METODOLOGIA PARA MONITORAMENTO INTELIGENTE DE CONDIÇÃO DE MÁQUINA: UMA ABORDAGEM USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

Iván Patricio Moreno Marcos

DISSERTAÇÃO DE MESTRADO EM SISTEMAS MECATRÔNICOS
DEPARTAMENTO DE ENGENHARIA MECÂNICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

METODOLOGIA PARA MONITORAMENTO INTELIGENTE DE CONDIÇÃO DE MÁQUINA: UMA ABORDAGEM USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

IVÁN PATRICIO MORENO MARCOS

ORIENTADOR: ALBERTO JOSÉ ÁLVARES

DISSERTAÇÃO DE MESTRADO EM SISTEMAS MECATRÔNICOS

PUBLICAÇÃO: ENM.DM – 10J/12
BRASÍLIA/DF: JANEIRO – 2012

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA MECÂNICA

METODOLOGIA PARA MONITORAMENTO INTELIGENTE DE CONDIÇÃO DE MÁQUINA: UMA ABORDAGEM USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

IVÁN PATRICIO MORENO MARCOS

DISSERTAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA MECÂNICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISÍTOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM SISTEMAS MECATRÔNICOS

APROVADA POR:

_____

Prof. Alberto José Álvares, Dr. Eng. (ENM-UnB)
(Orientador)

_____

Prof. Li Weigang, Dr. Eng. (CIC-UnB)
(Examinador Interno)

_____

Prof. Alexandre Ricardo Soares Romariz, Dr. Eng. (ENE-UnB)
(Examinador Externo)

BRASÍLIA/DF, 10 DE JANEIRO DE 2012

FICHA CATALOGRÁFICA

MARCOS, IVÁN PATRICIO MORENO
Metodologia para Monitoramento Inteligente de Condição de Máquina: Uma Abordagem usando Funções de Pertinência *Fuzzy* [Distrito Federal] 2012.
xxi, 153p., 210 x 297 mm (ENM/FT/UnB, Mestre, Sistemas Mecatrônicos, 2012).
Dissertação de Mestrado – Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Mecânica.
1. Funções de pertinência *fuzzy* 2. Manutenção preditiva
3. Regras de produção *fuzzy* 4. Monitoramento de condição
I. ENM/FT/UnB II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

MORENO, I. P. (2011). Metodologia para Monitoramento Inteligente de Condição de Máquina: Uma Abordagem usando Funções de Pertinência *Fuzzy*. Dissertação de Mestrado em Sistemas Mecatrônicos, Publicação ENM.DM-10E/12, Departamento de Engenharia Mecânica, Universidade de Brasília, Brasília, DF, 153p.

CESSÃO DE DIREITOS

AUTOR: Iván Patricio Moreno Marcos.

TÍTULO: Metodologia para Monitoramento Inteligente de Condição de Máquina: Uma Abordagem usando Funções de Pertinência *Fuzzy*.

GRAU: Mestre ANO: 2012

Iván Patricio Moreno Marcos
SCLN 407 BLOCO C, APARTAMENTO 25
70.855-530 Brasília – DF – Brasil.

DEDICATÓRIA

“Dedico esta humilde dissertação de mestrado a Deus em primeiro lugar, pois sem essa fé que me dá diariamente não poderia ter concluído esta pesquisa. Aos meus pais Lidia e Euclides, que estando longe, sempre tiveram palavras de coragem para mim, a minha adorável irmãzinha Liz, que ficando longe também, me ajudou a superar vários problemas talvez críticos que aconteceram durante a realização deste trabalho e por último à paixão da minha vida Beatriz”.

AGRADECIMENTOS

Agradeço...

Em primeiro lugar a Deus por essa inspiração que me dá para poder enxergar as trilhas mais férteis da minha vida.

À minha família por todos esses desejos e palavras de força que sempre me acompanham mesmo que eles fiquem temporariamente longe de mim.

Ao meu orientador o Prof. Dr. Eng. Alberto José Álvares por todas suas observações, dicas, críticas e transmissão de conhecimentos nas disciplinas que cursei com ele, e sobre tudo o que para mim é mais importante, a fé e confiança depositada em minha pessoa.

Ao laboratório GRACO (Grupo de Automação e Controle) e ao Departamento de Engenharia Mecânica da Universidade de Brasília (UnB) pela tecnologia disponibilizada.

Ao Eng. Antonio Araujo da Eletronorte e ao pessoal da Usina Hidrelétrica de Balbina pela estadia outorgada na última visita e pelos conhecimentos transmitidos respectivamente.

A todos os professores durante o primeiro ano de ensino por suas contribuições de conhecimento que me ajudaram a esboçar e delimitar gradativamente meus interesses profissionais e temas de pesquisa.

Aos meus colegas de casa e de laboratório por esses momentos de conversa que ajudaram a sanar algumas dúvidas e como é claro pela amizade brindada.

Como última instância, não sendo menos importante do que as outras, ao CNPQ (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelo apoio financeiro durante esses dois anos de pesquisa sob o Processo: 136028/2009-8.

RESUMO

METODOLOGIA PARA MONITORAMENTO INTELIGENTE DE CONDIÇÃO DE MÁQUINA: UMA ABORDAGEM USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

O alto desempenho requerido nas grandes usinas hidrelétricas no Brasil, com o objetivo de gerar um serviço de alta qualidade, tem suscitado, principalmente na área de manutenção de processos, conhecer e integrar nos seus sistemas, novas técnicas modernas de monitoramento de condição, diagnóstico e prognóstico para avaliação contínua de parâmetros chaves do estado de funcionamento de máquinas com o objetivo de detectar eventuais falhas ou defeitos e gerar um conjunto de ações de manutenção preditiva para incrementar a produtividade do processo. O presente trabalho se propõe a apresentar uma solução viável nesse contexto e tem como objetivo desenvolver uma metodologia para a concepção de um sistema de monitoramento de condição inteligente baseado em regras de produção usando funções de pertinência *fuzzy* visando melhorar a confiabilidade dos estados de máquinas para incrementar sua disponibilidade e produtividade. Para a implementação computacional da metodologia, criou-se um conjunto de regras de produção *fuzzy* na interface ECLIPSE usando a biblioteca *FuzzyJess*. Finalmente, um estudo de caso, de forma não integrada à base de conhecimento do SIMPREBAL (Sistema Inteligente de Manutenção Preditiva de Balbina), foi realizado com o histórico de dados das grandezas físicas monitoradas pelo SIMPREBAL na usina hidrelétrica de Balbina para avaliar o desempenho da nova base de regras *fuzzy* e sua futura integração na base de conhecimento do SIMPREBAL. Os resultados deste estudo sugerem que a base de regras *fuzzy* proposta, além de poder ser integrada no SIMPREBAL, possui um grande potencial na detecção do correto estado funcionamento de máquinas com o intuito de avaliar melhor as tendências em diagnóstico e prognóstico de possíveis falhas ou defeitos.

ABSTRACT

METHODOLOGY FOR INTELLIGENT MONITORING OF MACHINE CONDITION: AN APPROACH BY USING FUZZY MEMBERSHIP FUNCTIONS

The high performance needed in the large hydroelectric power plants from Brazil, demand knowing and integrating on their systems, principally on the process maintenance area, new modern techniques of condition monitoring, diagnostic and prognostic for continuous assessment of key performance parameters related to running state of machines with the aim of detecting possible failures or faults and generating a set of predictive maintenance actions to increase the productivity of the process. This work presents a feasible solution on that context and it has as main objective to develop a methodology for the conception of an intelligent condition monitoring system based on production rules by using fuzzy membership functions aiming to improve the reliability of the running states of machines for increase their availability and production. For the computational implementation of the methodology, we built a set of fuzzy production rules in the ECLIPSE interface by using FuzzyJess package. Finally, a study case, non-integrated to the knowledge base of SIMPREBAL (Balbina's Predictive Maintenance Intelligent System), was executed with the reading log of the monitored physical variables by SIMPREBAL on the Balbina's hydroelectric power plant to assess the performance of the new fuzzy rule base and its future integration on the knowledge base of SIMPREBAL. The results of this study suggest that the new fuzzy rule base, is not only able to integrate to SIMPREBAL, but also possess a great potential on the detection of correct running state of machines with the objective to assess better their trends on diagnosis and prognosis of failures or faults.

RESUMEN

METODOLOGÍA PARA MONITORAMIENTO INTELIGENTE DE CONDICIÓN DE MÁQUINA: UNA ABORDAJE USANDO FUNCIONES DE MEMBRECÍA *FUZZY*

El alto desempeño requerido en centrales hidroeléctricas del Brasil, con el objetivo de generar un servicio de alta calidad, ha originado, en el área de manutención de procesos, conocer e integrar en sus sistemas, nuevas técnicas modernas de monitoramiento de condición, diagnóstico y pronóstico para evaluar continuamente parámetros claves del estado de funcionamiento de máquinas con el objetivo de detectar eventuales fallas o defectos e generar un conjunto de acciones de manutención predictiva para aumentar la productividad del proceso. El presente trabajo se propone a presentar una solución viable en ese contexto e tiene como objetivo desenvolver una metodología para la concepción de un sistema de monitoramiento de condición inteligente basado en reglas de producción usando funciones de membrecía *fuzzy* apuntando a mejorar la confiabilidad de los estados de funcionamiento de máquinas para incrementar su disponibilidad y productividad. Para la implementación computacional de la metodología, se construyó un conjunto de reglas de producción *fuzzy* en la interface ECLIPSE usando la biblioteca *FuzzyJess*. Finalmente, un estudio de caso, de forma no integrada a la base de conocimiento del SIMPREBAL (Sistema Inteligente de Manutención Predictiva de Balbina), fue realizada con el histórico de datos de las variables físicas monitoreadas por el SIMPREBAL en la fábrica hidroeléctrica de Balbina con el objetivo de evaluar el desempeño de la nueva base de reglas *fuzzy* y su futura integración en la base de reglas del SIMPREBAL. Los resultados de este estudio sugieren que la base de reglas *fuzzy*, además de poder integrarse al SIMPREBAL, posee un gran potencial en la detección del correcto estado de funcionamiento de máquinas con el objetivo de evaluar mejor las tendencias en diagnóstico e pronóstico de fallas posibles.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE SÍMBOLOS, NOMENCLATURAS E ABREVIAÇÕES

JESS	- *Java Expert System Shell*.
API	- *Application Programming Interface*.
SE	- Sistema Experto ou Sistema Especialista.
CLIPS	- *C Language Integrated Production System*.
SIMPREBAL	- Sistema Inteligente de Manutenção Preditiva de Balbina.
UOD	- *Universe Of Discourse*.
OSA-CBM	- Open System Architecture for Condition Based Maintenance
CBM	- *Condition Based Maintenance*
CM	- *Condition Monitoring*
GUI	- *Graphical User Interface*
FF	- *Foundation Fieldbus*
IA	- Inteligência Artificial
JDBC	- *Java DataBase Connectivity*
SCADA	- *Supervisory Control and Data Acquisition*
OPC	- *Object Linking and Embedding for Process Control*
FMEA	- *Failure Mode and Effect Analysis*
UGH	- Unidade Geradora Hidráulica
AI	- *Artificial Intelligence*
KBS	- *Knowledge-Based System*
RCM	- *Realiability-Centered Maintenance*
ANN	- *Artificial Neural Network*
CBR	- *Case-Based Reasoning*
MCC	- Manutenção Centrada na Confiabilidade
MBC	- Manutenção Baseada em Condição
MPd	- Manutenção Preditiva
GA	- *Genetic Algorithms*
FLS	- *Fuzzy Logic Systems*
FNN	- *Fuzzy Neural Networks*
SPC	- *Statistical Process Control*
SVM	- *Support Vector Machine*
AR	- *Auto Regressive*

ARMA	- *Auto Regressive Moving Average*
ARIMA	- *Auto Regressive Integrated Moving Averages*
FFT	- *Fast Fourier Transformation*
PCA	- *Principal Component Analysis*
ICA	- *Independent Component Analysis*
HMM	- *Hidden Markov Models*
KPI	- *Key Performance Indicator*
OO	- *Oriented Object*
OMG	- *Object Management Group*
CORBA	- *Common Object Request Broker Architecture*
JNI	- *Java Native Interface*
ISO	- *International Organization for Standardization*
MIMOSA	- *Machinery Information Management Open System Alliance*
CMMS	- *Computerized Maintenance Management System*
PHM	- *Prognostic Health Management*
BSI	- *British Standards Institution*
CMMS	- *Computerized Maintenance Management System*
DCS	- *Distributed Control System*
IMS	- *Information Management System*
MLP	- *Multi Layer Perceptron*
RST	- *Rough Set Theory*
JVM	- *Java Virtual Machine*
RUL	- *Remaining Useful Life*
PA	- *Prognostics Assessment*
FST	- *Fuzzy Sets Theory*
TCF	- Teoria dos Conjuntos *Fuzzy*
ART	- *Adaptive Resonance Theory*
RNN	- *Recurrent Neural Network*
SIMAP	- *Intelligent System for Predictive Maintenance*
IPDSS	- *Intelligent Predictive Decision Support System*
IMMS	- *Integrated Maintenance Management System*
DFI	- *Fieldbus Universal Bridge*
FES	- *Fuzzy Expert System*
COG	- *Center of Gravity*

SAM	- *Standard Additive Method*
TSK	- *Takagi-Sugeno-Kang*
FK	- Foreign Keys
ER	- Employee Room
MC	- Monitoramento de Condição

TRABALHOS PUBLICADOS PELO AUTOR

Moreno, I.P., Álvares, A.J., Alape, L.F. (2011a). Una Abordaje Metodológica para Manutención Predictiva Basada em Lógica Fuzzy Aplicada a Plantas de Poder Hidroeléctrica, X Congresso Ibero-Americano em Engenharia Mecânica, CIBEM10, Porto, Portugal.

Moreno, I.P., Álvares, A.J., Alape, L.F. (2011b). Methodology for the Building of a Fuzzy Expert System for Predictive Maintenance of Hydroelectric Power Plants, Proceedings of COBEM 2011, 21st Brazilian Congress of Mechanical Engineering, October 24 – 28, Natal, RN, Brazil.

Alape, L.F. **Moreno, I.P.**, Álvares, A.J., Amaya, E.J. (2011a). A Methodology Based In Case-Based Reasoning to Build a Knowledge-Base Applied to Failure Diagnosis System of Hidrogenerators Machinery, Proceedings of COBEM 2011, 21st Brazilian Congress of Mechanical Engineering, October 24 – 28, Natal, RN, Brazil.

Alape, L.F. **Moreno, I.P.**, Álvares, A.J., Amaya, E.J. (2011b). Diseño y Construcción de una Base de Conocimiento para un Sistema Inteligente de Mantenimiento Aplicado a Equipos de Plantas Hidroeléctricas Basado en el Enfoque RBC, X Congresso Ibero-Americano em Engenharia Mecânica, CIBEM10, Porto, Portugal.

1 INTRODUÇÃO

"Os problemas mais significativos que enfrentamos não podem ser resolvidos no mesmo nível de pensamento em que estávamos quando os criamos"

Albert Einstein.

Nos últimos vinte anos, a filosofia da manutenção de máquinas ou *items*[1] tem mudado talvez mais do que outras disciplinas de gestão. Estas mudanças são devidas ao alto grau de incremento no número e variedade de máquinas (plantas, ativos e estruturas) que devem ser acondicionados em todo momento, desenhos de sistemas muitos mais complexos, novas técnicas de manutenção e pontos de vista cambiantes na organização da manutenção e responsabilidades, entre outras. Devido a esta avalanche de mudanças, administradores ou gerentes de todas as partes do mundo estão à procura de novas abordagens de manutenção; eles almejam evitar transtornos e perdas milionárias nas suas instalações e é por esse motivo procuram *frameworks* estratégicos que sintetizem os novos desenvolvimentos tecnológicos para padrões coerentes em virtude de poder avaliar a sensibilidade e tendências nas suas companhias para atingir uma geração de produtos de alta qualidade e baixo custo (Moubray, 1997).

