderiam ser geradas duas regras lingüísticas. *SE (ERRO é POSITIVO GRANDE) E (VARIAÇÃO DO ERRO é MÉDIA) ENTÃO (conseqüência)* ou também poderia ser gerada uma outra regra levando em consideração a pertinência da variável lingüística ERRO no membro *Positivo Pequeno*, *SE (ERRO é POSITIVO PEQUENO) E (VARIAÇÃO DO ERRO é MÉDIA) ENTÃO (conseqüência)*. Nesse caso foram geradas duas regras lingüísticas, mas nota-se que não há definição dos conseqüentes, essa ação será realizada pela rede neural na próxima etapa.

Uma vez definidos os pares antecedentes de cada caso esses pares são repassados à rede neural para que sejam percorridas suas conexões sinápticas e, finalmente, as saídas sejam obtidas; as saídas são os conseqüentes das regras lingüísticas que tem como antecedentes os pares fornecidos como entrada para a rede geradora desses conseqüentes.

A rede neural utilizada se vale apenas de duas camadas (entrada e saída) sem camada oculta intermediária. Existem onze neurônios de entrada incluindo o BIAS, os neurônios referentes à entrada dos antecedentes são dez, por exemplo, se os antecedentes forem *ERRO é NEGATIVO GRANDE E VARIAÇÃO DE ERRO é POSITIVO PEQUENO*, a entrada seria *(1, -1,-1,-1,-1,-1,-1,-1,1,-1)*.

A correspondência nos neurônios de saída é similar, a única mudança ocorre na variável lingüística, que ao invés de ser ERRO ou VARIAÇÃO DO ERRO é VARIÁVEL MANIPULADA.

No modelo implementado pode-se observar a presença do BIAS, polarização (ALMEIDA, 2000), que permite à rede neural uma capacidade maior de ajuste aos padrões fornecidos (GUIMARÃES e NETO, 2002). Caso o resultado de saída obtido seja *VARIÁVEL MANIPULADA é MÉDIA*, a matriz correspondente aos neurônios de saída seria *(-1 ,-1,1,-1,-1,-1,-1,-1,-1,-1,-1)* ou também *(-1,-1,-1,-1,-1,-1,-1,-1,1,-1,-1)*.

Mas para que a rede neural apresente valores satisfatórios ela deve ser treinada, deve ter suas 110 conexões sinápticas devidamente ajustadas. Então a rede neural foi submetida ao algoritmo de treinamento baseado na regra delta (KOVÁCS, 1996). Os valores definitivos de cada conexão são guardados em um

banco de dados externo de forma que quando o algoritmo é solicitado esses valores são carregados para a rede neural sem que haja a necessidade de novo treinamento, que, em alguns casos, pode ser demorado.

Para fins de teste foi estabelecida uma coleção de regras referentes à resolução de um determinado problema de controle. As regras têm antecedente e conseqüente bem definido, constituindo assim os padrões referentes ao treinamento da rede; o ajuste dos pesos para esse problema especifico se deu em apenas alguns segundos, extremamente aceitável para uma aplicação prática. O que não significa que um treinamento demorado fosse inaceitável em uma aplicação prática, levando em consideração que os valores do treinamento são armazenados em uma base de dados externa.

Quando se têm valores de conexões sinápticas armazenados externamente, pode-se selecionar as condições da rede neural de acordo com a lógica de controle a se implantar. Por exemplo, a rede neural pode ser treinada para trabalhar fornecendo valores de saída relativos ao controle de nível e, em uma outra situação, relativos ao controle de temperatura; quando uma situação similar a qualquer uma dessas surgir, basta selecionar no banco de dados a matriz referente à lógica de controle.

Após esta etapa o algoritmo estará apto a inferir a(s) regra(s) lingüística(s) formulada(s) com auxílio da rede neural, para isso utilizam-se mecanismos de inferência, no caso do algoritmo de controle proposto, mecanismo de inferência Mamdani.

