GERAÇÃO AUTOMÁTICA DA BASE DE REGRAS DE MODELOS FUZZY UTILIZANDO ALGORITMOS GENÉTICOS

ELAINE Y. NAGAI, LUCIA V. R. DE ARRUDA

Centro de Pós-Graduação em Engenharia Elétrica e Informárica Industrial; Centro Federal de Educação Tecnológica do Paraná
Avenida Sete de Setembro, 3165, CEP 80250-901, Curitiba, Paraná, BRAZIL
E-mails: eynagai@cpgei.cefetpr.br, arruda@cpgei.cefetpr.br

Resumo— Este artigo apresenta uma metodologia de geração de modelos fuzzy a partir de dados de entrada e saída extraídos (amostrados) do sistema que se deseja controlar. Esta identificação é realizada através de um algoritmo genético, que determina tanto os conjuntos fuzzy relacionados a cada variável, quanto a base de regras que descreve o funcionamento do sistema. O algoritmo de identificação foi aplicado no controle de posição de um pêndulo invertido para a verificação de sua funcionalidade.

Abstract— In this paper a methodology to automatically perform a fuzzy model identification is presented. This methodology is based on genetic algorithm techniques. The validity of the proposed identification method has been demonstrated by simulation example of inverted pendulum.

Keywords— Fuzzy systems, fuzzy modeling, genetic algorithms, system identification, intelligent control.

1 Introdução

Em geral, na implementação de um controlador é necessário, inicialmente, encontrar um modelo matemático que descreve o processo. Para tal, todo o comportamento do processo a ser controlado deve ser conhecido, em detalhes, o que, em muitos casos, é inviável. Em diversas situações, um volume considerável de informações essenciais só é conhecido de forma qualitativa. Do mesmo modo, critérios de desempenho só estão disponíveis em termos lingüísticos. Este panorama leva à imprecisão e à inexatidão que inviabilizam a maioria das técnicas matemáticas usualmente utilizadas (Moraga e Vergara, 1997).

A lógica fuzzy proporciona um meio efetivo para modelar adequadamente a natureza aproximada e inexata do mundo real (Lee, 1990). Desta forma, ao invés de calcular um controlador através da modelagem matemática do processo, o controle fuzzy define um controlador diretamente do conhecimento (ou domínio) de especialistas ou operadores do processo (Linkens e Nie, 1995). No entanto, existem situações nas quais o conhecimento especialista é ausente. Nestes casos, é necessário que o modelo fuzzy seja identificado a partir de dados de entrada e saída obtidos através da observação do comportamento do sistema.

A identificação de um modelo fuzzy pode ser realizada através do uso de técnicas de agrupamento (clustering), redes neurais artificiais, computação evolucionária ou da combinação destas técnicas.

Neste trabalho é apresentada uma metodologia de identificação de modelos fuzzy, quando não há a disponibilidade de informações a respeito deste sistema, utilizando-se os princípios da teoria dos algoritmos genéticos.

2 Descrição do Problema

Considere o modelo lingüístico fuzzy para sistemas MISO (multiple-input / single-output), onde a estrutura do modelo consiste de uma coleção de regras fuzzy do tipo Mamdani. Portanto, cada regra fuzzy é da forma:

$$R_i : \textbf{Se } x_1 \text{ é } A_{i1} \textbf{ e } \ldots \textbf{ e } x_n \text{ é } A_{in} \textbf{ Então } y \text{ é } B_i \quad (1)$$

Onde x_j, $j=1,\ldots,n$, onde n é o número de antecedentes de uma regra, são as variáveis de entrada, y é a variável de saída, e A_{ij}, B, $j=1,\ldots,n$ e $i=1,\ldots,m$, onde m é o número de regras da base de regras, são valores lingüísticos das variáveis de entrada e saída, respectivamente, na i-ésima regra fuzzy.

As variáveis de entrada e saída possuem seus valores no universo de discurso normalizado [0,1]. O significado (semântica) dos valores lingüísticos são caracterizados pelas funções de pertinência $\mu_{Aij}(x_j)$ e $\mu_{Bi}(y)$ definidos sobre os universos de discurso [0,1]. Neste artigo serão consideradas funções de pertinência triangulares.

