Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciências

Faculdade de Engenharia

Leandro da Costa Moraes Leite

Geração e simplificação da base de conhecimento de um sistema híbrido fuzzy-genético

Rio de Janeiro

2009

Leandro da Costa Moraes Leite

Geração e simplificação da base de conhecimento de um sistema híbrido fuzzy-genético

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientador: José Franco Machado do Amaral.

Rio de Janeiro

2009

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/CTCB

L533 Leite, Leandro da Costa Moraes.
Geração e simplificação da base de conhecimento de um sistema híbrido fuzzy-genético./ Leandro da Costa Moraes Leite. – 2009

110 p.

Orientador : José Franco Machado do Amaral.

Dissertação (mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1.Sistemas difusos. 2.Inteligência artificial.3. Sistemas especialistas I. Amaral, José Franco Machado do. II. Universidade do Estado do Rio de Janeiro. Faculdade de Engenharia. III. Título.

CDU 681.5

_____ _____

Assinatura Data

Leandro da Costa Moraes Leite

Geração e simplificação da base de conhecimento de um sistema híbrido fuzzy-genético

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em 17 de dezembro de 2009.

Banca Examinadora:

_____

Prof. Dr. José Franco Machado do Amaral (Orientador)
Faculdade de Engenharia - UERJ

_____

Prof. Dr. Jorge Luís Machado do Amaral.
Faculdade de Engenharia - UERJ

_____

Prof. Dr. Ricardo Tanscheit
Departamento de Engenharia Elétrica – PUC - Rio

Rio de Janeiro

2009

DEDICATÓRIA

Dedico esta obra ao meu saudoso avô, Alcy Vieira de Moraes, pelo exemplo de amor às coisas simples da vida.

AGRADECIMENTOS

Aos meus pais, Carlos Fernandes de Carvalho Leite e Meiry Terezinha da Costa Moraes Leite, pelo apoio, carinho e incentivo em todas as fases da minha vida.

Ao meu irmão, Marcus Vinícius da Costa Moraes Leite, pelo apoio e amizade.

A Danusa Viellas Rodrigues pelo amor e pelo companheirismo durante os momentos mais difíceis.

Ao amigo Luiz Eugênio de Andrade Segadilha, por ser um dos grandes incentivadores ao meu ingresso no mestrado.

Ao Prof. José Franco Machado do Amaral, pela importante orientação e por ter acreditado em meu trabalho.

Aos professores avaliadores desta dissertação, Prof. Jorge Luís Machado do Amaral e Prof. Ricardo Tanscheit, por aceitarem o convite para participação da banca examinadora deste trabalho e pelas importantes considerações e observações realizadas.

À UERJ, seus funcionários e professores pela ajuda e pelos ensinamentos transmitidos ao longo desta jornada.

Ao Instituto de Pesquisas da Marinha (IPqM), em especial a João Luís Marins (EN), pelo suporte concedido durante toda a fase do mestrado.

Uma jornada de duzentos quilômetros começa com um simples passo.

Antigo provérbio Chinês.

RESUMO

LEITE, Leandro da Costa Moraes. *Geração e Simplificação da Base de Conhecimento de um Sistema Híbrido Fuzzy-Genético*. 2009. 110 p. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2009.

Geração e Simplificação da Base de Conhecimento de um Sistema Híbrido Fuzzy-Genético propõe uma metodologia para o desenvolvimento da base de conhecimento de sistemas fuzzy, fundamentada em técnicas de computação evolucionária. Os sistemas fuzzy evoluídos são avaliados segundo dois critérios distintos: desempenho e interpretabilidade. Uma metodologia para a análise de problemas multiobjetivo utilizando a Lógica Fuzzy foi também desenvolvida para esse fim e incorporada ao processo de avaliação dos AGs. Os sistemas fuzzy evoluídos foram avaliados através de simulações computacionais e os resultados obtidos foram comparados com os obtidos por outros métodos em diferentes tipos de aplicações. O uso da metodologia proposta demonstrou que os sistemas fuzzy evoluídos possuem um bom desempenho aliado a uma boa interpretabilidade da sua base de conhecimento, tornando viável a sua utilização no projeto de sistemas reais.

Palavras-chave: Sistemas-Fuzzy. Computação evolucionária. Inteligência computacional. Multiobjetivo.

ABSTRACT

Genetic-Fuzzy System's Generation and Simplification of a Knowledge Base proposes a methodology to develop a knowledge base for fuzzy systems through the utilization of evolutionary computational techniques. The evolved fuzzy systems are evaluated considering two distinct criteria: performance and interpretability. Another Fuzzy Logic-based methodology for multiobjective problem analysis was also developed in this work and incorporated in GAs fitness evaluation process. The aforementioned systems were analyzed through computational simulations, and the results were compared to those obtained through other methods, in some applications. The proposed methodology demonstrated that the evolved fuzzy systems are capable of not only good performance, but also good interpretation of their knowledge base, thus showing that they can be effectively used in real world projects.

Key Words: Fuzzy systems. Evolutionary computation. Intelligence computation, Multiobjective.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

INTRODUÇÃO

A utilização de Sistemas Fuzzy para a solução de problemas do nosso cotidiano, nas mais variadas áreas de atuação, tem crescido significativamente desde o seu surgimento na década de sessenta. Desde então, a Lógica Fuzzy tem demonstrado a sua capacidade de representar aspectos do conhecimento humano de maneira simples e eficaz.

Fundamentada no conceito de variáveis lingüísticas, introduzido por Lotfi A. Zadeh, a Lógica Fuzzy construiu ao longo dos anos uma inquestionável história de sucesso. Parte desse sucesso pode ser explicado quando refletimos sobre a incrível capacidade humana em realizar uma ampla variedade de tarefas físicas e mentais, empregando, “apenas”, a nossa capacidade de percepção do meio ao qual estamos inseridos. De maneira natural, os nossos cérebros baseiam-se em nossas percepções sensoriais, que são informações incompletas e imprecisas sobre o meio, para a realização de inúmeras atividades complexas. Quando métodos computacionais tradicionais são utilizados para a solução de problemas complexos, cujo conhecimento seja decorrente de informações imprecisas e/ou vagas, os resultados obtidos são insatisfatórios. Assim sendo, a representação do problema real pelo modelo matemático obtido fica muito aquém da realidade do problema.

Para o desenvolvimento do conjunto de regras e do conjunto de funções de pertinência de um modelo fuzzy, é necessário um conhecimento prévio sobre o problema proposto (conhecimento subjetivo) ou a existência de dados históricos (conhecimento objetivo) que possibilitem a sua utilização para a geração da base de conhecimento do sistema fuzzy (CAMARGO, PIRES e CASTRO, 2004). A base de conhecimento está diretamente relacionada ao tipo de aplicação proposta, o que cria uma série de dificuldades, pois, o desempenho do sistema fuzzy dependerá da composição desta base de conhecimento, que foi definida especificamente para o problema tratado (CORDÓN, HERRERA e VILLAR, 2001). Como a maioria das técnicas convencionais utiliza métodos de tentativa e erro para a geração da base de conhecimento, o desenvolvimento dos sistemas fuzzy, principalmente os sistemas mais complexos, pode-se tornar uma tarefa árdua.

O termo *soft computing* é utilizado para designar uma família de técnicas computacionais, constituída pela Lógica Fuzzy (LF), Redes Neurais Artificiais (RNA) e pela Computação Evolucionária (CE), entre outras. Diferentemente das técnicas convencionais, as técnicas na *soft computing* apresentam, dentre outras características, um alto grau de

tolerância às imprecisões e incertezas. Essa tolerância possibilita que tais sistemas alcancem um melhor desempenho, grande autonomia, maior flexibilidade, baixo custo de implementação e melhor representatividade do problema real (ZADEH, 2001).

Os diferentes métodos que fazem da parte da família *soft computing* apresentam características inspiradas em diferentes áreas do conhecimento humano, como a lógica, a biologia, a fisiologia e a probabilidade. Por esse motivo, a escolha de qual método utilizar para a solução de um determinado problema ou aplicação está condicionada à natureza do problema envolvido.

A hibridização na *soft computing* possibilita a integração dos seus métodos e, por conseguinte, o surgimento de sistemas híbridos com características superiores aos demais sistemas baseados em uma única técnica.

Com relação aos sistemas fuzzy, muitos pesquisadores têm tentado integrar mais de uma técnica de Inteligência Computacional (hibridização) a fim de gerar um sistema com capacidade de aprendizagem (AMARAL, 2003). Tais sistemas possibilitam que a concepção da base de regras e das funções de pertinência de sistemas considerados complexos se torne uma tarefa menos árdua, especialmente, quando comparados aos métodos de tentativa e erro, muito utilizados no desenvolvimento dos sistemas fuzzy convencionais.

Dotar os sistemas fuzzy de uma capacidade de aprendizado foi, e continua sendo, o objeto de estudo de inúmeras pesquisas relacionadas à área de Inteligência Computacional. Um modelo fuzzy híbrido é um sistema adaptativo, caracterizado pelo ajuste interno de seus parâmetros, através do mapeamento entre o espaço de entrada e o espaço de saída (AMARAL et al., 2007). Dentre esses sistemas, podemos citar os Sistemas Neuro-Fuzzy como um modelo de hibridização de grande sucesso e tradição de pesquisa. Um enfoque diferente e mais recente, que tem apresentado grandes potencialidades, deu origem aos chamados Sistemas Fuzzy-Genéticos (CORDÓN et al., 2001c).

Os sistemas neuro-fuzzy agregam as vantagens inerentes aos sistemas de inferência fuzzy com a capacidade de aprendizagem comum às redes neurais artificiais. Dois modelos de sistemas neuro-fuzzy amplamente conhecidos são: o ANFIS (*Adaptative Network-Based Fuzzy Inference System*) e o NEFCON (*Neural Fuzzy Controler*). O sistema neuro-fuzzy ANFIS implementa um sistema de inferência fuzzy do tipo TSK (Takagi-Sugeno-Kang), e sua arquitetura possibilita a sua implementação em diferentes tipos de aplicações, como, por exemplo, sistemas de previsão, controle e aproximação de funções. O sistema neuro-fuzzy NEFCON foi desenvolvido especialmente para a área de controle, permitindo o aprendizado e

a otimização da base de regras de um controlador fuzzy do tipo Mamdani, de modo *online*, através da técnica de *reinforcement learning* (NÜRBERGER, 1999).

Nos sistemas híbridos do tipo Fuzzy-Genéticos, as técnicas baseadas na Computação Evolucionária possibilitam a aprendizagem total ou parcial da base de conhecimento de um sistema fuzzy, através da utilização dos princípios da evolução natural.

Alguns estudos demonstraram que os algoritmos genéticos estão sendo utilizados com sucesso no desenvolvimento da base de conhecimento dos sistemas fuzzy, possibilitando, dessa maneira, o aprendizado e a otimização tanto da base de dados quanto da base de regras do sistema. Há ainda trabalhos que apresentam o desenvolvimento de sistemas fuzzy, incluindo toda a base de conhecimento, além dos operadores que definem o mecanismo de inferência (Castro, 2004).

Neste trabalho, propomos uma metodologia que possibilita o desenvolvimento de toda a base de conhecimento de um sistema fuzzy, juntamente com a escolha do tipo de operador utilizado como *norma-t* no mecanismo de inferência fuzzy. Para alcançar tal objetivo, são aplicados métodos computacionais baseados na técnica dos algoritmos genéticos, gerando, assim, um sistema híbrido Fuzzy-Genético. Além do desenvolvimento de tais sistemas, a metodologia proposta possibilita o ajuste dos parâmetros das funções de pertinência dos conjuntos fuzzy, com o objetivo de melhorar o desempenho do sistema desenvolvido.

Outro aspecto interessante está relacionado à interpretabilidade dos sistemas fuzzy. Possibilitar que o sistema fuzzy evoluído seja facilmente compreendido, com relação ao problema e com relação à maneira com a qual o sistema deduz, com segurança, a sua solução, é fundamental para a sua validação. Por isso, a metodologia proposta neste trabalho tem como objetivo desenvolver um sistema fuzzy que possua um bom desempenho, aliado a uma boa interpretabilidade da sua base de conhecimento.

Os sistemas fuzzy desenvolvidos pelos algoritmos genéticos necessitam ser avaliados segundo dois objetivos distintos: desempenho e interpretabilidade. Existem na literatura diferentes técnicas para a análise de problemas que envolvam a solução de mais de um objetivo, também conhecidos como problemas multiobjetivo. Alguns estudos demonstraram que, dentre as diferentes técnicas existentes, o uso da Lógica Fuzzy proporciona resultados satisfatórios para a solução de tais problemas. Sua utilização apresenta certas vantagens quando comparada aos métodos tradicionais de solução multiobjetivo, como, por exemplo, a modelagem de problemas complexos pelo uso da linguagem natural, a possibilidade de incorporar as especificações do projetista na busca pelas melhores soluções, maior flexibilidade, etc.

Por esses motivos, uma nova técnica baseada na análise multiobjetivo, através da utilização de um modelo fuzzy, foi desenvolvida. Esse modelo fuzzy permite ao algoritmo genético avaliar os sistemas fuzzy evoluídos segundo os dois objetivos especificados, possibilitando, desta maneira, que o projetista seja capaz de desenvolver um sistema fuzzy que atenda aos critérios de desempenho e interpretabilidade estabelecidos, de uma maneira simples e eficiente.

Para demonstrar as potencialidades do método proposto neste trabalho, diversos experimentos envolvendo modelos matemáticos e simulações em software são apresentados, e os resultados são comparados aos obtidos pela utilização de outras técnicas.

Para a realização dos experimentos, os algoritmos e os modelos necessários às simulações foram desenvolvidos no ambiente MATLAB. Para as aplicações de controle e para a implementação do modelo fuzzy de avaliação multiobjetivo utilizou-se o SIMULINK, juntamente com o toolbox *Fuzzy Logic*, do próprio MATLAB/SIMULINK.

Esta dissertação está dividida em cinco capítulos. O capítulo um apresenta, de maneira sucinta, os conceitos básicos relacionados aos Sistemas Fuzzy e aos Algoritmos Genéticos. Esta apresentação visa esclarecer os conceitos básicos e fundamentais que fazem parte do contexto deste trabalho.

No capítulo dois descreve-se a hibridização das principais técnicas na área de Sistemas Inteligentes. Uma das técnicas mais tradicionais de hibridização que envolve o uso da lógica fuzzy e das redes neurais artificiais (Sistema Neuro-Fuzzy) é descrita de maneira sucinta. Os sistemas híbridos do tipo Fuzzy-Genéticos são descritos mais detalhadamente, por serem a base para a metodologia proposta. Ainda neste capítulo, os três principais métodos utilizados pelos algoritmos genéticos no processo de aprendizado são apresentados: o Método de Michigan, o Método de Pittsburgh e a Abordagem Interativa.

No capítulo três é apresentado o modelo híbrido proposto neste trabalho, sua arquitetura, suas características principais e o método de aprendizado evolucionário utilizado.

No capítulo quatro, são realizados os estudos de casos, onde os resultados obtidos são avaliados e comparados com as respostas dos sistemas originais utilizados.

No capítulo cinco temos a conclusão do trabalho, juntamente com uma relação de possíveis trabalhos futuros.

1. SISTEMAS INTELIGENTES

A busca do ser humano para compreender a natureza que o cerca e os seus diversos sistemas tem desempenhado um papel vital no avanço da ciência. A modelagem desses sistemas tem sido uma questão relevante na área de engenharia, especialmente quando se sabe que o mundo real é complexo (AMARAL, 2003).

Técnicas tradicionais que utilizam modelos matemáticos atribuem uma grande complexidade ao processo de representação dos sistemas reais. Em sistemas complexos, o uso de tais modelos matemáticos só é possível através de grandes simplificações do próprio modelo. Tais simplificações fazem com que, muitas vezes, os resultados obtidos experimentalmente não representem a realidade do sistema. Além disso, a capacidade em descrevermos o comportamento de determinados sistemas tende a reduzir, significativamente, em virtude da sua complexidade.

Para sobrepujar tais limitações, diferentes métodos baseados em Inteligência Computacional foram desenvolvidos nas últimas décadas, como, por exemplo: a Computação Evolucionária, as Redes Neurais Artificiais e a Lógica Fuzzy.

Este trabalho enfoca os sistemas híbridos Fuzzy-Genéticos, que possibilitam que os sistemas fuzzy convencionais sejam dotados de uma capacidade de aprendizagem das suas respectivas bases de conhecimento, através da análise de dados numéricos ou experimentais. O processo de aprendizado procura conciliar dois objetivos principais: desempenho e interpretabilidade. Infelizmente, esses dois objetivos não são simultaneamente alcançados de maneira fácil. Normalmente, um sistema fuzzy com uma boa performance possui uma base de conhecimento complexa, composta por um grande número de regras e funções de pertinência, em detrimento da sua interpretabilidade (CORDÓN et al.,2001a). Para solucionar tal problema, um novo método de avaliação baseado na lógica fuzzy foi proposto, possibilitando que os sistemas fuzzy fossem avaliados de maneira simples e intuitiva, com base nos critérios de desempenho e interpretabilidade estabelecidos pelo projetista.

Este capítulo aborda, sucintamente, os conceitos básicos da Lógica Fuzzy e da Computação Evolucionária, que formam a base para os fundamentos da metodologia proposta.

1.1 Sistemas Fuzzy

A teoria dos conjuntos fuzzy e a Lógica Fuzzy, desenvolvidas por Lotfi A. Zadeh na década de sessenta (ZADEH, 1965), fornecem um ferramental matemático para o tratamento de informações de caráter impreciso ou vago (TANSCHEIT, 1998). Os sistemas fuzzy, fundamentados na Lógica Fuzzy, têm demonstrado a sua eficiência nos mais diversos tipos de aplicações. Por ser capaz de incorporar o conhecimento de especialistas humanos, sua utilização em problemas de classificação, modelagem ou controle é destacada como uma das suas principais habilidades e motivos de seu sucesso (PEDRYCZ e GOMIDE, 1998) (CORDÓN et al., 2001c).

A lógica fuzzy encontrou inúmeros adeptos em todo o mundo nos mais diversos campos, como, por exemplo, na engenharia de controle, engenharia química, sistemas de auxílio à decisão, etc. Atualmente, são inúmeros os produtos desenvolvidos pelas indústrias que utilizam a lógica fuzzy para a execução de suas tarefas. A sua adoção pela indústria ocorreu inicialmente no Japão; devido ao grande sucesso, não demorou muito para que a lógica fuzzy fosse adotada nos principais centros industriais e de pesquisas dos principais países de todo o mundo (BONISSONE et al., 1995).

1.1.1 Sistemas Fuzzy Baseados em Regras

Nos dias de hoje, uma das principais áreas de aplicação da teoria dos conjuntos fuzzy, são os sistemas fuzzy baseados em regras (FRBSs – *Fuzzy Rule-Based Systems*). Os sistemas fuzzy utilizam uma estrutura de regras do tipo “Se-Então” similar às utilizadas nos sistemas clássicos de regras, diferenciando-se pelo fato de que, nos sistemas fuzzy, os antecedentes e os conseqüentes são constituídos por variáveis e conjuntos fuzzy.

A capacidade humana de resolver problemas está relacionada à nossa capacidade de inferir respostas com base em dados e informações imprecisas e/ou vagas. A lógica fuzzy, baseada na teoria dos conjuntos fuzzy, pode ser vista como uma extensão da lógica clássica, capaz de representar o conhecimento num ambiente de incerteza e imprecisão. Dessa maneira, os sistemas fuzzy, que utilizam um conjunto de regras fuzzy do tipo “Se-Então”, são capazes de representar o conhecimento humano de maneira concisa e eficaz.

Os dois principais tipos de sistemas fuzzy existentes na literatura são:

- Sistemas fuzzy do tipo Mamdani; e
- Sistemas fuzzy do tipo Takagi-Sugeno-Kang (TSK).

Esses dois tipos de sistemas serão descritos, resumidamente, nas duas próximas seções.

1.1.1.1 Sistemas Fuzzy do Tipo Mamdani

O sistema fuzzy do tipo Mamdani também é conhecido como um sistema com fuzzificador e defuzzificador ou, usualmente, como um controlador fuzzy (FLC – Fuzzy Logic Controller), como proposto pelo autor em seu artigo sobre o assunto (MAMDANI e ASSILIAN, 1975).

Na Figura 1 podemos observar a estrutura básica desse sistema.

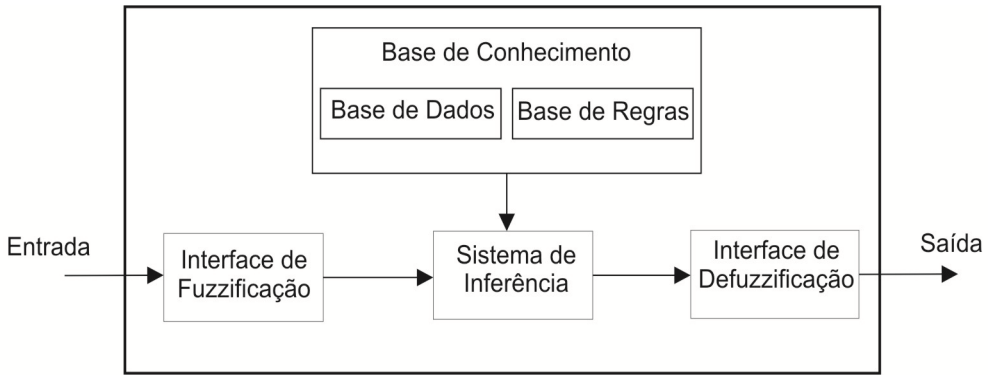

Figura 1- Estrutura Básica de um Sistema Fuzzy.

A base de conhecimento é responsável por armazenar o conhecimento específico sobre um determinado problema. Esse conhecimento está representado sob a forma de regras fuzzy do tipo “Se-Então”. Estas regras são formuladas a partir de variáveis lingüísticas, o que facilita o processo de interpretação do conhecimento obtido por especialistas humanos.

A base de dados e a base de regras são duas estruturas distintas que compõem a base de conhecimento. A base de dados contém os termos lingüísticos considerados nas regras lingüísticas e as funções de pertinência que definem a semântica dos termos que compõem cada uma das variáveis do sistema (CORDÓN et al., 2001c). Na Figura 2 observamos um exemplo de uma variável (temperatura), cujo universo de discurso é particionado por cinco termos lingüísticos (muito baixa, baixa, média, alta e muito alta).

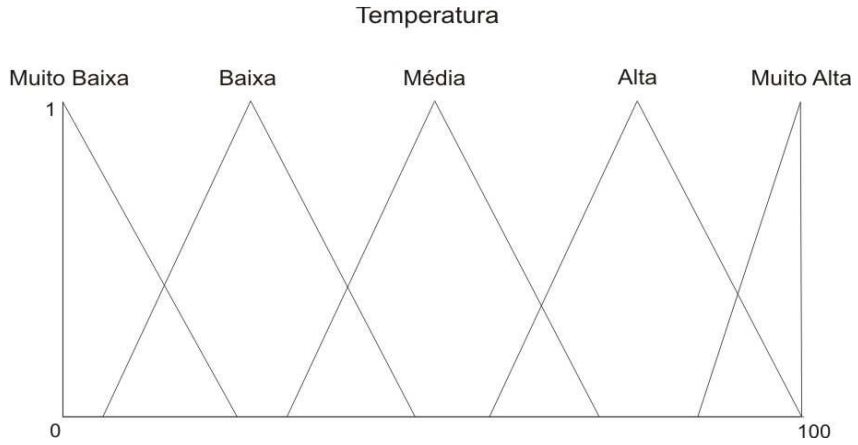

Figura 2 - Exemplo de uma variável fuzzy com cinco funções de pertinência.

A segunda estrutura na base de conhecimento é a base de regras, onde o conjunto de regras fuzzy está armazenado. Para um sistema com múltiplas entradas e uma única saída (MISO – Multiple Input Single Output), as regras possuem a seguinte forma:

$$\text{Se } X_1 \text{ é } A_1 \text{ e } \ldots X_n \text{é } A_n \text{ Então Y é B} \qquad (1)$$

onde, X_i e Y são as variáveis lingüísticas de entrada e saída, respectivamente, e A_i e B são os termos lingüísticos que definem os conjuntos fuzzy associados a cada uma das variáveis do sistema (CORDÓN et al., 2001c).

O mecanismo de inferência é composto por três componentes:

- Interface de Fuzzificação: responsável por transformar o dado de entrada em valores fuzzy.
- Sistema de Inferência: com base no conjunto de regras, estabelece a relação entre as variáveis de entrada e saída do sistema. O esquema de inferência fuzzy emprega o modus ponens generalizado, que é uma extensão do modo ponens clássico (ZADEH, 1973):

$$\begin{array}{l} \text{Se X é A Então Y é B} \qquad (2) \\ \text{X é } A' \\ \hline \text{Conclusão: Y é } B' \end{array}$$

- Sistema de Defuzzificação: responsável por converter os conjuntos fuzzy obtidos pelo processo de inferência numa saída de valor real (crisp) que representa a resposta do sistema. Dentre os vários métodos existentes, os

dois métodos mais utilizados são: o centro de gravidade e a média dos máximos.

A tarefa de agregação dos conjuntos de saída antecede ao processo de *defuzzificação*, e esta pode ser realizada de duas maneiras distintas: o modo FATI, do inglês *first aggregate than infer*, e o modo FITA, do inglês *first infer than aggregate* (BARDOSSSY e DUCKSTEIN, 1995) (CORDÓN, HERRERA e PEREGRÍN, 1997) (WANG, 1994).

No modo FATI, os conjuntos fuzzy Bi', originados das **n** diferentes regras do sistema, são agregados através de um operador G:

$$\mu_{B'}(y) = G\{\mu_{B1'}(y), \mu_{B2'}(y), \mu_{B3'}(y), \dots , \mu_{Bn'}(y)\} \qquad (3)$$

onde μ_{Bi} representa o conjunto fuzzy associado à variável lingüística y, que é a saída do sistema, e $\mu_{B'}(y)$ é o novo conjunto fuzzy implementado pelo operador G.

Em seguida, o operador D transforma o conjunto fuzzy $\mu_{B'}(y)$ num valor real y_0 (crisp):

$$y_0 = D(\mu_{B'}(y)) \qquad (4)$$

Normalmente, o operador G considerado é o valor máximo da t-conorma. O operador D usualmente utilizado é o centro de gravidade ou a média dos máximos, calculados, respectivamente, por:

$$y_0 = \frac{\int_0 \ y.u_{B'}(y)dy}{\int_0 \ u_{B'}(y)dy} \qquad (5)$$

$$y_0 = \frac{\sum_{m=1}^{M} y_m}{M} \qquad (6)$$

onde, y é o valor da variável de saída e $\mu_{B'}(y)$ é o seu grau de pertinência, y_m é o m-ésimo elemento do conjunto universo em que $\mu_{B'}(y_m)$ apresenta um valor máximo e M é o numero total desses elementos.

No modo FITA, a contribuição individual de cada conjunto Bi' é considerada separadamente e, ao final, um valor real (crisp) é obtido através de operador P, sem que haja a agregação dos conjuntos fuzzy. Usualmente, o operador P é a média ponderada dos valores individuais, dado por:

$$y_0 = \frac{\sum_{i=1}^{m} \mu_i \cdot y_i}{\sum_{i=1}^{m} \mu_i} \qquad (7)$$

onde, yi é obtido através da aplicação do operador D e μ_i é o grau de disparo da regra i.

A Figura 3 apresenta um exemplo de defuzzificação com os dois modos descritos anteriormente.

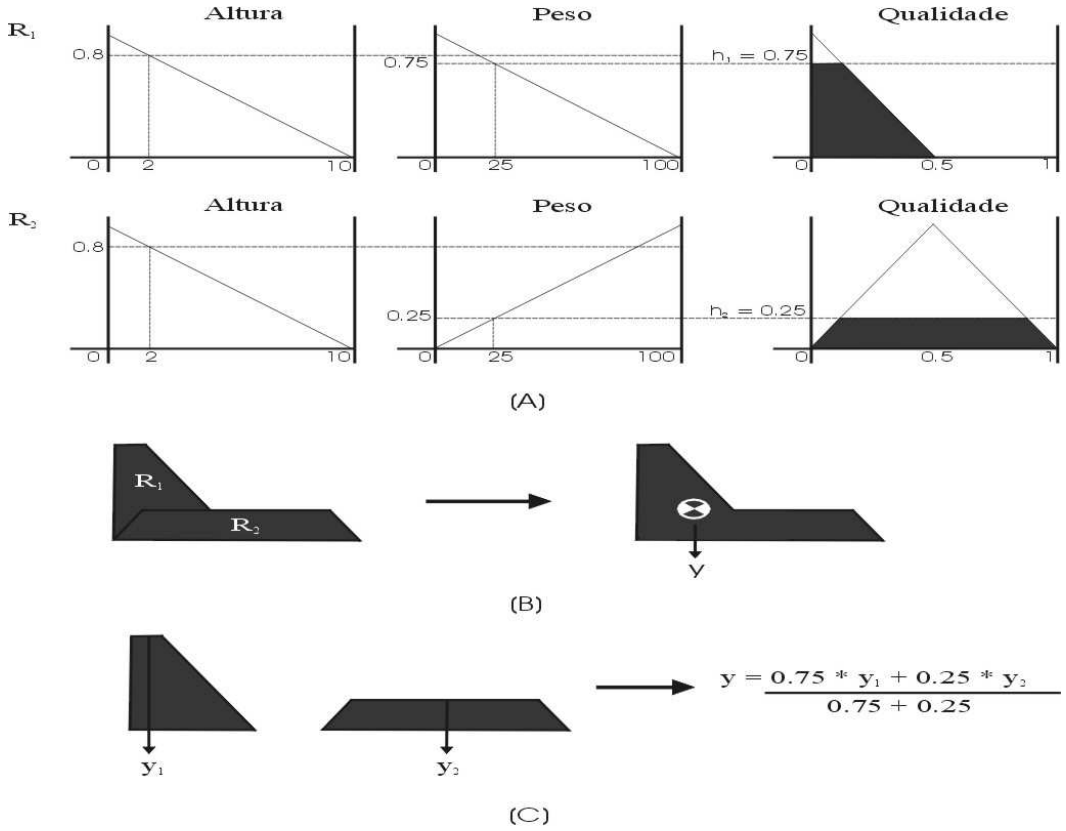

Figura 3 – Ilustração dos modos de defuzzificação FATI e FITA.

Na figura 3(A) é apresentado um exemplo com os dois conjuntos fuzzy de saída resultantes do acionamento das regras R_1 e R_2, respectivamente. A figura 3(B) apresenta o modo de dufuzzificação FATI, onde a saída y é o resultado do cálculo do centro de gravidade da figura, obtida pela agregação dos conjuntos R_1 e R_2. Na figura 3(C), o modo de defuzzificação FITA é empregado, calculando-se a média ponderada dos valores de y_1 e y_2, obtidos pelo cálculo das médias dos máximos dos conjuntos R_1 e R_2, respectivamente.

O modelo FITA, por não agregar os conjuntos fuzzy para posterior defuzzificação, é computacionalmente mais rápido do que o modelo FATI. Por esse motivo, o modelo FITA é

normalmente utilizado em aplicações executadas em tempo real e que necessitam de uma rápida resposta de saída do sistema (CORDÓN, HERRERA e PEREGRÍN, 1997) (DRIANKOV, HELLENDOORN e REINFRANK, 1993) (SUGENO e YASUKAWA, 1993).

1.1.1.2 Modelo Takagi-Sugeno-Kang (TSK)

Proposto por Takagi, Sugeno e Kang (1985,1988), esse modelo é uma alternativa ao modelo proposto por Mamdani. Similarmente ao modelo Mamdani, os termos antecedentes de cada regra do modelo TSK são compostos por variáveis lingüísticas, porém, o termo conseqüente de cada uma dessas regras é representado por uma função das variáveis de entrada do sistema.

A forma mais comum de se representar uma regra no modelo TSK é aquela em que a expressão do conseqüente constitui uma combinação linear das variáveis envolvidas no antecedente:

$$\text{Se } X_1 \text{ é } A_1 \text{ e } X_2 \text{ é } A_2 \text{ e .... } X_n \text{ é } A_n \text{ Então } Y = p_1 \cdot X_1 + p_2 \cdot X_2 + p_n \cdot X_n + p_0 \qquad (8)$$

onde, X_i e Y são as variáveis de entrada e saída do sistema, respectivamente, e p_i é um parâmetro real escolhido.