Determinada situação, simulada em testes, apresentou um erro de 40% e uma variação de erro de 70%, esses dados reais do processo foram interpretados respectivamente pela interface de fuzzificação como *ERRO 50% POSITIVO PEQUENO* e *VARIAÇÃO DO ERRO 100% POSITIVO GRANDE*, então a matriz correspondente aos antecedentes da regra lingüística repassada à rede neural foi *(-1, -1,-1,1,-1,-1,-1,-1,-1,1)*, a camada de saída da rede apresentou a seguinte matriz *(1,-1,-1,-1,-1,-1,-1,-1,-1,-1)*, o que significa que a conseqüência da regra lingüística foi *MUITO PEQUENO*. Nesse caso específico foi gerada apenas uma regra lingüística, mas isso ocorreu porque estes valores reais especificamente não permitem que após fuzzificados seja mais de um membro de variável lingüística, seja ela *ERRO* ou *VARIAÇÃO DO ERRO*. No que diz respeito aos valores de pertinência das variáveis lingüísticas de entrada, o resultado da intersecção de dois conjuntos difusos é o menor valor de pertinência (LUCENA, 2001). Com base na conseqüência definida pela rede neural e na operação das variáveis lingüísticas, chega-se à conclusão de que a variável manipulada será *50% MUITO PEQUENA*, onde a pertinência foi resolvida por uma operação de conjuntos fuzzy e o membro da variável lingüística pela rede neural.

Mas esse valor obtido ainda não é útil como saída real do processo, o valor *50% MUITO PEQUENO* não é compreensível pelas interfaces usuais de controle, então será executada a tarefa de defuzzificação sobre o valor final da conseqüência da regra lingüística obtido. Somente este valor interessa para que seja efetuada a defuzzificação com êxito, assim como a interface de fuzzificação fazia uma interação processo-algoritmo a interface de defuzzificação fará a interação algoritmo-processo, em outras palavras o caminho de volta.

Existem muitos métodos de defuzzificação, alguns mais adequados a determinadas aplicações e outros não, o que não significa que um método seja definitivamente melhor do que o outro. Um método bastante popular e citado no início desse artigo é o centróide, o método que foi utilizado no algoritmo de controle.

Para funções complexas a implementação do método de defuzzificação centróide torna-se mais difícil (HOLBERT e PANT, 1999), mas no caso explicitado não há problemas quanto a isso, pois são funções lineares. Findada a etapa de defuzzificação o algoritmo gera um valor passível de ser transmitido ao instrumento responsável pela ação de controle no processo.

O algoritmo é composto de duas interfaces com os valores reais do processo, uma interface referente às variáveis lingüísticas *ERRO* e *VARIAÇÃO DE ERRO* e outra interface referente à variável lingüística *VARIÁVEL MANIPULADA*, uma mapeando os valores reais do processo em valores tratáveis por lógica fuzzy e outra traduzindo um resultado gerado por ferramental de lógica fuzzy em um valor real aplicável ao processo. A estrutura do algoritmo pode ser considerada em cinco camadas diferentes: Fuzzificação, Agregação de regras, RNA, Inferência e Defuzzificação.

Enfim, o funcionamento do algoritmo segue a ordem dessas camadas especificadas acima, após os valores mapeados em membros de variáveis lingüísticas é feita a agregação dos termos antecedentes que posteriormente são repassados à RNA, então a RNA gera os conseqüentes das regras, a essa altura já completas. Com as regras completas o mecanismo de inferência é aplicado, o passo seguinte é a execução do método de defuzzificação sobre o membro da variável lingüística alusiva à ação de controle. E finalmente é passado um valor útil ao processo, a ação de controle.

## 6 CONCLUSÕES

O algoritmo “NEURO-FUZZY” faz uso de técnicas de modelagem muito populares, RNA e lógica fuzzy; dependendo do caso em que o algoritmo for aplicado, talvez sejam necessárias alterações nas funções de pertinência, quantidade de membros, método de defuzzificação e arquitetura da RNA.