Um modo eficiente para representar-se, computacionalmente, este tipo de função de pertinência é através de tuplas com dois elementos: (a_{ij}, b_{ij}) para as variáveis de entrada e (a_i, b_i), para a variável de saída, onde o primeiro parâmetro representa a posição do centro da base do triângulo e o segundo representa o comprimento da base do triângulo, como pode ser verificado na figura 1.

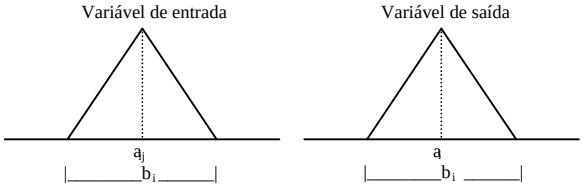


Figura 1

O procedimento de identificação está preocupado com: a identificação da estrutura do modelo, o que permite a obtenção da base de regras e, a estimação dos parâmetros do modelo, o que permite a obtenção da base de dados. A seguir são descritas as operações lógicas que são utilizadas no processo de inferência do modelo fuzzy a ser identificado.

Considerando-se um vetor de entrada $x = (x_1, x_2, ..., x_n)$ e uma base de regras composta por m regras lingüísticas fuzzy R_i, $i = 1,...,m$:

$$\begin{aligned} &Se\ x_1\ é\ A_{11}\ e\ \ldots\ e\ x_n\ é\ A_{1n}\ Então\ y\ é\ B_1 \\ &\vdots \\ &Se\ x_1\ é\ A_{m1}\ e\ \ldots\ e\ x_n\ é\ A_{mn}\ Então\ y\ é\ B_m \end{aligned} \quad (2)$$

O conectivo lógico **e** é utilizado para conectar os antecedentes em cada regra. Este conectivo é interpretado como o operador (t-norma) lógico min.

$$\mu_{A_i}(x) = \min\left(\mu_{A_{i1}}(x_1), \mu_{A_{i2}}(x_2), ..., \mu_{A_{in}}(x_n)\right) \quad (3)$$

A implicação fuzzy *Se-Então* é definida como uma conjunção das funções de pertinência do antecedente e do conseqüente de uma regra. Esta implicação é interpretada como:

$$\mu_{B_i^*}(y) = \min\left(\mu_{A_i}(x), \mu_{B_i}(y)\right) \quad (4)$$

É utilizado a s -norma max como conectivo de agregação das regras fuzzy:

$$\mu_{B^*}(y) = \bigcup_{i=1}^{m} \mu_{B_i^*}(y) \quad (5)$$

Onde $\mu_{B^*}(.)$ é a função de pertinência da saída inferida pelo sistema fuzzy. A saída numérica y^* é obtida através do método de defuzzificação COA (Center of Area), cujo funcionamento é descrito pela equação (6):

$$y^* = \frac{\sum_{k=1}^{d} y_k \cdot \mu_{B^*}(y_k)}{\sum_{k=1}^{d} \mu_{B^*}(y_k)} \quad (6)$$

Onde d é o número de discretizações do universo de discurso de saída, y_k é o valor representativo do *k-ésimo* intervalo de discretização do universo de discurso da variável de saída e $B^*(y_k)$ é a função de pertinência correspondente ao conjunto fuzzy resultante da inferência. Para a aplicação deste método de identificação é considerado um conjunto de dados de entrada/saída E_p composto por p tuplas de entrada/saída $e_i \in E_p$, chamados de exemplos, onde cada exemplo tem a seguinte forma (para um sistema com n entradas e uma saída):

$$e_l = (ex^l_1, ..., ex^l_n, ey^l),\ l = 1, ..., p \quad (7)$$

Em um modelo lingüístico fuzzy convencional, o conjunto de valores lingüísticos das variáveis de entrada e de saída é definido com antecedência. Além disso, o significado (semântica) de cada valor lingüístico A_{ij} é determinado pela função de pertinência $\mu_{Aij}(.)$ e o mesmo valor lingüístico pode aparecer em várias regras fuzzy. Porém, em cada regra fuzzy na qual este valor lingüístico aparecer, ele terá a mesma semântica, ou seja, a mesma função de pertinência. Este tipo de modelo lingüístico fuzzy é chamado de modelo lingüístico fuzzy descritivo. Neste trabalho é utilizado um modelo lingüístico fuzzy no qual as variáveis de entrada e saída não possuem valores definidos a priori, sendo por isso um modelo lingüístico fuzzy aproximativo. Quando este tipo de modelo é utilizado, as regras são ditas de semântica livre. Existem dois tipos de semântica livre: semântica livre irrestrita e semântica livre restrita. Esta implementação não impõe restrições com respeito à localização e formato das funções de pertinência, sendo, portanto, um modelo fuzzy com semântica livre irrestrita (Cordón e Herrera, 1997).