A saída do modelo TSK utilizando uma base de regras composta por *m* regras é obtida por uma soma ponderada das saídas individuais de cada uma dessas regras (CORDÓN et al., 2001c):

$$y_0 = \frac{\sum_{i=1}^{m} h_i \cdot Y_i}{\sum_{i=1}^{m} h_i} \qquad (9)$$

onde h_i é o resultado de um operador, do tipo norma-t, aplicado aos termos antecedentes da regra e às variáveis de entrada do sistema. Normalmente, o operador utilizado é o mínimo ou o produto algébrico.

1.2 Computação Evolutiva

A maioria das técnicas clássicas de otimização utiliza procedimentos determinísticos para encontrar uma nova solução, onde o resultado obtido é uma otimização do resultado anterior. Esses algoritmos, conhecidos como algoritmos de aproximação ponto-a-ponto, utilizam como solução inicial valores escolhidos aleatoriamente. A partir desses valores, com base numa regra de transição específica, o algoritmo sugere uma nova direção de busca, normalmente considerando informações locais. Uma busca unidirecional é então executada, tendo como referência à direção de busca sugerida previamente pelo algoritmo, na tentativa de encontrar o melhor resultado possível. Caso o resultado encontrado seja melhor do que o resultado anterior, esse passará a ser o novo resultado, e o algoritmo continuará a ser executado por mais um número determinado de vezes (DEB, 2004). A Figura 4 ilustra o procedimento descrito anteriormente.

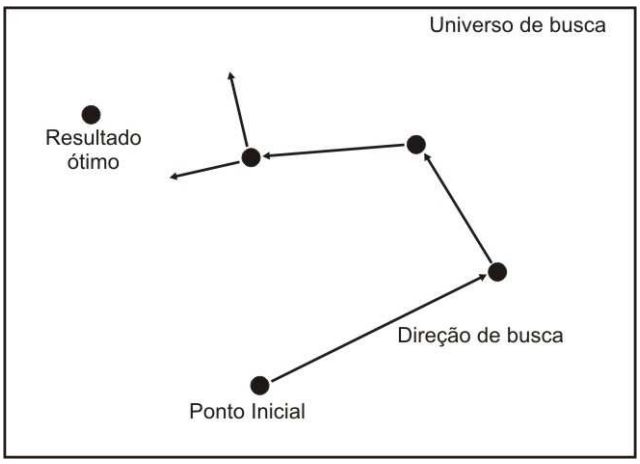

Figura 4 - Método clássico de otimização ponto-a-ponto.

Na computação evolutiva, a busca pela melhor solução para um determinado problema é inspirado na genética e nos princípios da evolução natural. Tais princípios foram introduzidos por Charles Darwin em 1859, no seu revolucionário livro A Origem das Espécies.

Segundo a evolução natural, os indivíduos mais aptos ao meio serão aqueles com maiores chances de sobrevivência e, conseqüentemente, com maiores chances de reprodução e geração de novos descendentes.

O termo computação evolutiva abrange diferentes técnicas, dentre as quais podemos destacar: estratégias evolutivas, programação evolutiva, algoritmos genéticos e programação

genética (BÄCK, FOGEL e MICHALEWICZ, 1997). Todas apresentam como características comuns a seleção dos indivíduos mais aptos da população através de técnicas de competição, métodos de combinação aleatória e alteração das estruturas genéticas, que potencialmente resultam em soluções melhores do que as existentes nas populações predecessoras.

Os algoritmos evolucionários proporcionam uma técnica de otimização universal, aplicada a uma grande variedade de problemas, tais como: otimização de parâmetros, problemas de análise combinatória, geração automática de programas computacionais, etc. Diferentemente dos métodos especialistas, desenvolvidos para tipos de tarefas específicas de otimização, não existe a obrigatoriedade do conhecimento específico com relação à estrutura do problema. É necessário, apenas, o conhecimento da função objetivo, capaz de mensurar as diferentes soluções para o problema proposto (GOLDBERG, 1989).

O próximo tópico aborda, de maneira sucinta, a técnica utilizada na otimização e no desenvolvimento dos sistemas híbridos fuzzy-genéticos, tema principal da metodologia apresentada neste trabalho. Esta é a técnica evolucionária utilizada para o aprendizado e simplificação da base de conhecimento dos sistemas fuzzy.

1.2.1 Algoritmos Genéticos

Nas últimas décadas, os algoritmos genéticos (AG) têm sido amplamente utilizados como ferramenta de busca e otimização nas mais variadas áreas, incluindo ciências, comércio e engenharia. Uma das razões principais para o sucesso dos algoritmos genéticos é a sua ampla aplicabilidade, facilidade de uso e perspectiva global (Mitchell, 1996). Os AGs diferem dos algoritmos de busca tradicionais em quatro pontos fundamentais (GOLDBERG, 1989):

1. Os AGs trabalham com a codificação do conjunto de dados e não com os próprios dados.
2. Os AGs buscam por uma população de pontos (soluções) e não por um único ponto.
3. Os AGs utilizam o valor da função objetivo para calcular os pontos mais promissores do espaço de busca, não necessitando de informações auxiliares sobre o problema ou cálculo de derivadas.
4. Os AGs utilizam regras de transição probabilísticas e não regras de transição determinísticas.

São estas as principais diferenças que tornam o AG um método de busca robusto, adequado a problemas em que o espaço de busca é complexo, extenso e desconhecido (ZEBULUM, 1999).

1.2.1.1 Funcionamento dos Algoritmos Genéticos

A idéia principal do algoritmo é que ao final de cada execução um novo conjunto de soluções com melhores aptidões seja criado, através da aplicação dos operadores inspirados na evolução biológica. Esse novo conjunto de soluções formará uma nova população que substituirá, totalmente ou parcialmente, a população anterior. Esse processo é inspirado na teoria da seleção natural de Darwin, onde os indivíduos mais aptos terão maiores chances de sobreviver e procriar, enquanto que, os indivíduos menos aptos terão maiores chances de "morrer". O algoritmo será executado inúmeras vezes, até que um critério de parada especificado seja satisfeito. Esse critério de parada pode ser um número máximo de gerações, saturação do valor de aptidão ou alcance de um valor pré-definido par a solução, por exemplo. A Figura 5 ilustra o procedimento descrito.

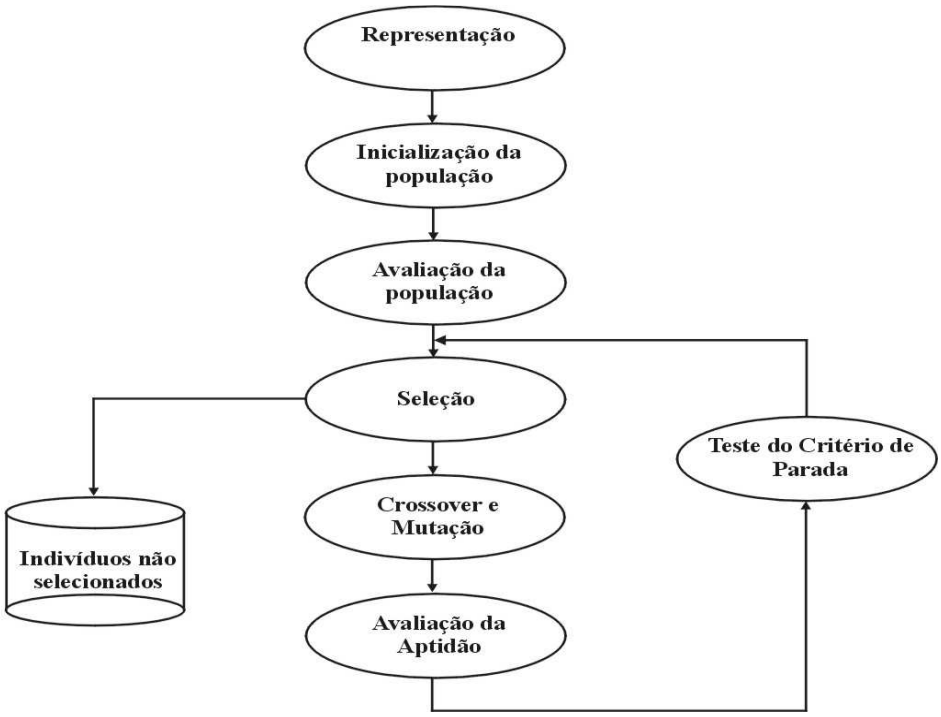

Figura 5 - Fluxograma de funcionamento de um AG.

Como podemos observar, ao final de cada execução, ao invés de ser apresentada uma única solução, o algoritmo produz um conjunto de possíveis soluções. Essa população de

soluções, obtida através de métodos probabilísticos e critérios inspirados na genética e na evolução natural, são as principais característica dos algoritmos genéticos.

No processo de utilização dos algoritmos genéticos, cinco critérios devem ser definidos (CORDÓN et al., 2001a):

1. Representação genética dos indivíduos (soluções);
2. Método de criação da população inicial;
3. Definição de uma função objetivo que descreva a aptidão de cada indivíduo da população;
4. Operadores genéticos que irão gerar as novas populações durante o processo de reprodução; e
5. Parâmetros gerais do AG, tais como tamanho da população, número máximo de gerações e probabilidade da aplicabilidade de cada operador genético.

O primeiro passo na utilização dos algoritmos genéticos é estabelecer como serão codificadas as soluções. Duas formas de representação podem ser utilizadas: as de tamanho fixo e as de tamanho variável (HARVEY, 1993). Na representação de tamanho fixo, os cromossomos que representam os indivíduos da população possuem um número de genes constante e iguais entre si. Já na representação com tamanho variável, o número de genes pode variar ao longo do processo evolutivo. Esta última representação proporciona uma maior flexibilidade, porém, a sua desvantagem é a necessidade de inclusão no algoritmo de mecanismos de controle de tamanhos, e de operadores de manipulação do tamanho dos genótipos (BANZHAF, FRANCONE e NORDIN, 1997) (HARVEY, 1993).

Usualmente, o método mais comum é a codificação binária, onde um *string* contendo **n** elementos representa um cromossomo. A Figura 6 apresenta um exemplo para a representação binária de um retângulo com quatro centímetros de base e dois centímetros de altura.

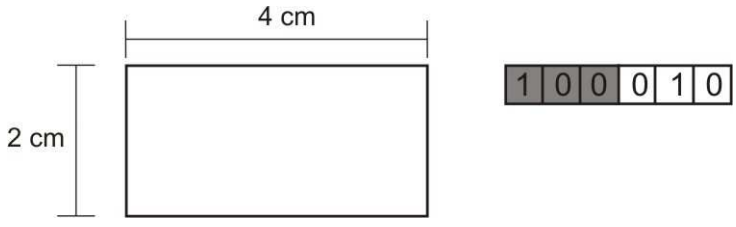

Figura 6 - Exemplo de representação binária do cromossomo.

Embora os AGs tenham sido inicialmente concebidos para utilizar a representação binária de tamanho fixo, é bastante usual a utilização de uma representação mais ampla, que utiliza números inteiros ou reais e cromossomos de tamanho fixo ou variável (ZEBULUM, 1999).

O segundo passo está relacionado à criação da população inicial de indivíduos. Tal processo consiste em gerar um número **i** de indivíduos que será a base da evolução do algoritmo genético. Esses indivíduos são gerados, tipicamente, de maneira aleatória, não seguindo qualquer tipo de relação determinística.

A escolha de uma função objetivo adequada para avaliar a aptidão de cada indivíduo é um dos pontos chave na execução bem sucedida dos algoritmos genéticos. O valor de aptidão é responsável por indicar quão boa uma determinada solução é para o problema em questão. A sua determinação é, em geral, realizada de maneira simples, quando apenas um único critério ou especificação deve ser atendido. Porém, a maior parte dos problemas de interesse prático deve atender a múltiplos objetivos, uma vez que diversas especificações devem ser levadas em consideração (FONSECA et al., 2003) (GOLDBERG, 1989). Neste último caso, a função objetivo deve utilizar métodos de conversão de uma medida de aptidão vetorial em um escalar (ZEBULUM, 1999).

O próximo critério a ser definido diz respeito aos tipos de operadores genéticos utilizados. O AG utiliza três tipos de operadores: seleção, *crossover* e mutação. O emprego desses operadores possibilita ao algoritmo genético evoluir, a cada geração, a sua população de indivíduos, na procura pelo indivíduo que represente a melhor solução.

O operador de seleção é responsável por selecionar os indivíduos que farão parte das próximas gerações. Sua função é semelhante ao processo da seleção natural biológica, onde os indivíduos mais aptos têm maiores chances de sobrevivência, enquanto que os indivíduos menos aptos têm maiores chances de ser excluídos (eliminados) das gerações futuras. Existem na literatura inúmeros algoritmos que executam a tarefa da seleção dentro do contexto dos algoritmos genéticos. Dentre eles, os métodos mais comuns são: seleção proporcional, seleção por torneio, seleção por truncamento e seleção escalonada (GOLDBERG e DEB, 1991) (BLICKLE, 1996).

O operador *crossover* proporciona expandir o critério de busca através da geração de dois novos indivíduos, com base na estrutura genética de seus antecessores. A taxa de probabilidade de o indivíduo ser selecionado pelo operador é usualmente alta, compreendida entre o intervalo [0.6, 0.95]. A Figura 7 apresenta um exemplo de aplicação do operador *crossover*, com um único ponto de cruzamento, entre dois indivíduos.

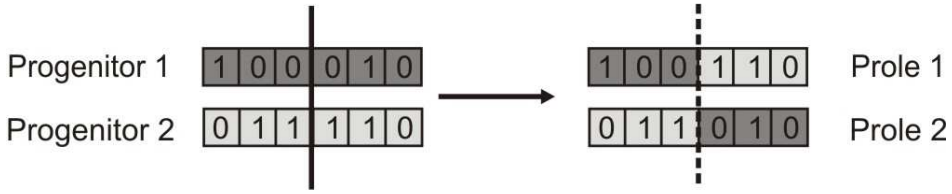

Figura 7 - Operador de *crossover* binário.

A mutação tem um papel importante na execução dos algoritmos genéticos. Sua utilização tem um caráter exploratório, dispersando a população pelo espaço de busca (PACHECO, 1996). Todos os indivíduos da população, obtidos por seleção e *crossover*, estão expostos à aplicação desse operador. Aplicado a cada elemento da estrutura do cromossomo, sua taxa de probabilidade usual está em torno de 1% (um por cento). Entretanto, diversos autores têm demonstrado que a utilização de valores entre 1% (um por cento) e 5% (cinco por cento) proporciona uma melhora no desempenho dos AGs em determinadas aplicações (MILLE, THOMSON e FOGATY, 1997) (ZEBULUM, 1999). A Figura 8 representa o resultado da aplicação do operador de mutação sobre um determinado indivíduo.

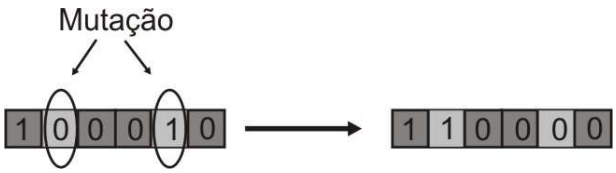

Figura 8 - Operador de mutação binário.

A literatura clássica de algoritmos genéticos considera o operador de *crossover* como sendo o principal no mecanismo de funcionamento dos AGs, ao passo que, o operador de mutação teria apenas um caráter secundário (GOLDBERG, 1989). Entretanto, muitos pesquisadores da área apresentam uma opinião contrária, acreditando que o operador de mutação é o mais importante no direcionamento de um algoritmo genético (HARVEY, 1993).

Considerando ainda os operadores genéticos, um aspecto importante diz respeito à substituição da população antiga pela nova geração. Esta substituição pode ser realizada de maneira total ou parcial. Na substituição parcial, apenas os indivíduos não selecionados são substituídos, recebendo o nome de reprodução *steady-state*. Também, é comum o uso do *elitismo*, que é uma estratégia que garante que o melhor indivíduo de cada geração será mantido para a próxima geração, evitando, dessa maneira, que os indivíduos mais aptos sejam perdidos nas novas gerações (PACHECO, 1996).

Por fim, o último critério a ser definido diz respeito os parâmetros gerais que sintonizam o AG, e estabelecem um critério de parada para execução do algoritmo. Alguns desses critérios, como tamanho da população, taxa de aplicação dos operadores, etc., terão uma influência significativa sob o desempenho do AG. As escolhas de tais parâmetros atendem aos critérios empíricos estabelecidos ou àqueles relacionados às características específicas do problema a ser solucionado.

2. SOFT COMPUTING

O termo *Soft Computing* se refere a uma família de técnicas computacionais, que inclui: a lógica fuzzy, a computação evolucionária, as redes neurais, a computação probabilística, dentre outras (BONARINE, MASULLI e PASI, 2003). De acordo com (MEUNIER, 1995), *Soft Computing* pode ser definida como uma técnica capaz de extrair, a partir de informações imprecisas, vagas e com um alto grau de incerteza, soluções consideradas robustas, tratáveis, de baixo custo e que mais se aproximam da realidade do problema.

As diferentes técnicas que pertencem à família *Soft Computing* possuem vantagens e desvantagens na sua utilização em determinados tipos de problemas. Apesar dos métodos compartilharem algumas características comuns, a presença ou ausência de determinadas características os tornam métodos complementares. Essa sinergia alcançada pela combinação de mais de uma técnica de *Soft Computing* recebe o nome de hibridização (BONISSONE et al., 1999). Na Figura 9 podemos observar alguns sistemas híbridos posicionados em função da interseção dos seus métodos individuais.

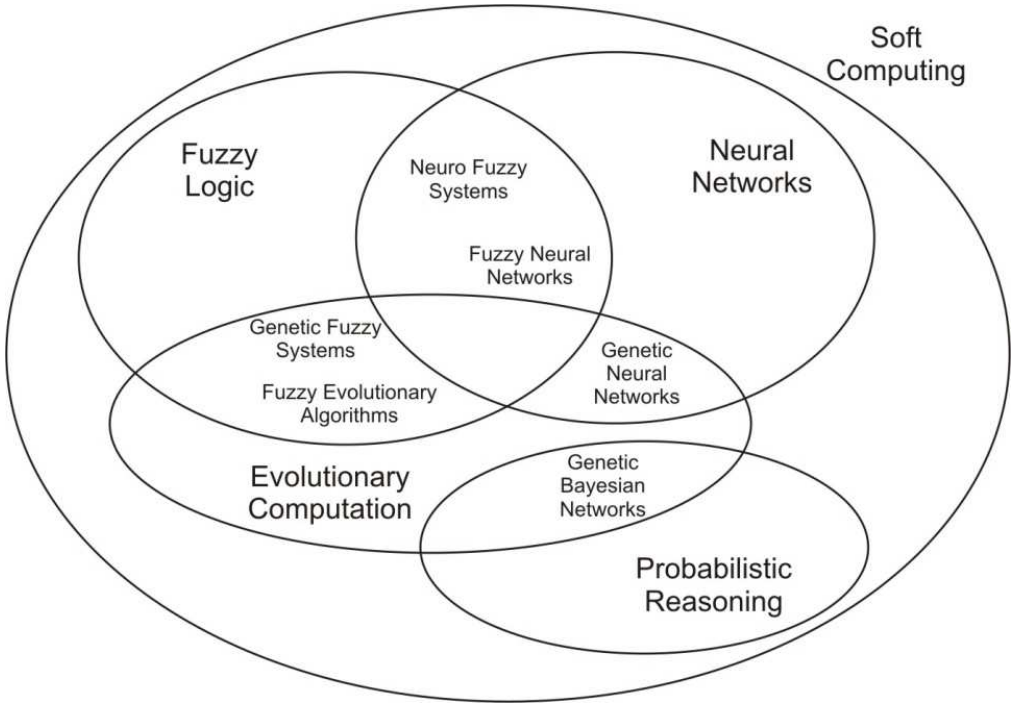

Figura 9 - Sistemas Híbridos da Soft Computing (CORDÓN et al., 2001a)

Nos próximos tópicos deste capítulo, descreveremos os sistemas híbridos do tipo neuro-fuzzy e fuzzy-genético, além dos métodos de Michigan, Pittsburgh e a abordagem Interativa, para o desenvolvimento dos sistemas híbridos fuzzy-genéticos. Concluindo este capítulo, apresentamos uma metodologia para a aprendizagem e a otimização dos SFG com base num único ciclo evolucionário.

2.1. Sistemas Neuro-Fuzzy

Os sistemas híbridos que combinam a Lógica Fuzzy (LF) com as Redes Neurais Artificiais (RNA) são reconhecidos na literatura como um dos primeiros exemplos da hibridização das técnicas de *Soft Computing* (CÓRDON et al., 2001b).

Devido a sua importância, e por utilizarmos neste trabalho exemplos que utilizam alguns modelos baseados nos sistemas do tipo Neuro-Fuzzy, descrevemos, de maneira sucinta, o referido método de hibridização e os dois principais sistemas a ele relacionados, que são: os Sistemas Neuro-Fuzzy ANFIS e os Sistemas Neuro-Fuzzy NEFCON.

O sistema neuro-fuzzy pode ser interpretado como um sistema fuzzy que possui a capacidade de adaptação e a habilidade de processamento paralelo similar às redes neurais artificiais (NAUCK e KRUISE, 1999) (LIN e LEE, 1996). Uma arquitetura típica de um sistema neuro-fuzzy é apresentada na Figura 10.

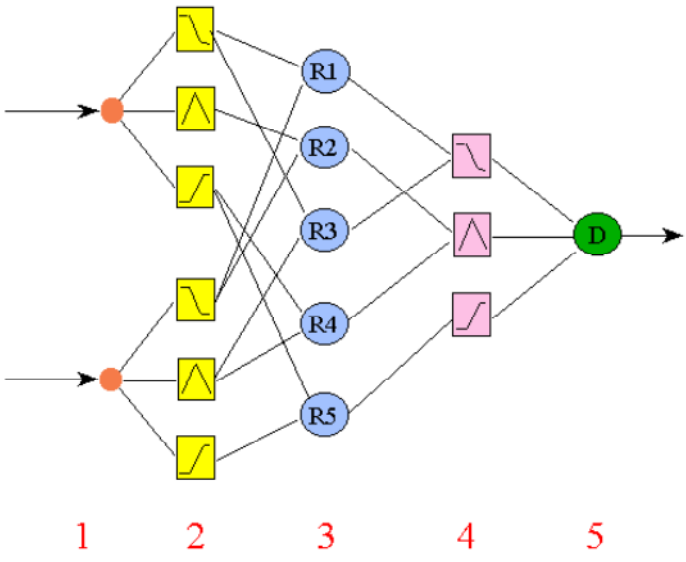

Figura 10 - Arquitetura de um Sistema Neuro-Fuzzy (SOUZA, 1999).

A numeração identifica as seguintes camadas: 1- Entradas, 2- Fuzzificação das entradas (pesos fuzzy), 3- Regras, 4- Conseqüentes das regras e 5- Defuzzificação.

2.1.1 Sistemas Neuro-Fuzzy ANFIS

O sistema Neuro-Fuzzy ANFIS (*Adaptive Network-based Fuzzy Inference System*) (JANG, 1993) implementa um sistema de inferência fuzzy TSK, normalmente utilizado na implementação de sistemas para previsão, aproximação de funções e controle (AMARAL, 2001). A Figura 11 ilustra um exemplo da arquitetura ANFIS.

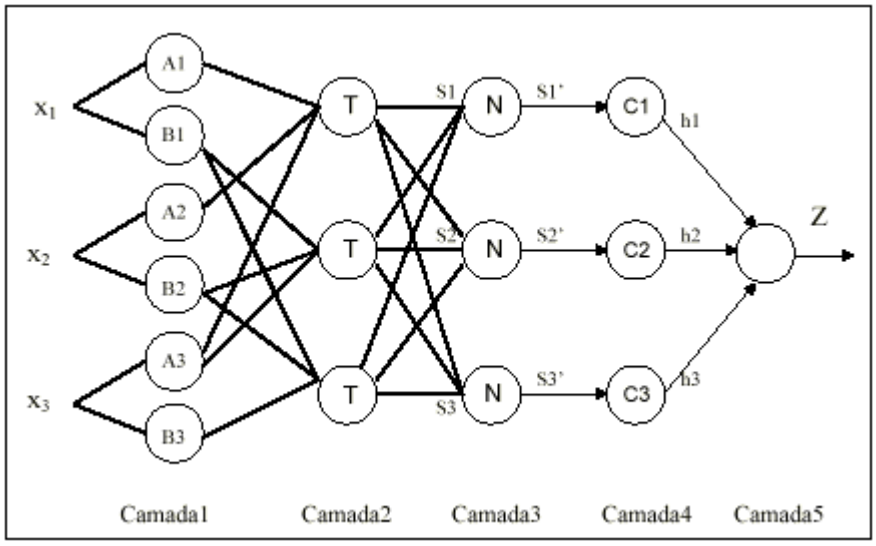

Figura 11 - Exemplo da Arquitetura ANFIS (SOUZA, 1999)

A camada 1 computa o grau de pertinência com que as entradas x_1 , x_2 e x_3 satisfazem os termos lingüísticos associados a esses nós (funções de pertinência). Na camada 2, cada nó corresponde a uma regra e computa o nível de disparo da regra, ou seja, calcula com que grau de pertinência o conseqüente da regra deve ser acionado (operação *norma-t*). A camada 3 realiza uma normalização nos níveis de disparo das regras. A normalização no sistema é utilizada como um pré-processamento para a defuzzificação. Na camada 4 as saídas dos nós são calculadas pelo produto entre os níveis de disparo normalizados e o valor do conseqüente da regra em si. Os conseqüentes são *singletons* ou combinações lineares das entradas. Finalmente, a camada 5 calcula a saída do sistema e promove a defuzzificação (AMARAL, 2001).

2.1.2 Sistemas Neuro-Fuzzy NEFCON

O Sistema Neuro-Fuzzy NEFCON (*NEural Fuzzy* CONtroler) foi especialmente desenvolvido para aplicações na área de controle. Um dos seus principais objetivos consiste no desenvolvimento e na otimização da base de regras de um controlador fuzzy do tipo Mamdani, de modo *online*.

O modelo NEFCON é derivado do modelo genérico de três camadas de um *perceptron* fuzzy. Ele consiste de uma camada de entrada, de uma camada escondida de regras e de uma camada de saída. As conexões entre as camadas são ponderadas por conjuntos fuzzy. Cada uma das camadas contém um certo número de unidades, na qual a camada 2 (camada escondida das regras) usa uma *norma-t*, enquanto que a unidade de saída combina os conjuntos nebulosos resultantes e realiza o procedimento de defuzzificação. As unidades de entrada contêm apenas os valores das entradas do modelo e não realizam processamento (AMARAL, 2003). A Figura 12 apresenta um exemplo do modelo NEFCON.

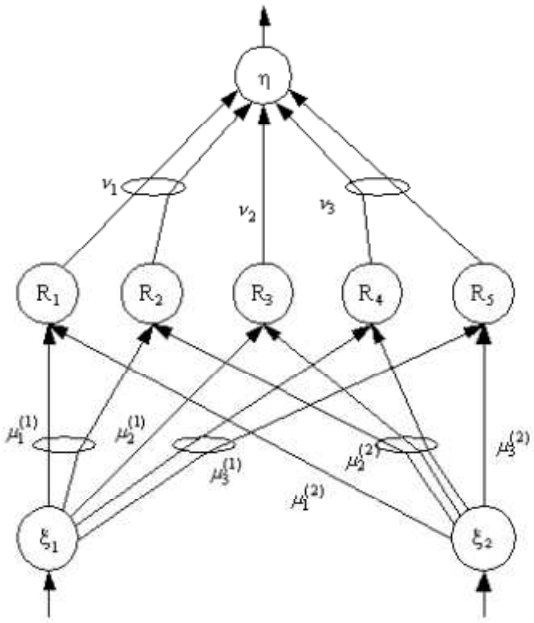

Figura 12 - Exemplo do modelo Neuro-Fuzzy NEFCON (AMARAL, 2001).

Usualmente, esse tipo de sistema utiliza o conhecimento de um especialista humano como conhecimento prévio para a inicialização do aprendizado do controlador fuzzy. Após o processo de aprendizagem, o controlador pode ser retirado do processo de desenvolvimento para ser utilizado em aplicações reais.

2.2 Sistemas Fuzzy-Genéticos

Uma das maiores dificuldades no desenvolvimento dos sistemas fuzzy consiste em gerar a sua base de conhecimento (CORDÓN e HERRERA, 2001). Geralmente, a base de conhecimento é gerada a partir do conhecimento extraído diretamente de um especialista. Porém, muitas vezes a representação explícita desse conhecimento, em regras do tipo "Se-Então", é difícil de ser realizada, devido à possibilidade de se inserir regras não utilizadas, incoerentes e contraditórias (TÚPAC et al., 1999).

Essa dificuldade motivou a utilização de técnicas de aprendizagem automática durante o processo de desenvolvimento dos sistemas fuzzy. Muitas vezes, a construção automática da base de conhecimento de um sistema fuzzy pode ser considerada um processo de busca no espaço de possíveis soluções (CORDÓN et al., 2001b). Dessa forma, as características dos algoritmos genéticos os tornam apropriados para a realização de tal tarefa. Assim, o sistema fuzzy que possui algum componente da sua base de conhecimento gerado ou aperfeiçoamento por algoritmos genéticos recebe o nome de Sistema Fuzzy-Genético (SFG) (CAMARGO, PIRES e CASTRO, 2004).

De acordo com (DEJONG, 1988), os algoritmos genéticos atuam com diferentes graus de complexidade, em função das alterações estruturais promovidas pela sua utilização nos diferentes tipos de problemas. Nos sistemas híbridos fuzzy-genéticos, a atuação do AG pode ocorrer desde a maneira mais simples, através da otimização de determinados parâmetros, até a maneira mais complexa, que envolve o aprendizado da base de regras e/ou base de dados do sistema fuzzy.

Portanto, os SFGs podem ser divididos segundo a sua aplicação: a otimização e aprendizado. Nos problemas de otimização, o AG atua sobre uma base de conhecimento já pré-definida, enquanto que nos problemas de aprendizado o AG é o responsável por gerar a base de conhecimento (base de dados e/ou base de regras) do sistema (CORDÓN, et al., 2001a).

Nos próximos tópicos, o uso do AG é apresentado na solução de problemas que envolvem tanto a otimização quanto a aprendizagem da base de conhecimento dos sistemas fuzzy.

2.2.1 Otimização de Parâmetros

A otimização de parâmetros de um sistema fuzzy pelo uso dos algoritmos genéticos pode ser realizada em duas instâncias do sistema (BONISSONE, KHEDKAR e CHEN, 1996):

- Na otimização da função de escalonamento das variáveis de entrada e saída do sistema; e
- No ajuste dos parâmetros que definem as funções de pertinência.

No primeiro caso, o AG atua sobre os elementos responsáveis pela normalização das variáveis de entrada e saída do sistema. O processo de otimização consiste em encontrar determinados parâmetros, com base no universo de discurso de cada uma das variáveis, adaptando os possíveis valores das variáveis de entrada e saída do sistema.

No ajuste de parâmetros que definem as funções de pertinência (FPs), o AG busca otimizar os parâmetros que codificam tais funções. Cada indivíduo da população é capaz de representar todas as FPs do sistema. A codificação é realizada por um único cromossomo, cujo número de genes é definido com base no tipo e no número de FPs, associadas às variáveis lingüísticas do sistema fuzzy. Normalmente, as FPs do tipo triangular são codificadas pelos valores dos seus três vértices, enquanto que as gaussianas são codificadas por dois atributos que definem o valor médio e o desvio padrão da função, por exemplo.

Observe o sistema com duas variáveis de entrada, X_1 e X_2, e uma variável de saída, Y, conforme apresentado na Figura 13. Podemos observar neste exemplo que as duas variáveis de entrada possuem três funções de pertinência cada e que a variável de saída possui quatro funções de pertinência. Todas as funções de pertinência são do tipo triangular.