Foi notada grande eficácia diante dos testes inferidos, logo dados indicam que existe a possibilidade de desenvolver controladores baseados no algoritmo NEURO-FUZZY.

## AGRADECIMENTOS

Agradeço toda a ajuda de meus familiares e amigos, meu orientador, da instituição a qual pertenço e dos autores das referências bibliográficas, extremamente úteis na composição deste artigo.

## REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, M.A.F. Introdução ao estudo de redes neurais artificiais, Florianópolis, Universidade Federal de Santa Catarina, 2000, 10.

BAPTISTA, R.P. Algoritmo do Back-Propagation - MLP (Multi-Layer Perceptron), UNISINOS, 2000, 1-7. BAUCHSPIESS, A.; FILHO, F.M.L; GOSMANN, H.L.; Controle fuzzy para sistema de nível de líquidos, XIV - Congresso Brasileiro de Automática, 2002, 3017-3018.

BITTENCOURT, J.R.; OSÓRIO, F. Sistemas Inteligentes baseados em Redes Neurais Artificiais aplicados ao Processamento de Imagens, I WORKSHOP DE INTELIGÊNCIA ARTIFICIAL UNISC - Universidade de Santa Cruz do Sul, 2000, 1-28.

CASTRO, F.C.C.; CASTRO, M.C.F. Redes Neurais Artificiais, 2001, PUCRS, Capítulo 1, 9-12.

FIALLOS, M.; MELCÍADES, W.; PIMENTEL, C. Paralelização do Algoritmo "Backpropagation" em Clusters de Estações de Trabalho, IV Congresso Brasileiro de Redes Neurais, 1999, 231-232.
FILHO, C.S. DDE - "Dynamic Data Exchange", 2000, 3-5.
FULLÉR, R. "Neural Fuzzy Systems", 1995, Abo Akademi University, 8-91, 118-141, 153-179.
GUIMARÃES, I.A.; NETO, A.C. Avaliação de Técnicas de Reconhecimento de Padrões Através do Procedimento de Lachenbruch, 2002, 5.
HOLBERT, K.E.; PANT, S.N. Fuzzy Logic in Decision Making and Signal Processing, 1999, Capítulos 4-6.
ICA, Cursos em inteligência computacional, Núcleo de pesquisa em inteligência computacional aplicada, PUC-Rio, 2001, 4-6.
KOVÁCS, Z.L. Redes Neurais Artificiais: Fundamentos e Aplicações, Editora Collegium cognitio, São Paulo, 2ª edição, 1996, 53-58.
LUCENA, P.D.; PAULA, M.F.D. Árvores de decisão fuzzy, 2001, 8-11
MEDEIROS, J.S. Bancos de Dados Geográficos e Redes Neurais Artificiais: Tecnologias de Apoio à Gestão de Território, 1999, Capítulo 4.
SOUZA, F.J. Modelos Neuro-Fuzzy Hierárquicos, Rio de Janeiro, 1999, 6-12.
TANSCHEIT, R. Fundamentos de lógica fuzzy e controle fuzzy, PUC-Rio, 2001, 2-15.
TANSCHEIT, R. Lógica fuzzy, raciocínio aproximado e mecanismos de inferência, Rio de janeiro, 2001, 1-9.
WIDROW, B.; WINTER, R. “Neural Nets for Adaptative Filtering and Adaptative Pattern Recognition” – IEEE Na introdution to Neuron and Eletronics Networks, 1988.

**Abstract**

*Based on fuzzy logic and artificial neural network principles, a control algorithm was created, whose functioning is based on control logic similar to the interpretation of human controllers. This control algorithm make use of the fuzzy logic techniques to work with linguistic rules and artificial neural networks to have autonomy to create the rules related to the current situation. In its development stage the algorithm was trained and tested with the help of a simulator developed specifically for this purpose.*

Keywords: *Fuzzy Modeling, Fuzzy Logic, Neuro-Fuzzy Systems.*