Segundo afirma Kothamasu *et al.* (2006), a única maneira de minimizar os custos de conserto, manutenção e a probabilidade de falha em máquinas, é desempenhando avaliação atual da sua saúde e predição de falhas futuras baseadas no estado atual, operação e histórico de manutenção. A grande motivação para prognósticos de falhas é minimizar o conserto, custos de manutenção e distúrbios operacionais associados, ao mesmo tempo em que minimiza-se o risco de paradas de máquinas não programadas.

[1] Segundo a ABNT no censo NBR-5462 (1994), *item* é definido como, qualquer parte, componente, dispositivo, máquina, subsistema, ativo, unidade funcional, equipamento ou sistema que possa ser considerado individualmente, eventualmente um item pode incluir pessoas.

O intuito da automação da eficiência das máquinas de produção de qualquer planta ou indústria é atingir o mais elevado nível de confiabilidade com o menor investimento em componentes e mão-de-obra. Nesse campo a tecnologia da informação tem um papel fundamental, que basicamente é identificar automaticamente sintomas de falhas, efetuar diagnósticos e tomadas de decisão com o intuito de direcionar a ação dos gestores através de utilização de redes de comunicação, sistemas de informação, aplicações de intranet e internet, sistemas de automação e sistemas de monitoramento periódico ou contínuo. Também, pode-se incluir nesta área de tecnologia da informação: sistemas de engenharia, gestão de manutenção, planejamento e financeiros (Souza, 2008).

A tecnologia de informação discutida deve permitir a integração total advinda de diversas fontes. Um sistema real de apoio à decisão deve fundir vários tipos de informação, como dos dados da instrumentação de campo (chão de fábrica), dos históricos das variáveis monitoradas (através de banco de dados) e dos relatórios de manutenção preventiva e corretiva (informação padronizada). Tal integração possibilita uma tomada de decisão apropriada com vistas ao planejamento de manutenção da planta ou processo.

É muito importância informatizar todas as informações relevantes contidas e que têm um papel importante na confiabilidade de geração de produtos de alta qualidade e desempenho, isto basicamente se traduz em centralizar a intensa informação advinda da instrumentação dos equipamentos em sistemas supervisórios (*SCADA – Supervisory Control and Data Acquisition*) ou banco de dados em tempo real com taxas variáveis de amostragem. Depois de uma análise prévia de integração, podem-se utilizar Sistemas Especialistas (SE), Redes Neurais (*ANN – Artificial Neural Network*), Lógica *Fuzzy*, Algoritmos Genéticos (AG), Raciocínio Baseado em Casos (*CBR – Case-Based Reasoning*), Sistemas Inteligentes Híbridos entre outras técnicas de inteligência computacional com o intuito de automatizar todos os sistemas de informação fornecidos pela planta ou processo para avaliar as tendências e custos dos produtos gerados em algum intervalo de tempo pré-definido.

O sistema SIMPREBAL (Amaya 2008, Souza 2008 e Tonaco 2008), baseia seus conceitos de concepção na abordagem de manutenção centrada na confiabilidade (*RCM – Reliability-Centered Maintenance*) que segundo Yu e Zhao (2005), deveria estar baseada na confiabilidade de máquinas e no efeito da suas falhas.

Na perspectiva de Yu e Zhao (2005), a abordagem RCM é desenhada para definir funções ou tarefas preventivas que são aplicáveis aos modos de falhas principais de máquinas que possuem um alto custo de efetividade segundo avaliação de métodos de manutenção existentes. Em outras palavras, a abordagem RCM tem a economia de operação como o seu ponto de partida através do desenho de informação relevante traduzido em modos e efeitos de falhas de máquinas. Esta ideia adere-se à proposta de Tonaco (2008), pois na primeira versão do sistema SIMPREBAL foi criada uma FMEA (*Failure Mode and Effect Analysis*) completa de todas as máquinas da usina hidrelétrica de Balbina com o fim de estabelecer os modos de falhas e seus efeitos para gerar uma base de conhecimento consistente usando esta ferramenta baseada em conceitos de RCM.

Inteligência Computacional (CI – *Computational Intelligence*) ou Inteligência Artificial (*AI – Artificial Intelligence*), em particular, Sistemas Baseados em Conhecimento (*KBS – Knowledge-Based Syste*ms) representam uma relativamente nova abordagem de programação e metodologia que tem envolvido e está ainda envolvendo gestão efetiva de manutenção[2] de máquinas em plantas industriais. As aplicações mais relevantes dos KBS, os quais emergiram nos últimos tempos, têm sido monitoramento de condição, diagnóstico, prognóstico de falhas[3] e localização/resolução de problemas em gestão de manutenção de máquinas industriais. Técnicas de CI são usadas amplamente em manutenção de máquinas começando com um mapeamento de ocorrências comuns de mau funcionamento até situações de emergência raramente acontecidas (Nadakatti *et al.* 2008).

A abordagem de CI é promissora para o domínio da manutenção de máquinas, por que captura o conhecimento ou habilidade que tem o especialista (operador de processo, técnico, engenheiro, etc.) para resolver eficientemente um problema. Isto ajuda o operador de processo em detectar rapidamente a falha da máquina para um fácil entendimento. Uma das características mais notáveis da aplicação de CI à manutenção de máquinas é ter a possibilidade de aprendizagem incremental, desde que o processo seja feito com uma metodologia adequada e um método padrão de inserir novo conhecimento ao sistema a controlar (Russell e Yuhui, 2007).

[2] Manutenção é definida pela ABNT (Associação Brasileira de Normas Técnicas) no censo NBR-5462 (1994) como, combinação de todas as ações técnicas e administrativas, incluindo as de supervisão, destinadas a manter ou recolocar um item em um estado no qual possa desempenhar uma função requerida.
[3] Falha é definida pela ABNT no censo NBR-5462 (1994) como, término da capacidade de um item desempenhar a função requerida.

Muitas das atividades na engenharia da manutenção, em particular monitoramento de condição, detecção de falhas, diagnósticos, prognósticos, tomadas de decisão e consertos, são tarefas ou funções baseadas na experiência e amplo conhecimento dos operadores da planta. Tal habilidade, além de usar a intuição e entendimento de como opera certa máquina, torna eficazes os operadores para resolver problemas em algumas condições estabelecidas. Baseado na experiência, um operador desenvolve habilidades como (Bond, 2010):

i. Um entendimento de como o sistema trabalha e a relação com o mundo exterior;
ii. Um entendimento intuitivo de como o sistema se comportará quando um ou vários subsistemas falharem;
iii. Um entendimento profundo dos sintomas de falhas em subsistemas.

A área de manutenção de qualquer indústria[4], segundo Nadakatti *et al.* (2008), é a mais destacável para aplicação dos KBS pois:

i. A habilidade do especialista (técnico, operador, engenheiro, etc.) é necessária para encontrar e reparar falhas em qualquer máquina antes que estas atinjam um estágio no qual possam acontecer grandes perdas econômicas à empresa;
ii. Existem várias ferramentas de *software* no mercado (Smar 2011, Siemens 2011), as quais ajudam o usuário a construir e manipular uma grande base de conhecimento para monitoramento, diagnóstico e prognóstico de falhas em máquinas.

Atualmente o bom funcionamento de um sistema depende da fase de monitoramento de condição de máquinas que visa determinar o tipo, tamanho e localização da falha, bem como o tempo de detecção. Procedimentos de monitoramento de condição de máquinas usam métodos heurísticos e analíticos, por este motivo deveriam ser apresentados de alguma forma unificada como números de confidência (fatores de segurança), funções de pertinência *fuzzy* ou funções de densidade de probabilidade (funções gaussianas), depois de alguma avaliação estatística em algum intervalo de tempo pré-definido (Isermann, 2006).

[4] Nesta dissertação as palavras indústria, planta, fábrica, organização serão usadas indistintamente.

1.1 MOTIVAÇÃO E DELIMITAÇÕES DO PROBLEMA

A principal motivação deste trabalho de pesquisa nasceu da necessidade de dar continuidade ao projeto de pesquisa ANEEL-Eletronorte, intitulado "Modernização das Áreas de Automação de Processos das Usinas Hidrelétricas de Balbina e Samuel" (número de contrato 4500052325 e número de projeto 128), o qual teve como objetivo o desenvolvimento de um sistema computacional para manutenção preditiva capaz de gerar diagnósticos e prognósticos de falhas visando auxiliar os funcionários ou operadores da usina hidrelétrica de Balbina na tomada de decisão com relação às ações operacionais de manutenção, sendo designado por SIMPREBAL (Sistema Inteligente de Manutenção Preditiva de Balbina).

O incentivo para a concepção desta nova proposta metodológica destaca-se em um estudo prévio sobre a viabilidade de dotar, futuramente, o sistema SIMPREBAL com uma nova base de regras de produção usando funções de pertinência *fuzzy*, para lidar com informação incompleta, duvidosa ou nebulosa[5] (Zadeh, 1965), que é inerente à medição de qualquer variável física monitorada, faixa ou nível de operação, no qual se almeja extrair alguma informação relevante para estabelecer, com certo grau de precisão, as condições de funcionamento de alguma máquina. Com isto pretende-se incrementar a confiabilidade de monitoramento de condição de máquinas para gerar melhores diagnósticos de falhas e ações operacionais de manutenção no sistema SIMPREBAL.

Embora esta proposta metodológica, para o monitoramento de condição inteligente de máquinas, seja aplicável a qualquer sistema automatizado dotado de algum método padronizado para aquisição de dados a partir de sensores, a implementação computacional da base de regras *fuzzy*, discutida nesta dissertação, foi direcionada para o monitoramento inteligente de condição de máquinas em usinas hidrelétricas, especificamente, para as unidades geradoras hidráulicas (UGH) da usina hidrelétrica de Balbina. A usina de Balbina está localizada no estado de Amazonas, sendo dotada de cinco UGH com uma capacidade total de geração de 250MW.

[5] A palavra *fuzzy* será usada neste trabalho, indistintamente com a palavra nebulosa(o).

1.2 OBJETIVOS DA PESQUISA

1.2.1 Objetivo geral

O intuito primordial desta dissertação é desenvolver uma metodologia genérica para a concepção de uma base de regras de produção usando funções de pertinência *fuzzy* implementando, principalmente, a fase de codificação[6] da lógica *fuzzy*. Com a novas base de regras *fuzzy*, pretende-se elaborar um estudo prévio de integração no sistema SIMPREBAL, com o intuito de incrementar a confiabilidade dos estados de máquinas e assim, gerar melhores diagnósticos de falhas e eventualmente ações operacionais de manutenção.

1.2.2 Objetivos específicos

Os objetivos específicos deste trabalho são apresentados a seguir:

- ✓ Apresentar a revisão bibliográfica desta pesquisa, ilustrando fundamentos relacionados à lógica *fuzzy*, manutenção preditiva de máquinas e alguns métodos complementários de inteligência computacional;

- ✓ Conceber e implementar, a partir da metodologia proposta, uma base de regras de produção usando funções de pertinência *fuzzy*, de forma não integrada ao sistema SIMPREBAL;

- ✓ Desenvolver um estudo prévio, sobre a viabilidade de integrar a nova base de regras *fuzzy* na camada de monitoramento de condição do sistema SIMPREBAL, usando o histórico de dados da usina de Balbina, comparando os resultados qualitativos e quantitativos obtidos com a base de regras *fuzzy* e com a base de regras original do SIMPREBAL. Com isto, almeja-se incrementar a confiabilidade de monitoramento de condição de máquinas.

[6] A palavra codificação é usada de forma indistinta com as palavras inglesas *fuzziness* ou *fuzzification.*

1.3 ESTRUTURA DA DISSERTAÇÃO

O presente trabalho de pesquisa está estruturado em oito capítulos

No primeiro capítulo, é apresentada uma breve introdução sobre o tema abordado, indicando o contexto no qual está inserido o trabalho e as justificativas para a escolha do tema pesquisado, bem como os objetivos a serem alcançados e o método escolhido para atingi-los. No final deste capítulo são descritas as delimitações do estudo.

O segundo capítulo está composto pelo estado-de-arte sobre os temas pertinentes à dissertação. Através da revisão de literatura, procura-se apresentar um mapeamento de forma clara e concisa da situação atual do processo de monitoramento de condição de máquinas e as principais técnicas de inteligência computacional usadas.

O terceiro capítulo descreve a abordagem metodológica proposta para a concepção de uma base de regras de produção usando funções de pertinência *fuzzy* para incrementar a confiabilidade de monitoramento de condição de máquinas.

No quarto capítulo é implementada na interface ECLIPSE usando a biblioteca *FuzzyJess*, a base de regras de produção usando funções de pertinência *fuzzy*, de forma não integrada ao sistema SIMPREBAL.

No quinto capítulo um estudo de caso é aplicado, usando o histórico de dados do sistema de mancais da usina de Balbina. Neste capítulo, é aplicado um estudo prévio sobre a viabilidade de integração da nova base de regras *fuzzy* na camada de monitoramento de condição do SIMPREBAL.

O sexto capítulo é destinado a estabelecer as conclusões e contribuições para trabalhos futuros com relação à proposta metodológica e sugestões para uma futura integração da base de regras *fuzzy* no sistema SIMPREBAL.

2 REVISÃO DE LITERATURA: MONITORAMENTO INTELIGENTE DE CONDIÇÃO DE MÁQUINA

"Eu não sou mais inteligente do que os outros. A única diferença é que passo mais tempo com os problemas"

Albert Einstein.

Na secção 2.1 é apresentado o estado-de-arte atual sobre a manutenção baseada em condição (MBC) ou manutenção preditiva (MPd) de máquinas, a arquitetura OSA-CBM e as principais abordagens de inteligência computacional aplicadas ao monitoramento de condição, diagnóstico e prognóstico inteligente de máquinas.

Subsequentemente na secção 2.2 uma breve introdução ao sistema SIMPREBAL é mostrada especificando o estado atual da suas camadas, em especial a camada de monitoração de condição. Com relação à introdução sobre a lógica *fuzzy* é apresentada no Apêndice A com o intuito de definir os conceitos de lógica *fuzzy* usados nesta dissertação.

2.1 BREVE INTRODUÇÃO ÀS TECNOLOGIAS EMERGENTES PARA MANUTENÇÃO PREDITIVA DE MÁQUINAS

Esta subseção é destinada a mapear o estado-de-arte atual da manutenção preditiva[1] e as técnicas modernas de monitoramento de condição de máquinas que abordam pontos como, confiabilidade dos estados de máquinas, diagnóstico e prognóstico inteligente, visando identificar estruturas e mecanismos de funcionamento de um plano estratégico completo de manutenção preditiva e as tecnologias emergentes nesta área.

[1] A ABNT (Associação Brasileira de Normas Técnicas) no censo NBR-5462 (1994) define manutenção preditiva ou manutenção baseada em condição como manutenção que permite garantir uma qualidade de serviço desejada, com base na aplicação sistemática de técnicas de análise, utilizando-se de meios de supervisão centralizados ou de amostragem, para reduzir ao mínimo a manutenção preventiva e diminuir a manutenção corretiva.