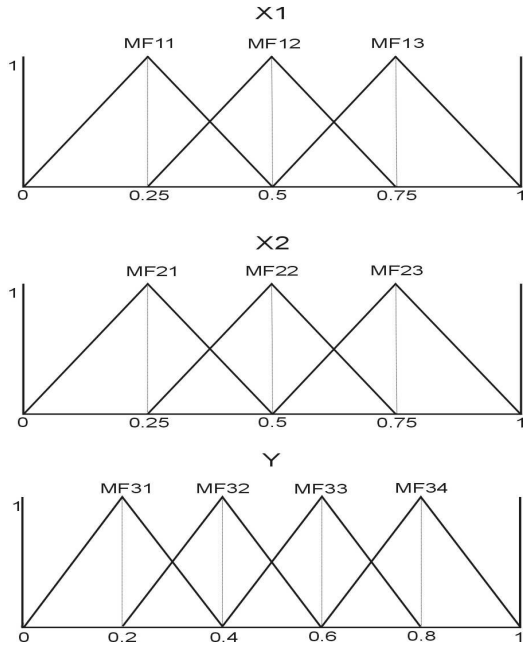

Figura 13 - Funções de pertinência de um sistema fuzzy.

Cada uma dessas funções de pertinência pode ser representada conforme o esquema abaixo:

MF_{11}: (0.0, 0.25, 0.5);	MF_{21}: (0.0, 0.25, 0.5);	MF_{31}: (0.0, 0.20, 0.40);
MF_{12}: (0.25, 0.5, 0.75);	MF_{22}: (0.25, 0.5, 0.75);	MF_{32}: (0.20, 0.40, 0.60);
MF_{13}: (0.5, 0.75, 1.0);	MF_{23}: (0.5, 0.75, 1.0);	MF_{33}: (0.40, 0.60, 0.80);
		MF_{34}: (0.60, 0.80, 1.00);

A Figura 14 apresenta o esquema de codificação do cromossomo para o caso do exemplo anterior.

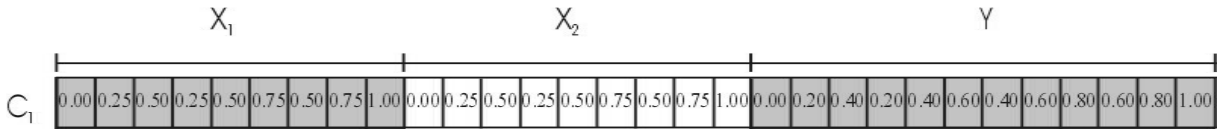

Figura 14 - Exemplo de codificação das FPs.

2.2.2 Aprendizagem da Base de Conhecimento

Os algoritmos genéticos podem ser utilizados para a aprendizagem parcial ou total da base de conhecimento de um sistema fuzzy. No processo de aprendizagem parcial, o AG considera uma base de dados já pré-definida, onde cada cromossomo deve codificar a base de

regras do sistema. A base de regras pode ser representada por uma matriz relacional, uma tabela de decisão ou por uma lista de regras. A Figura 15 apresenta um exemplo de uma matriz relacional com uma variável de entrada (X), uma variável de saída (Y), três FPs (A_1, A_2 e A_3) relacionadas à variável X e quatro FPs (B_1, B_2, B_3 e B_4) relacionadas à variável Y. A codificação da matriz relacional pode ser realizada através da representação binária (PHAM e KARABOGA, 1991) ou real (PARK, KANDEL e LANGHOLZ, 1994). Para o exemplo da matriz relacional R, da Figura 15, a codificação real do cromossomo é igual a C_1: (0.5, 0.6, 0.3, 0.0, 0.4, 0.3, 1.0, 0.8, 1.0, 0.0, 0.3, 1.0).

R	B_1	B_2	B_3	B_4
A_1	0.5	0.6	0.3	0.0
A_2	0.4	0.3	1.0	0.8
A_3	1.0	0.0	0.3	1.0

Figura 15 - Matriz Relacional (R)

Na aprendizagem total da base de conhecimento, tanto a base de regras quanto a base de dados devem ser desenvolvidas pelo algoritmo. Nesta última abordagem, a complexidade na utilização do algoritmo é superior à da aprendizagem parcial, uma vez que, nesse processo de aprendizagem, o AG deve ser capaz de lidar com um universo de busca heterogêneo, composto por cromossomos de diferentes tamanhos e representações.

O uso dos AGs para a aprendizagem da base de regras pode ocorrer com base em três diferentes metodologias (CORDÓN et al. 2001a): o Método de Michigan, o Método de Pittsburgh e o Método Interativo. Essas três abordagens são discutidas a seguir.

2.2.3 Sistema Fuzzy-Genético Baseado no Método de Michigan

Na utilização do Método de Michigan para o aprendizado da base de regras de um sistema fuzzy, cada indivíduo da população representa uma única regra. Assim, o algoritmo deve gerar uma população de regras que possibilite o aprendizado da tarefa a ser executada pelo sistema fuzzy, através da cooperação entre os indivíduos que constituem a base de regras do sistema. Devido à similaridade com os sistemas de classificação tradicionais (HOLLAND,

1976), esse método de aprendizado recebe a denominação de Sistema de Classificação Fuzzy (SCF) (CORDÓN et al., 2001a). A Figura 16 apresenta a estrutura geral do SFG baseado no Método de Michigan.

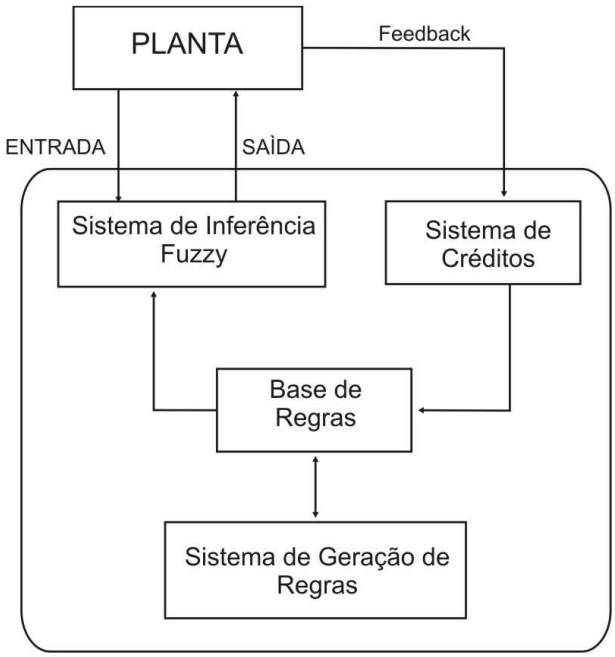

Figura 16 - Estrutura básica de um SFG baseado no Método de Michigan.

Inicialmente, todos os indivíduos da base de regras possuem o mesmo grau de aptidão. As regras ativadas durante a ação do sistema fuzzy serão recompensadas ou penalizadas pelo "Sistema de Créditos", através da comparação entre a resposta esperada e a resposta obtida pelo sistema. Dessa maneira, as regras terão seus valores de aptidão alterados proporcionalmente ao grau de contribuição de cada uma das regras, com relação à resposta obtida pela planta.

Após a execução de um determinado número de vezes, o AG é invocado com o objetivo de criar novas regras através das operações de *crossover* e mutação entre os indivíduos selecionados. As novas regras substituem parte da população atual (reprodução *steady-state*), na tentativa de proporcionar uma melhora no desempenho do sistema, através da eliminação das regras com valores de créditos mais baixos (menor aptidão).

Como a geração dessas novas regras pode ocasionar o surgimento de regras inválidas, isto é, regras cujo segmento de código faça referência a termos lingüísticos ou a variáveis inexistentes, o algoritmo deve verificar a sua validade antes de proceder à substituição de tais

regras. As regras consideradas inválidas serão descartadas e uma nova geração de regras será criada.

2.2.4 Sistema Fuzzy-Genético Baseado no Método de Pittsburgh

Ao contrário do método de Michigan, em que a solução é composta por toda a população de indivíduos, no método de Pittsburgh cada indivíduo da população representa a solução do sistema.

Essa característica possibilita que, nos SFGs baseados no método de Pittsburgh, o cromossomo codifique toda a base de conhecimento do sistema (base de dados + base de regras). A Figura 17 apresenta a estrutura geral de um SFG baseado no Método de Pittsburgh.

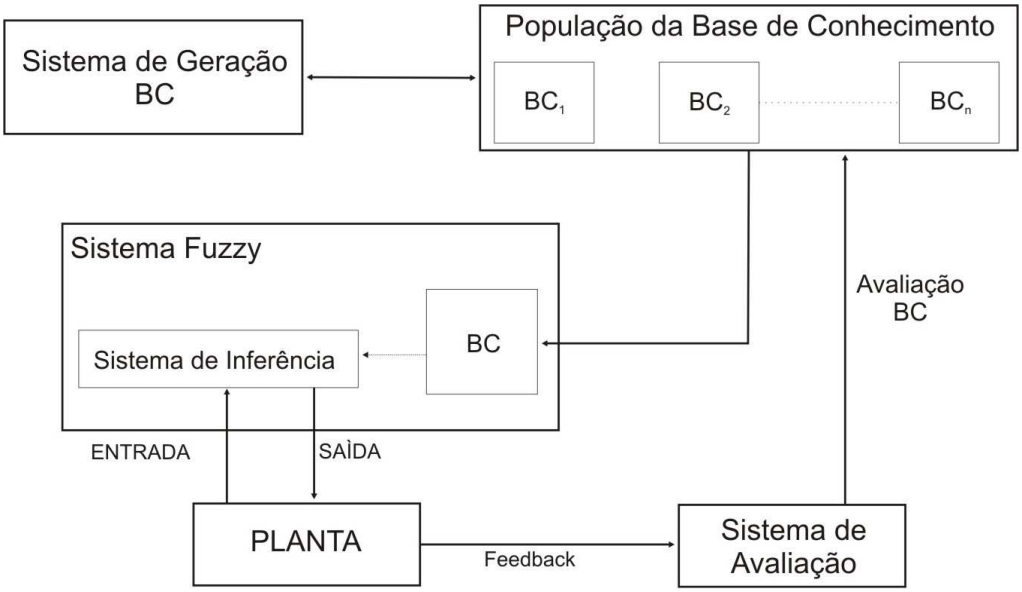

Figura 17 - Estrutura básica de um SFG baseado no Método de Pittsburgh.

A “População da Base de Conhecimento” contém os indivíduos que representam a base de conhecimento do sistema fuzzy. Para que o “Sistema de Avaliação” possa conceder os respectivos valores de aptidão a cada um dos indivíduos, o algoritmo deve ser executado *n* vezes para um mesmo valor de entrada, onde *n* representa o número de indivíduos da população. A aptidão de cada indivíduo pode ser calculada segundo uma das já tradicionais métricas de erro existentes na literatura, tomando por base o valor de *feedback* da planta e o valor esperado.

Após a avaliação dos indivíduos, o AG é invocado pelo “Sistema de Geração BC” para que uma nova geração de indivíduos seja criada. Como não existe interação entre os indivíduos com relação à resposta do sistema, toda a população de indivíduos com menor grau de aptidão pode ser substituída, sem que haja perda de desempenho.

A geração de novos indivíduos pode levar a uma discrepância entre a base de regras e a base de dados desse novo indivíduo. Assim como no método descrito anteriormente, o algoritmo deve possuir recursos para evitar ou corrigir tais distorções na base de conhecimento das novas gerações.

2.2.5 Sistema Fuzzy-Genético Baseado no Método Interativo

Usualmente, a Aprendizagem Interativa da base de regras do SFG ocorre em duas etapas distintas. Essa divisão busca solucionar o problema CCP, que será descrito na próxima seção.

Na primeira etapa, cada cromossomo codifica uma única regra, de maneira similar ao Método de Michigan. Porém, somente uma única regra é selecionada para compor a base de regras. Esse processo se repete até que a base de regras esteja completa ou que uma outra condição de finalização seja satisfeita.

Finda a primeira etapa, uma nova etapa de execução procura melhorar o desempenho da base de regras através da eliminação de regras desnecessárias e redundantes. Nesta etapa, as regras são “forçadas” a cooperarem entre si, num processo de refinamento e eliminação das regras consideradas excessivas. Ao final da execução, espera-se obter uma base de regras com um número menor de regras e que represente um ganho na performance do sistema.

2.2.6 Problema da Competição Versus Cooperação nos SFGs

Conforme (CORDÓN et al., 2001a), o principal problema no uso das técnicas evolucionárias para o desenvolvimento da base de conhecimento dos sistemas fuzzy deve-se aos seguintes fatos:

1. Nos sistemas fuzzy, a resposta a um determinado estado é obtida através da agregação das respostas individuais de cada uma das regras, cujos antecedentes

são ativados simultaneamente, em função dos valores das variáveis de entrada do sistema. Sendo assim, um sistema fuzzy que possui um bom desempenho é caracterizado por um conjunto de regras que cooperam entre si.

2. Os algoritmos evolucionários são caracterizados pela competição entre os membros da população, onde os indivíduos mais aptos sobrevivem em detrimento dos indivíduos menos aptos. Esta competição, induzida pela técnica dos algoritmos evolucionários, ocasiona o surgimento de melhores respostas à solução de um determinado problema, baseando-se no princípio da evolução natural.

Tal problema foi definido em (BONARINI, 1996) como o Problema da Competição versus a Cooperação (CCP – *Cooperation versus Competition Problem*). A sua solução está condicionada a um dos métodos de aprendizagem utilizado pelo SFG: Método de Michigan, Método de Pittsburgh ou Aprendizagem Interativa.

No Método de Michigan, a população é formada por regras que competem entre si, o que caracteriza o problema CCP. Segundo (BONARINI, 1996), num sistema fuzzy robusto as regras que cooperam entre si apresentam o mesmo antecedente. Sendo assim, a divisão da população em subgrupos (nichos), de acordo com a semelhança dos seus antecedentes, pode minimizar o problema CCP. As regras pertencentes a um determinado subgrupo competem entre si, enquanto que as regras que pertencem a diferentes subgrupos cooperam entre si para produzir a resposta final do sistema fuzzy.

Na abordagem Interativa, assim como no método de Michigan, as regras competem entre si, ocasionando o mesmo problema de competição descrito no método anterior. Para tentar solucionar tal problema, a execução do algoritmo é dividida em duas etapas distintas. Na primeira etapa, ocorre à competição entre as regras, enquanto que a segunda etapa procura resolver o problema CCP com base no estímulo à cooperação entre as regras.

No Método de Pittsburgh, cada indivíduo da população é constituído por um conjunto de regras. A aptidão de cada um dos membros da população é resultante da resposta obtida através da cooperação entre as suas próprias regras. Portanto, no método de Pittsburgh, o problema da cooperação versus a competição não existe.

Por evitar o problema da CCP, descrito anteriormente, e por possibilitar a representação de toda a base de conhecimento (base de dados e base de regras) de um sistema fuzzy num único cromossomo, o Método de Pittsburgh foi escolhido para ser utilizado na

metodologia desenvolvida neste trabalho. A descrição detalhada do método é apresentada no próximo capítulo.

2.3 Aprendizagem e Otimização dos SFGs

A aprendizagem automática da base de conhecimento dos sistemas fuzzy tem motivado diversas pesquisas relacionadas aos sistemas híbridos fuzzy-genéticos (SFG). Algumas dessas pesquisas têm demonstrado especial interesse na aprendizagem da base de conhecimento aliada à simplificação de tais sistemas.

Em (AMARAL, 2003) é proposto o desenvolvimento evolucionário de um sistema fuzzy através da técnica dos algoritmos genéticos, possibilitando que o sistema tenha a capacidade de aprendizado tanto da sua estrutura como dos seus parâmetros. O algoritmo proposto possui um caráter genérico, sendo capaz de criar e sintonizar um sistema do tipo Mamdani, cujo foco está relacionado à interpretabilidade do sistema fuzzy evoluído. Outros aspectos, como a variação dos operadores (*norma-t* e t-conorma) e dos métodos de defuzzificação são, também, abordados pelo algoritmo proposto.

A evolução de um sistema fuzzy que possua uma boa interpretabilidade é considerada de extrema importância. Para isso, o sistema fuzzy evoluído deve possuir uma base de regras com poucas regras e, conseqüentemente, apresentar um número reduzido de funções de pertinência. Considerando ainda o aspecto da interpretabilidade, o sistema fuzzy evoluído deve possuir um conjunto de funções de pertinência caracterizadas pela sua representatividade. Tal representatividade é garantida através da sobreposição controlada entre as funções de pertinência do sistema fuzzy (CASTRO, 2004) (DELGADO, 2002)(AMARAL,2003)

No algoritmo proposto para o desenvolvimento de tais sistemas híbridos fuzzy-genético definiram-se as seguintes características principais:

1. Interpretabilidade segundo o modelo de regras fuzzy;
2. Uma boa capacidade de se ajustar a um determinado conjunto de regras;
3. Permitir a criação da base de regras e o ajuste dos seus parâmetros;
4. O aprendizado de outras características importantes como a variação do número de funções de pertinência, a variação dos operadores *norma-t* e t-conorma e dos métodos de defuzzificação.

Analisando, resumidamente, o algoritmo evolucionário proposto em (AMARAL, 2003), podemos observar que o desenvolvimento dos sistemas fuzzy-genéticos ocorre em três etapas distintas dentro de um único ciclo de evolução.

Antes do início das três etapas que compõem o ciclo de evolução, as únicas características que o sistema apresenta são os números de funções de pertinência, definidas pelo projetista para cada uma das variáveis do sistema.

Na primeira etapa, os universos de discurso de cada uma das variáveis de entrada e saída são particionados de maneira uniforme, com base no número de funções de pertinência previamente definido. Esse particionamento pode ser realizado escolhendo-se um dos três possíveis tipos de funções de pertinência definidas: triangular, trapezoidal e gaussiana.

Na segunda etapa, a base de regras do sistema fuzzy é obtida através do uso dos algoritmos genéticos, bem como, os operadores e os métodos de defuzzificação são evoluídos simultaneamente. O sistema fuzzy apresenta as mesmas funções de pertinência da primeira etapa, com conjunto de regras, operadores e método de defuzzificação satisfatórios, de acordo com o critério estabelecido (AMARAL, 2003).

Na terceira e última etapa do ciclo evolucionário, realiza-se a sintonia dos parâmetros. Os algoritmos genéticos são utilizados para sintonizar os parâmetros das funções de pertinência, seguindo um critério de controle estabelecido pelo próprio projetista. Esse controle garante que as alterações ocorram dentro de uma determinada faixa especificada, sem que haja perdas relacionadas à representatividade de tais funções.

Ao final da terceira etapa, os critérios de parada são verificados. Caso não sejam satisfeitos, um novo ciclo evolucionário terá início e uma política de alteração deve ser aplicada. Neste caso, uma nova função de pertinência pode ser acrescentada a cada uma das variáveis do sistema, para, em seguida, iniciar-se um novo ciclo evolucionário, por exemplo.

A Figura 18 apresenta o fluxograma simplificado do algoritmo descrito anteriormente, como apresentado em (AMARAL, 2003).

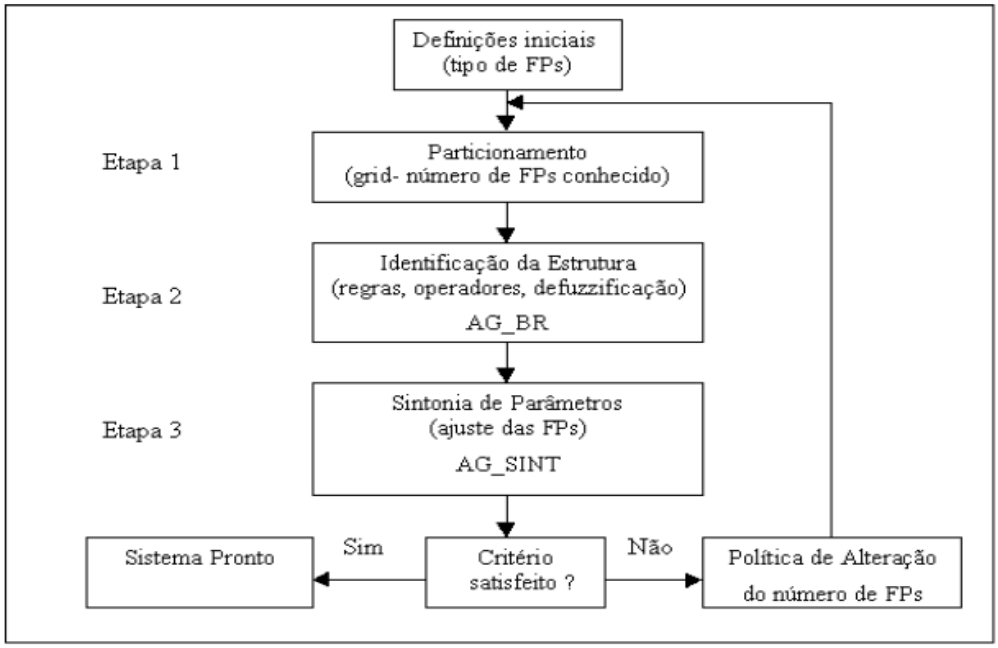

Figura 18 - Fluxograma simplificado do algoritmo de aprendizado (AMARAL, 2003).

No próximo capítulo, apresentamos o algoritmo evolucionário proposto neste trabalho para o desenvolvimento de um sistema híbrido do tipo fuzzy-genético, com capacidade de aprendizagem e simplificação da base de conhecimento, além da otimização de outros parâmetros do sistema.

Assim como a metodologia descrita neste tópico, o algoritmo evolucionário proposto baseia-se numa metodologia similar para o desenvolvimento e otimização da base de conhecimento de um sistema fuzzy, onde etapas distintas constituem o ciclo evolucionário.

3. PROJETO EVOLUTIVO DO SISTEMA FUZZY

Conforme dito anteriormente, uma das tarefas mais importantes e difíceis no desenvolvimento de um sistema fuzzy está relacionada à geração da base de conhecimento do sistema (CORDÓN e HERRERA, 2001).

Quando o conhecimento explícito dos especialistas humanos, necessário para a criação dos modelos fuzzy, não está disponível ou a sua interpretação é difícil de ser realizada, o desenvolvimento de tais modelos baseia-se na utilização de métodos de análise numéricos (informações objetivas sobre o problema). Nesse caso, a criação de tais modelos pode-se tornar uma tarefa árdua (AMARAL, 2003). Portanto, desenvolver um sistema fuzzy com capacidade de aprendizagem, através da análise automática das informações objetivas disponíveis, é de grande valia para os projetistas.

No presente trabalho o desenvolvimento dos sistemas fuzzy é realizado a partir da análise de dados numéricos e experimentais, valendo-se do uso da Computação Evolucionária para alcançar tal objetivo. Assim, a metodologia proposta utiliza a técnica dos algoritmos genéticos para desenvolver toda a base de conhecimento de um sistema fuzzy (base de regras e base de dados).

Além da geração da base de conhecimento, o ajuste de determinados parâmetros que possibilitam uma melhora no desempenho do sistema também é realizado pelo método proposto. Isso possibilita que o modelo obtido se ajuste da melhor maneira possível ao conjunto de dados disponíveis e analisados.

São utilizadas técnicas de aprendizagem e de otimização que facilitam o desenvolvimento de um sistema fuzzy com um bom desempenho e uma boa interpretabilidade lingüística da solução.

Conforme definido em outros estudos (KO et al., 2006) (AMARAL, 2001) (CORDÓN et al., 2001a), a interpretabilidade dos sistemas fuzzy pode ser classificada segundo dois fatores principais:

- Número de regras que compõem a base de regras, e
- Representatividade das funções de pertinência.

Sistemas fuzzy que possuem uma base de regras complexa têm a interpretabilidade prejudicada em virtude do número de regras apresentadas. O tamanho da base de regras é inversamente proporcional à interpretabilidade do modelo. A representatividade das funções

de pertinência também influencia diretamente na interpretabilidade do sistema. Neste trabalho a representatividade da base de dados está associada a dois parâmetros distintos (γ e κ), que definem, respectivamente, as seguintes propriedades (DELGADO, 2002):

- γ-completitude: Nesta propriedade fixa-se um grau mínimo (γ) de sobreposição entre as funções de pertinência para evitar falhas no particionamento do universo de discurso.
- κ-sobreposição: Um grau máximo (κ) de sobreposição na partição do universo de discurso é definido para evitar funções de pertinência completamente sobrepostas.

A Figura 19(A) apresenta um exemplo da utilização dos parâmetros γ e κ na definição da representatividade das funções de pertinência. Na Figura 19(B) é apresentado um exemplo onde interpretabilidade da base de dados é prejudicada em função da sobreposição exagerada entre as FPs e pela falha no particionamento do universo de discurso da variável *x*.

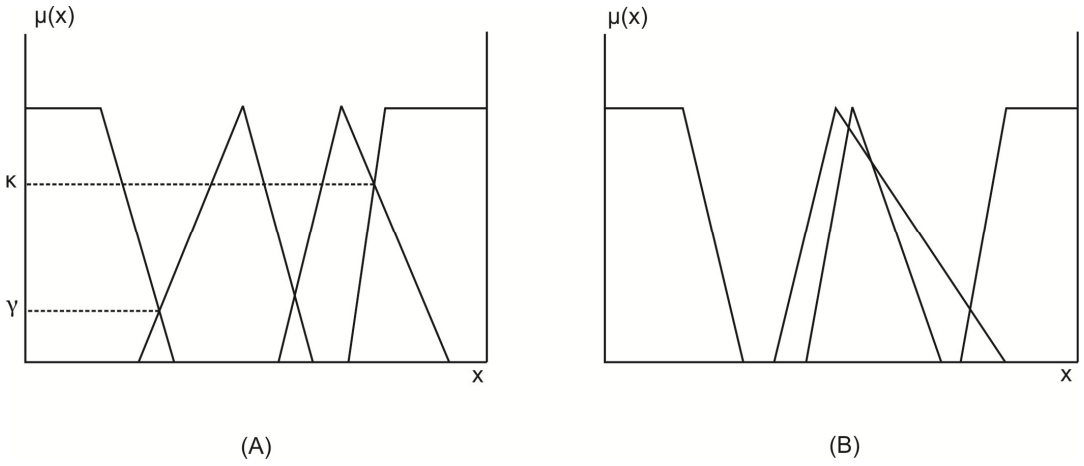

Figura 19 - Análise da representatividade das FPs. (A) Particionamento interpretável. (B) Particionamento não interpretável.

Por se tratar de um problema de otimização multiobjetivo, a presente metodologia requer o uso de técnicas apropriadas que permitam avaliar corretamente cada solução, levando-se em consideração os dois objetivos descritos anteriormente. Por propiciar que a análise de problemas multiobjetivo seja realizada de maneira eficiente e por facilitar o ajuste e a compreensão do problema envolvido, uma nova técnica baseada na lógica fuzzy foi desenvolvida e incorporada à função de avaliação do AG durante a fase de aprendizagem do sistema fuzzy.

Este capítulo apresenta a metodologia utilizada para a geração e otimização da base de conhecimento de um sistema híbrido fuzzy-genético. Sua execução é realizada um duas e etapas distintas: 1) geração da base de regras e da base de dados, e 2) otimização dos parâmetros relacionados às funções de pertinência de cada variável do sistema. Nas próximas seções, as etapas relacionadas à metodologia, bem como o método de análise multiobjetivo utilizando um sistema fuzzy são apresentados.

3.1 Características do Sistema Fuzzy

Um sistema do tipo MISO (*Multiple Input Single Output*) é capaz de mapear o espaço n-dimensional das variáveis de entrada para o espaço unidimensional da variável de saída. O uso da lógica fuzzy possibilita que esse mapeamento seja realizado de maneira simples, especialmente quando envolve sistemas complexos e funções não lineares muito complexas (AMARAL, 2003). Observação semelhante pode ser feita com relação ao uso da lógica fuzzy em sistemas do tipo MIMO (*Multiple Input Multiple Outpult*), onde os espaços n-dimensionais das variáveis de entrada e saída necessitam ser mapeados.

A metodologia proposta desenvolve um sistema fuzzy com aprendizado genético com múltiplas entradas e uma única saída (MISO), ou com múltiplas entradas e múltiplas saídas (MIMO). O sistema fuzzy desenvolvido é do tipo Mamdani, pois sua interpretação é mais direta do que a dos sistemas do tipo TSK (Takagi-Sugeno-Kang). Pelo mesmo motivo, todas as regras desenvolvidas durante o processo de aprendizagem possuem o mesmo grau de importância, isto é, todas as regras possuem peso igual a um.

Resumidamente, podemos definir as características do sistema fuzzy proposto da seguinte forma:

1. Tipo Mamdani.
2. Particionamento de entrada e saída do tipo fuzzy grid adaptativo.
3. Tipos de funções de pertinência: triangular, trapezoidal e gaussiana.
4. Operadores *norma-t* (min e prod).
5. Método de Defuzzificação: centro de gravidade.
6. Objetivos: Desempenho e Interpretabilidade.
7. Aprendizado baseado em Algoritmos Genéticos.

Vale ressaltar que a metodologia proposta visa a servir como uma ferramenta de auxílio aos projetistas de sistemas fuzzy. Como em qualquer outra técnica de aprendizado existente, a análise e validação do projetista não podem ser descartadas, uma vez que nem sempre é possível garantir um treinamento bem sucedido decorrente da utilização do algoritmo de aprendizagem.

3.2 Algoritmo de Aprendizagem

Nessa seção é apresentada a metodologia proposta para o desenvolvimento e a otimização da base de conhecimento de um sistema fuzzy, a partir da análise numérica ou experimental. Portanto, é necessário que um conjunto de exemplos, ou a representação matemática do problema, esteja disponível para análise e validação das respostas obtidas.

O algoritmo de aprendizagem desenvolvido neste trabalho utiliza a técnica dos algoritmos genéticos baseada no Método de Pittsburgh, sendo composto de duas etapas distintas:

- Etapa 1 - Geração da base de conhecimento (base de dados + base de regas): O AG gera os sistemas fuzzy com diferentes representações semânticas para as variáveis do sistema e as suas respectivas bases de regras.
- Etapa 2 - Otimização das funções de pertinência: O AG procura otimizar os parâmetros das funções de pertinência do melhor indivíduo selecionado ao término da etapa anterior. O objetivo é melhorar a performance do sistema fuzzy desenvolvido.

Embora ambas as etapas façam uso da técnica dos algoritmos genéticos para o desenvolvimento do sistema fuzzy, elas são independentes entre si. Inclusive, é possível que uma destas etapas deixe de ser executada ou que elas sejam aplicadas em separado. Neste último caso, uma outra metodologia pode ser empregada em conjunto com uma das duas etapas existentes.

A escolha do Método de Pittsburgh para o desenvolvimento do SFG possibilita que tanto a base de regras quanto a base de dados sejam evoluídas simultaneamente, sendo ambas codificadas num único cromossomo. Além disso, o uso do método de Pittsburgh evita o problema de competição individual entre as regras (CCP – *Cooperation versus Competition Problem*), descrito anteriormente no Capítulo 2.

Em cada uma das etapas, o AG utilizado apresenta o seu tipo de codificação, seus operadores e as suas funções de avaliação. Como na primeira etapa os indivíduos são avaliados segundo dois objetivos distintos (interpretabilidade e desempenho), uma nova técnica que utiliza um sistema fuzzy para a avaliação multiobjetivo é incorporada ao AG. A esta nova técnica damos o nome de Agregador Fuzzy. Para a segunda etapa, a função de avaliação se baseia numa das já tradicionais métricas de erro existentes, como o erro médio (*MSE – Mean Square Error*) ou erro médio quadrático (*RMSE – Root Mean Square Error*), por exemplo. Um fluxograma simplificado das duas etapas de execução do algoritmo pode ser observado na Figura 20.