3 Estrutura da Identificação Fuzzy Genético

O método de identificação fuzzy genético é composto por duas partes: um módulo de construção de regras fuzzy baseado em algoritmos genéticos, e um método de cobertura para um conjunto exemplo de dados de entrada e saída. Esta parte provê os parâmetros e a estrutura inicial do modelo fuzzy; e um módulo de simplificação e sintonia da base de regras, onde é verificada a completude do sistema.

3.1 Módulo de Construção Genética das Regras Fuzzy

Para identificar um conjunto de regras fuzzy R, descrevendo a estrutura de um modelo lingüístico fuzzy , é necessário que este conjunto de regras cubra todos os possíveis pares de entradas e saída, $e_i \in E_p$, ou seja, deve respeitar a propriedade da completude do modelo (completeness). A propriedade de completude de um sistema fuzzy assegura que este sistema é capaz de inferir uma saída apropriada, para qualquer entrada dentro de seu universo de discurso, ou seja, o sistema cobrirá todos os estados possíveis dentro de seu domínio (Lee, 1990).

Para satisfazer a completude do sistema é necessário definir-se o valor de cobertura de um exemplo e também o grau de compatibilidade entre uma regra e um exemplo, conforme descrito a seguir. Seja a união não vazia das funções de pertinência $\mu_{Aij}(.)$ e $\mu_{Bi}(.)$, tal que:

$$\mu_{Ai}(ex^l) = *\left(\mu_{Ai1}(ex^l_1), ..., \mu_{Ain}(ex^l_n)\right) \quad (8)$$

$$R_i(e_l) = *\left(\mu_{A_i}(ex^l), \mu_{B_i}(ex^l)\right) \quad (9)$$

Onde: * é uma t -norma (nesta implementação foi utilizada a t -norma min); $R_i(e_l)$ é o grau de compatibilidade entre a regra R_i e o exemplo e_l; $l=1,...,p$; $\mu_{Aij}(.)$ é a função de pertinência do *j-ésimo* antecedente da regra i; $\mu_{Bi}(.)$ é a função de pertinência do conseqüente de saída da regra i.

Dado um conjunto de regras R, o valor de cobertura de um exemplo e_l é definido como:

$$CV_R(e_l) = \bigcup_{i=1}^{m} R_i(e_l), \quad (10)$$

$$CV_R(e_l) \geq \tau \quad (11)$$

Onde: m é o número total de regras da base de regras; τ é o grau de cobertura mínimo; $CV_R(e_l)$ é o valor de cobertura do exemplo $e_l \in E_p$ em relação ao conjunto de regras R.

O conjunto de regras fuzzy deve satisfazer as duas condições apresentadas acima, ou seja, devem respeitar a propriedade de completude e ter um valor de cobertura adequado.

A construção genética das regras fuzzy consiste de um método de construção e um método de cobertura, onde ambos utilizam um dado conjunto de exemplo. O método de construção é realizado através de um algoritmo genético (AG) onde cada regra fuzzy é codificada em um cromossomo. O AG encontra a melhor regra fuzzy em cada rodada sobre um conjunto de exemplos de acordo com a função de fitness considerada. O método de cobertura é realizado como um processo iterativo; ele permite a identificação de uma base de regras fuzzy tal que esta base cubra o conjunto de exemplos. Em cada iteração, o método de construção escolhe o melhor cromossomo (regra fuzzy), considera o valor de cobertura relativa que esta regra tem em relação ao conjunto de exemplos e remove os exemplos que tenham um valor de cobertura maior ou igual a um dado valor τ (equação (11)).