2.1.1 Diferentes perspectivas na manutenção preditiva de máquinas

MBC ou MPd é uma alternativa às políticas de manutenção tradicional de programação de atividades ou manutenção baseada em uso e manutenção restaurativa. Em manutenção baseada em uso, manutenção preventiva é desempenhada depois de algum período específico de tempo ou histórico de uso da máquina. O intervalo de manutenção é baseado em estatísticas de mortalidade derivado desde o teste acelerado de sistemas mecânicos similares. Isto é usualmente governado por cenários pouco confiáveis que levam à perda da vida útil dos componentes. Ações de manutenção desnecessárias podem induzir outras falhas. Em manutenção restaurativa, concertos são feitos às máquinas depois de falhar ou exceder certos limites de operação estável. Em contraste, MBC é desempenhada na observação baseada na condição do sistema individual e em projeções de eventos de falhas críticas[2] (Garga *et al.* 2001).

Monitoramento de condição (MC) pode ser definido como uma técnica ou um processo de monitoramento das características de operação de uma máquina, de tal forma que mudanças e tendências das características monitoradas possam ser usadas para predizer as necessidades de manutenção da máquina. O anteriormente exposto, abraça o mecanismo de vida das partes individuais de todo o equipamento. O mecanismo de vida das partes individuais envolve monitoramento e analise de dados para predizer com certa exatidão as tendências da máquina (Han e Song, 2003).

MC é uma técnica que forma parte da MBC a qual visa monitorar o funcionamento de máquinas e indicar quando e que tipo de manutenção é necessário para reduzir o consumo de energia, bem como garantir que as paradas nunca serão acidentais senão planejadas. MBC tornar-se-á um ótimo serviço de manutenção sob a ajuda de um bem estruturado sistema de MC para prover informação útil e correta da condição da máquina. Um sistema de MC deveria lidar com a existência de interferência elétrica, informação incompleta, predizer, identificar e localizar os defeitos em detalhe e inclusive fazer uma estimação do tempo de vida restante das máquinas (Wang, 2003).

[2] Falha crítica é definida pela ABNT segundo o censo NBR-5462 (1994) como, falha que provavelmente resultará em condições perigosas e inseguras para pessoas, danos materiais significativos ou outras conseqüências inaceitáveis.

No trabalho de pesquisa de Ciarapica e Giacchetta (2006), elaborou-se um plano estratégico de manutenção baseado em condição (ver Figura 2.1), para uma planta de poder de ciclo combinado alocado em uma refinaria na Itália. Os principais objetivos que apóiam a implementação do plano de manutenção baseada em condição na refinaria italiana são os seguintes:

1. Prever falhas catastróficas e todas as conseqüências relacionadas em termos de segurança, impacto em disponibilidade e custos de manutenção das máquinas;
2. Identificar danos nas máquinas de forma antecipada e assim, priorizar as necessidades para manutenção delas com uma devida antecipação;
3. Prever danos encontrados através do funcionamento normal das máquinas para poder lidar só com os custos de ações de manutenção de falhas conhecidas, em vez de lidar com a manutenção de uma falha severa em condições desconhecidas;
4. Organização e distribuição dos recursos da planta destinada à reposição de peças das máquinas com antecipação programada;
5. Reduzir no estoque da planta a quantidade de peças de reposição, ordenando só as que se precisam e quando for necessário;
6. Melhorar a consciência e a documentação relacionada aos históricos dos equipamentos direcionando assim os esforços para atingir melhores resultados;
7. Desenvolver uma base de dados causa-efeito para usá-la como base para automações subsequentes do processo de gestão de diagnósticos da planta.

Em um plano de manutenção preditiva completo é importante revisar a metodologia proposta, avaliando principalmente os seguintes pontos (Ciarapica e Giaccheta, 2006):

1. Revisar a frequência atribuída inicialmente às atividades de monitoramento das máquinas da planta;
2. Corrigir valores iniciais e defeitos atribuídos aos níveis de alarme e alerta de cada máquina para a detecção de falhas em tempo real;
3. Verificar as correlações existentes entre os sinais emitidos pelos sensores alocados nas máquinas e o fenômeno de degradação de cada máquina em condições operacionais.

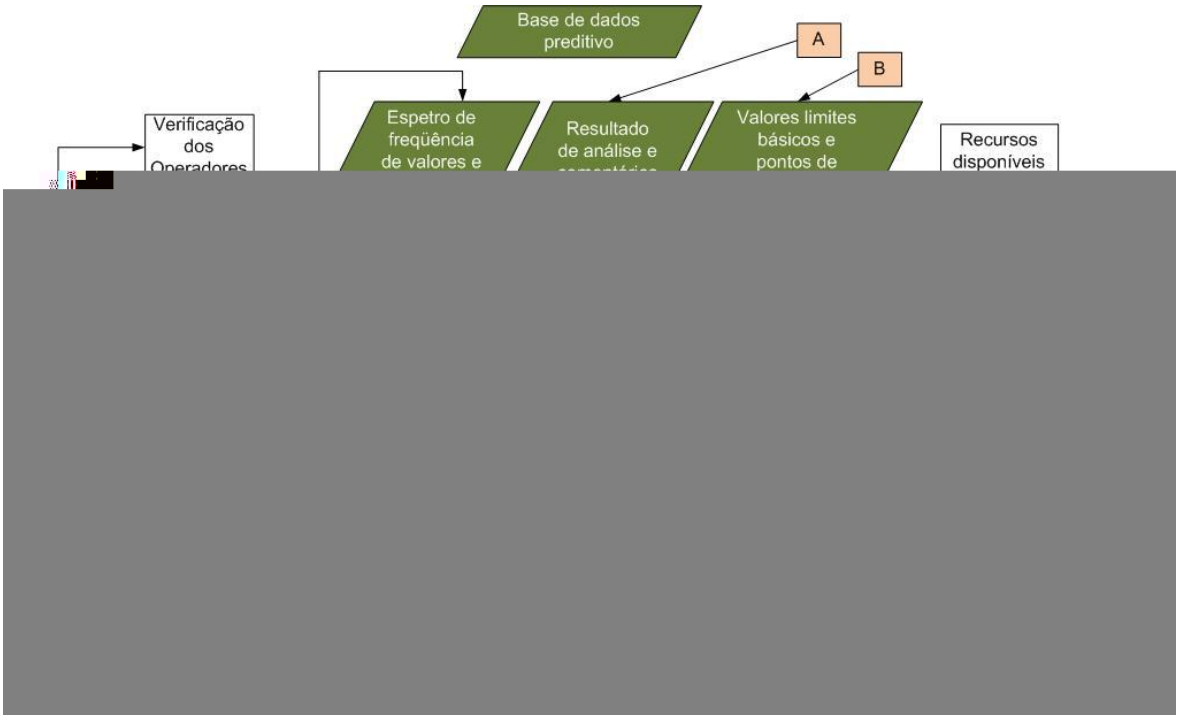

Figura 2.1 – Programação inicial de atividade de MBC (Ciarapica e Giaccheta, 2006)

Sistemas mecânicos devem ser monitorados por um ou mais sensores para controlar os componentes pertencentes aos seus subsistemas. Além disso, algum tipo de processamento com algoritmos computacionais é requerido para fazer inferência de diagnóstico e estimar a vida útil restante da máquina ou dispositivo. Em alguns sistemas, decisões em tempo real poderiam ser feitas para estender ou diminuir o tempo entre consertos de sistemas complexos, ou modificar o tempo real de uso do sistema para minimizar danos nas máquinas (Tasker *et al*, 2011).

Um sistema de controle distribuído (*DCS – Distributed Control System*) é usado para detectar condições de sinais de alerta e enviar algum tipo de comando de parada aos equipamentos que experimentam um defeito. Oferece um constante monitoramento dos estados das máquinas, por que recebe sinais, automaticamente administra a proteção em tempo real e habilita uma gravação dos valores dos sinais em um Sistema de Gestão de Informações (*IMS – Information Management System*). Isto implica que os dados contidos nesse sistema podem ser usados para propósitos de manutenção baseada em condição aplicando alguma técnica de monitoramento de condição, diagnóstico e prognóstico para calcular tendências das condições das máquinas sob controle (Ciarapica e Giacchetta, 2006).

Um ponto importante no planejamento de um projeto para manutenção baseada em condição é definir corretamente a FMEA (*Failure Mode and Effects Analysis*). Isto implica um mapeamento de todas as causas das falhas e seus efeitos de todas as máquinas integradas e que possuem um rol importante no sistema monitorado, em síntese é uma tabela padronizada (Dhillon, 2007).

No trabalho de pesquisa de Chang *et al* (1999), eles propõem um novo modelo de FMEA usando conceitos de lógica *fuzzy* para avaliação linguística dos fatores de criticidade e para obter número de prioridade de risco da tabela FMEA; esse modelo demonstrou e superou os inconvenientes que possui a tabela tradicional FMEA e o novo modelo pode eficientemente descobrir modos e efeitos potenciais de falhas das máquinas envolvidas em qualquer processo industrial.

Um efetivo programa de monitoramento de condição faz uso de uma quantidade de dados muito grande, incluindo dados de sensores como temperatura, pressão, nível, vazão, densidade e vibração. A importância da confiabiliade desses dados para as tarefas de manutenção facilitam a identificação da manutenção ótima, decisões de substituição, estabelecimento de programas de manutenção preventiva, programas de revisão e determinação de peças de recambio, entre outras (Tavner *et al*, 2008).

Atualmente pretende-se nas empresas e organizações modernas direcionar as informações de confiabilidade para serem disponibilizadas de forma digital num sistema de gestão de manutenção computarizada (CMMS – C*omputerized Maintenance Management System*). O objetivo é focar na função de gestão da manutenção, orientar o banco de dados para as análises de confiabilidade e conceber um modelo de banco de dados para integração de informações de confiabilidade. É conveniente dizer que o sistema SIMPREBAL é um CMMS em desenvolvimento (Bagadia, 2006).

Com a aparição dos conceitos de sistemas de gestão da manutenção e o uso de dados da planta, deu-se origem à manutenção eletrônica (*e-maintenance*) como análogo ao comércio eletrônico (*e-business*). A manutenção eletrônica definida como parte da gestão de manutenção onde as máquinas são monitorados e gerenciados pela internet. A *e-maintenance* é um campo emergente que aborda as necessidades fundamentais de ferramentas de inteligência preditiva para monitorar a degradação em vez de detectar

falhas num ambiente totalmente interconectado, optimizando a utilização das máquinas nas instalações (Lee *et al*, 2006).

Prognóstico inteligente é definido como uma abordagem sistemática que pode monitorar continuamente a degradação da saúde das máquinas e extrapolar o comportamento temporal de indicadores de saúde para predizer riscos de comportamento inaceitável através do tempo, bem localizar exatamente qual componente da máquina é mais provável a falhar. Tal entendimento contínuo da saúde atual e futura das máquinas e seus componentes e a estrutura do fluxo de informações habilitam o fluxo da *e-maintenance* baseado em prognósticos inteligentes, sincronizando ações de manutenção com a operação total do sistema, mas com os recursos de manutenção necessários e substituição de peças (Holsapple, 2003).

A gestão de prognóstico de saúde (*PHM - Prognostic Health Management*) é considerada como uma técnica de engenharia e tecnologia que pode ser usada para dar suporte à manutenção baseada em condição. A determinação do modelo PHM para uma peça de uma máquina normalmente precisa de dados do monitoramento de condição, da falha e outros dados de eventos coletados das máquinas em operação. Esses dados são geralmente armazenados em banco de dados de programas de MBC entre outros (Valavanis, 2009).

Sistemas de MBC são concebidos para prover sugestões avançadas de manutenção ou ações para desligar algum processo. Operadores são requeridos para avaliar a situação e tomar decisões apropriadas. Em outras palavras, sistema de MBC e operadores precisam colaborar juntos para resolver problemas mais complexos. A pesquisa de engenharia dos fatores humanos tem demonstrado que sob tais circunstâncias, operadores precisam entender o processo inteiro para efetivamente resolver problemas complexos (Kothamasu e Huang, 2007).

A PAS 55 da BSI (2011), define a necessidade de um plano estratégico global de manutenção que descreva a visão, políticas, objetivos e estratégias organizacionais. A implementação da norma de um sistema de gestão PAS 55, recomenda uma estrutura prática do PDCA (*plan-do-check-act*) para o modelo de gestão de ativos, como mostrado na Figura 2.2; onde são definidas quatro fases:

- ✓ ***Plan*** - estabelecer a estratégia de gerenciamento de máquinas, objetivo e planos necessários para entregar resultados nos termos do gerenciamento de máquinas da organização, política e plano estratégico organizacional;

- ✓ ***Do*** - estabelecer os ativadores de execução das máquinas e outros requisitos necessários para implementar os planos de gestão;

- ✓ ***Check*** - monitorar e medir resultados para compará-los com os valores esperados pela política de gestão de ativo da empresa (*benchmarking*): estratégia, objetivos, e outros requisitos;

- ✓ ***Act*** - tomar ações para garantir que o gerenciamento de máquinas e os objetivos sejam alcançados para continuamente melhorar o desempenho do sistema de gerenciamento de máquinas.

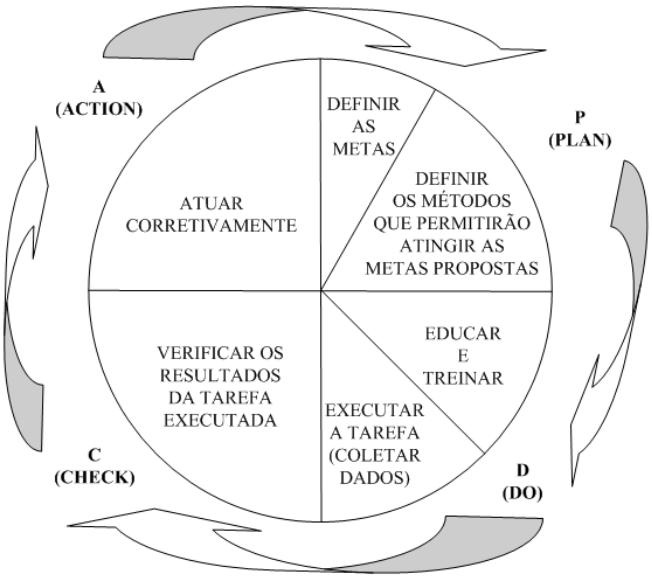

Figura 2.2 – Ciclo PDCA para a gestão de ativos (BSI, 2011)

Nos últimos anos as empresas de geração de energia elétrica, em particular, desejam integrar vários tipos de algoritmos de diagnóstico e prognóstico para criar um sistema completo de processamento híbrido. A arquitetura concebida deve ser o suficientemente geral para dar suporte a uma ampla gama de algoritmos e abordagens de diagnósticos e prognóstico oferecendo interoperabilidade e intercambiabilidade (Jing *et al*, 2007).

Como uma implementação da ISO (*International Organization for Standardization*), especificação funcional 13374, a aliança para sistemas abertos de gestão de maquinaria (MIMOSA, 2008) e a OSA-CBM, especificam uma arquitetura padrão e um *framework* para a implementação de sistemas de manutenção baseados em condição.

A arquitetura OSA-CBM especifica sete camadas funcionais (Aquisição de Dados, Processamento de Sinal, Monitoramento de Condição, Avaliação de Saúde, Avaliação de Prognósticos e Tomada de Decisão). É importante mencionar que a última camada (Apresentação) apresenta todas as informações relevantes das camadas anteriores como é mostrado na Figura 2.3.