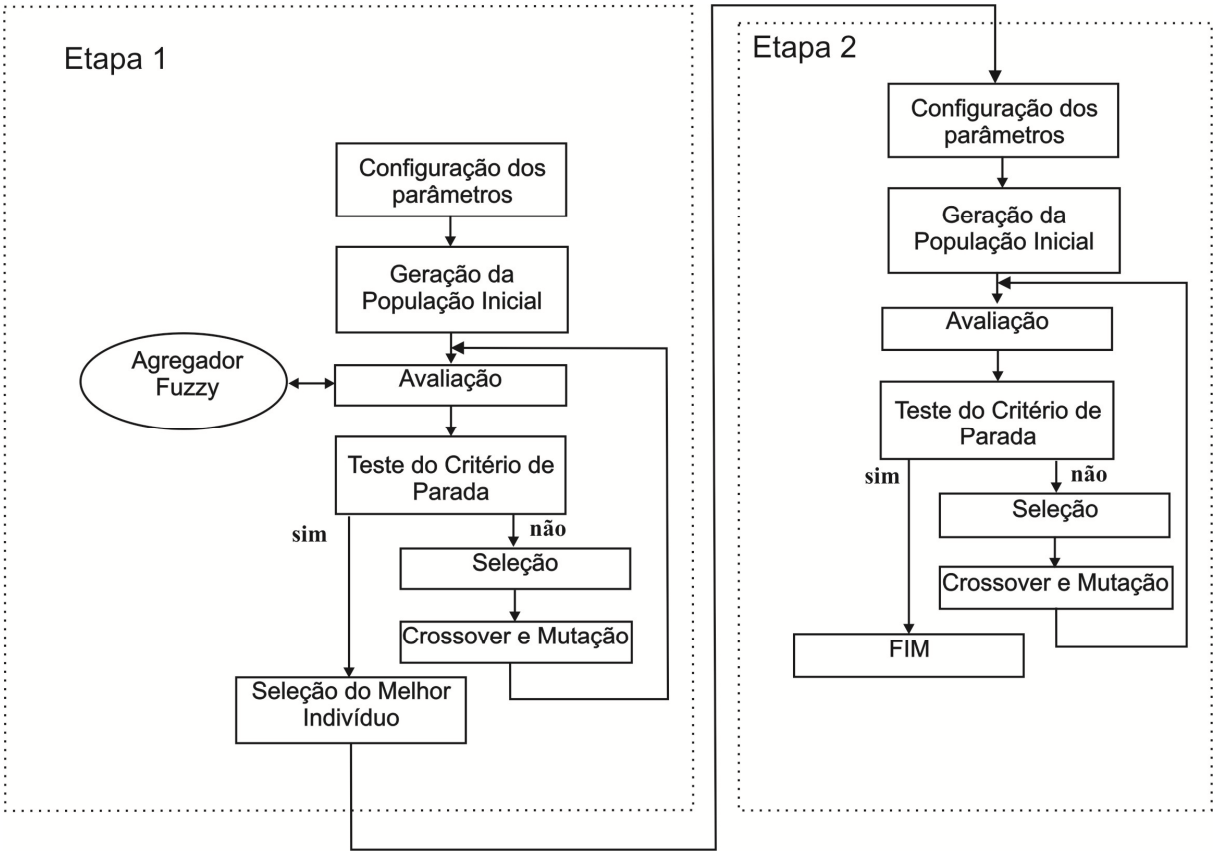

Figura 20 - Fluxograma de Execução do Algoritmo.

3.2.1 Etapa 1 - Geração da base de conhecimento

Nesta etapa, o AG é responsável por evoluir uma população de indivíduos, onde cada indivíduo representa a base de conhecimento de um sistema fuzzy. O algoritmo obedece ao seguinte esquema:

1. Configuração dos parâmetros gerais. Antes da execução do algoritmo, o projetista do sistema deve definir alguns parâmetros que serão utilizados durante todo o processo de evolução: número de variáveis do sistema, número mínimo e máximo de FPs por variável, tipo de função de pertinência, número de indivíduos e configuração dos operadores utilizados pelo AG.
2. Geração de uma população inicial de N indivíduos.
3. Utilização do Agregador Fuzzy para avaliar o desempenho do sistema, com base na comparação dos exemplos ou no comportamento das respostas do modelo utilizado, considerando, também, a interpretabilidade da base de regras evoluída.
4. Verificação do critério de parada: satisfeito o critério de parada especificado pelo projetista (número máximo de gerações, saturação da população, etc.), é retornado como resultado final da execução desta etapa o indivíduo que representa a melhor base de conhecimento para o sistema. Caso o critério de parada não seja atendido, o algoritmo prossegue sua execução.
5. Aplicar os operadores genéticos: seleção, *crossover* e mutação.
6. Retornar ao passo três.

- Codificação da Base de Conhecimento

Como a abordagem de Pittsburgh é adotada, a base de dados e a base de regras são codificadas em um único cromossomo. Isso possibilita que ambas sejam evoluídas e avaliadas simultaneamente. Além da codificação da base de dados e da base de regras, um terceiro elemento referente ou tipo de operador da *norma-t* (*mínimo* ou *produto*) também é codificado no cromossomo. Um exemplo da representação do cromossomo de um sistema MISO contendo estas três estruturas é mostrado na Figura 21.

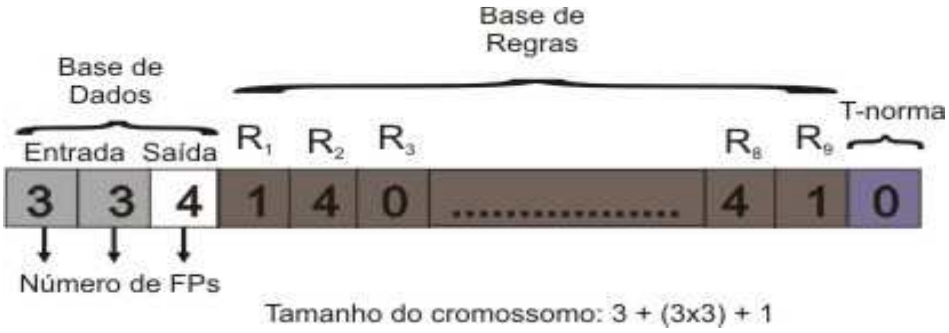

Figura 21 - Codificação do cromossomo (Base de Dados e Base de Regras).

Conforme observado na figura acima, o número de funções de pertinência e o conjunto de regras do sistema são codificados através da representação inteira dos seus parâmetros. O uso da representação inteira possibilita uma redução no tamanho do cromossomo, quando comparado ao tamanho do cromossomo obtido pelo uso da representação binária. Conseqüentemente, a representação inteira requer um esforço computacional menor.

A codificação do tipo de operador utilizado pela *norma-t* é realizado por um único gene, onde o valor zero (0) indica a utilização do operador *mínimo*, enquanto que o valor um (1) indica a utilização do operador *produto*.

O número de genes necessários para a codificação da base de dados é igual ao número de variáveis de entrada e saída do sistema fuzzy. No exemplo anterior, um sistema fuzzy do tipo MISO, com duas entradas e uma única saída, tem a sua base de dados codificada pelo uso de três genes, um para cada variável. Caso o sistema apresente mais entradas, ou o sistema evoluído seja do tipo MIMO, será acrescentado um número de genes de maneira proporcional ao número de variáveis de entrada e/ou saída do sistema.

Com base nas informações contidas em cada um dos genes, e na informação prévia do tipo de função de pertinência (triangular, trapezoidal ou gaussiana) definido pelo projetista, o algoritmo particiona uniformemente o universo de discurso de cada uma das variáveis. Cabe ressaltar que, no caso das funções triangulares, as FPs localizadas nos extremos inferior e superior do universo de discurso de cada variável serão sempre do tipo trapezoidal.

Para a codificação da base de regras, o número de genes necessários dependerá do número de FPs relacionado a cada uma das variáveis de entrada do sistema. Ainda utilizando a Figura 21 como exemplo, podemos observar que as duas variáveis de entrada do sistema utilizam três funções de pertinência cada. Assim, o número máximo de regras deve ser igual a nove. Como cada gene da base de regras representa uma única regra, podemos concluir que o

número de genes necessários para a codificação da base de regras também deve ser igual a nove.

O número de regras de um sistema fuzzy está diretamente relacionado ao número de funções de pertinência das variáveis de entrada. Por exemplo, num sistema com duas entradas, onde o espaço de cada variável é particionado por três e quatro funções de pertinência, há 3 * 4 = 12 regras possíveis.

Para propiciar uma melhora na interpretabilidade do sistema fuzzy, a metodologia proposta neste trabalho permite que a base de regras do sistema seja constituída por um número reduzido de regras. Essa redução no número de regras torna-se possível pela inclusão de um termo nulo no conseqüente da regra. Regras com um termo nulo no conseqüente são excluídas da base de regras. Considerando que no exemplo anterior existam quatro funções de pertinência relacionadas à variável de saída do sistema fuzzy, o conseqüente de cada regra pode assumir um valor inteiro compreendido no intervalo *C*, onde *C* = [0,4]. A Figura 22 apresenta a codificação da base de regras e as regras para o exemplo apresentado.

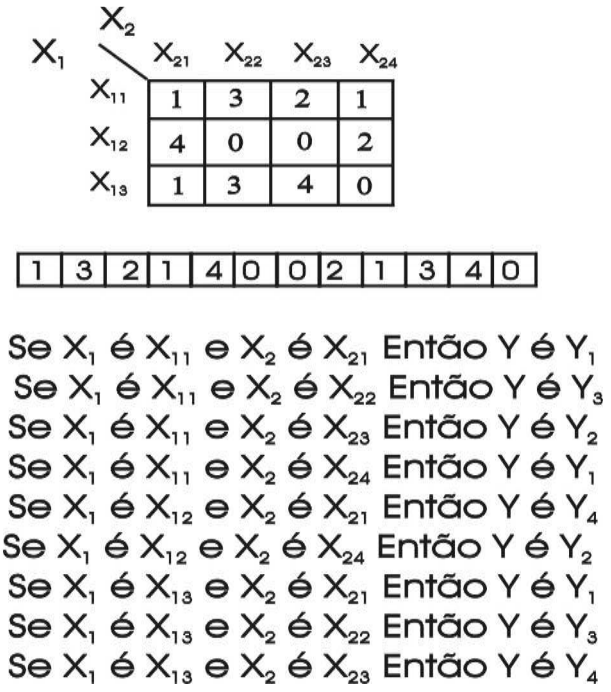

Figura 22 - Exemplo da codificação da base de regras.

- População Inicial

Antes da execução da primeira etapa do algoritmo, o projetista deve indicar o número mínimo e máximo de funções de pertinência associadas a cada uma das variáveis do sistema, bem como o tipo de função de pertinência que será utilizado (triangular, trapezoidal ou

gaussiana). A partir desses valores, o algoritmo define, de maneira aleatória, a estrutura relacionada ao número de FPs para cada um dos indivíduos da população. Definido este número, utiliza-se o método fuzzy-grid para particionar uniformemente o universo de discurso de cada uma das variáveis.

A escolha aleatória do número de FPs propicia o surgimento de cromossomos com diferentes tamanhos, uma vez que, conforme descrito no tópico anterior, o número de genes necessários a codificação da base de regras está diretamente relacionado ao número de FPs atribuído a cada uma das variáveis de entrada do sistema.

Esta diversidade genética torna o espaço de busca mais amplo, possibilitando que respostas consideradas ótimas sejam alcançadas. Porém, tal diversidade exige que o AG incorpore funcionalidades que evitem o surgimento de representações discrepantes entre a base de regras e a base de dados, após as operações de *crossover* e mutação entre os indivíduos da população. Este assunto será abordado em maiores detalhes mais adiante durante a descrição dos operadores genéticos.

- Função de Aptidão

Nesta etapa de execução do algoritmo, os indivíduos são avaliados segundo dois objetivos distintos: desempenho e interpretabilidade. A avaliação multiobjetivo, empregada com base na técnica denominada Agregador Fuzzy, é descrita em maiores detalhes na próxima seção deste capítulo.

- Operadores Genéticos

O operador de seleção utilizado pelo AG neste trabalho baseia-se no método da roleta. A probabilidade com que cada indivíduo da população pode ser selecionado está definido pela fórmula:

$$p_i = \frac{f_i}{\Sigma f} \qquad (10)$$

Onde, $\mathbf{p_i}$ é a probabilidade do indivíduo **i** ser selecionado, $\mathbf{f_i}$ é o grau de aptidão desse indivíduo e $\sum \mathbf{f}$ representa a soma da aptidão de todos os indivíduos da população atual.

Para evitar a convergência prematura da população em função do surgimento dos *super-indivíduos*, adotou-se a normalização linear dos valores de aptidão (*f'*), definido por (GOLDBERG, 1989):

$$f' = af + b \qquad (11)$$

Os coeficientes a e b podem ser definidos de diferentes maneiras. A Figura 23 apresenta graficamente os valores de aptidão após o processo de normalização linear. O valor de f'_{avg} deve ser igual ao valor de f_{avg} para garantir que os indivíduos medianos contribuam com um descendente nas gerações futuras (GOLDBERG, 1989). Para evitar o surgimento de valores negativos após o processo de normalização, o menor valor que f'_{min} pode assumir deve ser igual a zero (0).

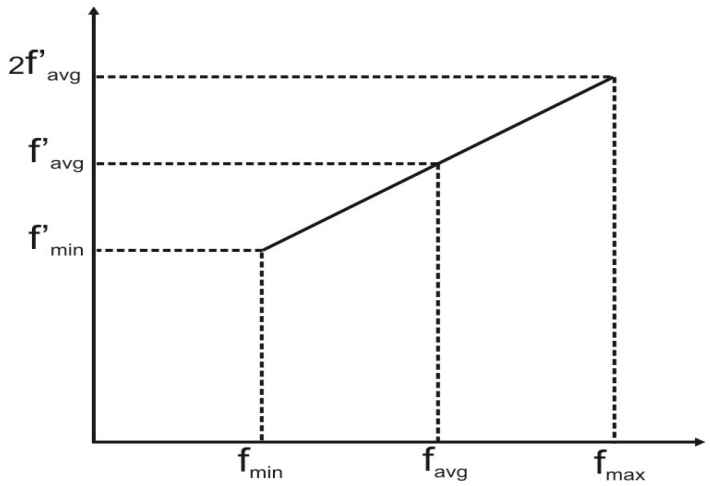

Figura 23 - Normalização linear dos valores de aptidão.

Juntamente com o método de seleção descrito, adotou-se o conceito de elitismo, em que o melhor indivíduo dentre todas as gerações é mantido e selecionado para a geração futura. Isso evita que a melhor solução seja perdida durante o processo de seleção.

Após a seleção, a operação de *crossover* é executada. Conforme observado anteriormente, o cromossomo codifica a base de conhecimento através da união de duas estruturas distintas, que codificam tanto a base de dados quanto a base de regras de um sistema fuzzy. Além disso, existe uma terceira estrutura, composta por um único gene, que serve para indicar o tipo de operador utilizado para *norma-t*. Nesse tipo de configuração, o operador de *crossover* atua de maneira independente nas três seções do cromossomo. Portanto, os indivíduos selecionados trocarão as informações associadas à base de dados, à base de regras e ao tipo de operador (*mínimo* ou *produto*) de maneira independente. A Figura 24 representa um exemplo da operação de *crossover* entre dois indivíduos da população.

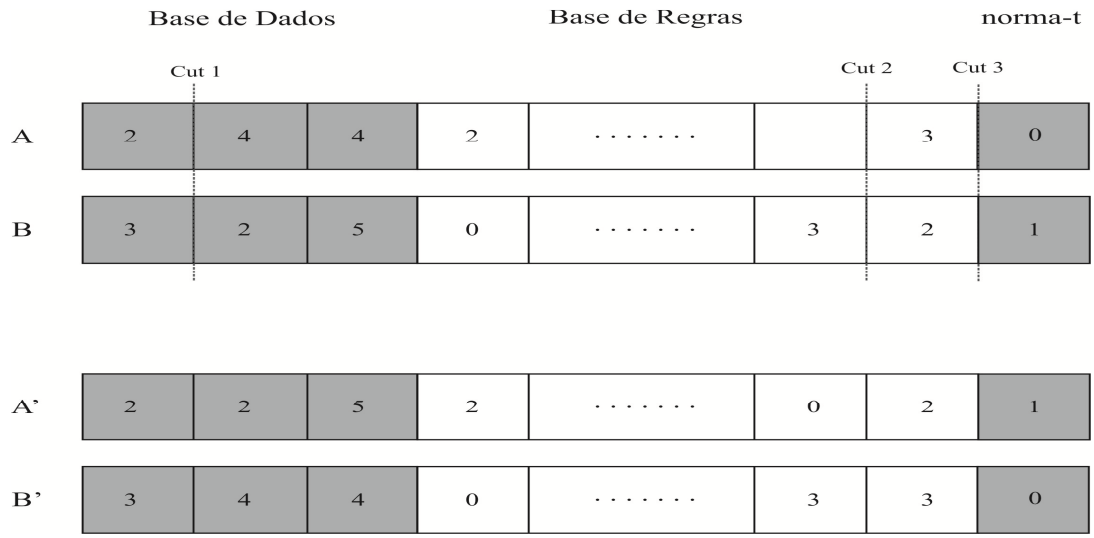

Figura 24 - Exemplo da Operação de *crossover*.

A última operação a ser realizada é a mutação, cuja probabilidade de ocorrência é verificada sobre cada um dos genes do cromossomo. Ocorrendo a mutação, os valores dos genes serão aleatoriamente acrescentados ou subtraídos de uma unidade. Caso o gene já apresente o valor mínimo definido, o operador de mutação poderá somente acrescentar uma unidade a esse valor. O inverso ocorre para o caso em que o gene já apresenta o valor máximo possível.

Conforme descrito anteriormente, a operação de *crossover* e mutação nos genes que constituem a base de conhecimento do sistema possibilitam a alteração no número de funções de pertinência dos indivíduos da população. Isso proporciona a exploração de novas áreas, além daquelas definidas durante a criação da população inicial. Porém, quando tal fenômeno ocorre, é necessário que o algoritmo garanta a integridade da base de regras relacionada a esta nova base de dados. Assim, o valor de cada gene da base de regras é verificado e modificado no caso de incoerências; novos genes podem ser acrescentados ou eliminados do cromossomo para manter a coerência entre a base de dados e a base de regras do sistema.

- Critério de Parada

O projetista do sistema pode adotar como critério de parada o número máximo de gerações ou definir como ponto de saturação um número máximo de gerações em que não haja melhora na aptidão da população.

Após a parada, o melhor indivíduo da população é retornado. Neste ponto, a execução do algoritmo pode ser interrompida ou, caso contrário, terá início a execução da segunda etapa do algoritmo.

3.2.2 Etapa 2 - Otimização das funções de pertinência

Durante a primeira etapa de execução do algoritmo, a base de conhecimento e o tipo de operador da *norma-t* foram evoluídos de maneira a encontrar a melhor solução que representasse um sistema fuzzy, com base em dois objetivos distintos: desempenho e interpretabilidade.

A segunda etapa visa a otimizar os parâmetros das funções de pertinência da melhor solução apresentada ao final da execução da etapa anterior. Assim, esta etapa é caracterizada pela otimização da base de conhecimento de um sistema fuzzy.

Na primeira etapa de execução a interpretabilidade da base de dados foi garantida através do particionamento uniforme do universo de discurso das variáveis. Para que durante o processo de otimização da base de dados as funções de pertinência mantenham a sua representatividade, deve-se estabelecer um limite mínimo (γ) e máximo (κ) para as sobreposições entre as FPs. Os limites (γ e κ) estão relacionados às propriedades de γ-completitude e κ-sobreposição descritas anteriormente no início deste capítulo. Os valores de tais parâmetros devem ser especificados pelo projetista ao início da execução da segunda etapa.

Nesta etapa, o algoritmo obedece ao seguinte esquema:

1. Configuração dos parâmetros gerais do algoritmo, tais como: valores de γ e κ; tamanho da população, critério de parada e configuração dos operadores genéticos.
2. Geração uma população inicial de N indivíduos com base no melhor indivíduo selecionado na etapa anterior.
3. Avaliação do desempenho do sistema com base na comparação dos exemplos ou no comportamento da resposta do modelo utilizado.
4. Caso seja atingido o critério de parada, o indivíduo com maior valor de aptidão é retornado, dando fim ao ciclo evolucionário. Caso contrário, o algoritmo continua a execução para o passo seguinte.
5. Aplicar os operadores genéticos: seleção, *crossover* e mutação.
6. Retornar ao passo três.

- Codificação da Base de Dados

As funções de pertinência são representadas através da codificação real dos seus parâmetros. O número de parâmetros necessários para a representação das funções de pertinência varia conforme o tipo definido (triangular, trapezoidal ou gaussiana). Conseqüentemente, o tamanho do cromossomo irá variar conforme o tipo de função de pertinência definido pelo projetista. Para o caso das funções triangulares, a Figura 25 apresenta um exemplo da estrutura cromossômica utilizada pelo algoritmo:

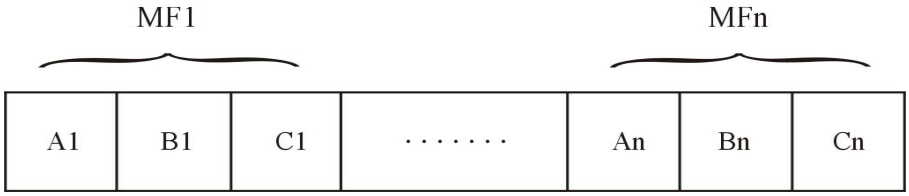

Figura 25 - Modelo da representação real das funções de pertinência do tipo triangular.

Os parâmetros A_i,B_i e C_i representam os vértices da função de pertinência n_i de uma determinada variável.

- População Inicial

A população inicial é criada tomando-se por base o indivíduo selecionado ao final da execução da primeira etapa. Este indivíduo serve como matriz para a geração da população inicial. A base de regras e o tipo de operador utilizado pela *norma-t* são idênticos em todos os indivíduos da população. A diferença entre os indivíduos está relacionada na maneira como o universo de discurso de cada variável está particionado.

Um número N-1 de indivíduos é criado a partir de variações nos parâmetros das FPs do indivíduo matriz, onde N é o tamanho da população. As variações nos parâmetros são realizadas de maneira aleatória pelo algoritmo, respeitando sempre o limite mínimo e o limite máximo para as sobreposições definidos pelos valores de γ e κ, respectivamente.

- Função de Aptidão

Nesta etapa de execução do algoritmo os indivíduos são avaliados segundo o desempenho apresentado. Para isso, uma das já tradicionais métricas de erro entre a resposta obtida e a resposta desejada pode ser utilizada, como, por exemplo, o erro médio (MSE) e o erro médio quadrático (RMSE).

- Operadores Genéticos

O primeiro operador genético a ser empregado é a seleção baseado no método da roleta. Como os indivíduos apresentam valores de aptidão muito próximos nesta etapa de execução, não há necessidade do uso da normalização do valor de aptidão.

Foram utilizados o operador de *crossover* de um ponto e o operador de mutação para valores reais. Após as operações de *crossover* e de mutação, o algoritmo avalia a estrutura do cromossomo e aplica as correções necessárias para manter a coerência entre os parâmetros que definem as FPs.

- Critério de Parada

Para estabelecer o fim da execução do algoritmo, os mesmos critérios de parada disponíveis na primeira etapa de execução podem ser utilizados. Ao fim da execução desta etapa, o melhor indivíduo da população é retornado.

3.3 Agregador Fuzzy

Um dos maiores desafios para a aplicação de algoritmos genéticos em problemas reais refere-se à concepção de estratégias de avaliação de aptidão, na qual diversos objetivos são levados em consideração (ZEBULUM, 1999). Essa dificuldade deve-se ao fato dos AGs utilizarem um valor escalar para determinar o grau de aptidão dos indivíduos de uma população, enquanto que nos problemas multiobjetivo, um “vetor de aptidões” associado a diversos objetivos representa a resposta ao problema.

Portanto, faz-se necessário o uso de métodos que possibilitem transformar esse “vetor de aptidões” em um escalar. A literatura identifica duas formas principais para a solução de problemas multiobjetivo por computação evolutiva: Conjunto Ótimo de Pareto (*Pareto Optimal Set*) e Agregação de Objetivos (FONSECA et al., 2003).

O Conjunto Ótimo de Pareto baseia-se no conceito de dominância, onde uma solução $x^{(1)}$ é dita dominante com relação a $x^{(2)}$ se ambas as condições (1) e (2) forem satisfeitas (DEB, 2004):

1. A solução $x^{(1)}$ não é pior do que $x^{(2)}$ em todos os objetivos analisados.
2. A solução $x^{(1)}$ é melhor do que a solução $x^{(2)}$ em pelo menos um único objetivo.

Do ponto de vista dos AGs, caso um indivíduo não seja dominado por nenhum outro indivíduo da mesma população, ele fará parte do conjunto Ótimo de Pareto. Os indivíduos pertencentes a esse conjunto terão maiores chances de ser selecionados para as próximas gerações, uma vez que a eles são atribuídos os maiores valores de aptidão.

O maior problema desta estratégia é o fato de não permitir uma forma direta de incorporar especificações do usuário na medida de aptidão, porque a medida de aptidão compara o desempenho do indivíduo em relação à população, e não em relação a especificações desejadas pelo usuário (ZEBULUM, 1999).

A estratégia por agregação de objetivos consiste na atribuição de uma medida escalar de aptidão, através do cálculo da média ponderada em relação a cada objetivo (FONSECA e FLEMING, 1995).

Apesar de ser uma das maneiras mais simples de se resolver um problema multiobjetivo, esta estratégia apresenta como grande desafio a escolha adequada dos pesos que devem ser atribuídos a cada um dos objetivos. Essa dificuldade torna-se ainda mais evidente quando o problema apresenta características não-lineares (DEB, 2004).

Alguns estudos demonstram que o uso da Lógica Fuzzy (LF) em problemas multiobjetivo tem apresentado bons resultados (AUCHARIYAMET e SIRISUMRANNUKUL, 2009) (MARWAHA et al., 2004) (SAGHIRI e ZARANDI, 2003). Nos problemas multiobjetivo complexos, as interações entre os diferentes objetivos são baseadas em informações incertas e vagas. Assim, a LF apresenta algumas vantagens quando comparada aos métodos tradicionais de solução, pois é capaz de lidar com informações imprecisas e vagas de maneira eficiente.

Além disso, o uso da lógica fuzzy possibilita que o problema seja compreendido de maneira mais simples, além de apresentar outras vantagens: maior flexibilidade, modelagem de problemas complexos pelo uso da linguagem natural, possibilidade de incorporar as especificações do projetista na busca pelas melhores soluções, etc.

Em (AMARAL, 2003) o autor sugere a utilização de um sistema fuzzy para agregar dois objetivos (erro da resposta do sistema e número de regras do sistema) a serem otimizados pela técnica dos AGs, tendo como saída à aptidão final do indivíduo. Nesse trabalho, optou-se por utilizar uma técnica semelhante, denominada Agregador Fuzzy, onde a LF é utilizada para converter um problema multiobjetivo em um problema com um único objetivo, capaz de ser facilmente solucionado e interpretado.

Conforme descrito anteriormente, durante a primeira etapa de execução do algoritmo a base de conhecimento de um sistema fuzzy deve ser evoluída atendendo a dois objetivos

distintos: desempenho e interpretabilidade. Assim, o Agregador Fuzzy é utilizado para avaliar cada indivíduo da população, com base no erro médio quadrático da resposta e na razão entre o número de regras existente e o número máximo de possíveis regras, retornando um valor escalar como medida de aptidão. A Figura 26 apresenta esquematicamente o mecanismo de avaliação do Agregador Fuzzy.

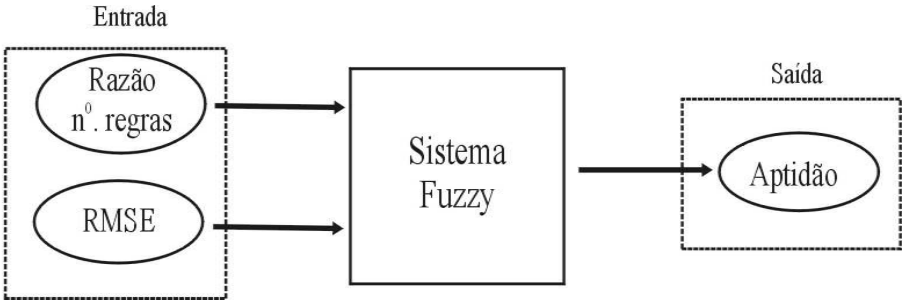

Figura 26 – Técnica fuzzy para avaliação de problemas multiobjetivo.

O uso desta técnica possibilita que o problema de otimização multiobjetivo seja abordado de maneira adequada, especialmente quando comparado às técnicas tradicionais de otimização utilizadas na resolução de problemas semelhantes. Além disso, sua implementação é rápida, possibilitando ao projetista realizar alterações de maneira simples, o que facilita a adaptação do algoritmo para melhor atender às suas necessidades.

No algoritmo de evolução, o modelo fuzzy para avaliação multiobjetivo foi desenvolvido utilizando-se o *toolbox Fuzzy Logic* para o ambiente *Matlab*. Na Figura 27 podemos observar o esquema geral de um modelo fuzzy utilizado em alguns experimentos desse trabalho. A Figura 28 apresenta as funções de pertinência, enquanto a Tabela 1 descreve a base de regras do modelo, ambas obtidas empiricamente durante a fase de testes realizados com o algoritmo.

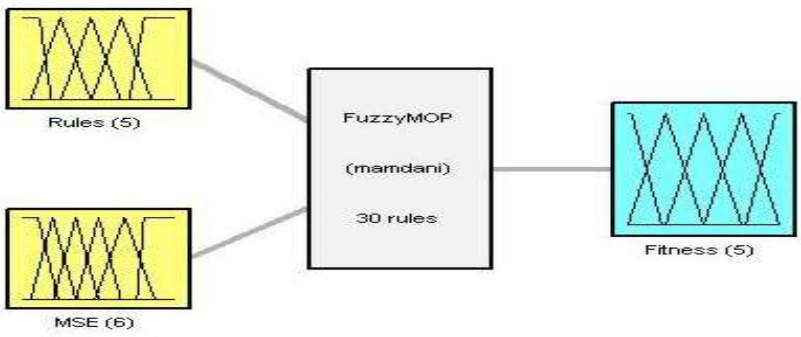

Figura 27- Modelo fuzzy para avaliação MO desenvolvido no *toolbox Fuzzy Logic*.

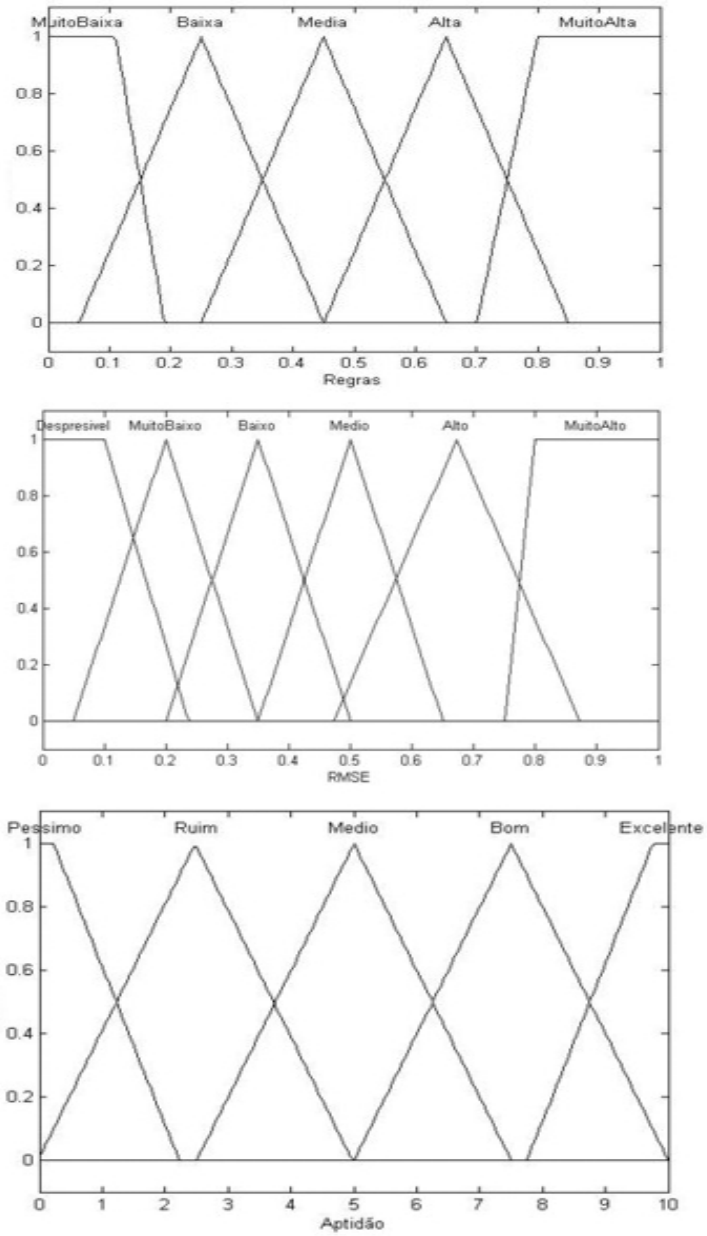

Figura 28 – Funções de pertinência do modelo fuzzy para avaliação MO.