Para verificar a qualidade de uma regra, é utilizada uma variação da medida do fator de confiança (confidence factor - *CF*). Seja $A \rightarrow B$ uma regra, onde *A* representa a parte antecedente da regra (uma conjunção de condições) e *B* é o conseqüente da regra. A medida *CF* é denotada por $|A \wedge B|/|A|$, onde $|x|$ representa a cardinalidade do conjunto *x*, ou seja, a medida *CF* é proporção do número de exemplos que satisfazem as condições contidas no antecedente da regra (Quinlan, 1987). A variação da medida *CF* utilizada neste trabalho é dada como:

$$(|A \wedge B| - 1/2)/|A| \geq \delta \quad (12)$$

Onde o operador $\wedge$ é definido com uma t-norma min e δ=0.4. A motivação para a subtração de ½ do numerador é favorecer a descoberta de regras mais gerais (Arruda et alii, 1999).

3.1.1 Geração da População Inicial

Como mencionado anteriormente, cada cromossomo representará uma regra, e cada um destes cromossomos será um vetor codificado como um vetor de números reais. Na população de regras fuzzy, um cromossomo C_r, $r=1,...,Q$, representa uma regra fuzzy do tipo:

Se x_1 *é* A_{r1} *e* … *e* x_n *é* A_{rn} *Então* y *é* B_r (13)

Onde os reais $(a_{rj}, b_{rj},)$, $(a_r, b_r,)$ são os vetores parâmetro das funções de pertinência $\mu_{Arj}(x_j)$ e $\mu_{Br}(y)$, respectivamente. C_r codifica estes vetores como:

$$(a_{r1}, b_{r1},..., a_{rn}, b_{rn}, a_r, , b_r,) \quad (14)$$

Uma população de Q cromossomos será representada por *C*:

$$C=(C_1, ..., C_Q) \quad (15)$$

Onde:

- A população inicial de cromossomos é criada parcialmente levando-se em consideração o conjunto de exemplos E_p (dados de entrada e saída) da seguinte forma:
- t cromossomos serão criados nesta fase, onde $t = \min\{|E_p|, (Q/2)\}$, sendo Q = tamanho da população;
- Seleciona-se aleatoriamente t exemplos $e_i \in E_p$, formando o conjunto E_t, e para cada um destes exemplos de E_t é determinado um cromossomo (que pertencerá à população inicial);
- Considerando-se o exemplo $e_i \in E_p$, e o seu componente $ex_j^l \in [0,1]$, $\Delta ex_j^l = \min\{(ex_j^l - 0), (1 - ex_j^l)\}$. Seja $\gamma(ex_j^l)$ um valor aleatório pertencente ao intervalo $[0, \Delta ex_j^l]$. A função de pertinência será: $(a_j^l = ex_j^l, b_j^l = 2 * \gamma(ex_j^l))$, $(a_j^l = média, b_j^l = dispersão)$. O procedimento é o mesmo para todos os componentes de e_l;
- Os $Q - t$ cromossomos restantes da população inicial são escolhido aleatoriamente, com cada cromossomo em seu respectivo intervalo.

Função de Fitness. A função de fitness utilizada neste algoritmo genético é uma composição de três diferentes critérios:

(i) Valor de Alta Freqüência: a freqüência de uma regra fuzzy R_i, no conjunto de exemplos E_p, é definido como:

$$\Psi_{E_p}(R_i) = \frac{1}{p} \sum_{l=1}^{p} R_i(e_l) \quad (16)$$

Onde: $R_i(e_l)$ é o grau de compatibilidade entre R_i e $e_i \in E_p$; p é total de tuplas pertencente a E_p;

(ii) Grau Médio de Cobertura Sobre Exemplos Positivos: o conjunto de exemplos positivos para R_i com grau de compatibilidade maior ou igual a w é definido como:

$$E_w^+(R_i) = \{e_l \in E_p / R_i(e_l) \geq w\} \quad (17)$$

O grau médio de cobertura sobre $E_w^+(R_i)$ é definido como:

$$G_w(R_i) = \sum_{e_l \in E_w^+(R_i)} (R_i(e_l) / n_w^+(R_i)) \quad (18)$$

$$n_w^+(R_i) = |E_w^+(R_i)| \text{ e } w \in [0, 1] \quad (19)$$

(iii) Conjunto Reduzido de exemplos negativos: o conjunto de exemplos negativos para R_i é definido como:

$$E_w^-(R_i) = \{e_l \in E_p / R_i(e_l) = 0 \cdots e \cdots A_i(ex^l) > 0\} \quad (20)$$