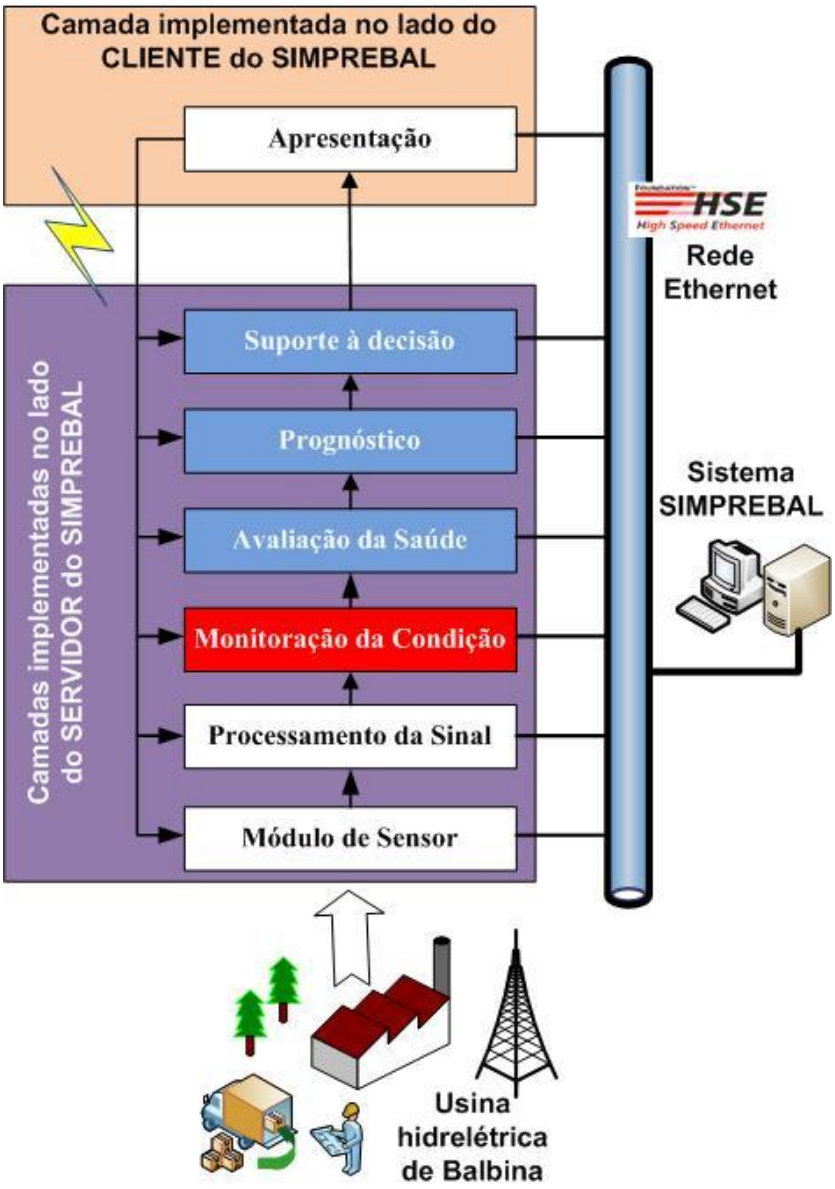

Figura 2.3 – Arquitetura OSA-CBM do sistema SIMPREBAL

O padrão OSA-CBM estabelece uma via para a integração de muitos componentes de diversos fabricantes, especificando as entradas e as saídas entre tais componentes. Embora a arquitetura OSA-CBM preconize os fundamentos para arquiteturas de sistemas abertos com processamento híbrido de saúde e prognósticos de máquinas, mesmo assim, as camadas estão mais relacionadas a funções de manutenção do que funções de processamento inteligente.

O padrão OSA-CBM não especifica a forma como deve ser implementado em cada módulo ou camada, além disso, não define que tipo de tecnologias e algoritmos desenvolver em cada camada. O padrão simplesmente define o tipo de dados recebidos na entrada e produzidos na saída de cada módulo, por último define a maneira de transmissão entre camadas oferecendo uma independência entre os sete módulos.

Análogo ao sistema SIMPREBAL existe no mercado um Sistema Inteligente de Manutenção Preditiva (SIMAP – *Intelligent System for Predictive Maintenance*). O SIMAP é semelhante ao SIMPREBAL sendo um software orientado para diagnóstico de falhas em tempo real. No trabalho de Garcia *et al* (2006), fez- se um estudo de caso com o SIMAP aplicado ao monitoramento de condição da caixa de câmbios de um gerador originando corretos diagnósticos e prognósticos de falhas.

2.1.2 Inteligência computacional aplicada à manutenção preditiva de máquinas

Considerando um pouco de história, redes neurais foram aplicadas desde inícios do ano 1990 a sistemas de potência elétrico. A primeira conferência em aplicações de redes neurais artificiais a sistemas de potência foi realizada no ano 1991. Na metade dos anos noventa, a lógica *fuzzy* foi aplicada a sistemas de potência, tais como, controle, operações, diagnóstico, prognóstico e planejamento.

Soft Computing, que em termos simples é uma abordagem emergente que usa ferramentas de inteligência computacional como lógica *fuzzy*, redes neurais e algoritmos genéticos, para lidar com algum problema da vida real que não tenha uma estrutura completamente definida, foi aplicada a sistemas de potência nos anos noventa.

Recentemente, a computação evolucionaria (CE) tem sido usada principalmente para resolver problemas de planejamento, controle e operações em sistemas de potência, pois, sistemas de potência são tipicamente sistemas de grande porte e complexos.

Por outro lado, a tecnologia *data mining*, que essencialmente envolve a procura de padrões estáveis com informações relevantes e facilmente interpretável em uma grande base de dados usando alguma técnica de *Soft Computing* é amplamente usada. Este tipo de tecnologia talvez seja usado para resolver certos problemas inerentes em sistemas de potência, como poderia ser o caso de uma planta de geração de energia hidrelétrica.

Existe *Software* de *data mining*, atualmente disponível para qualquer organização. Em estações de potência é usado para aperfeiçoar o desempenho, reduzir o consumo de energia e identificar antecipadamente ações para reduzir custos de operações, (Dote e Ovaska, 2001).

No trabalho de pesquisa de Niu *et al* (2010), é proposto uma nova arquitetura para manutenção de máquinas através da integração da MBC e técnicas de fusão de dados. Esta integração pode ser graficamente visualizada na Figura 2.4.

Na Figura 2.4, primeiro os sinais advindos da planta são coletados, e a seguir ocorre o processamento de sinal. Depois, as características apropriadas são calculadas e é extraída essa dada informação com relação ao estado de funcionamento das máquinas. A seguir, as características que podem permitir reconhecer falhas diferentes claramente são selecionadas para análise de diagnóstico, enquanto que outras características indicando tendência de degradação da saúde da máquina são escolhidas para monitoramento e funções de prognóstico para predizer a vida útil restante (*RUL – Remaining Useful Life*).

Um sistema alternativo de suporte de decisão preditiva inteligente para manutenção baseada em condição é proposta por Yam *et al* (2001), no qual propõe um modelo ou Sistema de Suporte de Decisão Preditiva Inteligente (IPDSS – *Intelligent Predictive Decision Support System*) baseado na abordagem de ANN (*Artificial Neural Network*). Esta metodologia foi validada em uma planta de geração de energia hidrelétrica para o teste do funcionamento de máquinas críticas. Os resultados desta pesquisa mostraram que

o sistema gera diagnósticos de falhas confiáveis e possui um forte poder de predição para a tendência com relação à degradação das máquinas.

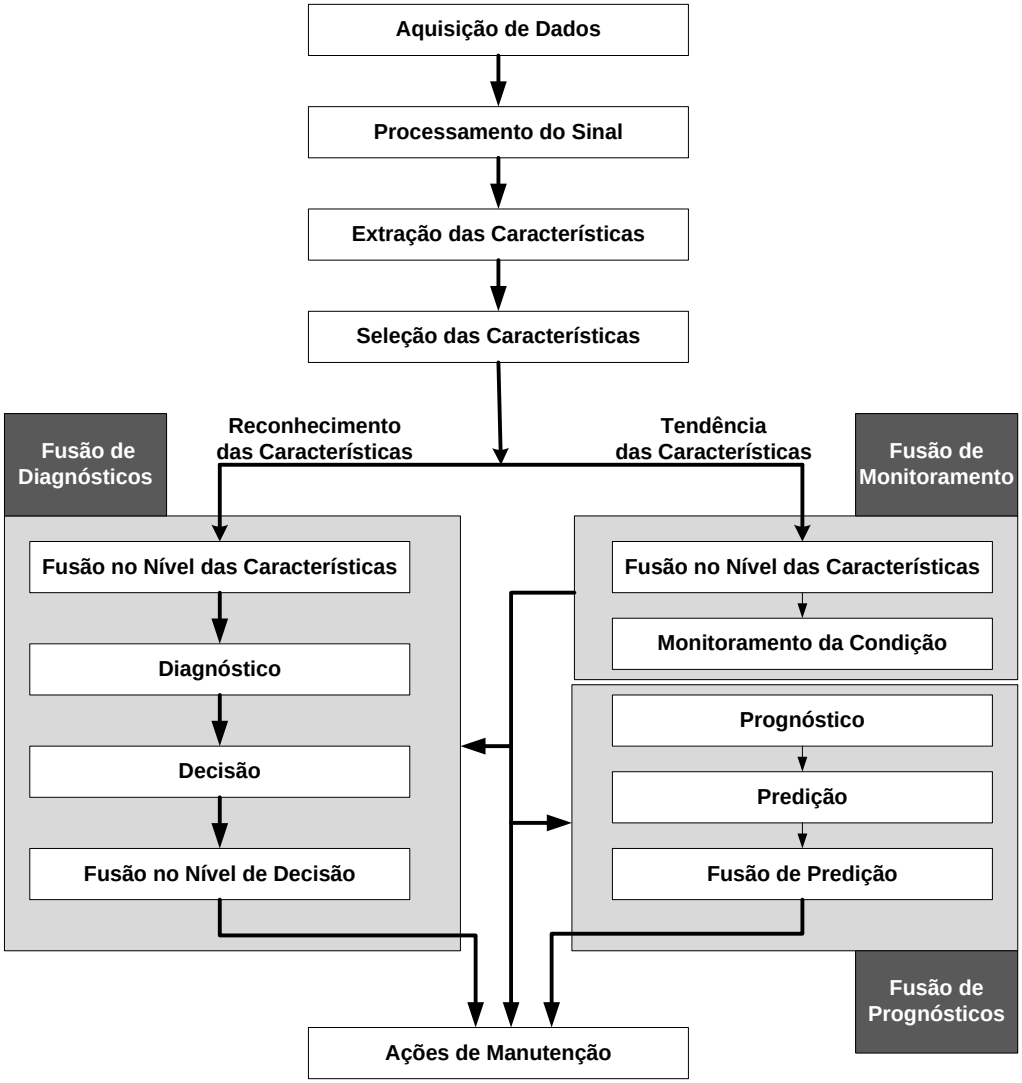

Figura 2.4 – Fluxo de informações de MBC integrando fusão de dados (Niu *et al*, 2010)

Prognóstico de falhas é o processo de predizer a confiabilidade futura de uma máquina através da avaliação do grau de deslocamento da degradação em relação as suas condições de operação normal. Abordagens para processamento de prognósticos de falhas podem ser classificadas em quatro categorias (Niu *et al* 2010):

- Modelos físicos;
- Sistemas baseado em casos e **baseado em regras de produção**;
- Métodos de aprendizagem estatísticos dirigidos por modelos;
- Métodos de aprendizagem estatísticos dirigidos por dados.

Um Sistema Gestão de Manutenção Integrada (IMMS – *Integrated Maintenance Management System*) pode estabelecer em uma organização trabalhos mais confiáveis com relação ao pré-planejamento e pré-programação de atividades de manutenção de máquinas, reduzir o impacto de custos de inventário para reposição ou troca de peças, reduzir as atividades de manutenção não planejadas e reduzir ao mínimo o risco de falhas catastróficas (Telang, 2010).

Segundo Momoh e Tomsovic (1995), as áreas mais frutíferas para aplicação da lógica *fuzzy* ou a Teoria dos Conjuntos *Fuzzy* (TCF) em sistemas de potência, podem ser citadas como:

- ✓ As áreas relacionadas com o planejamento do sistema, abordando problemas de programação de atividades e confiabilidade;
- ✓ As áreas relacionadas com a operações do sistema, abordando problemas de análise de contingência, avaliação da instabilidade, tomadas de decisão, controle e monitoramento de condição, suporte, coordenação, entre os mais importantes;
- ✓ As áreas relacionadas ao controle do sistema, abordando problemas de estabilizador baseado em lógica *fuzzy*, conversor e impulsor de energia, entre outros controles;
- ✓ As áreas de diagnóstico do sistema, abordando problemas de diagnóstico de transformadores, diagnósticos de rede e diagnósticos de máquinas.

Uma seqüência de passos é proposta por Ross (2010), quando é pretendido usar a lógica *fuzzy* para resolver algum problema do mundo real:

A. Descrição total do problema original: O problema a ser abordado deveria estar primeiro definido matematicamente e linguisticamente;

B. Definição de todos os limiares para todas as variáveis envolvidas no problema: Para uma variável, existe um valor específico com o mais alto grau de satisfação avaliado desde o conhecimento empírico mais certo grau de desvio aceitável até um valor que é completamente inaceitável. Os dois valores correspondentes ao maior e menor grau de satisfação, são limiares relacionados;

C. Quantização *fuzzy*: Baseado nos valores limiares obtidos desde o passo B, formas adequadas de funções de pertinência são construídas. Muitas formas de funções de

pertinência são disponíveis, tais como, lineares, lineares por partes, trapezoidais, parabólicas e assim por diante. As funções de pertinência deveriam refletir a mudança ou tendência do grau de satisfação com a mudança em variáveis avaliadas pelos expertos (operários de chão de fábrica, engenheiros de manutenção, etc);

D. Seleção das operações *fuzzy* adequadas: Em termos de processo de tomada de decisão prática efetuada por especialistas, uma operação *fuzzy* adequada é selecionada de maneira que os resultados obtidos sejam semelhante àqueles obtidos por especialistas. A interpretação dos resultados usando sistemas *fuzzy* está baseada no processamento do especialista no domínio do conhecimento. Por conseguinte, neste nível uma arquitetura ou plano baseado num sistema experto *fuzzy* híbrido é desejável. Isto ajuda a remover qualquer ambiguidade que possa acontecer no processo de inferência do problema. As operações mais comumente usadas são: Mamdani e Zadeh.

Fazendo um breve parêntese, paralelamente a este trabalho de dissertação usando conceitos de lógica *fuzzy*, está sendo desenvolvida uma base de conhecimento aplicando a técnica de Raciocínio Baseado em Casos (CBM – *Case Based Reasoning*), como uma abordagem alternativa para a evolução metodológica do sistema SIMPREBAL (Alape *et al*, 2011a e Alape *et al*, 2011b).

Seguindo com o raciocínio, o conhecimento do especialista tem um papel importante na localização e resolução de problemas ou análise de falhas. Em sistemas de potência, é necessário diagnosticar o mau funcionamento das máquinas através de monitoramento inteligente de condição e seus distúrbios. O mau funcionamento das máquinas é causado por muitos fatores, um deles é a incerteza contida na informação disponível para desempenhar diagnóstico e prognóstico de falhas. Além disso, as condições que induzem falhas talvez mudem com o tempo. Conjeturas subjetivas baseadas na experiência são de vital importância. Por conseguinte, abordagens de sistemas expertos são exitosamente úteis neste tipo de raciocínio. Um sistema especialista baseado em lógica *fuzzy* oferece uma representação e construção do conhecimento mais sofisticado (Grabys *et al*, 2005).

Lógica *fuzzy* e redes neurais artificiais (RNA) possuem uma qualidade interpolativa. A combinação dos conjuntos *fuzzy* com redes neurais multicamada (sistema híbrido) podem melhorar o desempenho das redes neurais artificiais em si. Esta abordagem híbrida pode

ser usada para aqueles problemas onde as amostras de treinamento numéricos não podem ser construídas, mas só conhecimento humano é disponível. Por outro lado, a introdução de conceitos *fuzzy* podem incrementar a quantidade de informação disponível nas saídas dessas redes neurais. Por tanto, aplicando conceitos de conjuntos *fuzzy*, de forma adequada, talvez ofereçam um melhor uso das saídas das redes neurais, atingindo uma eficiente representação da informação (Abraham *et al*, 2006).