Tabela 1- Base de regras do sistema fuzzy MO.

		Razão entre o nº. de regras				
		Muito Baixa	Baixa	Média	Alta	Muito Alta
RMSE	Desprezível	Excelente	Excelente	Excelente	Bom	Médio
	Muito Baixo	Excelente	Bom	Bom	Bom	Médio
	Baixo	Bom	Bom	Médio	Médio	Ruim
	Médio	Médio	Médio	Médio	Ruim	Ruim
	Alto	Ruim	Ruim	Péssimo	Péssimo	Péssimo
	Muito Alto	Péssimo	Péssimo	Péssimo	Péssimo	Péssimo

No próximo capítulo são apresentados os estudos de caso utilizando a metodologia proposta, para o desenvolvimento de sistemas híbridos do tipo fuzzy-genéticos em diferentes tipos de aplicações.

4. ESTUDOS DE CASOS

Com o objetivo de avaliar a proposta apresentada, foram realizados diversos experimentos envolvendo diferentes tipos de aplicações. Os sistemas fuzzy evoluídos a partir da aplicação da metodologia proposta foram analisados tomando-se por base o desempenho de suas respostas e o grau de interpretabilidade da base de conhecimento evoluída.

Para a execução dos testes de simulação, foi desenvolvido um programa computacional para a execução dos processos evolutivos utilizando a técnica dos algoritmos genéticos. O programa foi todo desenvolvido no ambiente de programação do software MATLAB.

A avaliação da aptidão, tomando-se por base o desempenho e a interpretabilidade do sistema, é realizada através do uso de uma técnica multiobjetivo baseada na lógica fuzzy, denominada, neste trabalho, de Agregador Fuzzy. O sistema fuzzy desenvolvido para este fim foi implementado utilizando-se o *toolbox* Fuzzy Logic do SIMULINK. O uso desta ferramenta possibilitou que o sistema fuzzy para avaliação multiobjetivo fosse facilmente incorporado à metodologia do AG proposta.

Os estudos de casos podem ser divididos em três partes principais. Na primeira parte, são realizados experimentos relacionados a aproximação de funções. Na segunda parte, são desenvolvidos controladores fuzzy e as respostas são analisadas segundo o seu comportamento transiente e estacionário (tomando-se por base a entrada de um determinado sinal de controle). Finalmente, na terceira parte é realizado um experimento que visa a desenvolver um controlador MIMO capaz de controlar uma planta complexa, que envolve uma aplicação real para o controle de velocidade de uma embarcação naval com hélice de passo variável (HPC).

Em todos os experimentos os resultados são comparados com as respostas obtidas pela utilização de outras técnicas desenvolvidas em estudos anteriores. Para o experimento do controle de velocidade da embarcação naval, os resultados são comparados aos resultados obtidos pelo uso do controlador que atualmente realiza o controle real da planta.

Como os sistemas fuzzy desenvolvidos pela metodologia proposta são evoluídos segundo dois objetivos distintos, tais sistemas foram identificados pela nomenclatura SFEMO (Sistema Fuzzy Evolucionário Multiobjetivo). O uso desta nomenclatura pode ser observado nas respostas obtidas nos diferentes testes realizados nas próximas seções desse capítulo.

4.1 Ambiente de Simulação

O sistema para a realização das simulações foi totalmente desenvolvido em ambiente MATLAB. Para as aplicações de controle, o SIMULINK pode ser utilizado para analisar o desempenho dos controladores fuzzy evoluídos.

Dois *scripts* (arquivos *.m*) principais são os responsáveis pela execução dos dois ciclos evolucionários que compõem a metodologia proposta. O arquivo *GA_exec.m* é responsável por conceber diferentes sistemas fuzzy a partir de um conjunto de parâmetros pré-definidos. Os parâmetros são configurados através da utilização de uma Interface Homem Máquina (IHM), desenvolvida para utilizar o próprio ambiente de comando do MATLAB. Tais parâmetros são:

- **Número de variáveis:** Representa o número de variáveis de entrada e de saída do sistema fuzzy.
- **Limites para o número de MFs:** Número mínimo e máximo de funções de pertinência admissíveis para cada uma das variáveis do sistema fuzzy.
- **Universo de discurso:** Intervalo fechado que compõe o universo de discurso de cada variável.
- **Tipo de FP:** Define o tipo de função de pertinência utilizado pelo algoritmo, podendo ser do tipo triangular, trapezoidal ou gaussiana.
- **Modelo e FO:** Em aplicações de controle que utilizam o SIMULINK, o sistema solicita ao projetista o nome do arquivo que representa o modelo a ser controlado (*nome_do_arquivo.mdl*). O mesmo ocorre com a função objetivo (FO) responsável pela análise da resposta do sistema (*nome_do_arquivo.m*).
- **Tamanho da população:** Número de indivíduos que formam o conjunto de possíveis soluções.
- **População Inicial:** A população inicial pode ser criada aleatoriamente ou pode ser utilizar uma população já existente. Neste último caso, o nome do arquivo contendo as configurações necessárias (*nome_do_arquivo.mat*) deve ser fornecido ao sistema.
- **Taxa de Crossover e Mutação:** Determina o valor probabilístico para a ocorrência de cruzamento e mutação nos cromossomos, respectivamente.

- **Valor de Saturação:** O número máximo de gerações em que o ciclo evolucionário ocorre antes de ser interrompido, caso não haja qualquer tipo de evolução relacionada à melhor solução encontrada.

A população inicial pode ser salva num arquivo (*nome_do_arquivo.mat*) para os casos em que esta é gerada aleatoriamente pelo programa. O mesmo ocorre para o final deste primeiro ciclo evolucionário, onde a população final (conjunto de sistemas fuzzy) e uma série de outros parâmetros que permitem a análise gráfica de todo o processo evolutivo são armazenados no disco rígido.

A segunda etapa do ciclo evolucionário ocorre por conta da execução do arquivo *RefineGA.m*. Utilizando uma interface gráfica semelhante à utilizada pelo ciclo anterior, o projetista deve configurar os limites mínimo e máximo (γ e κ) das sobreposições entre as funções de pertinência, além dos parâmetros de configuração do AG, tais como: taxa de crossover e mutação, número máximo de gerações, etc.

4.2 Aproximação de Funções

O Sistema Fuzzy Evolucionário MultiObjetivo (SFEMO) proposto foi utilizado em duas aplicações de aproximação de funções. Os resultados obtidos foram comparados com os obtidos através da aplicação de outros modelos fuzzy. Uma das aplicações selecionadas utiliza a conhecida função *Sinc*, enquanto a outra aborda a aproximação de uma função com dados ruidosos.

Função *Sinc*

Esta função é representada por:

$$Sinc(x,y) = \frac{\sin(x)\sin(y)}{(x)(y)} \qquad (12)$$

Foi utilizado um total de 850 amostras, divididas em 225 amostras para treinamento e 625 amostras para testar a eficiência do sistema fuzzy evoluído. As entradas *x* e *y* variaram dentro da área limitada pelo quadrado [-10, 10] x [-10, 10], e a função *Sinc(x,y)* variou no intervalo [-0.21, 1].

Funções do tipo gaussianas foram utilizadas para representar as funções de pertinência do sistema. Para calcular a métrica de erro do sistema, o RMSE foi utilizado.

A configuração do sistema durante as duas etapas de execução do algoritmo pode ser observada na Tabela 2.

Tabela 2 – Configuração do AG para função *Sinc*.

	Etapa 1	**Etapa 2**
Tamanho da População	100	100
Número de Gerações	200	100
Taxa de *Crossover*	0.65	0.6
Taxa de Mutação	0.02	0.1
γ	-	0.1
κ	-	0.8

A Tabela 3 apresenta o erro médio quadrático (RMSE) do conjunto de treinamento e de testes para o sistema SFEMO. Também, são relacionados os resultados obtidos na mesma aplicação para os sistemas SFE (AMARAL, 2003), Wang-Mendel (WANG e MANDEL, 1998) e para os sistemas neuro-fuzzy NEFPROX (NAUCK e KRUSE, 1999), FSOM (VUORIMA, 1994), ANFIS (JANG, 1993) e NFHQ (SOUZA, 1999).

Tabela 3 – RMSE para a função *Sinc*.

Sistema	RMSE (treinamento)	RMSE (teste)
Wang-Mendel	0,0735	0,0766
NEFPROX	0,0696	0,0733
SFEMO	**0,0692**	**0,0699**
SFE	0,0390	0,0403
FSOM	0,0314	0,0319
ANFIS	0,0226	0,0268
NFHQ	0,0150	0,0150

Na Figura 29 podemos observar a evolução da aptidão do melhor indivíduo e a aptidão média da população no decorrer das gerações, durante a execução da primeira etapa do algoritmo. O processo de evolução foi encerrado pelo critério de saturação do valor de aptidão do melhor indivíduo.

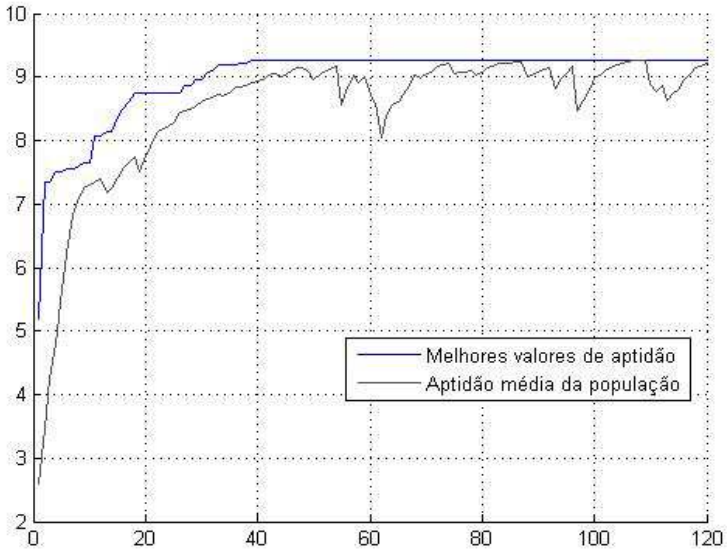

Figura 29 – Gráfico de aptidões da função *Sinc*.

O sistema proposto obteve bons resultados dentre os sistemas que apresentam uma mais fácil interpretação. Considerando este critério de avaliação, o SFEMO apresentou resultados inferiores ao sistema SFE, porém, obteve melhor desempenho quando comparado aos métodos NEFPROX e Wang-Mendel.

Por não ter havido melhora no desempenho durante a segunda etapa de execução do algoritmo, o universo de discurso do sistema fuzzy manteve-se particionado de maneira uniforme. A base de regras do SFEMO é constituída por 16 (dezesseis) regras. A Tabela 4 apresenta uma comparação entre os sistemas obtidos pela aplicação dos métodos SFEMO e o SFE. Podemos observar que a base de regras do SFEMO apresenta uma maior simplicidade quando comparada ao SFE em função do menor número de regras.

Tabela 4 - Configuração dos sistemas SFEMO e SFE da função Sinc.

Sistema	Funções de Pertinência	Nº de Regras
SFEMO	5_x5_x7	16
SFE	5_x5_x6	25

Função com Ruído:

Neste experimento são analisados os desempenhos de diferentes sistemas na aproximação de uma determinada função quando dados ruidosos são utilizados durante a fase de treinamento. A função a ser aproximada é (DELGADO, 2002):

$$F_1: \Omega \rightarrow \Re, F_1(x_1, x_2) = f(x_1, x_2) + N[\mu, \sigma] \qquad (13)$$

Onde,

$$f_1(x_1, x_2) = 1.9(1.35 + exp(x_1 - x_2)sen(13(x_1 - 0.6)^2)sen(7x_2) \qquad (14)$$

com Ω = [0,1], μ = 0 e σ = 0.3. As amostras ruidosas foram igualmente espaçadas. Para o treinamento foram utilizadas 225 amostras ruidosas e o teste foi realizado com 2500 amostras igualmente espaçadas em Ω.

A configuração utilizada para o desenvolvimento do sistema SFEMO pode ser observada na Tabela 5.

Tabela 5 - Configuração do AG para função ruidosa.

	Etapa 1	**Etapa 2**
Tamanho da População	200	100
Número de Gerações	100	100
Taxa de *Crossover*	0.65	0.7
Taxa de Mutação	0.01	0.1
γ	-	0.1
κ	-	0.8

Para avaliar o desempenho do sistema SFEMO comparamos os resultados obtidos com os sistemas SFE (AMARAL, 2003), Wang-Mendel (WANG e MENDEL, 1998), NEFPROX (NAUCK e KRUSE, 1999) e ANFIS (JANG, 1993). A Tabela 6 apresenta os resultados obtidos com a utilização dos diferentes sistemas para a aproximação da função ruidosa.

Tabela 6 - RMSE para a função ruidosa.

Sistema	RMSE (treinamento)	RMSE (teste)
Wang-Mendel	0,70	0,62
NEFPROX	0,64	0,55
SFEMO	**0,40**	**0,39**
SFE	0,41	0,32
ANFIS	0,25	0,13

A Figura 30 apresenta a evolução do melhor indivíduo e da aptidão média da população durante as duas etapas de execução do algoritmo. A análise gráfica permite observar a melhora obtida durante a execução da segunda etapa do AG, resultante da

otimização dos parâmetros das funções de pertinência do melhor indivíduo, selecionado ao final da primeira etapa de execução.

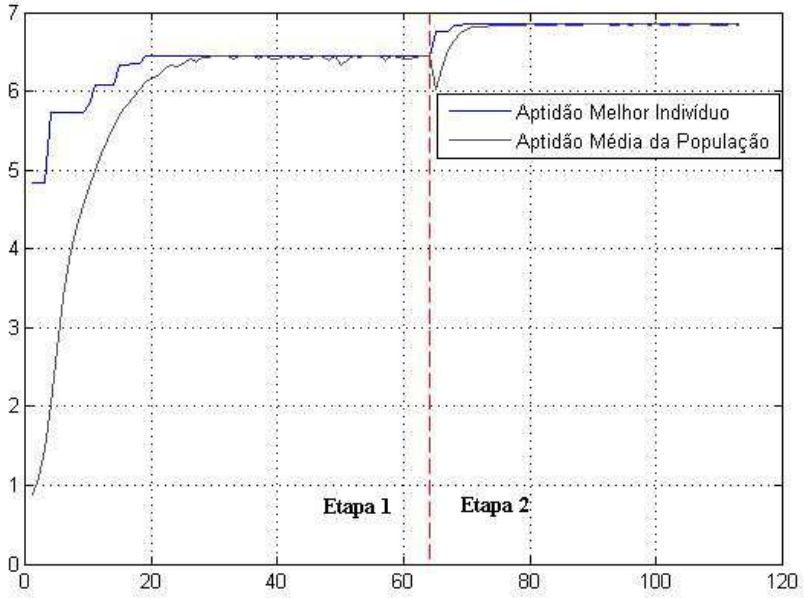

Figura 30 - Gráfico de Aptidões da função ruído.

A Tabela 7 apresenta a comparação entre os sistemas SFEMO e o SFE. Apesar de apresentar um desempenho inferior ao SFE durante a fase de testes, conforme observado na Tabela 6, quando comparamos tais sistemas podemos observar que o sistema SFEMO possui uma base de regras mais simples do que a apresentada pelo SFE, levando-se em consideração que ambos os sistemas possuem o mesmo número de funções de pertinência para as variáveis de entrada. Essa redução no número de regras contribui para o ganho na interpretabilidade do SFEMO.

Tabela 7 - Configuração dos sistemas SFEMO e SFE da função ruído.

Sistema	Funções de Pertinência	Nº de Regras
SFEMO	4_x4_x6	10
SFE	4_x4_x9	16

Na Figura 31 estão apresentadas graficamente as funções de pertinência do sistema fuzzy obtido ao final da execução do algoritmo. Nela podemos constatar que a representatividade das funções de pertinência foi mantida, graças às alterações controladas nos parâmetros que as definem.

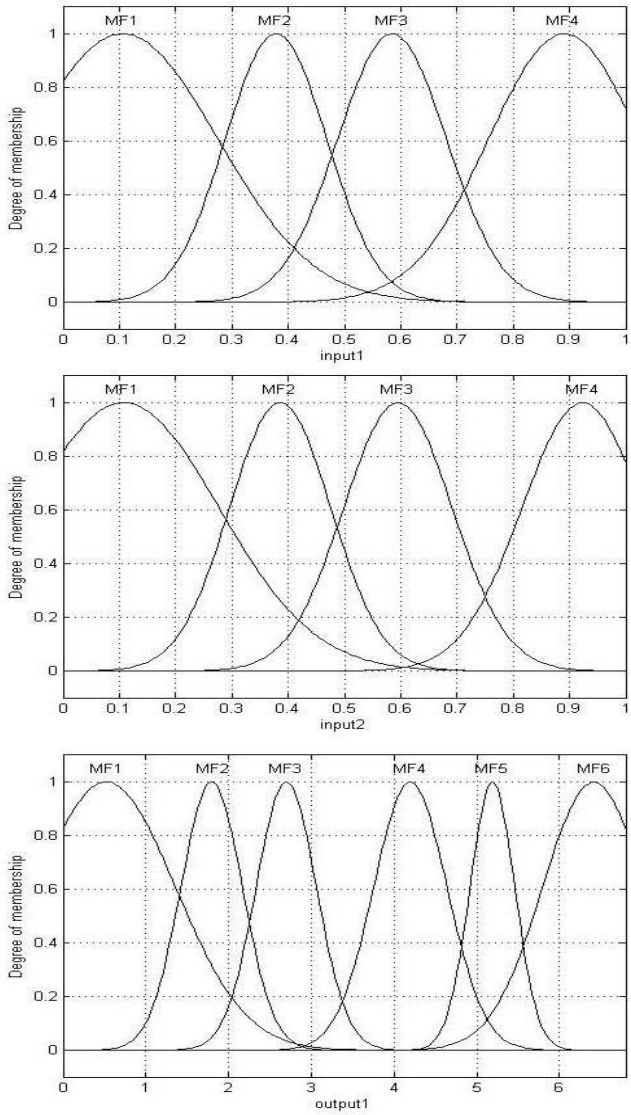

Figura 31 - Funções de pertinência do sistema SFEMO para a função ruído.

4.3 Controle

A Lógica Fuzzy tem sido amplamente utilizada no desenvolvimento de controladores nos mais diversos tipos de aplicações. Porém, o desenvolvimento de um controlador fuzzy pelo “método de tentativa e erro” é uma tarefa que, normalmente, requer uma considerável quantidade de trabalho e tempo.

Apesar de ser possível desenvolver rapidamente um “protótipo” de um controlador fuzzy, os ajustes dos parâmetros necessários para que o controle apresente o comportamento desejado não é uma tarefa trivial.

O método proposto simplifica a tarefa de desenvolvimento do controlador fuzzy, atuando sobre o desenvolvimento de toda a base de conhecimento, além de realizar os ajustes necessários para uma melhora no desempenho do sistema, utilizando, para tal, as técnicas de aprendizado baseadas na Computação Evolucionária.

Para o projeto de sistemas de controle linear, as técnicas analíticas convencionais têm sido aplicadas com sucesso. Nesse tipo de metodologia, o primeiro passo para a análise de um sistema de controle deve ser a obtenção de um modelo matemático do sistema (OGATA, 2003). Obtido o modelo matemático, é possível analisar o seu desempenho a partir de diversos métodos disponíveis. Na análise e teste de sistemas de controle, utilizam-se sinais de entrada de testes específicos que possibilitem a comparação de desempenho entre os diferentes sistemas de controle, com base em determinadas especificações (OGATA, 2003). Dentre as especificações no domínio do tempo, podem-se destacar como as mais importantes: o tempo de subida (t_r), o máximo valor de ultrapassagem (M_p) e o tempo de acomodação (t_s) (AMARAL, 2003). Tais parâmetros são apresentados na Figura 32.

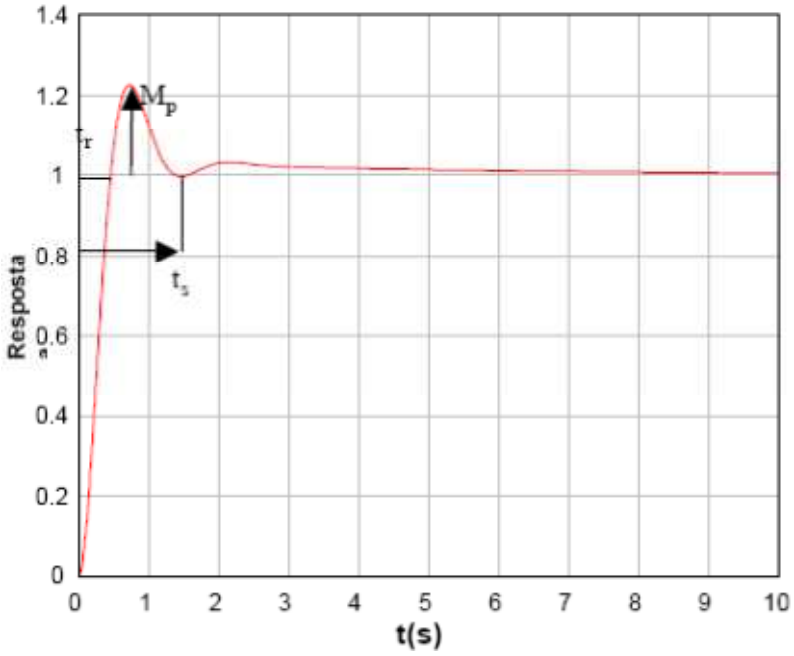

Figura 32 – Exemplo de resposta transiente de um sistema (AMARAL, 2003)

Três aplicações de controle são utilizadas para analisar o desempenho do controlador fuzzy, desenvolvido pelo método Fuzzy-Evolucionário proposto (SFEMO). As duas primeiras consideram sistemas lineares de segunda e terceira ordens, respectivamente. Os resultados foram comparados aos obtidos por outros métodos.

A terceira aplicação é considerada mais complexa que as anteriores, por representar uma planta com características não-lineares: o controle pela armadura de um motor CC com características não-lineares de saturação. Os resultados foram novamente comparados com os obtidos por outros métodos.

4.3.1 Experimento 1 – Planta de 2ª Ordem

Este experimento refere-se ao controle de uma planta de segunda ordem, representada pela seguinte função de transferência:

$$G_1(s) = \frac{4}{s(s+0.5)} \qquad (15)$$

A análise da resposta transiente e de regime estacionário foi realizada aplicando-se um degrau unitário na entrada do sistema em malha fechada. É desejável que o controlador utilizado possibilite que a resposta do sistema siga a resposta de entrada, apresentando um baixo sobre-sinal (*overshoot*) e alcançando a estabilidade no menor intervalo de tempo possível (t_s). Um diagrama representando o sistema de controle é apresentado na Figura 33.

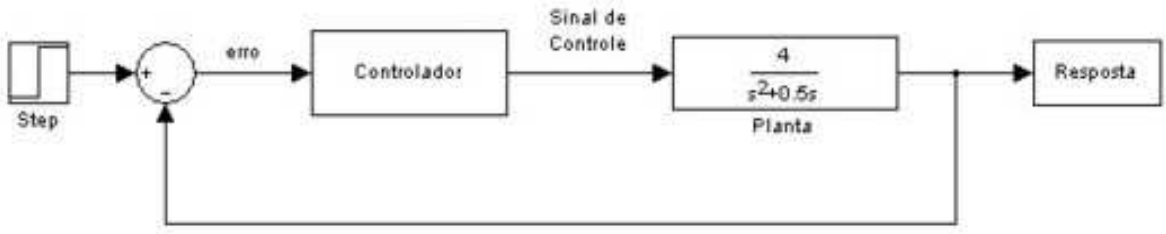

Figura 33 – Diagrama de controle para a planta de segunda ordem.

A configuração do sistema utilizada durante o experimento pode ser observada na Tabela 8.

Tabela 8 - Configuração do AG para a planta de 2ª ordem.

	Etapa 1	**Etapa 2**
Tamanho da População	60	100
Número de Gerações	50	80
Taxa de *Crossover*	0.65	0.6

Taxa de Mutação	0.01	0.1
γ	-	0.1
κ	-	0.8

Para avaliarmos o desempenho do sistema, comparamos as respostas obtidas utilizando controladores desenvolvidos por outros três métodos. Um controlador fuzzy evoluído pelo uso da computação evolucionária (SFE) foi utilizado em (AMARAL, 2003) para o controle desta mesma planta. Também foram comparadas as respostas obtidas pelo controlador NEFCON e por um compensador analítico $G_c(s)$, cuja função de transferência resultante é (OGATA, 2003):

$$G_c(s) = \frac{10(2s+1)(5s+1)}{(0.1992s+1)(80.19s+1)} \qquad (16)$$

Os resultados obtidos são apresentados na Figura 34. Podemos observar o bom desempenho obtido pelo SFEMO. Apesar de apresentar uma reposta com um tempo de subida (t_r) um pouco maior do que o obtido pelo controlador analítico e pelo NEFCON, o controlador SFEMO apresentou o menor valor de *overshoot* e um tempo de acomodação (t_s) próximo ao obtido pelo controlador analítico. Além disso, foi o único a apresentar erro nulo da resposta em regime estacionário, juntamente com o controlador analítico.

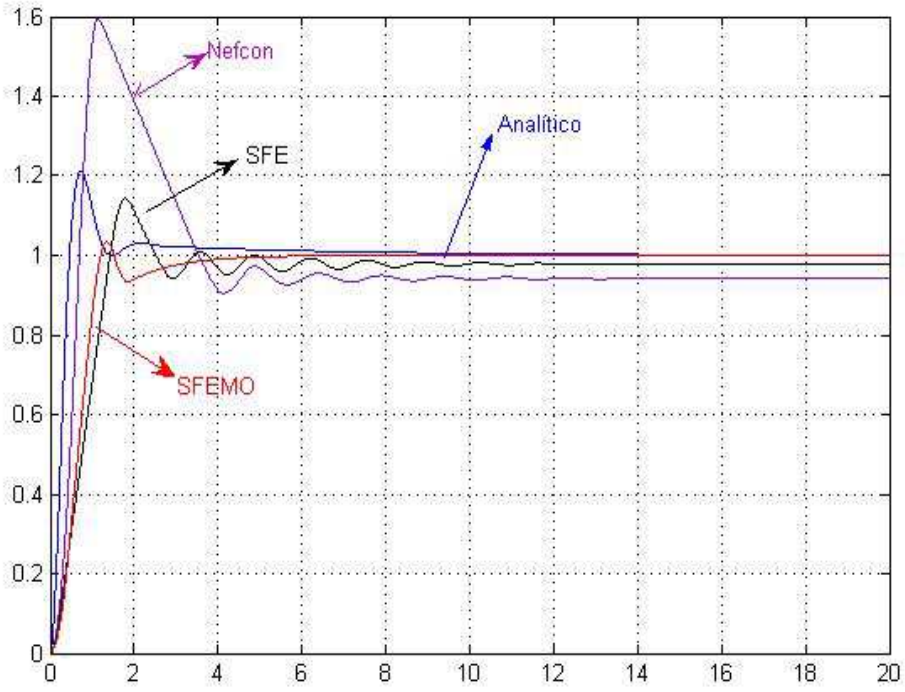

Figura 34 - Respostas ao degrau da planta de 2ª ordem.

Além do bom desempenho obtido pelo SFEMO, o sistema também apresenta uma boa interpretabilidade da sua base de conhecimento, obtida graças ao desenvolvimento de uma base de regras com um número reduzido de regras e de sobreposições controladas (definidas pelas propriedades de γ-completitude e de κ-sobreposição) entre as funções de pertinência do sistema. A Tabela 9 compara a base de dados e a base de regras do SFEMO e do SFE.

Tabela 9 - Características do SFEMO para planta de 2ª ordem.

	SFEMO	**SFE**
Nº de FPs para a variável de entrada 01	3	3
Nº de FPs para a variável de entrada 02	3	3
Nº de FPs para a variável de saída	3	5
Nº de regras da base de regras	5	9

A tabela de decisão do SFEMO é apresentada na Tabela 10. A Figura 35 apresenta os conjuntos fuzzy obtidos ao final da execução da segunda etapa do algoritmo.

Tabela 10 – Tabela de decisão do SFEMO para o controle da planta de 2ª ordem.

	Variação do Erro		
RMSE	X21	X22	X23
X11	-	Y1	-
X12	Y1	-	-
X13	Y3	Y3	Y2

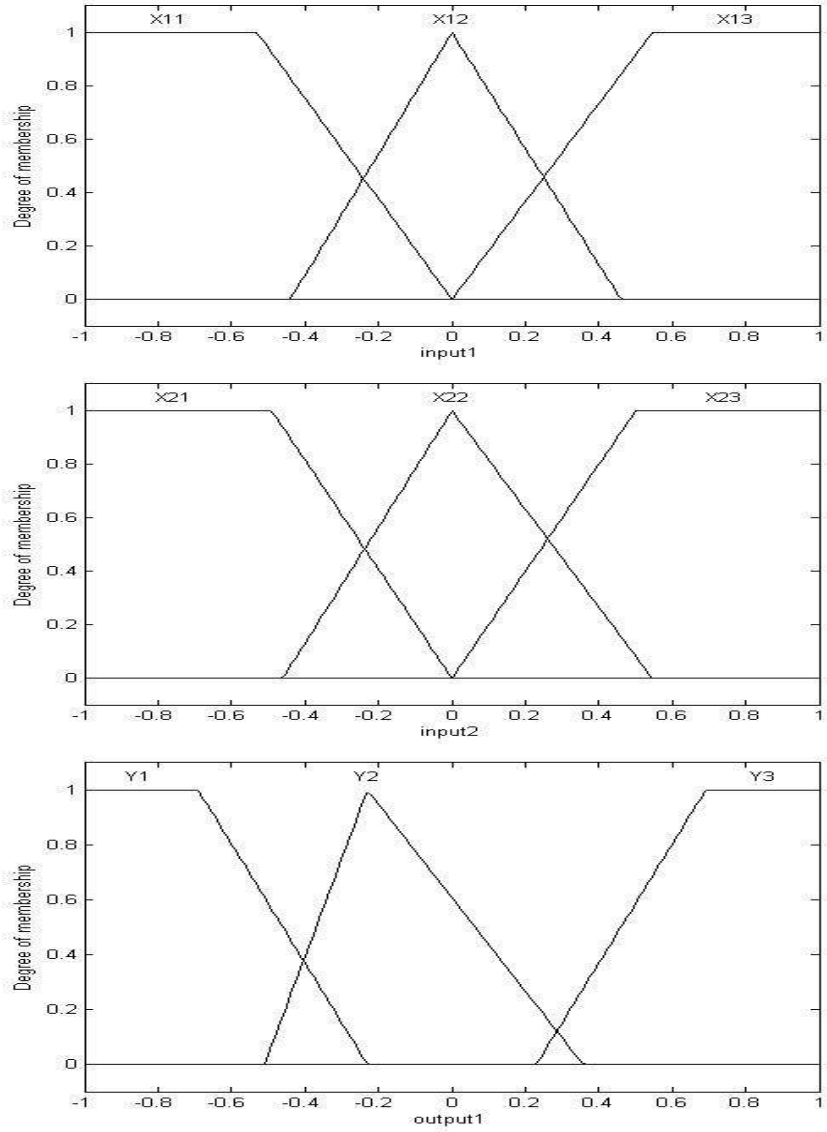

Figura 35- Funções de pertinência para o controlador SFEMO da planta de 2ª ordem.

4.3.2 Experimento 2 – Planta de 3ª Ordem

Este experimento considera uma planta de terceira ordem, representada pela seguinte função de transferência:

$$G(s) = \frac{1}{s(s+1)(s+5)} \qquad (17)$$

Mais uma vez, para permitir a análise da eficiência do método proposto, novas comparações entre as respostas obtidas pela aplicação de diferentes técnicas e metodologias foram realizadas. Neste experimento compararam-se as respostas obtidas por um controlador

PID sintonizado pelo método de Ziegler-Nichols (K_p = 18, K_i = 12,81 e K_d = 6,32) (OGATA, 2003) e por um controlador fuzzy-genético (SFE), desenvolvido e sintonizado por uma diferente metodologia evolucionária proposta em (AMARAL, 2003).

A configuração do sistema utilizada para este experimento pode ser observada na Tabela 11.

Tabela 11 - Configuração do AG para a planta de 3ª ordem.