Onde: $\mu_{Ai}(ex^l) = *(\mu_{Ai1}(ex_1^l), ..., \mu_{Ain}(ex_n^l))$;

Um exemplo é considerado negativo para uma regra fuzzy quando é melhor "coberta" por outra regra com o mesmo antecedente, mas com um consequente diferente. Os exemplos negativos são sempre considerados sobre um conjunto completo de exemplos (no caso o conjunto E_p). Seja $n_{R_i}^- = |E^-(R_i)|$, a função de penalidade sobre o conjunto de exemplos negativos é:

$$g_n(R_i^-) = \begin{cases} 1, \cdots se \cdots n_{R_i}^- \le kn_w^+(R_i) \\ \dfrac{1}{n_{R_i}^- - kn_w^+(R_i) + \exp(1)}, \cdots casocontrário \end{cases} \quad (21)$$

Onde é permitido, sobre uma determinada porcentagem do número de exemplos positivos $kn_w^+(R_i)$, um número de exemplos negativos sem qualquer penalidade. Esta porcentagem é definida pelo parâmetro $k \in [0, 1]$. Desta forma, a função de fitness é a maximização da seguinte equação:

$$Z(R_i) = \Psi_{E_p}(R_i) \cdot G_w(R_i) \cdot g_n(R_i^-) \quad (22)$$

Especificações do Algoritmo Genético .Para a implementação deste trabalho foi utilizado o Matlab, com o auxílio do toolbox de algoritmos genéticos (gaot – genetic algorithms for optimization toolbox).

Cada cromossomo representa uma regra (como demonstrado na equação (14)), e cada função de pertinência é representada por uma tupla de dois elementos. Se o objetivo do modelo for a aproximação de uma função de duas entradas e uma saída, cada cromossomo será composto por seis genes. Como pode ser observada, a equação (23):

$$C_i = (a_{i1}\ b_{i1}\ a_{i2}\ b_{i2}\ a_i\ b_i) \quad (23)$$

Utilizando-se o gaot, um cromossomo é um vetor com *n+1* elementos, onde n é o número de parâmetros de interesse. No exemplo dado acima, n=6, portanto, um cromossomo terá 7 elementos, sendo o último elemento o valor de fitness deste cromossomo, como pode ser observado na equação (24).

$$C_r = (a_{i1}\ b_{i1}\ a_{i2}\ b_{i2}\ a_i\ b_i\ fitness) \quad (24)$$

Os operadores genéticos são utilizados para criar novas soluções baseadas nas soluções existentes na população. Existem dois tipos básicos de operadores genéticos: crossover e mutação. O crossover produz dois novos indivíduos a partir de dois indivíduos, enquanto a mutação altera um indivíduo para produzir um novo indivíduo. A chamada destas funções no gaot é realizada da seguinte forma:

[c1,c2] = crossover(p1,p2,bounds,params) (25)

[c1] = mutação(p1,bounds,params) (26)

Onde *p1* e *p2* são os cromossomos pais, bounds é uma matriz com os limites de cada um dos elementos que compõem o cromossomo e *params* é o vetor (geração_corrente, par), sendo que *par* são os parâmetros do operador de crossover/mutação. O primeiro valor de *par* é a freqüência de aplicação deste operador. Os operadores genéticos utilizados foram: mutação não-uniforme (`'nonUnifMutation'`) com o parâmetro *b=5* e probabilidade de mutação igual a 0,07 (7% da população), e crossover aritmético (`'arithXover'`) com probabilidade igual a 0,6 (60% da população). O método de seleção utilizado foi o método do torneio (`'tournSelect'`) de tamanho (popsize) $^{1/2}$, onde popsize é o tamanho da população.

3.3 Fase de Simplificação da Base de Conhecimento

Nesta fase, ocorre uma modificação na representação de uma regra na base de regras. Agora, uma função de pertinência é denotada por uma tupla de três elementos (a,b,c), como ilustra a figura 2.