Existem várias abordagens para o monitoramento inteligente de condição de máquina industriais Rutkowski e Cpalka (2003), Mitra e Hayashi (2000), Siler e Buckldey (2005) e Javadpour e Knapp (2003), que desembocam na implementação de uma rede neural *fuzzy* para monitoramento inteligente de condição de máquinas em ambientes de manufatura automatizados para incrementar a confiabilidade dos diagnósticos, prognóstico e tomadas de decisão.

Outro tipo de abordagem é feita usando uma rede *fuzzy* ARTMAP, que baseia-se em conceitos de: combinação de aprendizagem supervisionada e não supervisionada baseada na arquitetura da Teoria de Ressonância Adaptativa (ART – *Adaptive Resonance Theory*) de Grossberg (Carpenter e Grossberg, 2003). Essa rede foi aplicada para analisar os dados de vibração, em series de tempo e em tempo real, de máquinas rotativas. Uma das vantagens que se consegue, usando esta rede sobre uma arquitetura de rede neural convencional é que pode ser incrementalmente treinada (Jain *et al* ,2008).

Existem outras duas abordagens bastante conhecidas: os sistemas *neuro-fuzzy* e as redes neurais recorrentes (RNN – *Recurrent Neural Networks*), as quais foram avaliadas no trabalho de pesquisa de Wang *et al* (2004), com o intuito de fazer um estudo comparativo para escolha com relação ao desempenho para predição de condição sobre a saúde de máquinas industriais. Segundo os testes feitos nesse estudo, os sistemas *neuro-fuzzy* apresentaram um melhor desempenho do que a RNN em ambos os casos de critérios de avaliação: precisão de predição e eficiência de treinamento.

Diagnóstico de falhas em máquinas, baseados no monitoramento de condição de vibrações usando conceitos de lógica *fuzzy* incorpora um número de técnicas de detecção de falhas e diagnóstico sofisticado. No trabalho de pesquisa de Mechefske (1998), é proposto o uso de técnicas de lógica *fuzzy* para classificar o espectro de frequencias representando várias

falhas de elementos em mancais de rolamento. Basicamente aqui, o espectro de frequencias representando um número de diferentes condições de falhas tem sido processado usando uma variedade de formas de conjuntos *fuzzy*.

A idéia proposta por Mechefske (1998) poderia ser aplicada ao sistema de mancais da usina hidrelétrica de Balbina para controlar mais um parâmetro, a vibração dos mancais através de futuros sensores instalados nas cinco unidades geradoras hidráulicas. Desta forma pode-se incrementar a confiabilidade do monitoramento das máquinas da usina com o intuito de gerar melhores diagnósticos, prognósticos, ações operacionais de manutenção e programação de atividades no sistema SIMPREBAL.

Entre os algoritmos ou modelos de diagnósticos e prognósticos mais usados em máquinas industriais, segundo Pandian e Ali (2009), deve-se considerar os seguintes aspectos: com relação às técnicas de aquisição de dados, estas são usadas para colecionar e guardar dados de uma máquina em estado de operação. A técnica de processamento de sinais usualmente analisa a forma de onda e dados multidimensionais. A análise no domínio do tempo está baseada no tempo da forma de onda. A análise no domínio do tempo é feita pelos modelos AR (*Auto Regressive*), ARMA (*Auto Regressive Moving Average*) e ARIMA (*Auto Regressive Integrated Moving Averages*).

Seguindo a análise estabelecida por Pandian e Ali (2009), a análise no domínio da frequencia está baseada na transformada do sinal para o domínio na freqüência. A análise mais amplamente usada é a transformada rápida de Fourier (*FFT – Fast Fourier Transformation*). A análise no domínio da frequencia e tempo pesquisa ambos domínios para sinais de forma de onda não estacionaria. Por outro lado, Bases de dados contendo um grande volume de dados usam técnicas de análise de dados multivariável tais como, Análise de Componente Principal (*PCA – Principal Component Analysis*) e Análise de Componente Independiente (*ICA – Independent Component Analysis*).

Três tipos de abordagem categorizam as técnicas de diagnóstico de falhas: abordagens estatísticas, abordagens de inteligência artificial e abordagens baseadas em modelos. O Controle de Processos Estatísticos (*SPC – Statistical Process Control*), *Cluster Analysis*, HMM e *Support Vector Machine* (SVM) são técnicas mais comumente conhecidas como

abordagens estatísticas. Os Modelos de Markov Ocultos (*HMM – Hidden Markov Models*) são usados para eventos e monitoração da condição conjuntamente (Pandian e Ali, 2009).

Em Inteligência Artificial (IA), tal como as Redes Neurais Artificiais (*ANN – Artificial Neural Network*), Sistemas Baseados em Lógica *Fuzzy* (*FLS – Fuzzy Logic Systems*), Sistemas *Neuro-Fuzzy* (*FNN – Fuzzy Neural Networks*) e Algoritmos Genéticos (*GA – Genetic Algorithms*) são algumas das técnicas mais usadas para abordagem em monitoramento de condição, diagnóstico e prognóstico de falhas em máquinas.

As técnicas baseadas em modelos utilizam modelos matemáticos explícitos e físicos da máquina monitorada. Baseados neste modelo, métodos de geração residual tais como, Filtro Kalman (estimador Bayesiano da média atual), estimação de parâmetros e relações parciais são usados para obter sinais, chamados residuais, os quais indicam presença de falha nas máquinas. Este tipo de abordagem só é efetivo se o modelo é preciso e se torna muito complexo para sistemas de grande porte (Pandian e Ali, 2009).

No trabalho de pesquisa de Fast e Palmé (2009), fez-se um estudo para a criação de um sistema online para monitoramento e diagnóstico das máquinas de uma planta de geração na Suécia. O modelo implementado está composto por um modelo de rede neural representando cada componente principal da planta conectado a uma interface de usuário gráfica *(GUI – Graphical User Interface).* O modelo de rede neural foi integrado a um servidor administrador de informações da planta, e a GUI ficou disponível para uma arquitetura cliente-servidor. A rede neural foi treinada com dados operacionais dos componentes da planta originando bons resultados de diagnóstico de falhas.

No trabalho de pesquisa de Kumar *et al* (2009), é proposta uma abordagem para avaliação da saúde de motores elétricos de grandes dimensões (sistema com geração >500KW), usando os dados de monitoramento de condição para estabelecer inferência *fuzzy* hierárquica em períodos de tempo.

No trabalho de pesquisa de Li e Sum (2008), propõe-se um modelo para seleção de plantas de geração baseado numa rede neural *fuzzy* melhorada. Basicamente neste trabalho é adotado um algoritmo de redução baseado na teoria dos conjuntos aproximados (*RST – Rough Set Theory*), proposto por (Pawlak, 1982). O objetivo desse novo modelo é reduzir

os fatores de influência de seleção de plantas de geração e eliminar as atribuições não correlacionadas. É implementada uma rede neural que usa o algoritmo de aprendizagem *backpropagation*. O modelo foi dotado de um método de inferência *fuzzy* para calcular os graus de pertinência das amostragens típicas as quais são as entradas para a rede neural.

No trabalho de Prazzo *et al* (2010), propõe-se um modelo usando a técnica de lógica *fuzzy* para classificação de severidade da temperatura em máquinas de calefação. Foram usados três critérios para classificação da severidade de calefação dos componentes de cada máquina: calefação corrigida, diferença máxima de temperatura entre componentes e temperatura máxima admissível. Cada critério foi definido e delimitado graficamente com quatro possíveis estados de operação de cada componente: normal, quente, muito quente e severamente quente, estas variáveis linguisticas foram implementadas computacionalmente com funções trapezoidais *fuzzy*.

Um sistema completo de diagnósticos de falhas em mancais de rolamentos foi proposto por Fujimoto e Padovese (2001), onde se apresenta um sistema de diagnóstico automático para detecção e classificação de falhas em mancais de rolamento baseado em uma análise de lógica *fuzzy*. O sistema proposto lida com o espectro de sinal de vibração advindo dos mancais, coletados em diferentes condições de operação com diferentes velocidades do eixo e diferentes cargas. Neste caso foram definidas sete variáveis lingüísticas de entrada cada uma delas com os seus adequados conjuntos *fuzzy* (trapezoidais e triangulares). Foram criados 72 regras *fuzzy* com processamento de inferência estilo Mamdani e o método de descodificação usado foi o centro de gravidade.

Em sistemas de geração de energia elétrica, especificamente na área de controle e monitoramento, vários trabalhos usando conceitos de lógica *fuzzy* e outras técnicas de inteligência computacional foram desenvolvidos:

Wen *et al* (1988) aplicaram Computação Evolucionaria (*EC – Evolutionary Computing*) para desenhar um controlador ótimo de excitação baseado em lógica *fuzzy* para um gerador assíncrono. Além disso, os resultados atingidos foram usados para o desenho automatizado de controladores *fuzzy*.

Cho e Park (1997) desenvolveram um sistema experto usando relações *fuzzy* para lidar com as incertezas impostas em diagnósticos de falhas de sistemas de potência. Os resultados experimentais para sistemas de potência reais demonstraram a utilidade da técnica proposta para diagnóstico de falhas que tinham uma considerável quantidade de informação com incerteza.

Sistemas *fuzzy* e *neuro-fuzzy* têm sido usados em várias aplicações de MBC, incluindo os seguintes (Kothamasu e Huang, 2007):

- ✓ Diagnóstico de motores e bombas hidráulicas;
- ✓ Diagnóstico de falhas em controle de fluxo de válvulas;
- ✓ Predição de vibrações em bombas alimentadas por calefação de alta pressão;
- ✓ Determinação do fluxo de falhas relativo a transformadores de alta voltagem;
- ✓ Diagnóstico de falhas em motores de grande porte;
- ✓ Diagnóstico de falhas em mancais rotativos;
- ✓ Detecção de falhas em robôs;
- ✓ Reconhecimento de falhas incipientes em transformadores de potência.

Sistemas *fuzzy* podem efetivamente lidar com ambas informações: qualitativa e quantitativa. Eles são apropriados para aplicações de manutenção baseada em condição, já que, conhecimento qualitativo experto pode ser incorporado no processo de construção do modelo. O modelo pode ser bem sintonizado usando dados experimentais, históricos quantitativos, etc. Intuitivamente, resultaria em um sistema mais robusto, preciso e facilmente interpretável. A interpretabilidade do modelo é muito importante em aplicações de MBC (Cigolini *et al*, 2009).

Existem dois tipos principais de sistemas *fuzzy*, o tipo Sugeno e o tipo Mamdani. Diferem basicamente no formato dos conseqüentes da suas regras. Os consequentes de um sistema *fuzzy* Sugeno estão na forma de uma função enquanto que em um sistema *fuzzy* Mamdani os consequentes estão na forma de termos linguisticos. Sistemas *fuzzy* Mamdani são mais compatíveis com o processo de raciocínio de operadores e mais intuitivos em aplicações de MBC (Mellin e Castillo, 2005).

Abordagens *neuro-fuzzy* implementam um sistema *fuzzy* em uma estrutura de rede neural, erro *backpropagation* para procurar ajustar parâmetros em suas funções de pertinência. Outra abordagem aplicativa usa Algoritmos Genéticos (AG) para ajustar os mesmo parâmetros (Kothamasu e Huang, 2007).

Em modelamento *neuro-fuzzy*, existe uma tendência de sacrificar interpretabilidade por precisão do modelo. Não é difícil encontrar aplicações que usem sete ou mais termos lingüísticos para descrever completamente algum significado físico ou parâmetro. Parece que em tais aplicações esquecem que a principal razão da escolha de um sistema *fuzzy* é devido a sua interpretabilidade. Esta situação é percebida quando regras são extraídas automaticamente desde dados usando algum método popular como *subtractive clustering*, onde o número de termos linguisticos é igual ao número de regras ou *clusters* (Rutkowski, 2010).

No trabalho de pesquisa de Kothamasu e Huang (2007), é apresentada uma abordagem de modelamento *neuro-fuzzy* usando regras do tipo IF-THEN e é demonstrada uma grande utilidade em aplicações de MBC. Os benefícios que podem ser conseguidos usando esta abordagem podem ser citados como:

- ✓ Representação do conhecimento baseado em regras junto à extração de regras: oferece um caminho para integrar modelamento baseado em dados com modelamento baseado em modelos físicos;
- ✓ Modelo baseado em regras: brinda compatibilidade com o processamento heurístico humano, permitindo ao especialista direitamente contribuir com a construção do modelo. Obviamente está é uma característica que precisa ser depurada antes de oferecer uma contribuição à base de conhecimento do sistema em estudo. Em lógica *fuzzy* seria equivalente a caracterizar os termos linguisticos contidos em todo um processo;
- ✓ Modelo transparente para interpretação do usuário: decisões feitas no sistema podem ser claramente explicadas de modo que o sistema rapidamente possa ganhar veracidade do usuário. Isto é especialmente importante em aplicações criticas de segurança onde vidas humanas estão em jogo;

- ✓ Processamento aproximado: oferece uma abordagem computacional paramétrica, por conseguinte abre a possibilidade de sintonizar automaticamente os parâmetros de modo que o sistema possa adaptar-se a mudanças do seu entorno.

No trabalho de Rigoni (2009), é proposta uma metodologia para implantação da Manutenção Centrada na Confiabilidade (MCC) usando sistemas baseados em conhecimento e lógica *fuzzy*. Basicamente esta metodologia é orientada por sistemas baseados em conhecimento *fuzzy* que incorporam critérios para diagnóstico e tomada de decisão, levando em conta os aspectos técnicos, gerenciais, experiência de programas consolidados de MCC e o conhecimento institucional. Com esta metodologia é possível minimizar os riscos de insucesso das metodologias tradicionais de implantação da MCC, considerando e tratando as incertezas de imprecisão léxica do processo com uma visão holística das interações e necessidades da MCC.

2.2 O SISTEMA DE MANUTENÇÃO PREDITIVA DE BALBINA (SIMPREBAL)

Esta subsecção oferece uma visão geral do sistema SIMPREBAL para a compreensão do estado atual da suas camadas, em especial a camada de "**monitoração de condição**", onde é proposta uma nova base de regras de produção usando funções de pertinência *fuzzy* para avaliar sua futura implementação nessa camada.

2.2.1 Concepção metodológica do sistema SIMPREBAL

A metodologia que deu origem ao SIMPREBAL baseia-se no modelo de referência OSA-CBM, apresentado em sete camadas (a sétima camada apresenta informações das camadas anteriores). A integração de técnicas de inteligência computacional às seis primeiras camadas são apresentadas e desenvolvidas em: Amaya (2008), Tonaco (2008) e Souza (2008).

Nos trabalhos de pesquisa anteriormente expostos, basicamente, delimitou-se e distribuiu cada sistema e subsistema que compõe a usina hidrelétrica de Balbina, para um estudo mais específico e avançado, a saber: Sistema de Mancais, Sistema da Turbina e Sistema do Gerador. O objetivo em comum é propor uma metodologia baseada no padrão OSA-CBM para manutenção preditiva de máquinas em tempo real, com o intuito de gerar ações

operacionais de manutenção de máquinas que auxiliem à tomada de decisão dos operários da área de manutenção da usina de Balbina. Os três trabalhos de mestrado fizeram um mapeamento das funcionalidades de cada sistema e subsistema e geraram três propostas metodológicas.

No trabalho de pesquisa de Amaya (2008), é proposta uma metodologia não implementada no sistema SIMPREBAL, para o desenvolvimento de um sistema inteligente de manutenção preditiva baseada em sistemas especialistas e um sistema inteligente híbrido baseado em lógica *fuzzy* e redes neurais artificiais.