	Etapa 1	**Etapa 2**
Tamanho da População	60	100
Número de Gerações	50	80
Taxa de *Crossover*	0.65	0.6
Taxa de Mutação	0.01	0.1
γ	-	0.1
κ	-	0.8

Na Figura 36 são apresentadas as respostas obtidas pelos diferentes controladores utilizados. Pode-se constatar o bom desempenho apresentado pelo controlador SFEMO. A resposta transiente apresenta um tempo de subida (t_r) um pouco maior do que a resposta mais rápida do sistema, proporcionada pelo controlador PID. Entretanto, esta última apresenta um *overshoot* de 46,22% superior ao apresentado pelo SFEMO, e um tempo de acomodação (t_s) de, aproximadamente, 5s superior ao obtido pelo uso do controlador SFEMO. O controlador SFE apresentou uma resposta mais lenta, porém com pouca ultrapassagem (*overshoot*) e menos oscilatória do que a resposta obtida pelo PID.

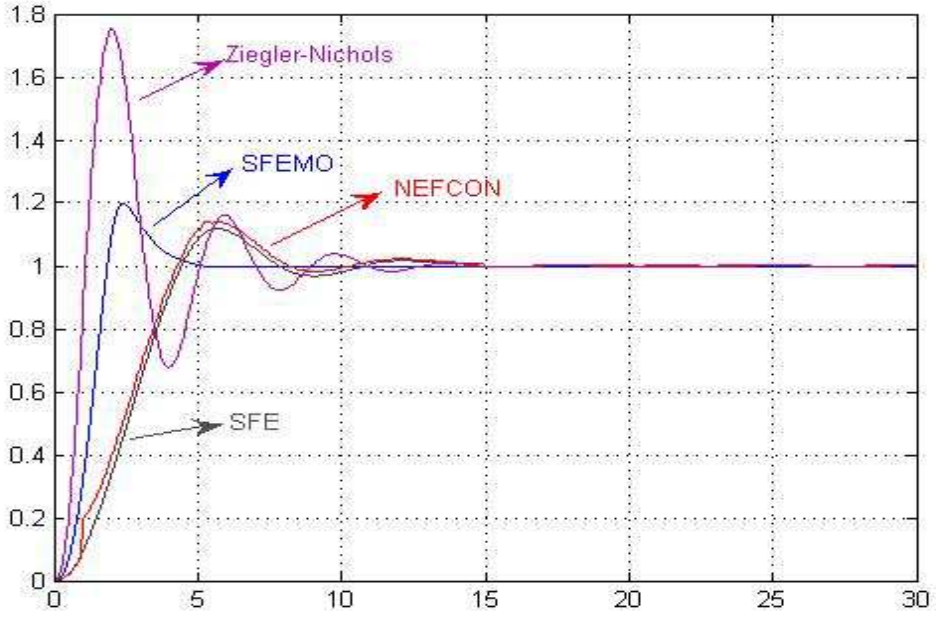

Figura 36 – Respostas ao degrau da planta de 3ª ordem.

A Tabela 12 apresenta as características do controlador SFEMO obtido ao final do processo de evolução.

Tabela 12 - Características do SFEMO para a planta de 3ª ordem.

	SFEMO	**SFE**
Nº de FPs para a variável de entrada 01	3	3
Nº de FPs para a variável de entrada 02	3	3
Nº de FPs para a variável de saída	3	5
Nº de regras da base de regras	4	9

Além do bom desempenho, o SFEMO apresenta uma base de regras formada por apenas 4 (quatro) regras, enquanto o SFE possui 9 (nove) regras para um número semelhante de FPs relacionadas às variáveis de entrada. Os valores de γ e κ (propriedades de γ-completitude e de κ-sobreposição) evitam que as funções de pertinência evoluídas apresentem sobreposições que prejudiquem as suas representatividades. Tais características contribuem para que o SFEMO seja um sistema que possua uma boa interpretabilidade da sua base de conhecimento.

A Tabela 13 apresenta a tabela de decisão do SFEMO e a Figura 37 apresenta os conjuntos fuzzy obtidos ao final da execução da segunda etapa do algoritmo.

Tabela 13 - Tabela de decisão do SFEMO para o controle da planta de 3ª ordem.

	Variação do Erro		
RMSE	X21	X22	X23
X11	-	-	-
X12	Y1	-	-
X13	Y1	Y2	Y3

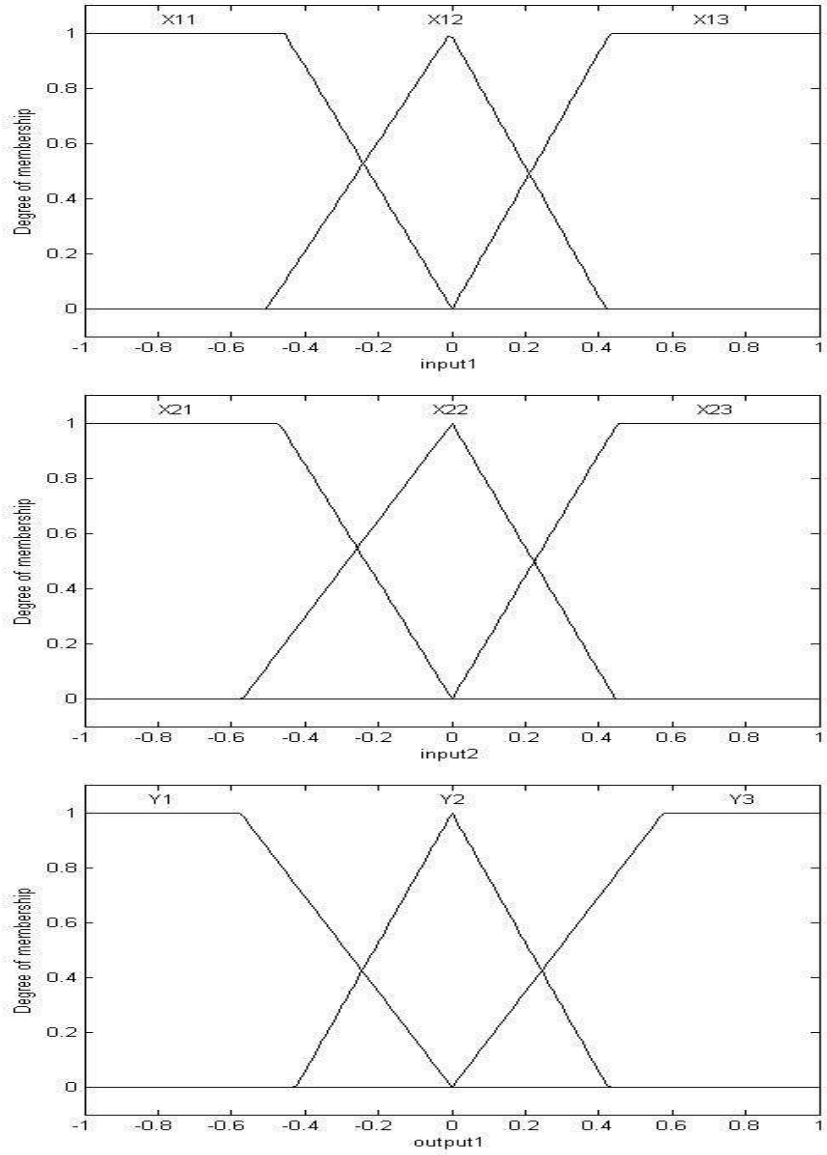

Figura 37 - Funções de pertinência do controlador SFEMO para a planta de 3ª ordem.

4.3.3 Experimento 3 – Controle de um Motor CC

Este experimento tem por objetivo controlar de um servomotor de corrente contínua controlado pela armadura com características não-lineares de saturação. Os procedimentos tradicionais para determinar as soluções de problemas envolvendo sistemas não-lineares são, geralmente, bastante complexos. Para reduzir esta complexidade, muitas vezes utiliza-se um processo de linearização no qual o sistema é substituído por um sistema linear equivalente, válido para uma determinada faixa de operação. Por outro lado, os controladores fuzzy são

intrinsecamente não-lineares e o processo de aprendizagem, baseado nos algoritmos genéticos, já leva em conta esta não-linearidade do sistema (AMARAL, 2003).

A Figura 38 apresenta o sistema de controle para o motor DC. A não-linearidade do sistema é representada pela inserção de um bloco de saturação, que limita o sinal de acionamento da armadura do motor em -0.8 e 0.8. A função de transferência do motor é dada por:

$$G_{DC}(s) = \frac{1}{s(1.3s+1)} \qquad (18)$$

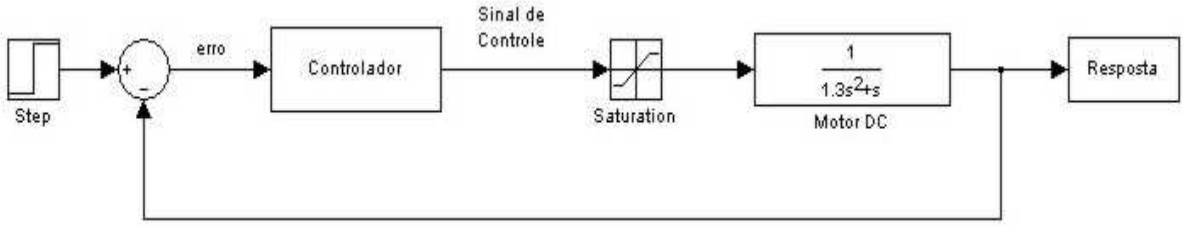

Figura 38 - Esquema de controle para motor DC com não-linearidade.

A configuração utilizada para a evolução do controlador fuzzy pode ser observada na Tabela 14.

Tabela 14 - Configuração do AG para o controle do motor DC.

	Etapa 1	**Etapa 2**
Tamanho da População	60	100
Número de Gerações	50	80
Taxa de *Crossover*	0.65	0.6
Taxa de Mutação	0.01	0.1
γ	-	0.1
κ	-	0.8

A Figura 39 apresenta as respostas transientes à entrada de um degrau unitário para o controle do motor DC com características não-lineares. Comparando as respostas obtidas com o uso dos controladores SFEMO, SFE e NEFCON (AMARAL, 2003), podemos observar o bom desempenho obtido pelo controlador SFEMO. A resposta apresenta um tempo de subida e um máximo *overshoo*t semelhante aos outros dois sistemas, porém, o SFEMO alcançou o estado estacionário num intervalo de tempo inferior aos demais sistemas (tempo de acomodação).

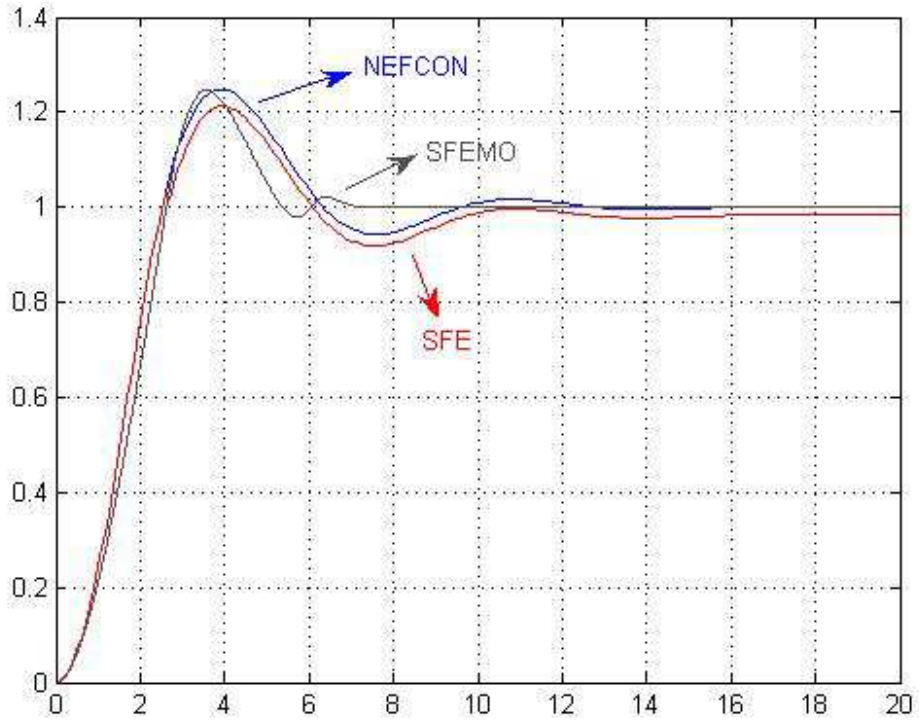

Figura 39 - Respostas ao degrau para o motor dc.

A Tabela 15 apresenta as características do controlador SFEMO obtido ao final do processo de evolução, comparando-o com o controlador evoluído pelo SFE (Amaral, 2003).

Tabela 15 - Características do SFEMO para o controle do motor dc.

	SFEMO	**SFE**
Nº de FPs para a variável de entrada 01	3	3
Nº de FPs para a variável de entrada 02	3	3
Nº de FPs para a variável de saída	4	5
Nº de regras da base de regras	5	9

Comparando a interpretabilidade da base de regras entre o SFEMO e o SFE, podemos constatar que o primeiro possui uma melhor interpretabilidade por conter 4 (quatro) regras a menos, considerando o mesmo número de funções de pertinência para ambas as variáveis de entrada. Além da interpretabilidade da base de regras do SFEMO, o particionamento do universo de discurso das variáveis com base nas propriedades de γ-completitude e de κ-sobreposição garantem uma boa interpretabilidade da base de dados do sistema. Tais características podem ser observadas pela análise da tabela de decisão da base de regras e pelas funções de pertinência do sistema, apresentadas na Tabela 16 e na Figura 40, respectivamente.

Tabela 16 - Tabela de decisão do SFEMO para o controle do motor CC.

	Variação do Erro		
RMSE	X21	X22	X23
X11	Y2	Y1	-
X12	-	-	Y3
X13	-	Y4	Y3

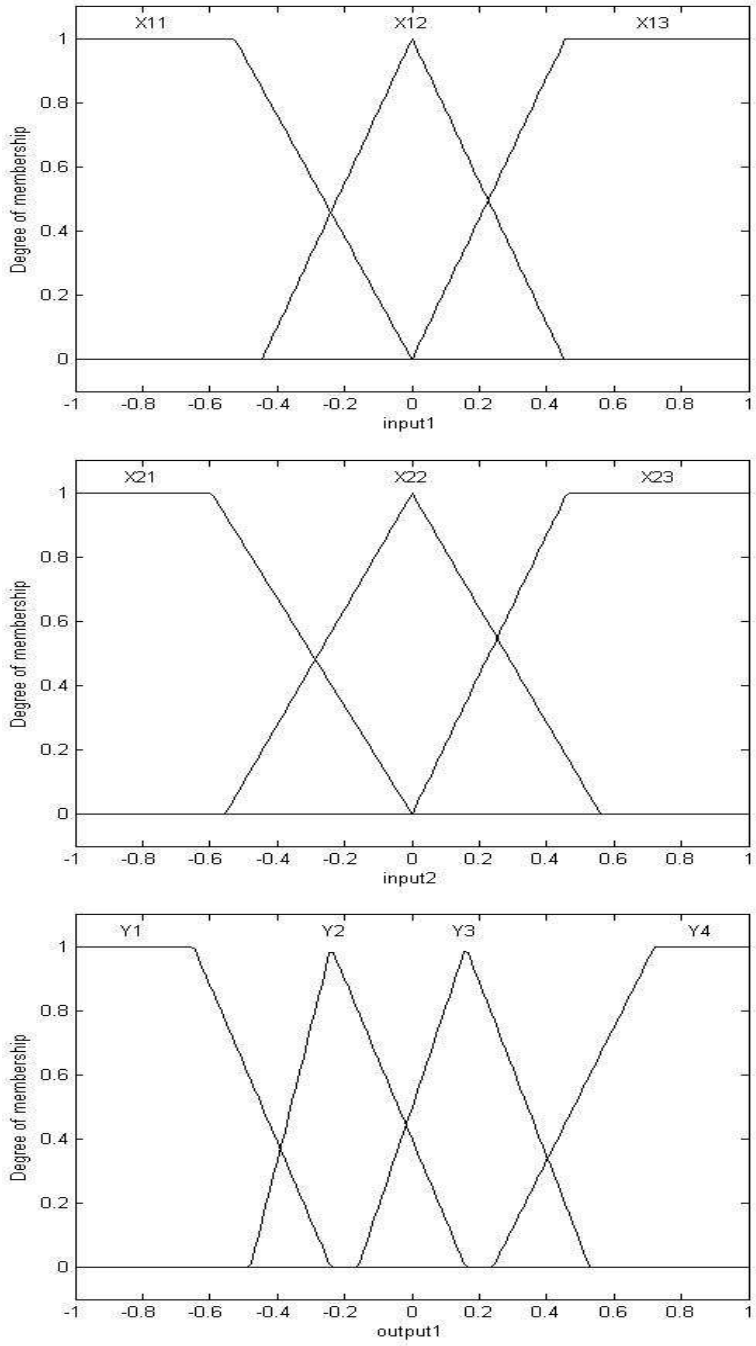

Figura 40 - Funções de pertinência do controlador SFEMO para o motor cc.

4.4 Controle de Velocidade de uma Embarcação de Passo Controlável

A construção moderna de embarcações normalmente envolve o uso de hélices de passo variável e reversível no arranjo do sistema propulsor. Esse conjunto permite um movimento axial de suas hélices, com influência direta no deslocamento da embarcação. A Figura 41 apresenta esquematicamente o movimento de uma das pás do hélice de passo variável e reversível. Observe o sinal negativo em θ_{MIN} indicando a reversão do passo e, conseqüentemente, alterando o sentido de deslocamento da embarcação para ré. Apesar de aumentar a complexidade do sistema, tal arranjo possui vantagens em relação aos sistemas que utilizam a configuração de passo fixo.

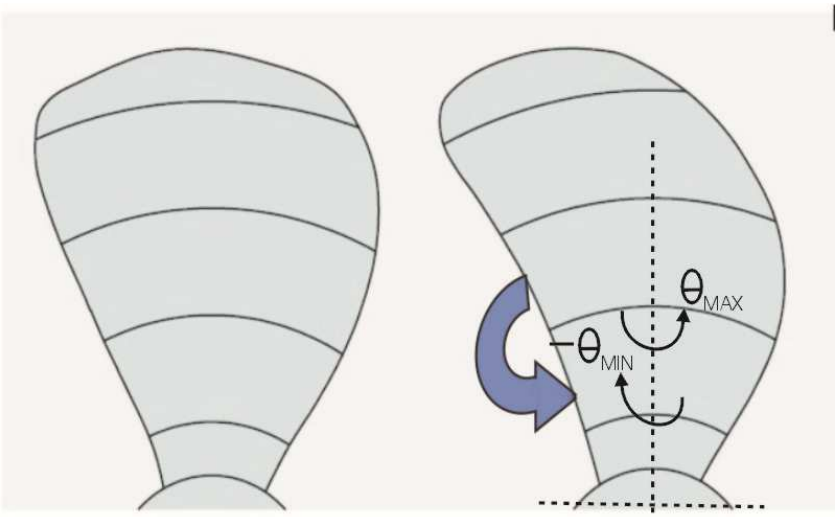

Figura 41 - Representação do deslocamento da pá de um hélice de passo variável.

O controle do posicionamento das pás do hélice (controle de passo) permite um melhor aproveitamento da potência da máquina de acionamento em toda a sua faixa de operação. Além disso, a inversão no sentido de movimentação do navio é realizada pela atuação no passo do hélice, permitindo, dessa forma, a utilização de máquinas unidirecionais como as turbinas a gás, sem que haja a necessidade de instalação de caixas de reversão (CHAGAS,1983).

Um exemplo do modelo da planta de propulsão com estas características pode ser observado na Figura 42.

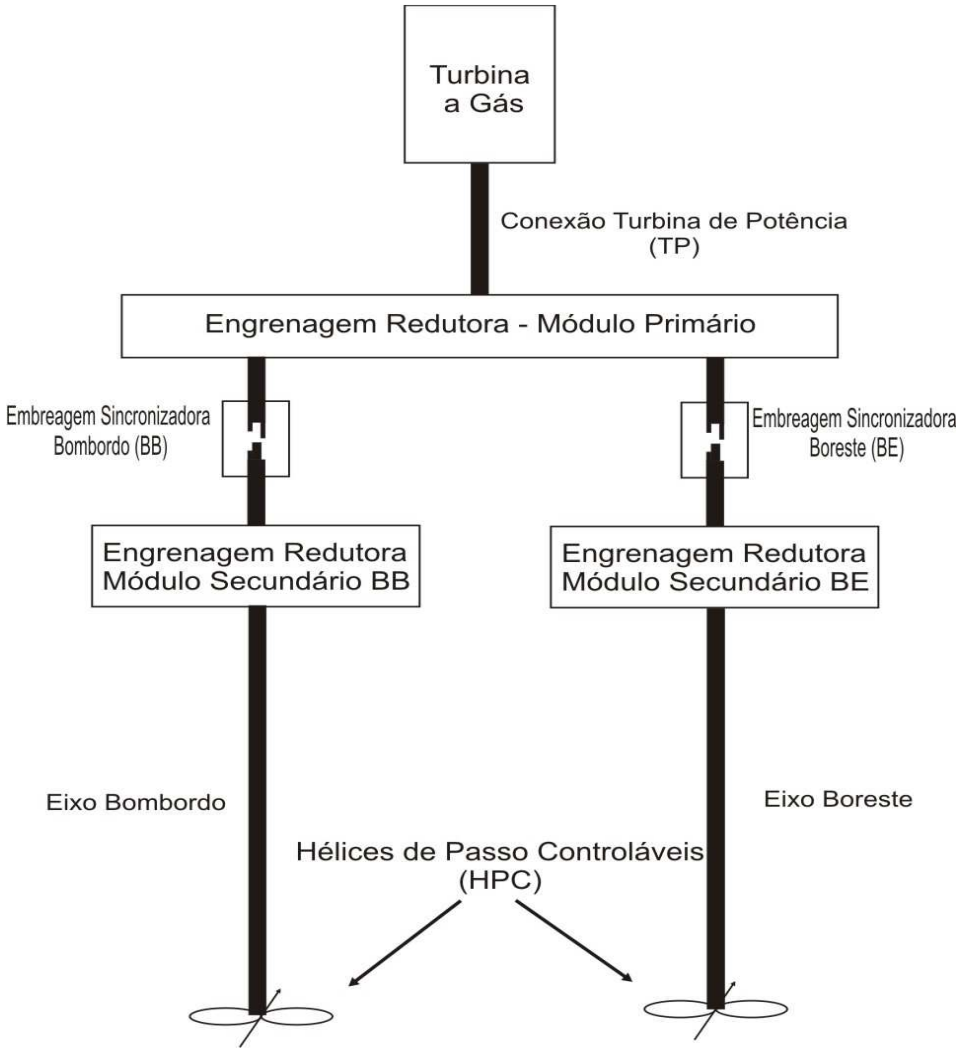

Figura 42 – Exemplo da planta de propulsão utilizando turbina a gás.

O objetivo do sistema de propulsão é prover torque e rotação aos hélices do navio de forma a produzir o empuxo necessário para causar movimento. O sistema de controle utilizado deve garantir que as velocidades desejadas do navio sejam alcançadas num intervalo de tempo razoável sem, contudo, exceder aos limites de torques especificados para cada equipamento.

No exemplo apresentado na Figura 42, um sistema composto por três engrenagens redutoras (módulo primário e módulo secundário de bombordo e boreste – BB e BE) é o responsável pelas reduções necessárias entre a máquina propulsora e os eixos dos hélices. Um conjunto de duas embreagens permite que os eixos dos hélices sejam conectados ao eixo propulsor da turbina de potência (TP). Cada embreagem possui uma entrada (eixo TP) e uma saída (eixo carga – hélice). O acoplamento ocorre quando a rotação equivalente da turbina de

potência for superior à rotação de saída da embreagem. Essa configuração ainda permite que a turbina a gás acione apenas um único eixo ou nenhum dos dois eixos propulsores do navio.

Os hélices de passo controlável (HPC) produzem o empuxo necessário para movimentar a embarcação. O passo do hélice é fixado por um sistema de controle local, que recebe o *setpoint* de ângulo e posiciona as pás através de um sistema hidráulico.

A turbina a gás (TG) possui um controlador de combustível (MFC – *Main Fuel Controller*) que recebe um *setpoint* do ângulo da válvula de combustível, denominado PLA (*Power Level Angle*), e fornece como saída uma vazão de combustível a ser consumida pela TG.

4.4.1 Modelo Matemático do Navio

Para que seja possível realizar o estudo de desempenho de uma instalação propulsora é necessária a inclusão de todos os modelos dos componentes da instalação que possuem uma participação significativa na geração e no fluxo de energia (CHAGAS, 1983).

Com base nesses modelos, e utilizando determinadas equações de movimento, é possível realizar simulações computacionais que descrevam a dinâmica de movimento do navio. Segundo (MORISHITA e BRINATI, 1984) as seguintes equações representam a dinâmica de um sistema propulsor:

$$\frac{dN_h}{dt} = \frac{1}{2\pi J_t}\left(Q_m - Q_h - Q_f\right) \qquad (19)$$

Onde,

- N_h é a rotação do eixo propulsor;
- J_t é o momento de inércia total do sistema;
- Q_m é o torque motor fornecido ao sistema de propulsão;
- Q_h é o torque de oposição do hélice; e
- Q_f é o torque perdido em atrito.

$$\frac{dV}{dt} = \frac{1}{M}\left(T_h(1-t) - R\right) \qquad (20)$$

Onde,

- *V* é a velocidade do navio;
- *M* é a massa total do navio;
- *Th* é o empuxo produzido pelo hélice;
- *R* é a resistência ao deslocamento do navio; e
- *t* é o coeficiente de redução do empuxo.

As equações (19) e (20) são particularizadas para a configuração do sistema de propulsão apresentado na Figura 42, que é o objeto de estudo deste trabalho. Os modelos que constituem tal sistema foram formulados através de informações obtidas junto aos fabricantes dos equipamentos ou com base em diferentes estudos técnicos realizados. Nas próximas seções deste capítulo descreveremos, sucintamente, os modelos que compõem a instalação propulsora estudada.

4.4.2 Turbina a Gás GE LM2500

Nesta seção são apresentadas as equações do modelo da turbina a gás, bem como a sua implementação computacional em Matlab/Simulink. O modelo matemático utilizado teve como referência o estudo técnico apresentado em (MORISHITA e BRINATI, 1984). Segundo esse estudo, as seguintes equações de estado representam o modelo da TG:

$$\frac{dN_g}{dt} = \frac{1}{J_g}.F_{11}\left(\frac{N_g}{N_{gmax}\sqrt{\theta_0}}\right).\frac{1{,}356\,.60}{\theta_0^{0{,}719}}.F_{12}(\mathrm{H}_0).\delta_0\theta_0^{0{,}719}.0{,}45359 \qquad (21)$$

$$Q_{pt} = \frac{F_7(\gamma_0).\delta_0.1{,}356 + F_{10}\left(\frac{N_g}{N_{gmax}\sqrt{\theta_0}}\right)\frac{1}{\theta_0^{0{,}719}}F_{12}(\mathrm{H}_0)}{\left(1{,}6889.10^{-4}\left(0{,}0045156\frac{\theta_0}{\delta_0^2}W_{54}^2\right)+1\right)\left(0{,}91889.10^{-4}\left(0{,}0067733\frac{\theta_0}{\delta_0^2}W_{54}^2\right)+1\right)} \qquad (22)$$

$$T_{51} = \left(F_{17}\left({N_g}\Big/{N_{gmax}\sqrt{\theta_0}}, {N_{pt}}\Big/{\sqrt{\theta_0}}\right)\theta_0 + F_{16}\left(\frac{N_g}{N_{gmax}\sqrt{\theta_0}}\right)\frac{\theta_0^{0{,}281}}{\delta_0}.F_{12}(\mathrm{H}_0) - 39\right)\frac{5}{9} \qquad (23)$$

$$W_{54} = 0{,}453259.F_{19}\left(\frac{N_g}{N_{gmax}\sqrt{\theta_0}}\right)\frac{\delta_0}{\sqrt{\theta_0}} \qquad (24)$$

$$P_{s3} = \left(F_6 \left({N_g}/{N_{gmax}}, {N_{pt}}/{\sqrt{\theta_0}} \right) \delta_0 + F_9 \left(\frac{N_g}{N_{gmax}\sqrt{\theta_0}} \right) \frac{1}{\theta_0^{0,719}} \right) F_{12}(\mathrm{H}_0).\, 0{,}0689 \qquad (25)$$

$$\mathrm{H}_0 = \frac{W_f - F_4 \left({N_g}/{N_{gmax}}, {N_{pt}}/{\sqrt{\theta_0}} \right)}{0{,}45359} \qquad (26)$$

$$\delta_0 = \frac{P}{1{,}01325} \qquad (27)$$

$$\theta = \frac{T + 273{,}15}{288{,}15} \qquad (28)$$

Onde,

- N_g é a rotação do gerador de gás da turbina (rpm);
- W_f é a vazão de combustível de entrada (lb/h);
- N_{pt} é a rotação da turbina de potência (rpm);
- Q_{pt} é o torque na turbina de potência (N.m);
- T_{54} é a temperatura na saída do gerador de gás (^{o}R);
- W_{54} é a vazão do gás de combustão na saída do gerador de gás (lb/s);
- P_{s3} é a pressão de gás na saída do gerador de gás (psi);
- P é a pressão ambiente (bar);
- T é a temperatura ambiente (graus Celsius); e
- J_g é o momento de inércia do gerador de gás (23.89 kgm^2).

No Apêndice A as funções que aparecem nas equações acima (F_4, F_6, F_7, F_9, F_{10}, F_{11}, F_{12}, F_{16}, F_{17} *e* F_{19}) são apresentadas graficamente.

O diagrama de blocos que implementa as equações de estado do modelo da turbina no Matlab/Simulink pode ser observado na Figura 43. O torque gerado por esse bloco representa o torque fornecido pela TG em função da demanda de PLA e da rotação da própria turbina de potência. Para calcular a rotação da turbina de potência o modelo da TG é integrado ao modelo da redutora, conforme observado na figura abaixo.

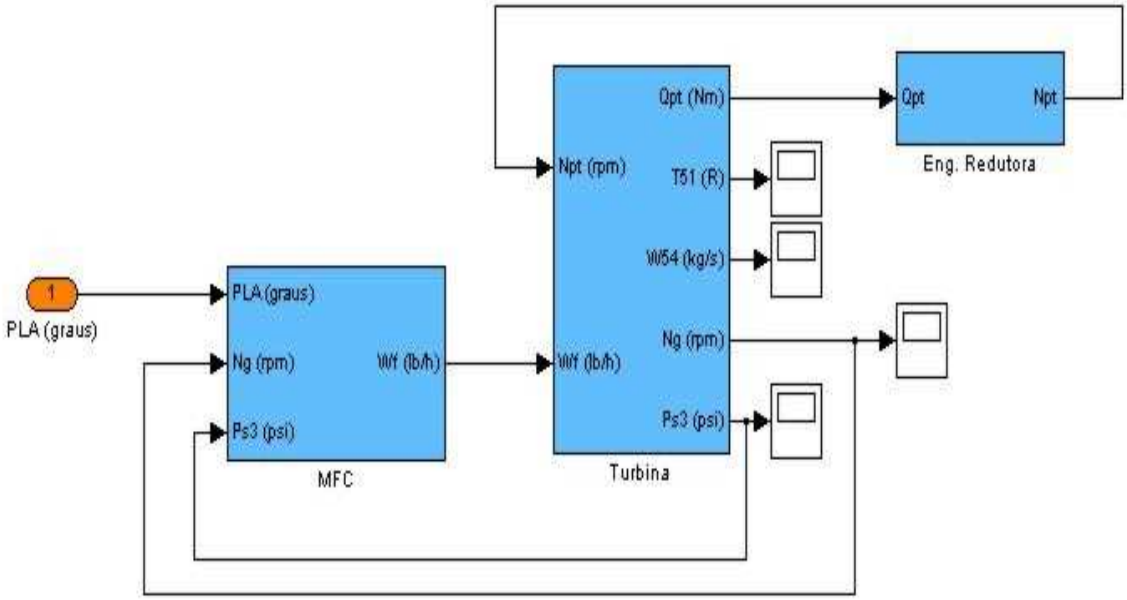

Figura 43 - Diagrama de blocos da TG.