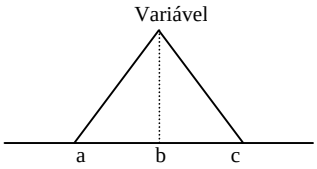


Figura 2 – Nova representação de uma função de pertinência

Esta modificação é necessária para corrigir possíveis falhas na cobertura do universo de discurso das variáveis do sistema, assegurando, assim, a propriedade de completude na base de dados. Ao representar as funções com três parâmetros, torna-se possível a existência de funções de pertinência triangulares não-simétricas. O número de funções de pertinência de cada uma das variáveis não é pré-determinado. Desta forma, há a possibilidade de existirem funções de pertinência sobrepostas e/ou concêntricas, fazendo com que o número de regras seja muito elevado. A fase de simplificação agrupa estas funções sobrepostas (ou concêntricas) de forma a tentar reduzir a quantidade de funções de cada uma das variáveis, visando manter a consistência do modelo. Considerando-se que o número de regras geradas geralmente é muito grande, devido ao elevado tamanho do conjunto de dados de treinamento (E_p), é provável que existam regras conflitantes entre si, isto é, regras que possuem o mesmo antecedente e consequentes diferentes. Para resolver este conflito, é definido um grau de veracidade para cada uma destas regras conflitantes, e a regra com o maior grau de veracidade é mantida e as outras são eliminadas. Desta forma, não somente as regras conflitantes são eliminadas, como também é reduzido o tamanho da base de regras, levando-se em conta a propriedade de completude desta base. O grau de veracidade de uma regra é definido como segue (Babuska e Verbruggen, 1997).

$$GrauVeracidade(R_i) = \prod_{l=1}^{n} R_i(e_l) \quad (27)$$

Onde: *n* é o número de pares de dados de treinamento; $R_i(e_l)$ é o grau de compatibilidade da Regra R_i em relação à tupla $e_i \in E_p$.

4 Algoritmo de Identificação

Seja E_p o conjunto de dados de treinamento, o método de identificação fuzzy genético é composto pelos seguintes passos:

Passo 1: Estipular os conjuntos de dados de treinamento e dados de teste, sendo estes dois conjuntos disjuntos. Costuma-se utilizar 1/3 dos dados disponíveis para o treinamento e 2/3 para o teste;

Passo 2: Executar o módulo gerador da base de regras;

Passo 2.1: Enquanto $|E_p| \neq 0$ seguir os 2.2 a 2.6

Passo 2.2: Executar o algoritmo genético (AG). O indivíduo resultante desta execução será uma regra candidata a integrar a base de regras;

Passo 2.3: Calcular o CF (vide equação (12)) (confidence factor) da regra candidata;

Passo 2.4: Caso o CF desta regra seja maior que o valor desejado e esta regra ainda não pertencer à base de regras, ela é inserida na base;

Passo 2.5: Calcular o valor de cobertura, $CV_R(.)$ (vide equação (10) e equação(11)), da regra em relação a cada par de dados do conjunto E_p;

Passo 2.6: Remover todos os pares de dados cujo valor de cobertura é maior ou igual à constante τ pré-estabelecida;

Passo 3: Executar o módulo de simplificação e sintonia (passos 3.1 a 3.4);

Passo 3.1: Modificar a representação da regra;

Passo 3.2: Agrupar as funções concêntricas e/ou sobrepostas para cada uma das variáveis;

Passo 3.3: Alinhar as funções de pertinência para garantir a total cobertura do universo de discurso;

Passo 3.4: Verificar e tratar a existência de regras inconsistentes;

Passo 4: Verificar a taxa de erro na utilização do modelo fuzzy obtido. Esta verificação é realizada com os dados de teste.

Passo 5: Caso a taxa de erro esteja abaixo limite aceitável, o algoritmo é encerrado, e a base de regras e os demais parâmetros do modelo fuzzy são salvos.

5 Resultados Experimentais

Esta seção demonstra os resultados obtidos na aplicação do algoritmo desenvolvido no controle de posição de um pêndulo invertido, descrito em Hui et alii (1993). Foi suposto um modelo fuzzy com 3 entrada ($y(t-1)$, $u(t-1)$, $u(t-2)$) e uma saída $y(t)$. Desta forma, o objetivo é identificar funções de pertinência destas quatro variáveis e as regras que descrevem o comportamento deste modelo. Para realizar a identificação, foram coletados 201 amostras do simulador, e destas amostras, 1/3 foram utilizados para o treinamento e 2/3 para o teste. Os universos de discurso de entrada e de saída foram normalizados no intervalo de [0,1]. O índice de desempenho utilizado na avaliação dos resultado obtidos é definido tal como segue:

$$MSE = \frac{1}{p}\sum_{l=1}^{p}\left(y(l) - y_f(l)\right)^2 \quad (28)$$

$$NRMSE = \frac{p}{\sum_{l=1}^{p} y(l)} \cdot \sqrt{MSE} \quad (29)$$

Onde: MSE (Mean-Square-Error) é a média do erro quadrático entre a saída desejada e a saída obtida, (ou saída fuzzy); $NRMSE$ (Normalized Root Mean Square Error) é o erro normalizado entre a saída fuzzy e a saída desejada; p é o número de dados; $y(l)$ é a saída desejada; $y_f(l)$ é a saída fuzzy.

5.1 Modelo Fuzzy Obtido

Os resultados obtidos estão demonstrados nas tabelas 1, 2 e 3. A tabela 1 contém a descrição do erro (quadrático – *MSE* e normalizado - *NRMSE*), o número de regras do modelo (*R*) e o número de partições no universo de discurso de cada uma das variáveis do modelo (ou o número de conceitos que define cada variável) (*NCE1*),(*NCE2*),(*NCE3*) e (*NCS*), para as variáveis de entrada 1, entrada 2, entrada 3 e saída, respectivamente.

Tabela 1 – Resultados Obtidos

MSE	NRMSE	R	NCE1	NCE2	NCE3	NCS
0.0026	0.0809(8.09%)	7	3	2	2	4

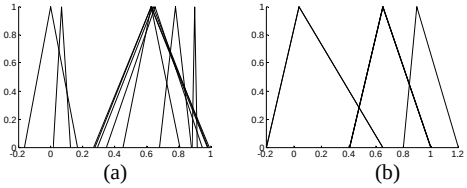
A base de regras obtida pode ser observada na tabela 2. Cada linha corresponde a uma regra com três antecedentes ($y(t-1),. u(t-1), u(t-2)$) e um conseqüente($y(t)$). A tabela 3 apresenta uma descrição numérica da partição do universo de discurso de cada uma das variáveis do modelo. A primeira coluna desta tabela contém a identificação lingüística do conjunto fuzzy, a segunda coluna identifica a qual variável este conjunto pertence e a terceira coluna apresenta os parâmetros (a sua posição no universo de discurso) deste conjunto. As figuras 3, 4, 5, 6 apresentam a partição do universo discurso de cada uma das variáveis do modelo. A figura 7 apresenta a saída obtida através do modelo fuzzy gerado pelo algoritmo de identificação (linha contínua) e a saída esperada (linha tracejada).

Tabela 2 – Base de Regras Obtida

E11	E22	E31	S1
E11	E22	E32	S2
E12	E21	E31	S3
E12	E21	E32	S3
E12	E22	E31	S3
E12	E22	E32	S3
E13	E22	E32	S4

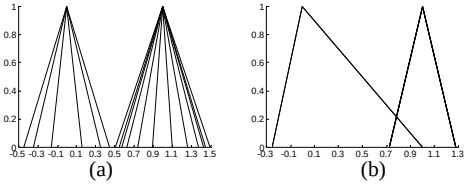
Tabela 3 - Descrição da Partição das Variáveis do Modelo

Conjunto Fuzzy	*Variável a qual este Conjunto Pertence*	*Parâmetros do Conjunto*
E11	$y(t-1)$ - entrada 1	[-0.2000, 0.0350, 0.6488]
E12	$y(t-1)$ - entrada 1	[0.4035, 0.6488, 1.0000]
E13	$y(t-1)$ - entrada 1	[0.8000, 0.9000, 1.2000]
E21	$u(t-1)$ - entrada 2	[-0.2543, 0, 1.0000]
E22	$u(t-1)$ - entrada 2	[0.7234, 1.0000, 1.2766]
E31	$u(t-2)$ - entrada 3	[-0.2216, 0, 1.0000]
E32	$u(t-2)$ - entrada 3	[0.7535, 1.0000, 1.2465]
S1	$y(t)$ - saída	[-0.2000, 0, 0.1600]
S2	$y(t)$ - saída	[0.0827, 0.1600, 0.6375]
S3	$y(t)$ - saída	[0.4811, 0.6375, 0.7939]
S4	$y(t)$ - saída	[0.7707, 0.8000, 1.2000]