Fez-se um estudo de caso relativo ao Sistema do Gerador (gerador elétrico principal), no qual é proposta, de forma não integrada, a modelagem de um sistema híbrido *fuzzy* ARTMAP para o sistema SIMPREBAL, detalhando a arquitetura conceitual, as fases de treinamento da FAM (*fuzzy* ARTMAP), o pseudocódigo para treinamento da FAM e as fases de desempenho da FAM.

Por outro lado, no trabalho de Souza (2008) é proposta uma metodologia para a construção de um sistema de manutenção baseada em condição visando dar suporte às atividades de manutenção de ativos de usinas hidrelétricas e melhorar a confiabilidade desses ativos.

A metodologia proposta desempenhou um estudo de caso no Sistema de Mancais de Balbina, apresenta conceitos de manutenção centrada na confiabilidade e técnicas de sistemas especialistas que desembocam em um sistema inteligente de apoio à decisão.

Como no trabalho anteriormente exposto, baseia-se no modelo de referência OSA-CBM e propõe adotar a abordagem Markoviana (cadeias de Markov) para predição de falhas na camada de prognósticos do padrão OSA-CBM, pois, este método é amplamente difundido, facilmente encontrado na literatura, e principalmente, por possuir a capacidade de modelar vários fenômenos naturais e artificiais.

Quando se aplicam as cadeias de Markov, é definida uma série de estados para modelar as fases de transição das máquinas antes de atingir um estado de falha, são definidos também parâmetros de estimação de mantenabilidade e confiabilidade das máquinas. Foram calculados e delimitados esses parâmetros através das equações clássicas de Chapman-

Kolmogorov. O principal inconveniente desta abordagem é o pouco tempo de funcionamento do sistema SIMPREBAL em Balbina (Souza, 2008).

Segundo o anteriormente exposto, o número de falhas identificadas pelo SIMPREBAL é reduzido para calcular estatisticamente a vida útil restante de cada máquina por que as equações de Chapman-Kolmogorov pressupõem um tempo de observação suficientemente longo para que as falhas possam ser modeladas como eventos aleatórios (distribuições exponenciais).

No trabalho de Tonaco (2008) é proposta uma metodologia que descreve um método detalhado, que vai desde a coleção de dados até a implementação de regras de produção. Basicamente desta coleta resultou a catalogação das possíveis falhas e as intervenções de manutenção adotadas para corrigir as falhas, em consequencia, foi definida a estrutura da base de conhecimento do sistema SIMPREBAL e finalmente a implementação das regras de produção.

Este trabalho foi validado com um estudo de caso no Sistema da Turbina de Balbina, o qual permitiu monitorar os dados dos sensores instalados nas máquinas da usina, sugerir tomadas de decisão direcionando intervenções de manutenção, aumentar a segurança do processo de geração de energia a partir da monitoração dos dados e tratar e interpretar as informações adquiridas a partir dos sensores instalados nas máquinas monitoradas pelo sistema SIMPREBAL.

Uma das contribuições mais importantes deste trabalho foi o desenvolvimento da FMEA para todos os sistemas analisados (sistema de mancal, sistema da turbina e sistema do gerador).

2.2.2 O servidor SIMPREBAL

O servidor SIMPREBAL é basicamente uma aplicação Java *standalone*, cuja tarefa principal é obter os dados dos equipamentos da usina hidrelétrica de Balbina. Esta aquisição de dados pode ser feita de duas formas: através do banco de dados via o *driver* JDBC ou através das máquinas em tempo real via servidor OPC e processá-los através de uma máquina de inferência de um sistema especialista.

O seu processamento inteligente subjaz na detecção de situações de manutenção preditiva e eventualmente em indicações de tomadas de decisão com relação à atuação daquela falha potencial nos sistemas e subsistemas da planta.

Quando o sistema inteligente detecta alguma determinada falha nos sistemas ou subsistemas, em qualquer uma das cinco unidades geradoras hidráulicas, ele envia automaticamente um email informando a equipe responsável pela manutenção, subsequentemente armazena o diagnóstico no banco de dados via o *driver* JDBC e finalmente envia ao módulo de comunicação cliente-servidor informações e sugestões de manutenção para serem visualizadas em tempo real na camada de apresentação dos clientes conectados (Souza, 2008).

Um estudo mais profundo de como foi concebido e implementando tanto o servidor quanto o cliente do sistema SIMPREBAL pode ser encontrado nos trabalhos de Álvares e Amaya (2010), Álvares *et al* (2009), Álvares *et al* (2007a), Álvares *et al* (2007b), Álvares e Amaya (2007), Álvares *et al* (2007c) e Tonaco *et al* (2007).

2.2.2.1 Camada de aquisição de dados

Nesta camada é realizada a aquisição de dados pelo sistema SIMPREBAL referente às grandezas físicas dos equipamentos monitorados nas cinco unidades geradoras hidráulicas de Balbina, sendo possível esta aquisição de três formas: via servidor OPC através do *driver* JNI, via banco de dados através do *driver* JDBC e através de variáveis simuladas (aplicável só para execução de testes quando o programa de manutenção preditiva estava em desenvolvimento) como é mostrada na Figura 2.5.

Na Figura 2.5 embora o servidor OPC e o sistema SIMPREBAL possam estar instalados em uma mesma máquina, eles também podem ser instalados em máquinas diferentes.

A vantagem de estarem em máquinas diferentes é a distribuição do processamento de dados entre dois computadores, entretanto a desvantagem é o aumento do fluxo de informações na rede.

A instrumentação da usina de Balbina é formada por transmissores inteligentes da SMAR, os quais são controlados por componentes de hardware multifuncionais DFI302 *Foundation Fieldbus*.

Estes componentes, designados como DFI, são dotados de um servidor OPC denominado DFI *OLE Server*. As DFI adquirem, através de seus módulos de entrada e saída, dados diretamente dos transmissores inteligentes pertencentes à rede de instrumentação e os disponibiliza na rede de supervisão por meio do servidor OPC.

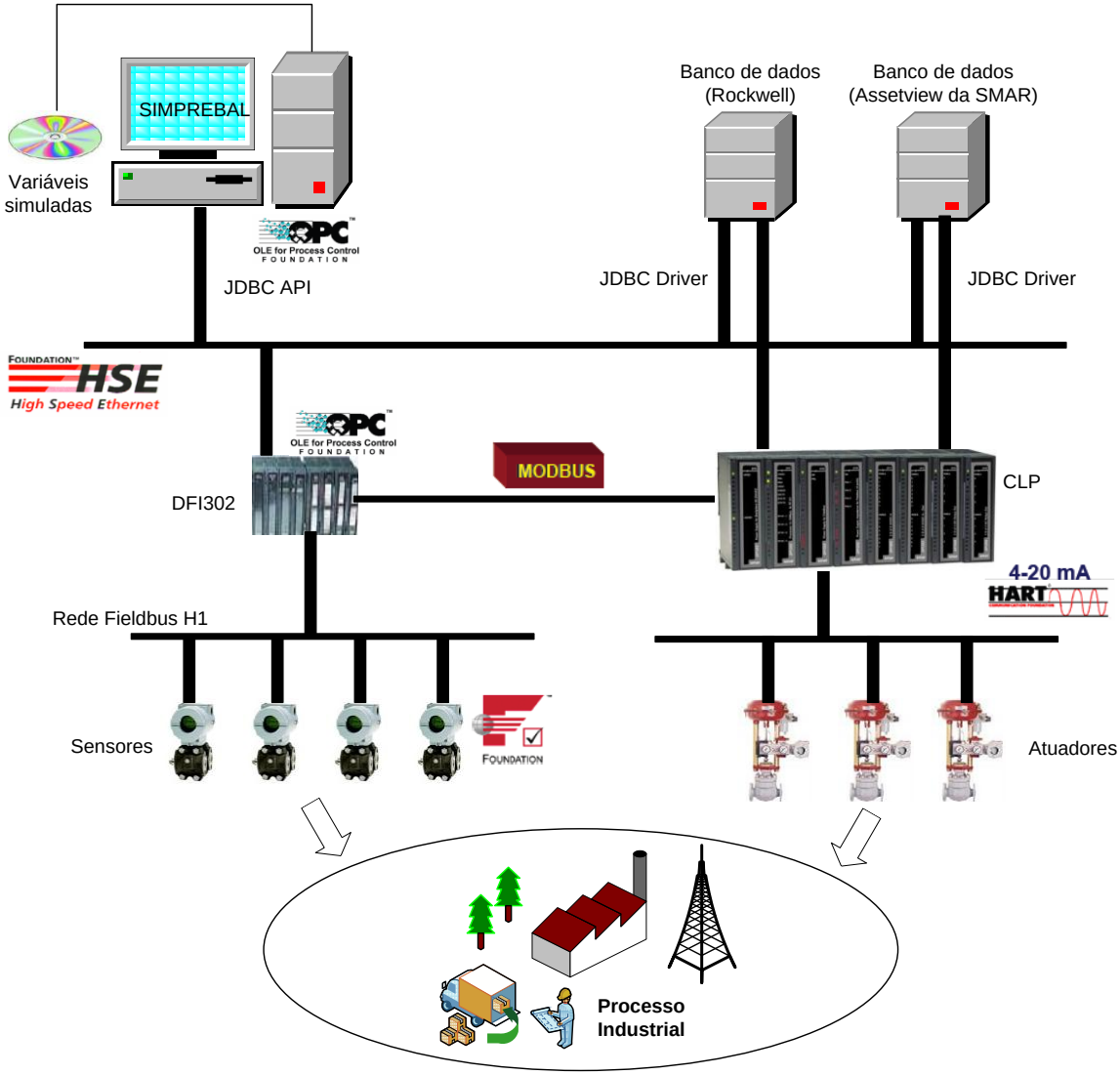

Figura 2.5 – Mecanismo de aquisição de dados do sistema SIMPREBAL (Souza 2008, modificado)

O servidor SIMPREBAL, por sua vez, possui um cliente OPC, desenvolvido por meio da biblioteca comercial *JOPCClient*, que comunica-se com o DFI *OLE Server* e recebe de forma *online* os referidos dados. São trocadas, portanto, informações sobre o valor e a qualidade de variáveis de processo tais como temperatura, nível, vazão e pressão (Souza, 2008).

2.2.2.2 Camada de processamento de sinal

Neste módulo o sistema SIMPREBAL realiza a seguinte sequencia (Souza, 2008):

- ✓ Testa a conectividade com o servidor OPC caso mesmo esteja em um computador remoto;
- ✓ Testa a conectividade com as *Bridges Universais Fieldbus* (DFI);
- ✓ Testa a dinamicidade das *tags* recebidas, isto é, verifica se os valores das *tags* recebidas se alteram corretamente;
- ✓ Processa informações sobre a qualidade do sinal OPC recebido;
- ✓ Processa informações sobre a qualidade do sinal *Fieldbus*.

Os testes de comunicação são basicamente desempenhados pelo comando PING. Para uma melhor compressão sobre o processamento de informações sobre a qualidade do sinal nesta camada, é necessário explicar o método de representação das variáveis monitoradas pelo sistema SIMPREBAL como mostrado nos seguintes parágrafos.

Cada variável de processo como temperatura do óleo na cuba do mancal combinado é descrita a partir de duas *tags*, ou dois itens OPC, sendo um do tipo VALUE e outro do tipo STATUS. O primeiro carrega informações sobre o valor propriamente dito da grandeza monitorada e sobre a qualidade do dado obtido via OPC, e o último traz informações sobre a qualidade do sinal transmitido pela instrumentação *Fieldbus*.

A partir destas informações é construído no sistema SIMPREBAL uma instância da classe *Tag*. Deste modo, cada objeto *Tag*, que posteriormente será enviado como fato inicial para o processamento das regras de produção do sistema inteligente, possui uma estrutura formada a partir dos dois itens OPC supracitados (Souza, 2008).

A Figura 2.6 apresenta o processo de representação da variável de temperatura do óleo na cuba do mancal combinado. Conforme pode-se enxergar na Figura 2.6, no *item* VALUE utiliza-se o campo *Quality* e obtém-se a qualidade do sinal OPC, e no item STATUS, utiliza-se o campo *Value* e obtém-se a qualidade do sinal *Fieldbus*.

Após o processamento de sinal, obtém-se um objeto da classe *Tag* com os seguintes atributos principais (Souza, 2008):

- ***ID*** – Nome fornecido ao parâmetro monitorado. Trata-se de uma sequencia de letras e números segundo um padrão específico visando identificar brevemente e de forma única um determinado padrão;
- ***Label*** – Pseudônimo do parâmetro monitorado. Trata-se de um mnemônico associado à grandeza monitorada, criado para garantir inteligibilidade e flexibilidade ao *software*;
- ***Value*** – Valor da grandeza medida;
- ***Quality*** – Qualidade do sinal obtido pelo servidor OPC;
- ***SubQuality*** – Informações adicionais sobre a qualidade do sinal obtido pelo servidor OPC;
- ***Status*** – Estado geral da comunicação dos instrumentos *Fieldbus*;
- ***SubStatus*** – Informações adicionais sobre o estado da comunicação *Fieldbus*.

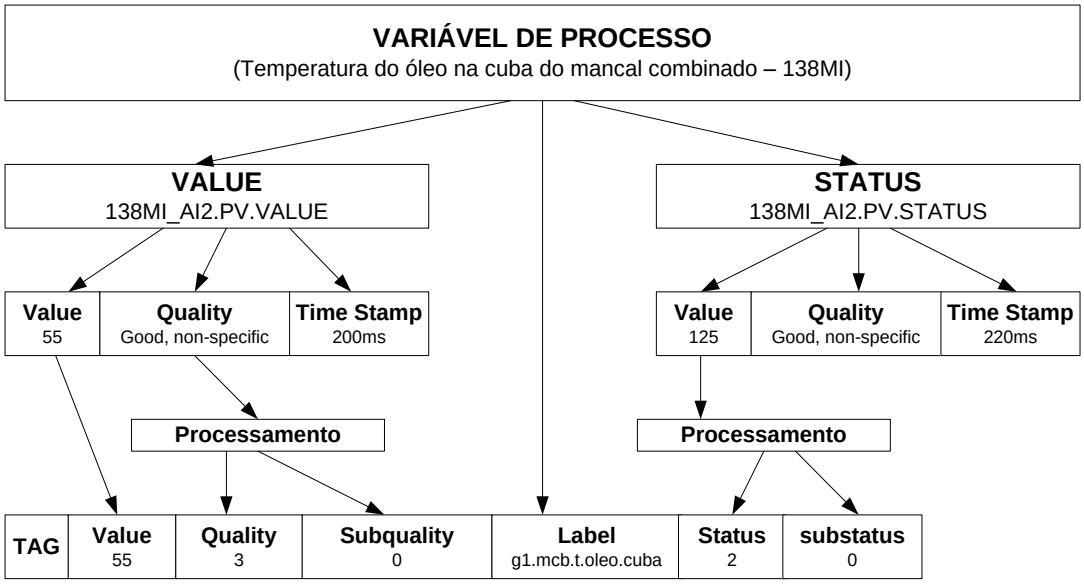

Figura 2.6 – Construção do objeto *Tag* para a temperatura do óleo na cuba do mancal combinado (Souza, 2008)

Como uma observação final nesta camada, uma lista de objetos *Tags* é enviada à memória do sistema especialista, sob a forma de fatos. As regras de produção desenvolvidas na camada de processamento de sinal solicitam informações a respeito de um determinado atributo da classe *Tag* (atributos *Quality* e *Subquality* para o processamento de sinal OPC e atributos *Status* e *SubStatus* para o processamento de sinal *Fieldbus*).

Como última instância um exemplo de como é construída as regras de produção para o processamento nesta camada é apresentado na Figura 2.7.