4.4.3 Atuador e Controlador do Passo

O atuador de passo é o dispositivo responsável por posicionar as pás do hélice de acordo com o valor do passo demandado (*setpoint*). Por ser um dispositivo hidráulico, é o óleo injetado no cilindro o responsável pelo posicionamento desejado das pás do hélice. A Figura 44 representa esquematicamente o atuador de passo descrito.

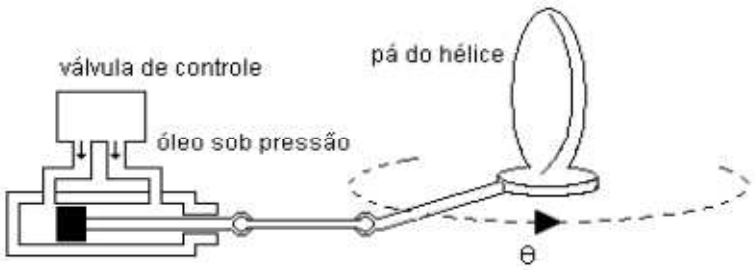

Figura 44 - Representação esquemática do atuador de passo (EPUSP, 2006).

Segundo (MORISHITA e BRINATI, 1984), a equação dinâmica do atuador de passo é dada por:

$$\frac{d\theta}{dt} = \begin{cases} F(N_h)[\theta - \theta_{ref}(t-2), & para\ |\theta - \theta_{ref}(t-2)| \leq 0.5^0 \\ F(N_h), & para\ |\theta - \theta_{ref}(t-2)| > 0.5^0 \end{cases} \quad (29)$$

Onde θ é o ângulo de passo real do hélice em graus, θ_{ref} é o valor de passo demandado (ângulo em graus) e $F(N_h)$ é o fator de aceleração que depende da velocidade angular do HPC. O sinal θ_{ref} tem um atraso de dois segundos que é próprio de um sistema hidráulico (EPUSP, 2006).

O bloco que representa o atuador de passo no ambiente Matlab/Simulink está representado na Figura 45.

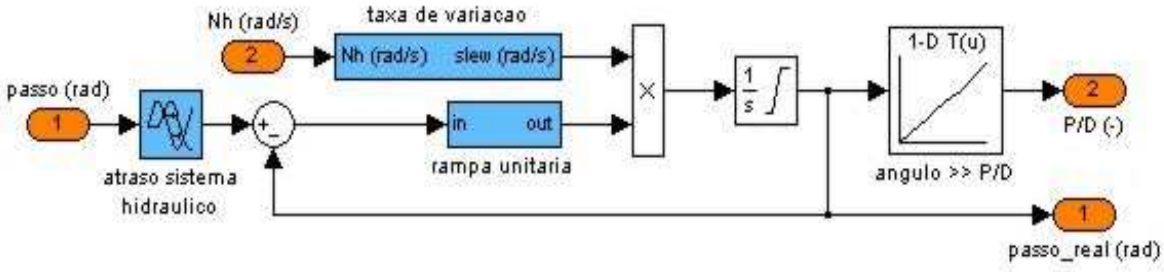

Figura 45 - Diagrama de blocos do HPC.

Conforme observado na Figura 45, além do posicionamento do passo do hélice, o modelo apresenta como saída a razão passo/diâmetro, denominada P/D. A razão P/D é uma grandeza adimensional, utilizada pelo modelo para integrar o hélice ao casco do navio com o objetivo de calcular o deslocamento da embarcação. A curva característica que a representa dependerá das configurações de fabricação do hélice utilizado.

4.4.4 Casco-Hélice do Navio

O modelo casco-hélice integra os modelos do casco do navio e do hélice acoplado ao eixo propulsor. Pela integração de tais modelos é possível simular a dinâmica de deslocamento do navio (MORISHITA e BRINATI, 1984) (HEINSLER, 2002) (CHAGAS, 1983).

As fórmulas de torque e do empuxo do hélice, utilizadas para o cálculo do deslocamento dinâmico do navio (equações 18 e 19), são definidas, respectivamente, por (MORISHITA e BRINATI,1984):

$$Q_h = \frac{(1+J^2)\rho N_h^2 D^5 KQ'(J', P/D)}{err(V)} \qquad (30)$$

$$T_{h\_BE} = T_{h\_BB} = (1 + J^2)\rho N_h^2 D^4 KT'(J', P/D)(1 - C(P/D)\tau(V) \qquad (31)$$

Onde,

- *J* é coeficiente de avanço do navio;
- *J´* é o coeficiente de avanço modificado do navio;
- ρ é a densidade da água do mar;
- N_h é a velocidade de rotação do eixo do hélice;
- *D* é o diâmetro do hélice;
- *KQ´(J´,P/D)* é o coeficiente de torque modificado, que é um adimensional obtido em ensaios de tanque de provas;
- *err(V)* é a eficiência relativa rotativa do hélice (obtida junto ao fabricante do hélice);
- *KT´(J´,P/D)* é o coeficiente de empuxo modificado, que é um adimensional obtido em ensaios de tanque de provas;
- *τ(V)* é o coeficiente de redução do empuxo em função da velocidade de deslocamento do navio; e
- *C(P/D)* é um coeficiente que corrige *τ(V)* para diferentes valores de P/D.

O diagrama de blocos com a representação do modelo casco-hélice e atuador de passo é apresentado na Figura 46.

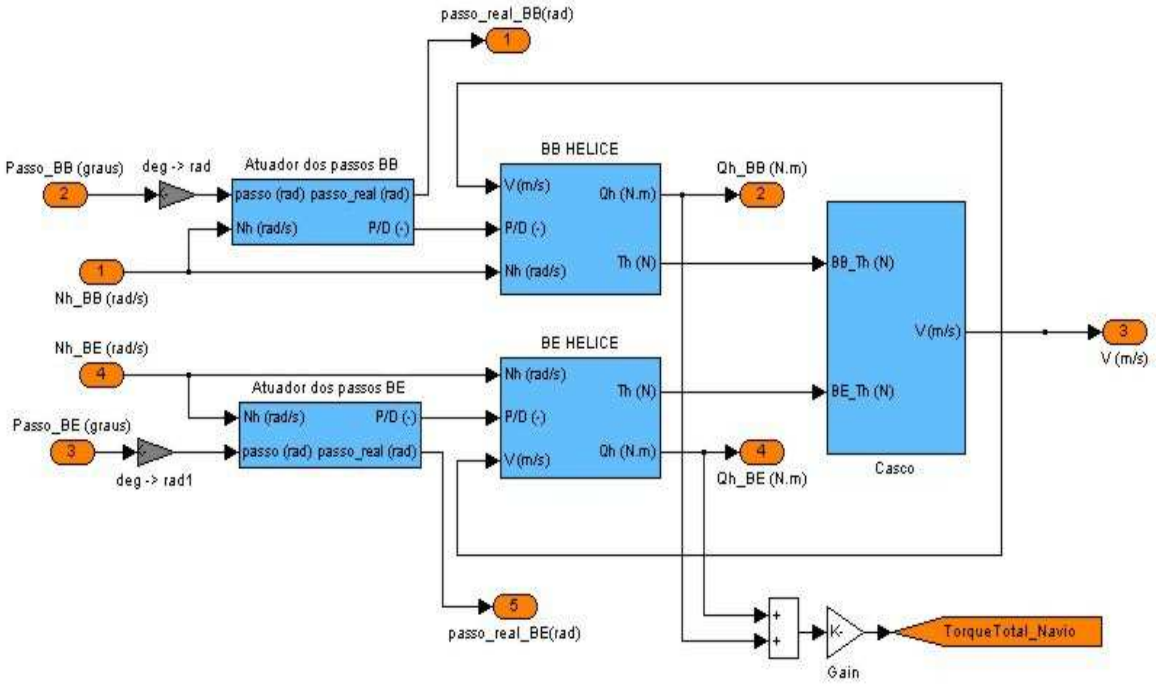

Figura 46 - Diagrama de blocos Casco-Hélice-HPC.

4.4.5 Controle de Velocidade do Navio

Em algumas aplicações navais que envolvem embarcações com sistemas de passo controlável, um dos objetivos principais do sistema de controle é possibilitar que a velocidade desejada da embarcação seja alcançada no menor intervalo de tempo possível durante as manobras de aceleração, sem, contudo, exceder os limites de torques máximos estabelecidos.

Em situações de não emergência, é desejável que as manobras de aceleração também levem em consideração o consumo mínimo de combustível pela máquina propulsora. Essa economia pode ser alcançada através da combinação propícia entre a potência demandada pela máquina propulsora e a posição das pás do hélice.

Usualmente, alguns dos métodos utilizados consistem em estabelecer uma matriz de valores, capaz de relacionar as possíveis velocidades do navio com o par de parâmetros “rotação/passo”, através de simulações e interpolações entre os valores obtidos. A partir desse conjunto de dados, os controladores analíticos são desenvolvidos, de maneira que a relação “passo/rotação” satisfaça às exigências estabelecidas pelo projetista ou a alguma outra necessidade do sistema.

Nesse experimento, utilizamos a metodologia proposta para desenvolver um controlador fuzzy que possibilite ao navio alcançar a velocidade final desejada no menor intervalo de tempo possível, sem que os limites de torques máximos permitidos pela instalação propulsora sejam ultrapassados.

O sistema de controle utilizado deve atender a determinados requisitos de aceleração especificados pelo projetista. Para isso, o controlador comanda os valores de PLA enviados ao controlador interno da turbina e os valores de passo enviados aos controladores do HPC de ambos os bordos do navio. Assim, o controlador fuzzy é caracterizado por possuir duas entradas (erro entre a velocidade final desejada e a velocidade atual do navio e a taxa de variação do erro) e duas saídas (demandas de PLA e passo). O diagrama de blocos implementado em Matlab/Simulink de todo o sistema pode ser observado na Figura 47, abaixo.

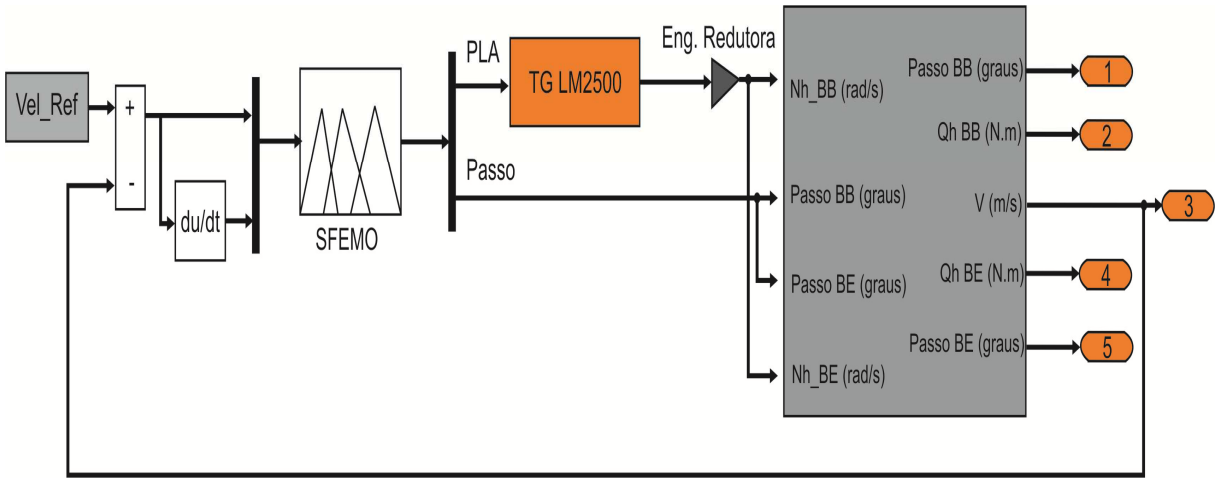

Figura 47 - Diagramas de bloco com controlador fuzzy para a planta do navio.

Para verificar a eficiência do método proposto, as respostas obtidas com o uso do SFEMO em circunstâncias específicas foram comparadas às respostas obtidas pelo uso de um controlador analítico convencional (EPUSP, 2006) através de simulações computacionais. Foram realizados testes de aceleração para diferentes velocidades do navio e alguns desses resultados são apresentados a seguir.

Durante as simulações os seguintes parâmetros foram considerados:

Tabela 17 - Parâmetros de simulação para controle do navio.

		Etapa 1	**Etapa 2**
AG	**Tamanho da População**	60	50
	Número de Gerações	50	30
	Taxa de *Crossover*	0.7	0.65
	Taxa de Mutação	0.02	0.1
	γ	-	0.1
	κ	-	0.8
Navio	**Máxima velocidade de projeto**	25 nós	
	Máximo torque no eixo do hélice	500 kN.m	
	Massa total do navio	2657 ton	

Aceleração de 0-12 nós:

A Figura 48 apresenta a resposta do SFEMO para a manobra de aceleração do navio de zero a doze nós. Na Figura 49 e na Figura 50 podemos observar a variação no torque vista pelo eixo do hélice e a potência instantânea na turbina de potência, respectivamente.

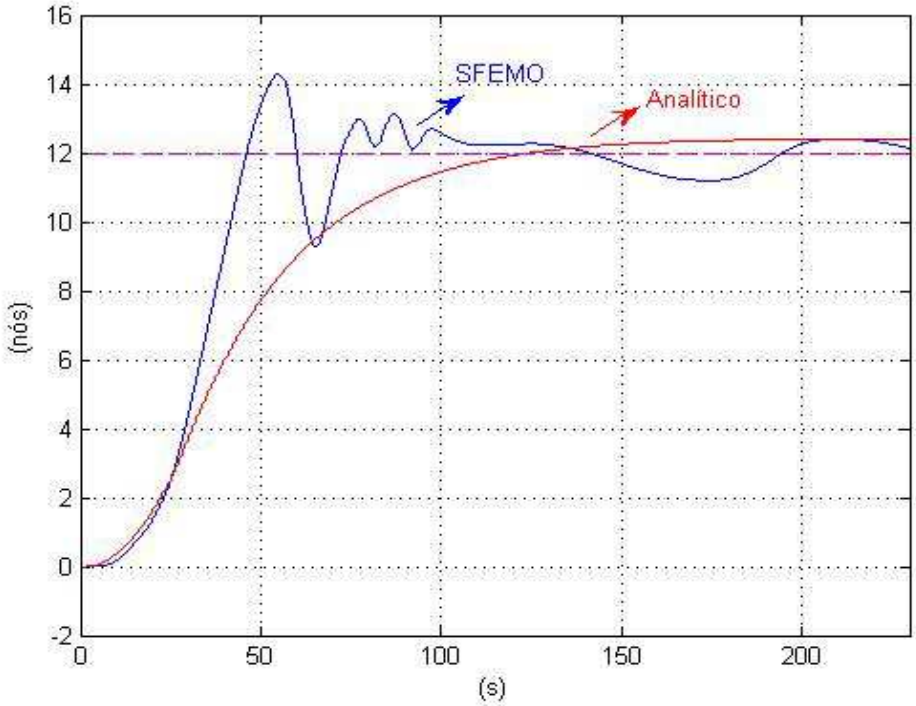

Figura 48 – Velocidade do Navio (0-12 nós).

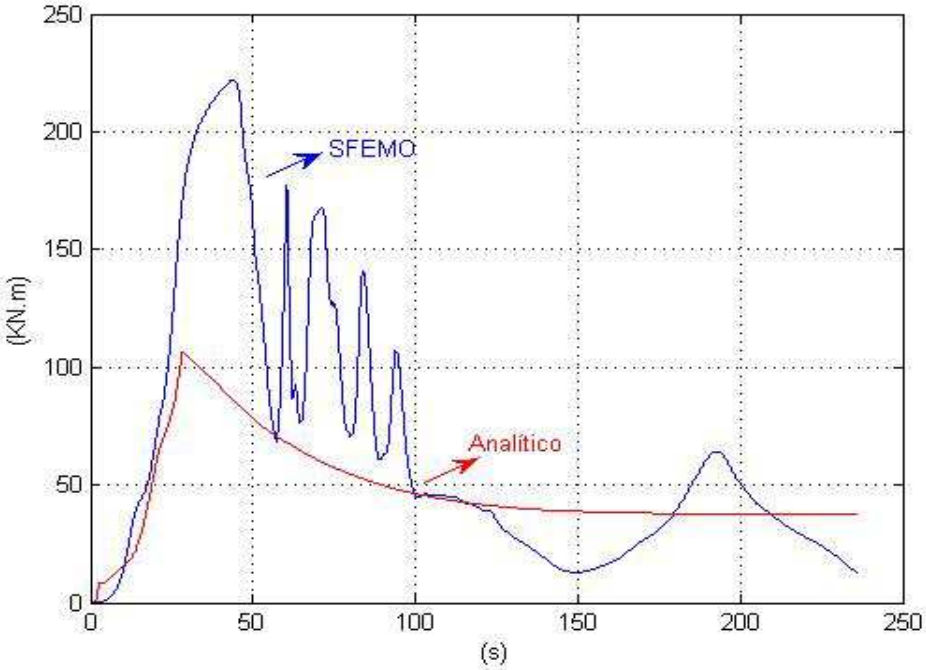

Figura 49 - Torque no eixo do hélice (0-12 nós).

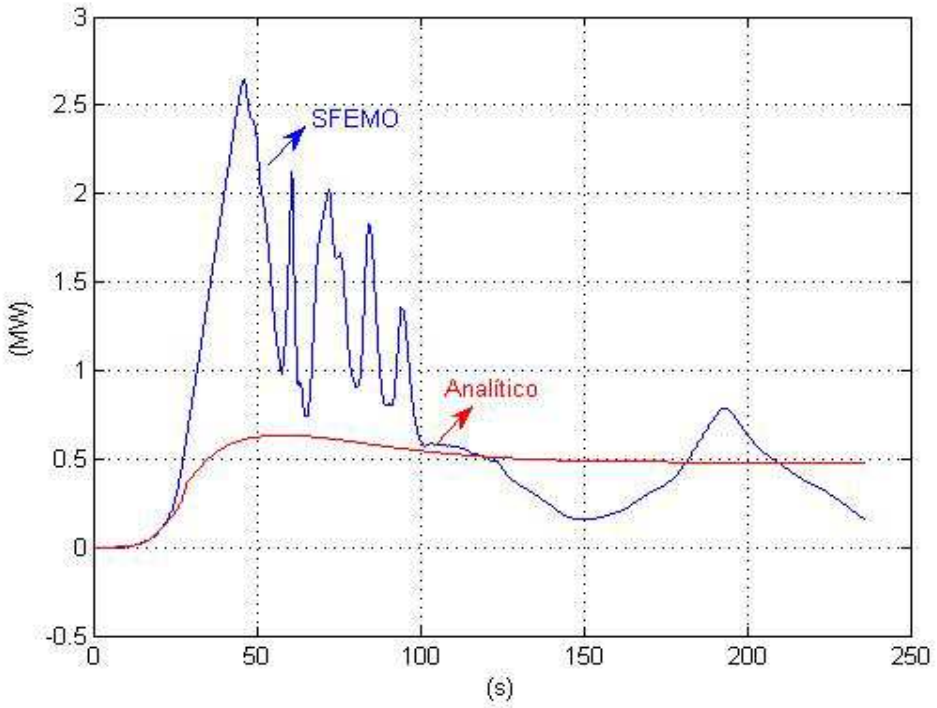

Figura 50 - Potência instantânea da TG (0-12 nós).

Crash-Stop:

A manobra de *crash-stop* consiste em demandar a parada do navio quando esse se encontra na máxima velocidade avante. O desempenho do controlador nessa manobra é avaliado com base na distância percorrida pelo navio até que esse alcance a velocidade nula após o comando de *crash-stop*. A Figura 51, a Figura 52 e a Figura 53 apresentam as respostas do SFEMO para a referida manobra.

A Tabela 18 apresenta uma comparação entre o controlador SFEMO e o controlador analítico convencional com relação à distância total percorrida e o tempo de parada do navio.

Tabela 18 - Distância percorrida (crash-stop).

Controlador	Distância Percorrida (m)	Tempo (s)
SFEMO	555.3	86
Analítico	480.3	150

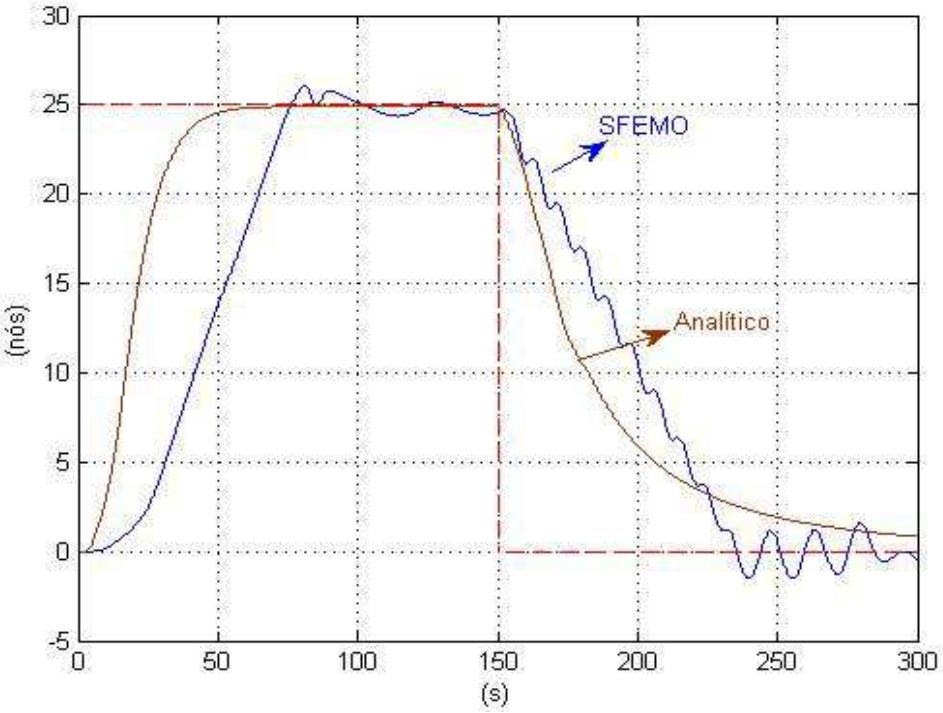

Figura 51 - Velocidade do navio (*crash-stop*)

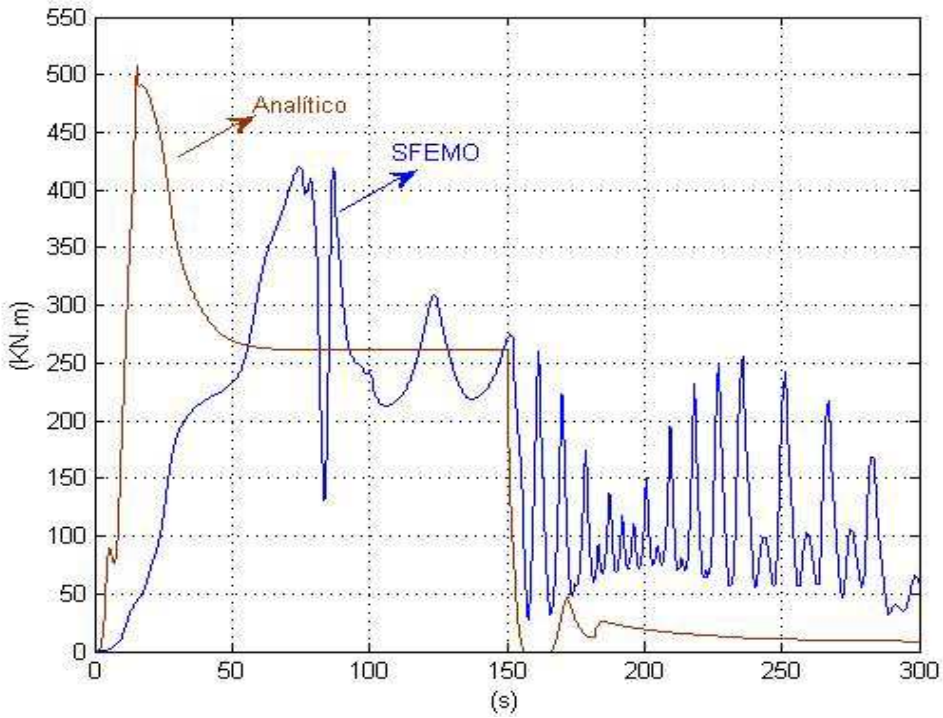

Figura 52 - Torque no eixo do hélice (crash-stop).

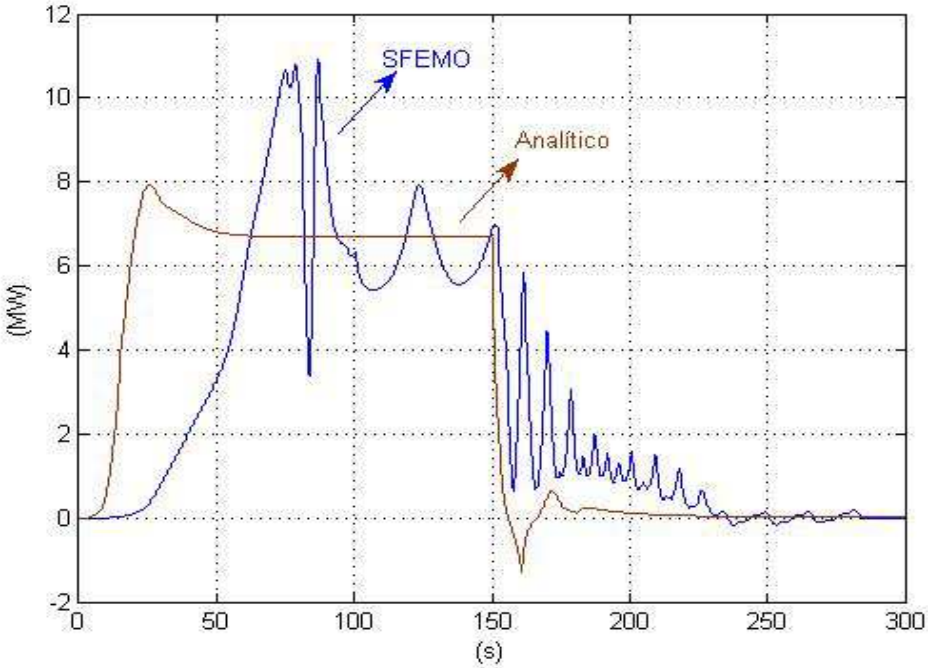

Figura 53 - Potência Instantânea na TG (crash-stop).

A Tabela 19 apresenta o conjunto de regras que constituem a base de regras do controlador SFEMO desenvolvido. Na Figura 54 podemos observar as respectivas funções de pertinência para as variáveis de entrada e saída do sistema.

Tabela 19 - Base de Regras para controle do navio.

	Posição de PLA (Y1)					Posição de Passo (Y2)				
	X21	X22	X23	X24		X21	X22	X23	X24
X11	Y12	Y11	-	-	X11	Y23	Y21	-	Y24
X12	-	Y11	Y12	-	X12	Y21	Y23	Y24	-
X13	Y11	Y13	-	-	X13	Y23	Y22	Y24	-

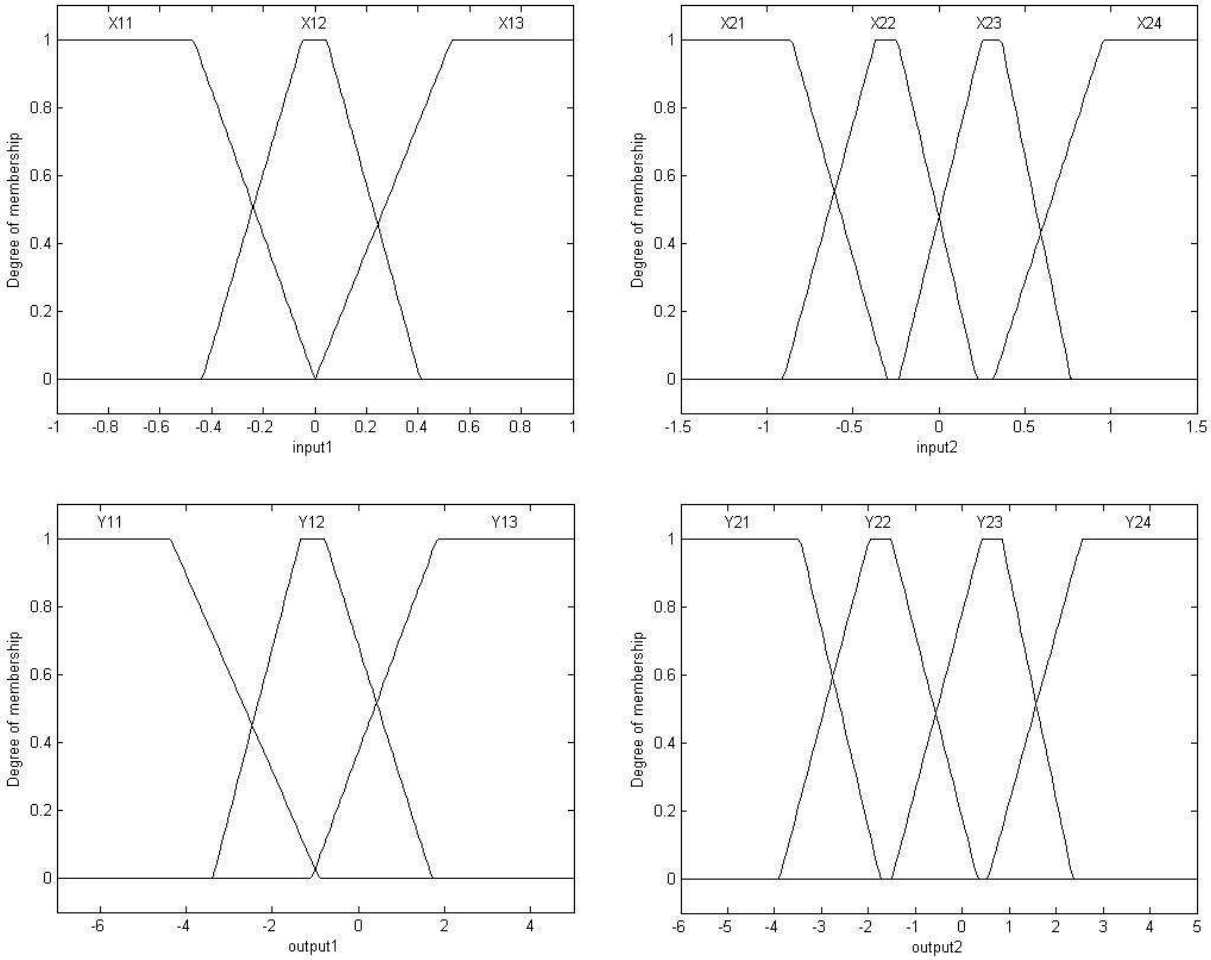

Figura 54 - MFs do controlador SFEMO para controle de velocidade do navio.

As respostas do SFEMO apresentaram muitas oscilações, principalmente quando comparadas às respostas obtidas pelo uso do controlador analítico. Porém, considerando os critérios de análise que foram definidos durante o processo de aprendizagem, podemos observar que o sistema fuzzy evoluído apresenta um bom desempenho, possibilitando que o navio alcance a velocidade final no menor intervalo de tempo possível, sem exceder aos limites de torques toleráveis pelo sistema de propulsão.

Cabe ressaltar que foi necessário um número máximo de quatro (4) funções de pertinência, além de uma base de regras constituída por nove (9) regras, para que o SFEMO fosse capaz de controlar este sistema complexo. Esta simplificação do controlador fuzzy é de grande valia para o entendimento do funcionamento e comportamento do sistema de propulsão analisado.

5 CONCLUSÕES E TRABALHOS FUTUROS

O inquestionável sucesso da lógica fuzzy pode ser constatado pelo crescente emprego dos sistemas fuzzy nas mais variadas áreas de atuação. Esse sucesso pode ser explicado pela habilidade que tais sistemas têm em representar o conhecimento humano de maneira simples e eficaz.

Porém, os métodos tradicionais de desenvolvimento de tais sistemas, baseados nas informações provindas de especialistas humanos ou através de técnicas de tentativa e erro, podem tornar esta tarefa bastante árdua. Em sistemas complexos, esses métodos podem até mesmo inviabilizar o desenvolvimento dos sistemas fuzzy.