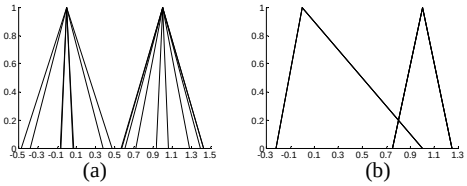
(a) (b)

Figura 3 – Partição do universo de discuso da variável de entrada 1 (y(t-1)) antes(a) e depois (b)da fase de simplificação



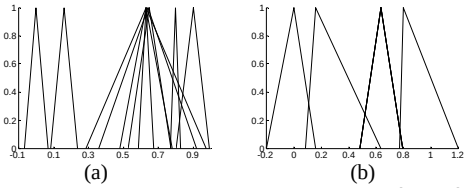
(a) (b)

Figura 4 – Partição do universo de discurso da variável entrada 2 (u(t-1)) antes (a) e depois (b)da fase de simplificação



(a) (b)

Figura 5 – Partição do universo de discurso da variável entrada 3 (u(t-2)) antes (a) e depois (b)da fase de simplificação



(a) (b)

Figura 6 – Partição do universo de discurso da variável saída (y(t)) antes (a) e depois (b)da fase de simplificação

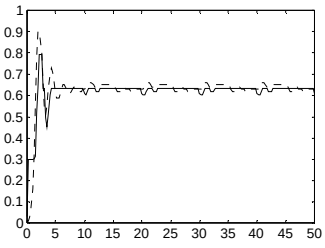


Figura 7 – Saída fuzzy obtida a partir da base de regras simplificada

6 Considerações Finais

Este artigo apresentou uma metodologia para identificação de modelos fuzzy utilizando AG. O método proposto é composto de duas partes: um módulo baseado em AG que gera a base de regras para o modelo fuzzy e um módulo de simplificação que realiza uma redução da base de regras obtida, garantindo ainda a completude do modelo obtido. Para validar o método, foi realizada a identificação, baseado em dados de simulação, de um pêndulo invertido. O modelo obtido reproduziu adequadamente o comportamento do sistema. No entanto, para validação da técnica é necessário, ainda, realizar estudos comparativos com outros métodos fuzzy de identificação existentes na literatura.

7 Referências Bibliográficas

Arruda, L. V. R.; Ferting, C. S.; Freitas, A. A. e Kaestner, C. *A.* (1999). Fuzzy Beam -Search Rule Induction Algorithm. *3rd European Conference on Principles and Practice of Knowledge Discovery in Database* , Praga, Rep. Tcheca, pp. 341 – 347.

Babuska, R. e Verbruggen, H. B. (1997). Constructing Fuzzy Models by product Space Clustering. In H. Helledoorn e D. Driankov (Eds.), *Fuzzy Model Identification: Selected Approach*, pp. 53 – 90, ed. Springer Verlag, Germany.

Cordón, O. e Herrera, F. (1997). Identification of Linguistic Fuzzy Models by Means of Genetic Algorithms. In H. Helledoorn e D. Driankov (Eds.), *Fuzzy Model Identification: Selected Approach*, pp. 215 – 250, ed. Springer Verlag, Germany.

Hui, S.; Kusche wski, J.G. e Zak, S.H. (1993). Application of Feedforward Neural Networks to Dynamical System Identification and Control. *IEEE Transactions on Control Systems Technology*, Vol. 01, pp. 37 – 49.

Lee, C. C. (1990). Fuzzy Logic in Control Systems: Fuzzy Logic Controller, Part I and II. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 20, pp.404-435.

Linkens, D. e Nie, J. (1995). *Fuzzy-Neural Control: Principles, Algorithms and Applications,* Prentice-Hall Int.Ltd.

Moraga, C. e Vergara, V. (1997). Optimization of Fuzzy Models by Global Numeric Optimization. In H. Helledoorn e D. Driankov (Eds.), *Fuzzy Model Identification: Selected A pproach*, pp. 251 – 278, ed. Springer Verlag, Germany.

Quinlan, J. R. (1987). Generating Production Rules from Decision Trees . *Proc. Int. Joint Conf. AI (IJCAI-87)*, pp. 304 – 307.