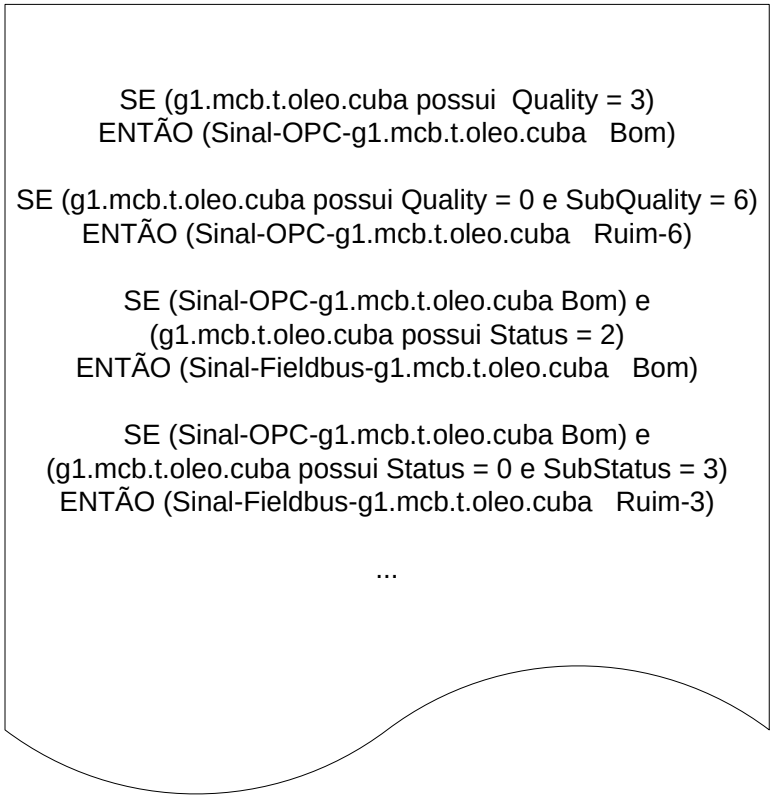

Figura 2.7 – Regras de produção para o processamento de sinal do óleo no mancal combinado (Souza, 2008)

2.2.2.3 Camada de monitoração de condição

Basicamente esta camada recebe os dados advindos da camada de processamento de sinal, os quais são objetos da classe *Tag*, para comparação com valores previamente estabelecidos pela equipe do sistema SIMPREBAL e os operários da usina, de modo a atribuir às variáveis monitoradas uma condição específica.

Na primeira versão do sistema SIMPREBAL, esta camada foi implementada somente com regras de produção do tipo SE-ENTÃO em formato Jess, onde se verifica o valor do parâmetro monitorado, isto é, o atributo *Value* de um item OPC. Foram definidas quatro faixas de operação para o valor desse atributo, os quais são citados a seguir (Souza, 2008):

- **Normal** – Trata-se de uma faixa de valores que indicam um funcionamento dentro do previsto para um determinado equipamento, isto é, uma condição normal de operação;
- **Alerta** – Trata-se de uma faixa de valores dentro da qual a variável monitorada indica um defeito incipiente no equipamento. Esta faixa de valores foi estabelecida visando alcançar respostas para toda e qualquer alteração da condição normal de operação de um determinado equipamento;
- **Alarme** – Consiste de uma faixa de valores previamente estabelecida pelos gestores da automação da usina e indica uma situação de risco ao equipamento monitorado. Exige-se que, ao atingir a faixa de alarme, sejam tomadas atitudes preventivas para evitar danos ao equipamento ou prejuízos decorrentes de uma parada inesperada;
- **Trip** – Consiste de uma faixa de valores que caracterizam uma condição de operação inaceitável para os equipamentos da usina. Ao se atingir a faixa de *trip*, os equipamentos são desligados automaticamente.

Uma representativa gráfica das quatro faixas de operação, para a variável temperatura do enrolamento do estator, com código 49GA, pertencente ao sistema gerador elétrico principal é mostrado na Figura 2.8:

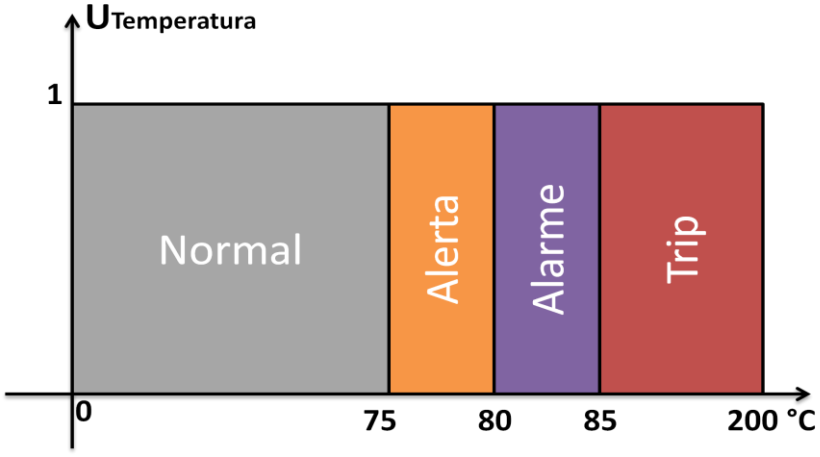

Figura 2.8 – Esboço das faixas de operação para a variável temperatura do enrolamento do estator com código 49GA

Algumas observações com relação às faixas de operação da Figura 2.8 são:

- As faixas de operação não possuem nenhum nível de transição. Por exemplo, para valores muito próximos a 75°C a condição da máquina ainda é considerada como sendo NORMAL, o que não é uma boa forma de monitoramento de condição.
- Qualquer valor contido nessas faixas possui o máximo nível de pertinência a essa faixa. Por exemplo, para o valor de 79.785°C, a condição da máquina ainda segue sendo ALERTA. Não existe nenhum índice que mensure uma pertinência parcial.

A Figura 2.9 mostra uma estrutura geral de regra, com relação à sintaxe, para a temperatura do óleo no mancal combinado (a mesma estrutura é aplicável para as demais variáveis).

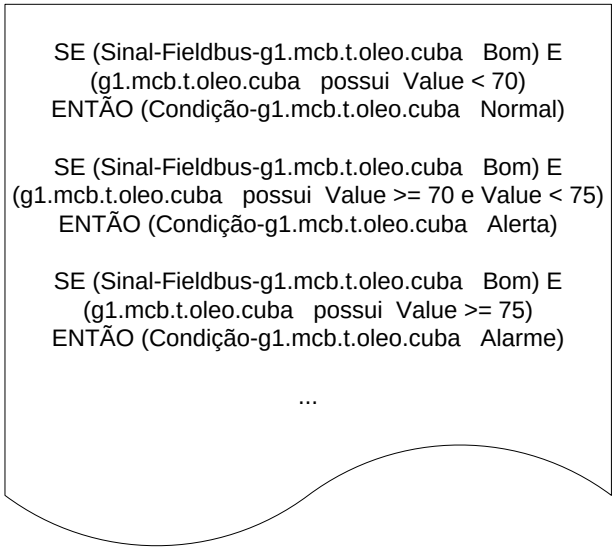

Figura 2.9 – Regras de produção para o monitoramento de condição da temperatura de óleo do mancal combinado (Souza, 2008)

2.2.2.4 Camada de avaliação de saúde (diagnóstico)

A camada de diagnóstico utiliza a ferramenta FMEA (Análise dos Modos e Efeitos de Falhas) para encontrar relações entre as grandezas monitoradas e as falhas do sistema. Para a criação de tal FMEA para a usina hidrelétrica de Balbina, também foram usadas instruções técnicas de operação, instruções técnicas de manutenção e planejamento de manutenção autônoma que contenham descritivos funcionais dos equipamentos, bem como, descrições dos procedimentos operacionais e procedimentos de manutenção.

Outros documentos como ordens de serviço foram utilizados também para a elaboração da FMEA. Um ponto importante a mencionar com relação à FMEA é que ela tem um desenvolvimento dinâmico e deve ser atualizada sempre que necessário de forma a registrar todas as modificações de procedimentos efetuados e todas as ocorrências inéditas de falhas (Souza, 2008). Como um exemplo, uma parte do formulário ou tabela FMEA aplicado ao sistema mancal guia do gerador (SMGG) é mostrado na Tabela 2.1.

Tabela 2.1 – Tabela FMEA para o SMGG (Souza, 2008)

FORMULÁRIO DE FMEA – ANÁLISE DOS MODOS E EFEITOS DE FALHA					
Objeto de estudo: SISTEMA MANCAL GUIA DO GERADOR (SMGG)					
Função: Transferir os esforços radiais do eixo do gerador ao concreto.					
Componente	**Função**	**Modo potencial de falha**	**Efeito potencial de falha**	**Causa potencial de falha**	**Controles atuais**
1. Cuba	Armazenar óleo de modo a imergir as partes ativas do mancal guia	1.1. Armazenamento insuficiente de óleo na cuba	-Atuação de TRIP na unidade geradora.	-Vazamento de óleo (perda de carga) pelas vedações e conexões do mancal, da tubulação dos filtros, dos transmissores, dos trocadores de calor e das motobombas.	-Transdutor de temperatura (71GMO)
			-Atrito excessivo entre as sapatas e o eixo.		
			-Aquecimento anormal.	-Vazamento de óleo pelas folgas no mancal.	
		1.2. Perdas das características físico-químicas do óleo	-Corrosão do metal patente.	-Contaminação do óleo com água.	-Análise do óleo
			-Má formação do filme de óleo.	-Má qualidade do óleo.	
			-Atrito excessivo entre as sapatas e o eixo.		
			-Aquecimento anormal.		
2. Sapatas	Permitir a formação e manutenção de um filme de óleo (efeito hidrodinâmico) entre o metal patente e o eixo da turbina durante o funcionamento da unidade geradora.	2.1. Má formação do filme de óleo.	-Atuação TRIP na unidade geradora.	-Folga ou desalinhamento das sapatas.	-Transdutores de temperatura do mancal (38GMM1,2,3)
			-Atrito excessivo entre as sapatas e o eixo.	-Corrosão do metal patente.	
			-Maior vibração do eixo.	-Contaminação do óleo com sujeira e fragmento de metal patente.	
			-Aquecimento anormal.		

É importante mencionar que as regras de produção desenvolvidas a partir das tabelas FMEA das cinco unidades geradoras hidráulicas são regras de diagnóstico de falhas relacionados aos fatos produzidos na camada de monitoração de condição. Além das tabelas FMEA, outra saída da camada de avaliação da saúde é a determinação do risco potencial associado a cada falha.

2.2.2.5 Camada de prognóstico

Nesta camada existem duas contribuições essenciais, a dizer, uma proposta por Amaya (2008), o qual propõe uma arquitetura ou metodologia FAM (*fuzzy* ARTMAP) o qual foi previamente discutido no item 2.2.1. A segunda proposta é apresentada por Souza (2008), a qual propõe o uso das cadeias de Markov para definir os estados de transição das máquinas e assim prognosticar um estado de falha iminente.

Com relação às cadeias de Markov, no qual os prognósticos de falha são estimativas de tempo médio entre falhas calculados por meio dessa modelagem Markoviana, é necessário obter no banco de dados do SIMPREBAL os registros de ocorrências de ALERTA, ALARME e TRIP para cada *tag* de um determinado equipamento, além dos tempo de início e de término de cada falha. Uma representativa desta modelagem pode ser vista na Figura 2.10.

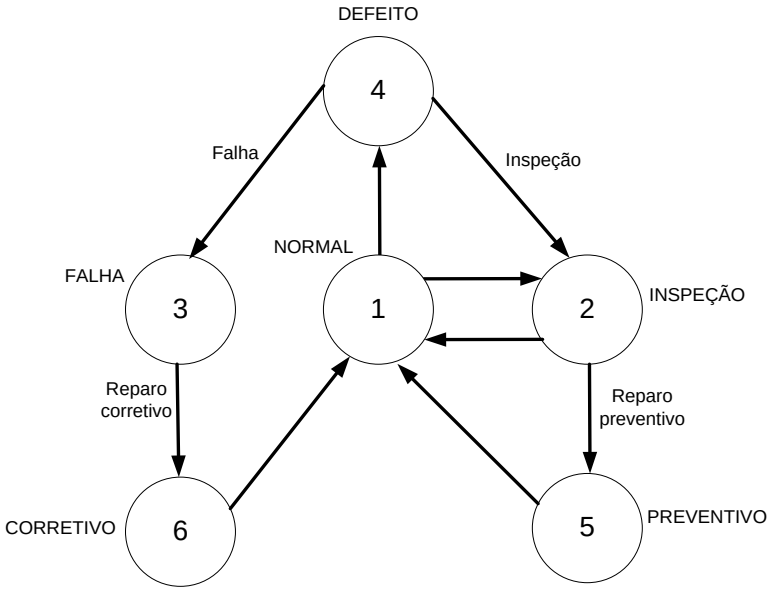

Figura 2.10 – Modelo Markoviano de manutenção adaptado ao sistema SIMPREBAL (Souza, 2008)

Onde os estados de cada equipamento são definidos a continuação (Souza, 2008):

- **Normal** – Apto para exercer sua função, sem restrições;
- **Inspeção** – Avaliação das condições de funcionamento;
- **Falha** – Indisponível, após uma falha funcional;
- **Defeito** – Disponível, mas com falha potencial;
- **Preventivo** – Em manutenção preventiva, após uma falha potencial;
- **Corretivo** – Em manutenção corretiva, após uma falha funcional.

As transições, ou eventos que mudam o estado do equipamento são definidos por Souza (2008), como:

- **Inspeção** – Inicio de inspeção de um equipamento;
- **Reparo corretivo** – inicio de manutenção corretiva para corrigir a falha funcional;
- **Reparo Preventivo** – Início de manutenção preventiva para corrigir a falha potencial.

O cálculo relativo aos parâmetros de estimação da mantenabilidade tais como, freqüência de disponibilidade forçada, tempo médio de manutenção preventiva, entre outras, e para a estimação de parâmetros de confiabilidade tais como, tempo de permanência no estado de defeito, taxa de falha funcional, taxa de falha potencial, entre outras, pode ser encontrado no trabalho de Souza, (2008).

2.2.2.6 Camada de tomada de decisão

A camada de tomada de decisão é, em sua forma mais simples, uma sugestão de ordem de serviço desenvolvida a partir do estudo prévio da FMEA.

Todas as tomadas de decisão, associadas a cada eventual diagnóstico de falha, estão previamente inseridas no banco de dados do sistema SIMPREBAL em tabelas denominadas *decisões_ughX*, onde *X* é o número da unidade geradora hidráulica.

2.2.3 O cliente SIMPREBAL

No cliente SIMPREBAL está desenvolvida a camada de apresentação do modelo de referencia OSA-CBM. Este cliente foi pensada par ser uma aplicação *web*, deste modo disponibilizando o fluxo de informações para toda a empresa da Manaus Energia.

Foi desenvolvida em páginas HTML, nas quais estão inseridos um *applet* Java e estruturas em PHP e *javascript*. A Figura 2.11 mostra a página principal de acesso ao sistema SIMPREBAL.

Figura 2.11 – Janela para acesso ao sistema inteligente de manutenção preditiva.

2.2.3.1 Camada de apresentação

Na Figura 2.12 é apresentada a janela principal no qual são monitorados os ativos que têm sob controle o sistema SIMPREBAL. Esta figura apresenta as áreas importantes que compõem o sinótico do sistema desenvolvido em Java.

Em contraste a Figura 2.13 apresenta algumas das demais funcionalidades que possui o sinótico de monitoramento do SIMPREBAL, o qual tem sob controle os três sistemas que conforma cada unidade geradora hidráulica. Com relação ao manual de operações, descrições mais minuciosas entre outras funcionalidades do sistema inteligente, podem ser encontrados em trabalhos de Amaya (2008), Souza (2008) e Tonaco (2008).