Este trabalho apresenta uma metodologia que serve de auxílio aos projetistas de sistemas fuzzy, através do desenvolvimento de toda a base de conhecimento do sistema, constituída pela base de dados e pela base de regras. Para viabilizar a aprendizagem da base de conhecimento de tais sistemas foi empregada a técnica evolucionária dos algoritmos genéticos, pois sua eficiência e êxito no desenvolvimento de sistemas similares são amplamente reconhecidos na literatura científica.

Além da eficiência do sistema fuzzy evoluído, a metodologia proposta preocupou-se em desenvolver um sistema cuja base de conhecimento seja caracterizada por possuir uma boa interpretabilidade. Para isso, a interpretabilidade da base de conhecimento foi avaliada segundo a representatividade das funções de pertinência associadas às variáveis do sistema e ao número de regras que constituem a base de regras do sistema fuzzy. Durante a primeira etapa de execução do algoritmo, a representatividade das funções de pertinência é garantida pelo particionamento uniforme do universo de discurso de cada variável; durante a segunda etapa, na fase de otimização do sistema fuzzy, a representatividade é mantida pela escolha adequada, por parte do projetista, dos parâmetros que definem as propriedades de γ-completitude e κ-sobreposição das funções de pertinência. A interpretabilidade da base de regras, por sua vez, foi alcançada graças à possibilidade de inclusão de um termo nulo no conseqüente de cada regra, com a sua conseqüente exclusão da base de regras do sistema.

Como os sistemas fuzzy evoluídos foram avaliados com base em dois objetivos distintos, uma nova técnica de agregação foi incorporada ao processo evolucionário. Esta técnica de análise multiobjetivo, denominada Agregador Fuzzy, permitiu que os indivíduos fossem avaliados de uma maneira simples e eficiente. A utilização de termos lingüísticos

permite, dentre outras coisas, que as exigências e necessidades do projetista sejam mais facilmente incorporadas ao processo de aprendizagem do sistema fuzzy evoluído.

A metodologia proposta teve a sua eficiência comprovada em diferentes experimentos realizados durante este trabalho. Foram realizados experimentos relacionados à aproximação de funções e, principalmente, na área de controle. Neste último caso, respostas de diferentes plantas com diferentes graus de complexidade foram analisadas. Em todos os casos, a metodologia proposta proporcionou bons resultados quando comparada aos diferentes métodos existentes na literatura, principalmente, considerando os com maior interpretabilidade.

Por apresentar um amplo campo de estudo, diferentes possibilidades de trabalhos futuros podem ser incorporados à metodologia aqui proposta. Uma das possibilidades poderia ser um estudo que viabilize alterar a atual estrutura do algoritmo desenvolvido, para uma estrutura que permita a sua execução em paralelo. Esse paralelismo computacional permitiria que o processo de validação e simulação fosse acelerado, possibilitando que sistemas mais complexos sejam avaliados mais facilmente e que populações com um maior número de indivíduos sejam utilizadas.

Estudos futuros também poderiam alterar a metodologia de maneira que, ao final do ciclo evolucionário, um conjunto de possíveis soluções ótimas fosse apresentada ao projetista para análise. Para isso, o conceito de Conjunto Ótimo de Pareto deve ser incorporado e os resultados comparados.

Outro aspecto importante seria o estudo da implementação desta metodologia em hardware. Por permitir o desenvolvimento de sistemas fuzzy com um número reduzido de regras, o circuito eletrônico resultante teria o seu projeto simplificado, o que facilitaria o seu desenvolvimento em uma plataforma de hardware.

Apêndice A

Seguindo a nomenclatura utilizada em (MORISHITA e BRINATI, 1984), as funções F_4, F_6, F_7, F_9, F_{10}, F_{11}, F_{12}, F_{16}, F_{17} e F_{19} são utilizadas nas equações de estado do modelo matemático da turbina a gás GE LM2500. As figuras a seguir representam graficamente tais funções.

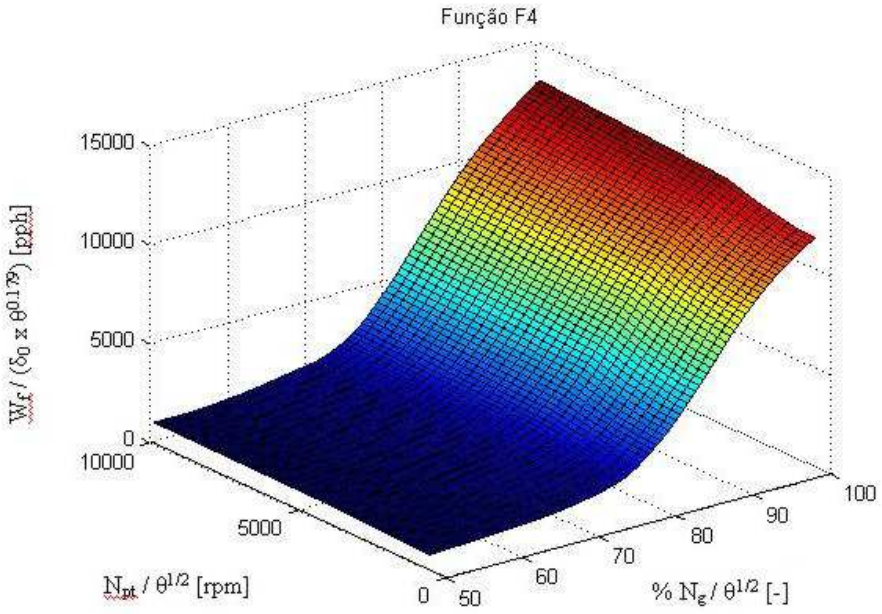

Figura 55 - Função *F4* do modelo da TG

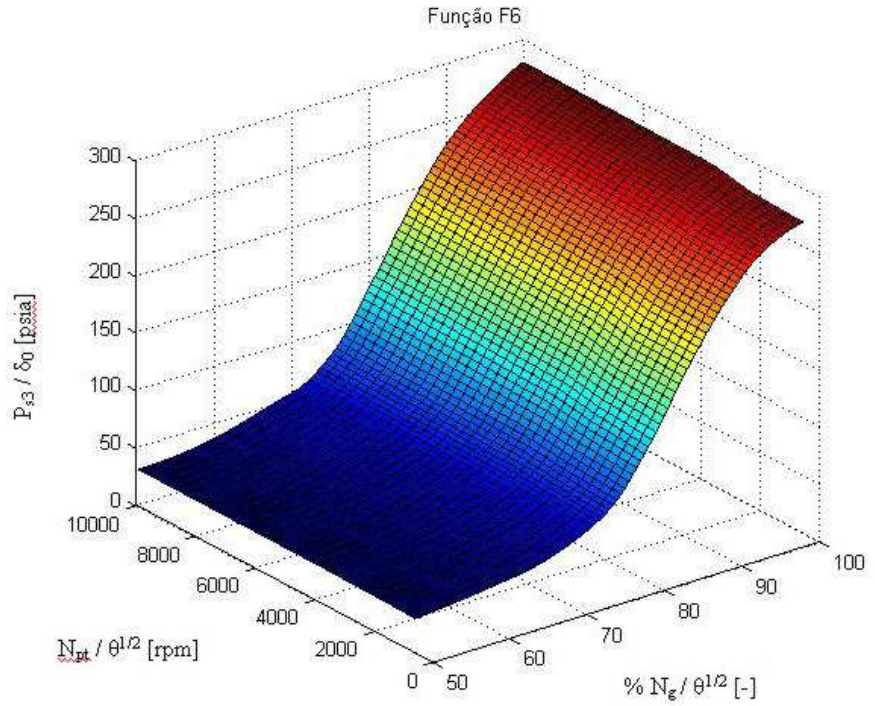

Figura 56 - Função *F6* do modelo da TG

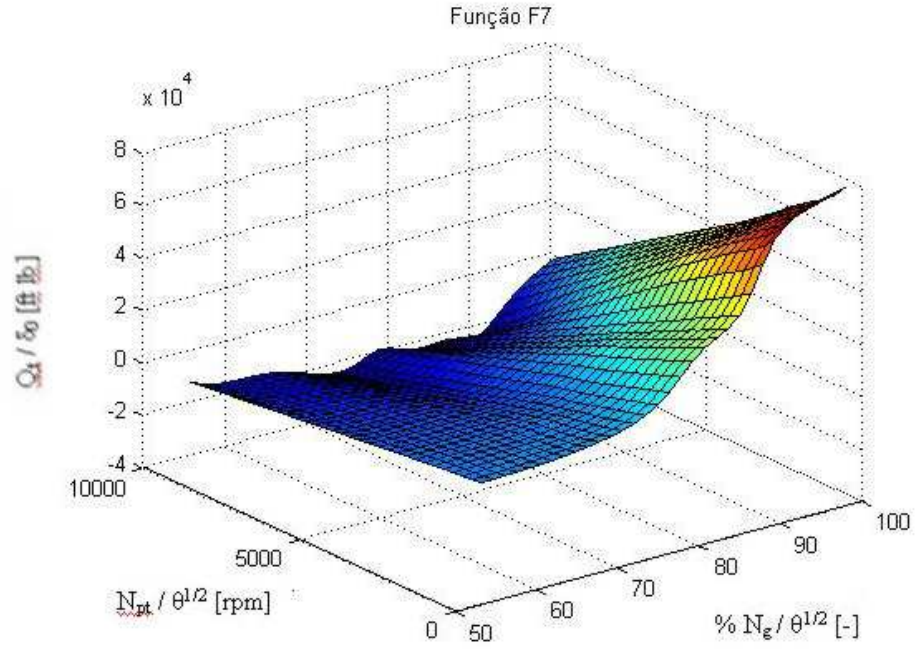

Figura 57 - Função *F7* do modelo da TG

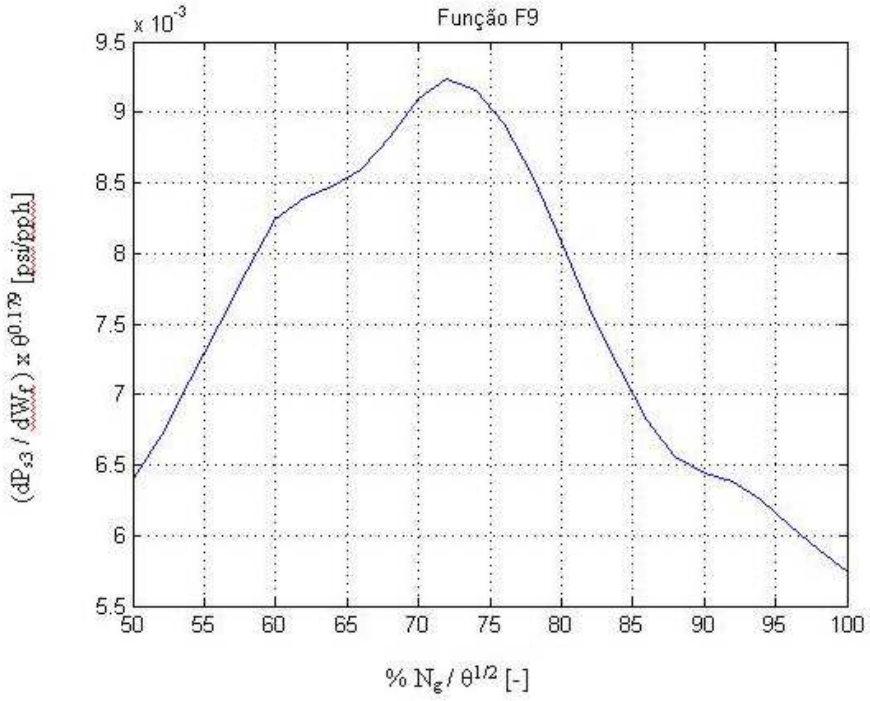

Figura 58 - Função *F9* do modelo da TG

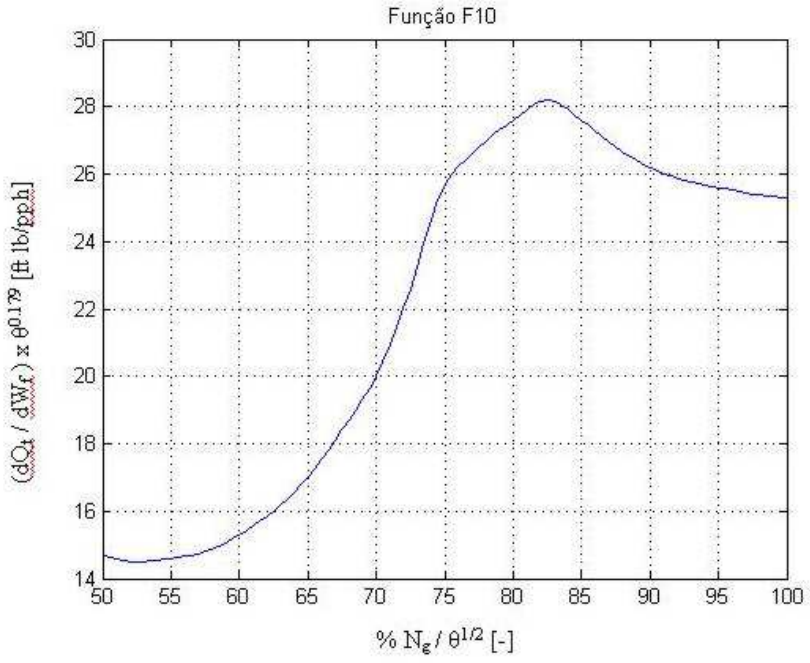

Figura 59 - Função *F10* do modelo da TG

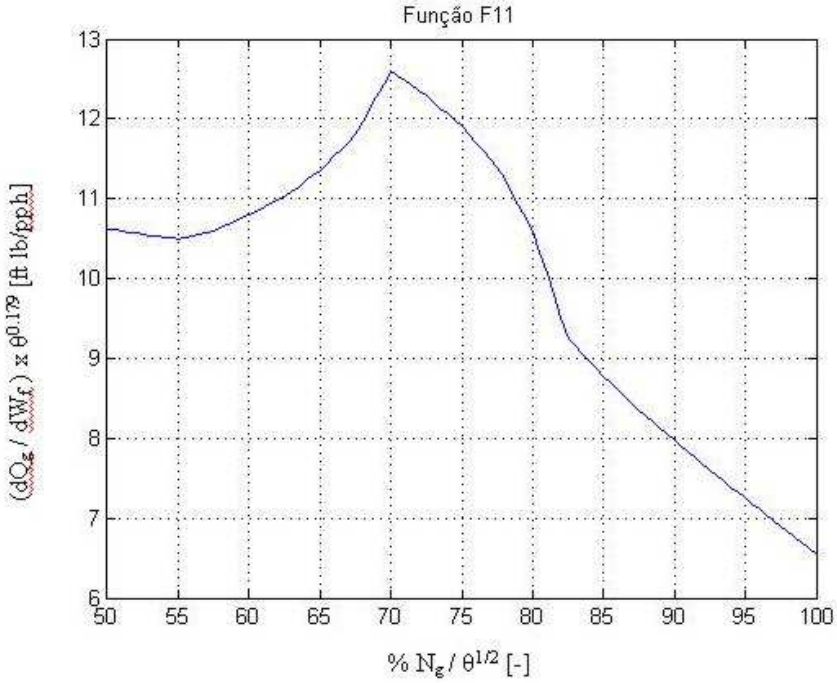

Figura 60 - Função *F11* do modelo da TG

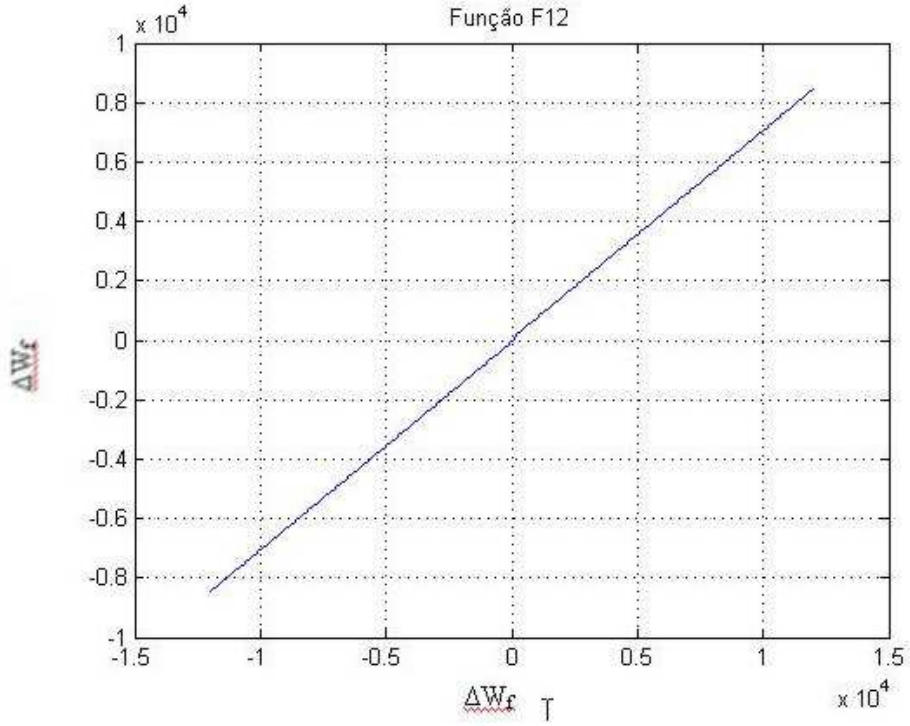

Figura 61 - Função *F12* do modelo da TG

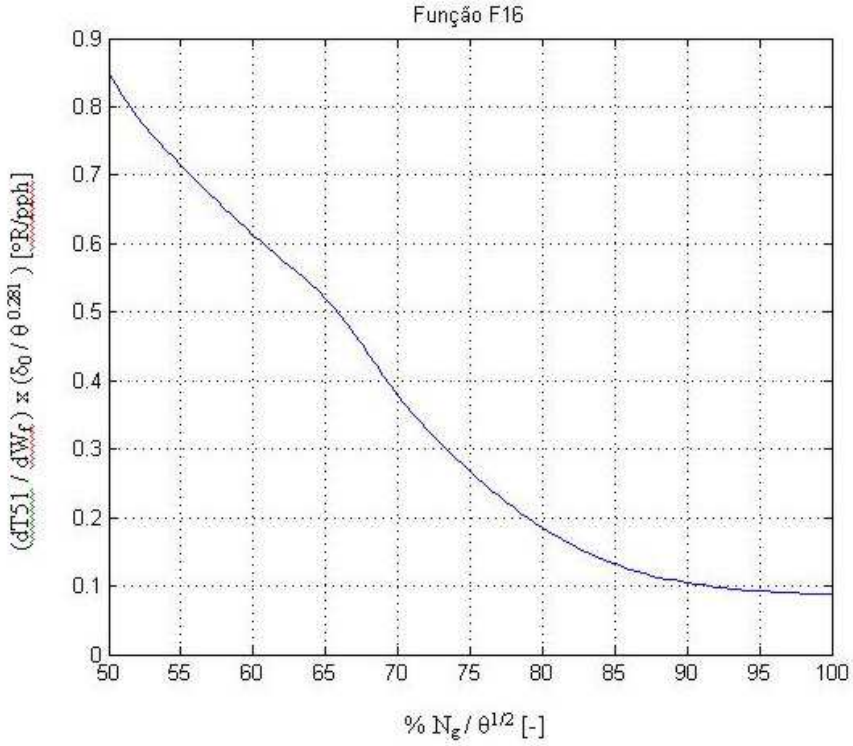

Figura 62 - Função *F16* do modelo da TG

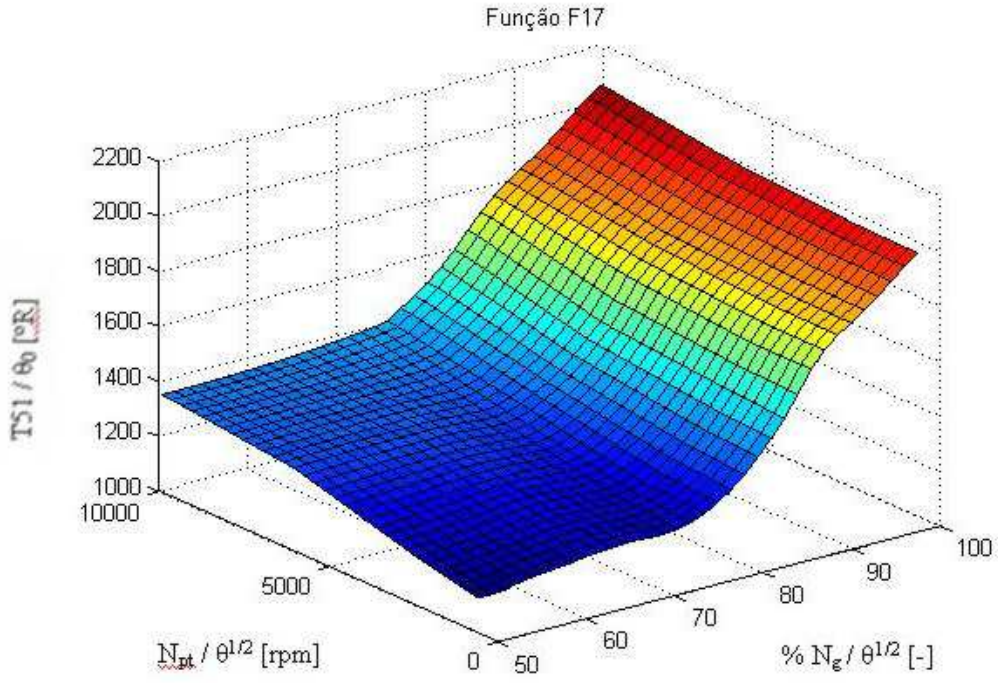

Figura 63 - Função *F17* do modelo da TG

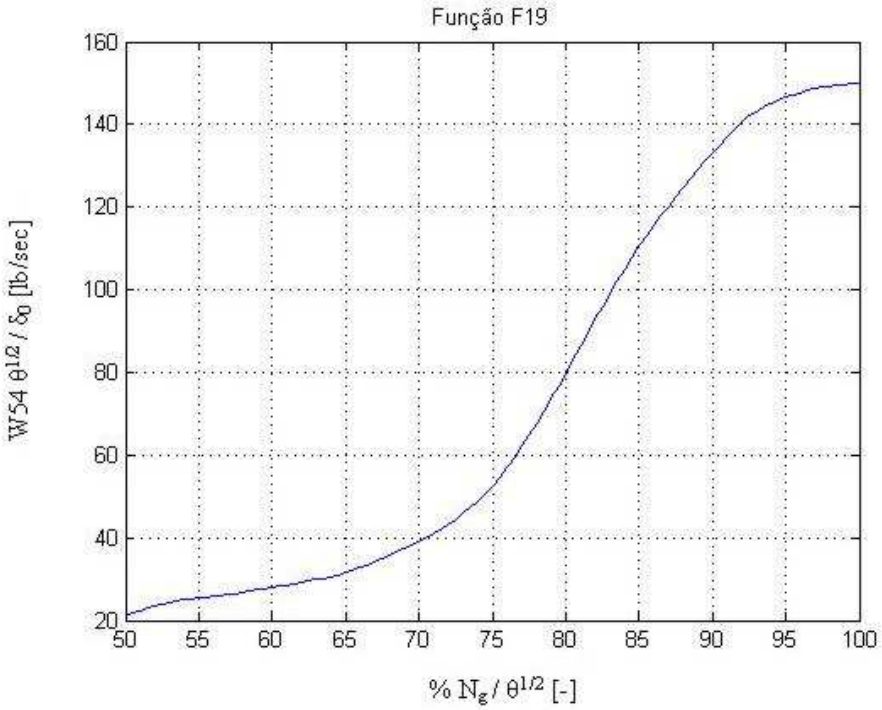

Figura 64 - Função *F19* do modelo da TG

REFERÊNCIAS

AMARAL, J. F. (2003). **Síntese de Sistemas Fuzzy por Computação Evolucionária**. *Tese de Doutorado, Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro (DEE-PUC-Rio)* , 132 p. Rio de Janeiro.

AMARAL, J. F., et al. (2002). **Evolutionary Fuzzy System Design and Implementation**. *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP '02), 4 v.* , 1872 - 1876 p. Singapore.

AUCHARIYAMET, S., & SIRISUMRANNUKUL, S. (2009). **Volt/VAr Control in Distribution Systems by Fuzzy Multiobjective and Particle Swarm**. *6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2009). 1 v.*, 234 - 237 p.

BANZHAF, W., FRANCONE, F., & NORDIN, P. (1997). **On Some Emergent Properties of Variable Size Evolutionary Algorithms**. *ICGA'97 Workshop on Evolutionary Computation with Variable Size Representation*.

BARDOSSY, A., & DUCKSTEIN, L. (1995). **Fuzzy Rule-Based Modeling With Application to Geophisical, Biological and Engineering Systems**. 256 p. Boca Raton, Florida: CRC Press.

BEZERRA, R. A., VELLASCO, M. M. e TANSCHEIT, R. (2005). **Hierarchical neuro-fuzzy BSP Mamdani system**. *XI International Fuzzy Systems Association World Congress (IFSA'05)*. 1321-1326 p. Beijing.

BLICKLE, T. (1996). **Theory of Evolutionary Algorithms and Application to System Synthesis**. *Dissertação de Doutorado, Swiss Federal Institute of Technology* . Zurique.

BONARINE, A., MASULLI, F., & PASI, G. (2003). **Soft Computing Applications (Advances in Intelligent and Soft Computing)**. 229 p. Physica-Verlag Heidelberg.

BONISSON, P. P., et al. (1999). **Hybrid Soft Computing Systems: Industrial and Commercial Applications**. *Proceedings of the IEEE , 87 v. , 9 ed.*, 1641 - 1667 p.

BONISSONE, P. P. (1999). **Soft Computing Systems: Commercial and Industrial Applications**. *IEEE International Fuzzy Systems Conference Proceedings*. Seoul.

BONISSONE, P. P., et al. (1995). **Industrial Applications of Fuzzy Logic at General Electric**. *Proceedings of the IEEE , 83 v. , 3 ed.*, 450 - 465 p.

BONISSONE, P. P., KHEDAKR, P. S., & CHEN, Y. (1996). **Genetic Algorithms for Automated Tuning of Fuzzy Controllers: A Transportation Application**. *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems , 1 v.* , 674 - 680 p.

CAMARGO, H. A., PIRES, M. G., & CASTRO, P. A. (2004). **Genetic Design of Fuzzy Knowledge Bases – a study of different approaches**. *IEEE Annual Meeting of the Fuzzy Information (NAFIPS '04) , 2 v.* , 954- 959 p.

CHAGAS, P. S. (1983). **Proposta de Modelagem, Simulação e Determinação de Parâmetros de Controle para uma Instalação Propulsora do tipo C.O.D.O.G.**. *Dissertação de Mestrado. Escola Politécnica da USP*. São Paulo.

CORDÓN, O., HERRERA, F., & VILLAR, P. (2001). **IEEE Transactions on Fuzzy Systems**. *9 v.* , 667 - 674 p.

CORDÓN, O., et al. (2001a). **A Multiobjective Genetic Alorithm for Feature Selection and Granularity Learning in Fuzzy-Rule Based Classification Systems**. *IFSA World Congress and 20th NAFIPS International Conference , 3 v.* , 1253 - 1258 p.

CORDÓN, O., et al. (2001b). **Ten Years of Genetic Fuzzy Systems: Current Framework and New Trends**. *IFSA World Congress and 20th NAFIPS International Conference , 3 v.* , 1241 - 1246 p.

CORDÓN, O., et al. (2001c). **Genetic Fuzzy Systems – Evolutionary Tuning and Learning of Fuzzy Knowledge Bases**. *19 v.* , 462 p. World Scientific Publishing Company.

CORMEN, T. H., LEISERSON, C. E., & RIVEST, R. L.. **Introduction to Algorithms**. *3 ed.*, 1312 p. The MIT Press.

DE JONG, K. (1988). **Learning with Genetic Algorithms: An Overview**. *Machine Learning Journal , 3 v.* , 121-138 p. Springer Netherlands.

DEB, K. (2001). **Multi-Objective Optimization Using Evolutionary Algorithms**. 518 p. Wiley.

DELGADO, M. R. (2002). **Projeto Automático de Sistemas Nebulosos: Uma Abordagem Coevolutiva**. *Tese de Doutorado. Faculdade de Engenharia Elétrica e Computação. UNICAMP*. Campinas.

DRIANKOV, D., HELLENDOORN, H., & REINFRANK, M. (1996). **An Introduction to Fuzzy Control**. *(2 ed.)* , 316 p. New York: Springer.

EPUSP. (2006). **Modelagem e Simulação do Sistema de Propulsão - Corveta Barroso**. *Laboratório de automação e Controle da Escola Politécnica da USP*. São Paulo.

FONSECA, C., et al. (2003). **Evolutionary Multi-Criterion Optimization**. *Second International Conference (EMO 2003). Lecture Notes in Computer Science* , 812 p. (Springer, Ed.).

HARVEY, I. (1993). **The Artificial Evolution of Adaptive Behavior**. *Tese de Doutorado. University of Sussex, School of Cognitive and Computing Sciences (COGS).* England.

HAUPT, R. L. (1998). **Practical Genetic Algorithms**. *2 ed*, 272 p. Wiley-Interscience.

HEISLER, H. (2002). **Advanced Vehicle Technology**. *2 ed.*, 654 p. Society of Automotive Engineers Inc.

HOLLAND, J. H. (1992). **Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence**. 228 p. The MIT Press.

KOZA, J. (1992). **Genetic Programming: On the Programming of Computers by Means of Natural Selection**. 840 p. MIT Press.

LAZO, et al. (1999). **Controle Nebuloso de um Manipulador Robótico com Ajuste Heurístico da Base de Regras**. *Encontro Nacional de Inteligência Artificial, XIX Congresso Nacional da Sociedade Brasileira de Computação. 4 v.*, 471-480 p. Rio de Janeiro.

MEUNIER, B. B., YAGER, R. R., & ZADEH, L. A. (1994). **Fuzzy Logic and Soft Computing (Advances in Fuzzy Systems : Applications and Theory)**. *4 v.* , 497 p. World Scientific Publishing Company.

MILLER, J. F., THOMSON, P., & FOGARTY, T. (1997). **Genetic Algorithms Recent Advancements and Industrial Applications**. *Designing Electronic Circuits Using Evolutionary Algorithms. Arithmetic Circuits: A Case Study* , Cap. 6. D. Quagliarella, J. Periaux, C. Poloni and G. Winter, Wiley.

NÜRNBERGER, A., NAUCK, D., & KRUSE, R. (1999). **Soft Computing**. *Neuro-Fuzzy Control Based on the NEFCON-Model , 2 v.* , 182 - 186 p. Berlin: Springer.

OGATA, K. (2003). **Engenharia de Controle Moderno**. *4 ed.*, 788 p. São Paulo: Prentice Hall.

PACHECO, M. A. (1996). **Notas do Curso de Algoritmos Genéticos**. *Departamento de Engenharia Elétrica, PUC-RJ*. Rio de Janeiro.

PEDRYCZ, W. (1993). **Fuzzy Control and Fuzzy Systems (Control Theory and Applications, 3) (Electronic and Electrical Engineering Research Studies: Control Theory and Applications)**. 2 ed, 368 p. Research Studies Pre.

ROSS, T. J. (1995). **Fuzzy Logic with Engineering Applications**. 2 ed., 650 p. Wiley.

SMITH, S.F. (1980). **A Learning System Based on Genetic Adaptive Algorithms**. *Doctoral Dissertation, Department of Computer Science, University of Pittsburgh.*USA.

SOUZA, F. J. (1999) **Modelos Neuro-Fuzzy Hierárquicos.** *Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro*. Rio de Janeiro.

TAKAGI, T. e SUGENO, M. (1985). **Fuzzy Identification of Systems and Its Applications**. *IEEE Transactions on Systems, Man and Cybernetics. 15 v.*, 116-132 p.

TANSCHEIT, R. (1998). **Lógica Fuzzy, Raciocínio Aproximado e Macanismos de Inferência.** *17º Encontro Nacional de Automática*. 46-55 p. Brasil.

TANSCHEIT, R. (2003). **Sistemas fuzzy**. *VI Simpósio Brasileiro de Automação Inteligente (SBAI'03), Anais de Minicursos*. 35 p. Brasil.

ZEBULUM, R. S. (1999). **Síntese de Circuitos Eletrônicos por Computação Evolutiva**. *Tese de Doutorado, Departamento de Engenharia Elétrica, PUC-Rio* . Rio de Janeiro.