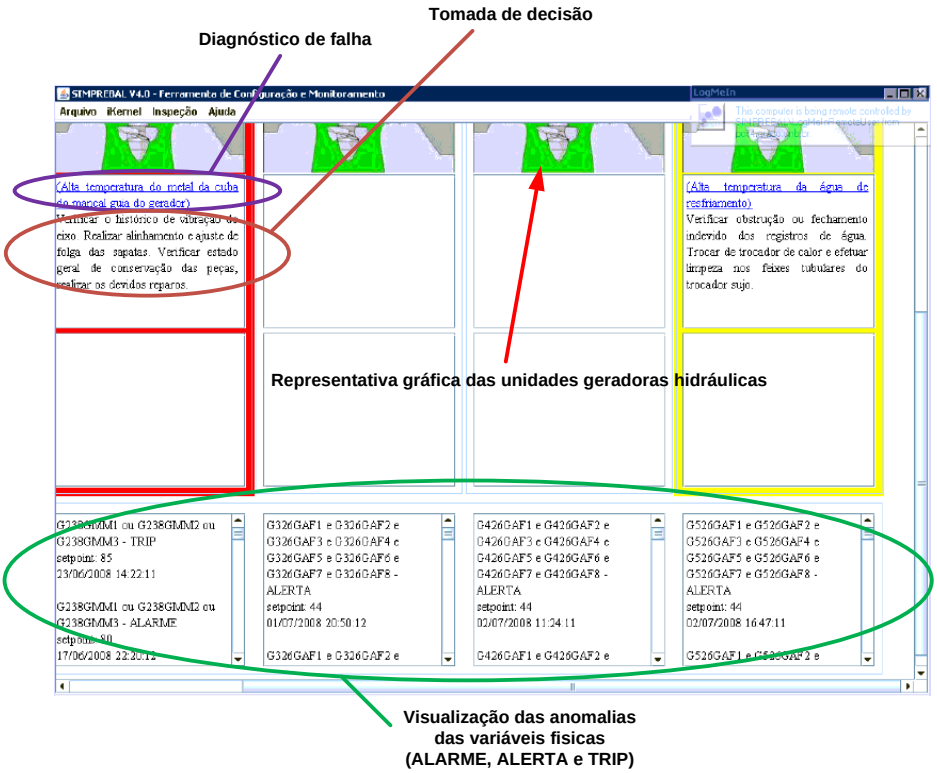

Figura 2.12 – Interface usuário-máquina do sistema SIMPREBAL

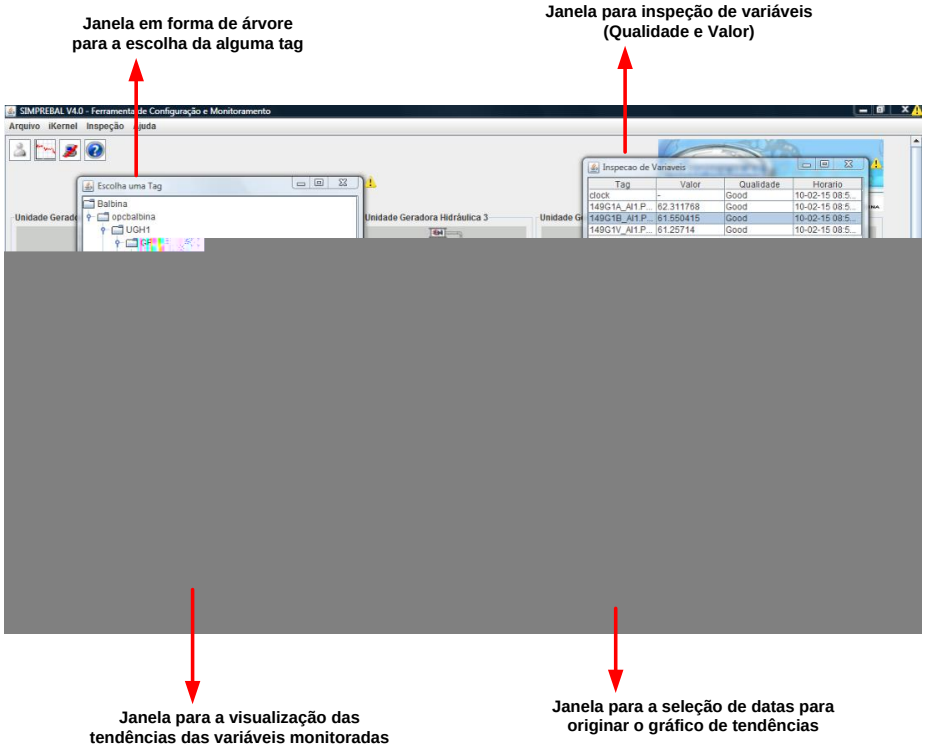

Figura 2.13 – Outras funcionalidade do sinótico de monitoramento do sistema SIMPREBAL

2.3 CONSIDERAÇÕES E SÍNTESE DO CAPÍTULO

Neste capítulo foram abordados os conceitos de MBC, a arquitetura OSA-CBM e as tecnologias emergentes usadas no desenvolvimento de um sistema de manutenção preditiva de máquinas completo.

Segundo a revisão de literatura analisada, aplicações de técnicas de inteligência computacional são as mais promissórias para implementações nas camadas do padrão OSA-CBM, entre as principais, redes neurais artificiais, algoritmos genéticos, lógica *fuzzy* e sistemas neuro-*fuzzy* para monitoramento de condição, diagnóstico e prognóstico inteligente de máquinas.

Através da visão geral do sistema SIMPREBAL visou-se oferecer as funcionalidades atuais de suas camadas e das lacunas que possui atualmente para avaliar a viabilidade da implementação do uso da lógica *fuzzy* no sistema de manutenção preditiva.

A camada de monitoramento de condição do sistema SIMPREBAL, por ser uma das primeiras, é uma camada crítica, pois é onde se estabelece a condição atual das máquinas para que nas camadas seguintes sejam desempenhados diagnósticos e eventuais tomadas de decisão. Uma solução viável aos problemas que apresenta a Figura 2.8 é abordada pela lógica *fuzzy*, pois lida com a incerteza contida nas informações.

Pelo anteriormente exposto sobre a camada de monitoração de condição, é proposta uma base de regras de produção usando funções de pertinência *fuzzy* de forma não integrada ao sistema SIMPREBAL, para um estudo prévio de sua futura integração na camada de monitoração de condição, com o intuito de lidar com a incerteza contida nas faixas de operação das máquinas que a base de regras original do sistema SIMPREBAL não possui.

3 ABORDAGEM METODOLÓGICA

"The tools researchers use to probe certain AI problems, says this Berkeley professor, are sometimes too precise to deal with the "fuzziness" of the real world."

Lotfi A. Zadeh

Este capítulo apresenta o núcleo desta proposta metodológica concebida para a implementação de uma base de regras de produção usando funções de pertinência *fuzzy*, implementando a fase de codificação (*fuzzification*), para monitoramento inteligente de condição de máquinas.

A base de regras *fuzzy* tem sua implementação descrita no Capítulo 4, na interface ECLIPSE usando a biblioteca *FuzzyJess*. No Capítulo 5 é feito um estudo prévio ou preliminar com a nova base de regras *fuzzy*, para avaliar sua integração futura na camada de monitoramento de condição do SIMPREBAL, usando o histórico de dados das variáveis físicas monitoradas.

3.1 INTRODUÇÃO

A abordagem metodológica proposta neste trabalho nasceu da necessidade de lidar com a incerteza inerente às faixas de operação de qualquer grandeza física, cujos valores são mensurados através de sensores que encontram-se habitualmente instalados em qualquer processo industrial, e no caso específico de estudo da usina hidrelétrica de Balbina.

As máquinas que conformam as unidades geradoras hidráulicas de Balbina são monitoradas pelo sistema SIMPREBAL, através das leituras das variáveis físicas advindas dos sensores instalados nessas máquinas.

A proposta metodológica é desenvolvida através de uma série de passos genéricos que são apresentados a seguir:

a) Apresentar o cenário e elaborar um mapeamento de todas as variáveis físicas (leituras de sensores) do sistema que controlam o estado ou condição de cada máquina do processo industrial em estudo (Balbina);

b) Considerando as variáveis em estudo obtidas em (a), estabelecer tabelas das faixas de operação de cada variável, leituras do sensor, propriedades do sensor e uma análise estatística (gráfico de tendências e histórico de anomalias dos valores de cada variável) para analisar o comportamento de cada variável e por último, codificar as leituras e faixas de operação dessas variáveis usando funções de pertinência *fuzzy* (processo de codificação);

c) Implementar uma base de regras de produção usando o tipo de funções de pertinência *fuzzy* obtidas em (b), e simulá-las com o histórico de leituras dos sensores do sistema, para avaliar as tendências dos estados das máquinas e comparar os resultados com os obtidos pela base de regras original do sistema, com o intuito de estabelecer um estudo prévio sobre a viabilidade de futura integração do conjunto de regras *fuzzy* na camada de monitoramento de condição do SIMPREBAL.

Algumas observações com relação à lista de passos descritas são apresentadas a seguir:

1. Os passos (a) e (b) são desenvolvidos neste capítulo, mas a codificação é implementada no Capítulo 4;

2. No passo (c), o raciocínio para a concepção da base de regras *fuzzy* é desenvolvido neste Capítulo, mas tem sua implementação descrita no Capítulo 4;

3. Os passos desde (a) até (c) serão validados no Capítulo 5 (estudo de caso sistema de mancais) com os valores do histórico de leituras dos sensores, para avaliação de uma futura integração da base de regras *fuzzy* no SIMPREBAL.

O conjunto de passos propostos nesta metodologia são apresentados a partir do item 3.2.

3.2 DEFINIÇÃO E AQUISIÇÃO DAS ENTRADAS OU VARIÁVEIS DO SISTEMA

Processos e modelos baseados em lógica *fuzzy* interatuam com o mundo exterior. Na verdade funções de pertinência *fuzzy* ou conjuntos *fuzzy* são entidades abstratas que não existem no mundo real como entidades físicas, mas eles são muito mais reflexivos da maneira de como é percebido o mundo real, de como são construídos seus modelos operacionais apropriados e informação do processo em algum nível desejado de abstração (granularidade) que oferece o mais apropriado nível de conceituação.

Segundo o anteriormente exposto, qualquer comunicação e interoperabilidade com o mundo exterior requer algum mecanismo bem desenvolvido de interação o interfaceamento (*interfacing*).

Para o estudo de caso, usando a base de históricos de dados da usina hidrelétrica de Balbina, o interfaceamento do sistema SIMPREBAL com a usina o estabelece a DFI (ver Figura 3.1), disponibilizando todos os dados advindos da instrumentação (leituras dos sensores instalados nas 5 unidades geradoras) através de um servidor OPC para que o sistema SIMPREBAL possa obtê-los através de uma rede Ethernet.

Na Figura 3.1 é mostrada uma configuração de rede industrial típica de cada unidade geradora hidráulica (conjunto de sensores conectados às máquinas para monitorar sua condição e enviar informações sobre a condição da máquina monitorada para seus transmissores).

A usina hidrelétrica de Balbina usa a tecnologia *Foundation Fieldbus*, onde os instrumentos (transmissores de temperatura) são agrupados em uma rede por canais e conectados a uma DFI (*Fieldbus Universal Bridge*).

Cada DFI tem quatro canais de comunicação, geralmente uma é de reserva. Um servidor OPC publica os *itens* (valores e outros parâmetros de configuração da instrumentação inteligente). Neste cenário podem-se obter informações dos instrumentos através da conexão com o servidor OPC. Com relação ao banco de dados é armazenado o histórico de anomalias, tomadas de decisão para uma eventual detecção de falha, as variáveis do processo, registro de usuários e dados das máquinas.

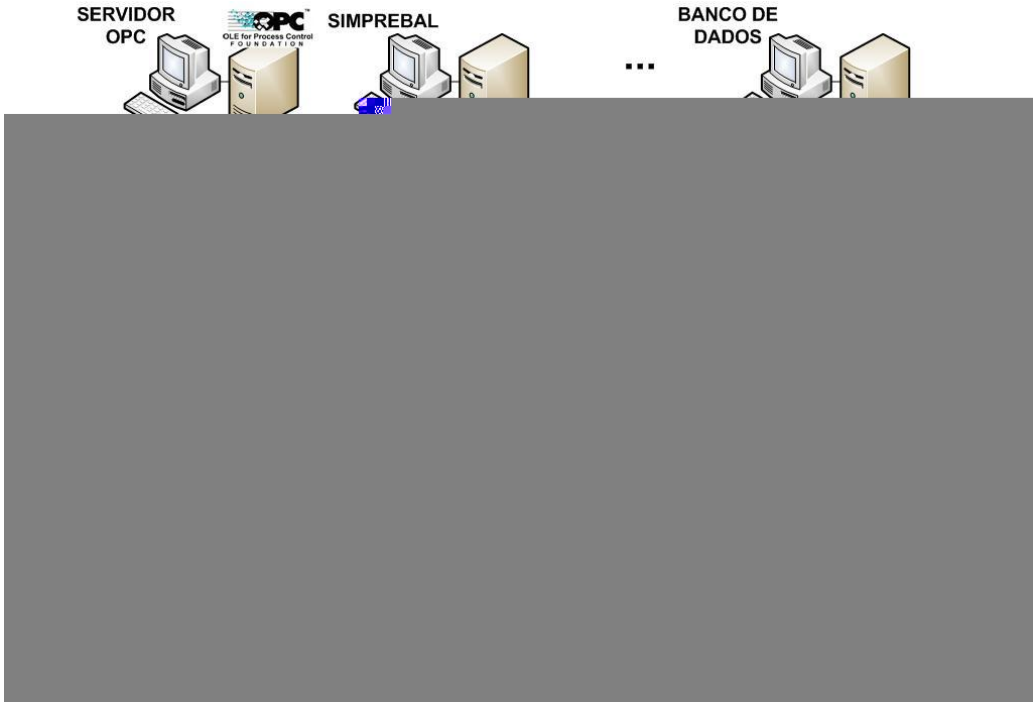

Figura 3.1 – Arquitetura e interação do sistema SIMPREBAL com a instrumentação da usina de Balbina em cada unidade geradora.

Os sinais obtidos pelos sensores da Figura 3.2, como produto de monitorar os estados de condição das máquinas, são inerentemente análogos e precisam ser digitalizados usando algum método de conversão analógico-digital.

Os processos de interfaceamento no nível de sensor são: conversão analógico-digital e conversão digital-analógico. A conversão analógico-digital é realizada na camada de **aquisição de dados** do modelo OSA-CBM, isto é feito com precisão pelos sensores inteligentes *Foundation Fieldbus* instalados estrategicamente nas cinco unidades geradoras hidráulica.

Basicamente os sensores de tecnologia SMAR, mapeiam as variáveis do processo, como temperatura, nível, fluxo ou vazão e traduzem esses sinais análogos em sinais digitais usando técnicas de conversão analógico-digital para oferecer um sinal digital com parâmetros próprios que apresente à variável digitalizada para o seu subsequente processamento na **camada de processamento de sinal** do padrão OSA-CBM (Smar 2001, Smar 2005).

Para o estudo de caso (apresentado no Capítulo 5), o Subsistema Mancal Guia do Gerador (SMGG) possui sapatas[1] e uma cuba[2], e segundo o histórico de anomalias desse subsistema no ano 2009, as sapatas apresentaram anomalias e foi por esse motivo o eleito para validação do conjunto de regras *fuzzy* desenvolvidas nesta metodologia com o objetivo de avaliar uma futura integração na camada de monitoramento de condição do sistema SIMPREBAL.

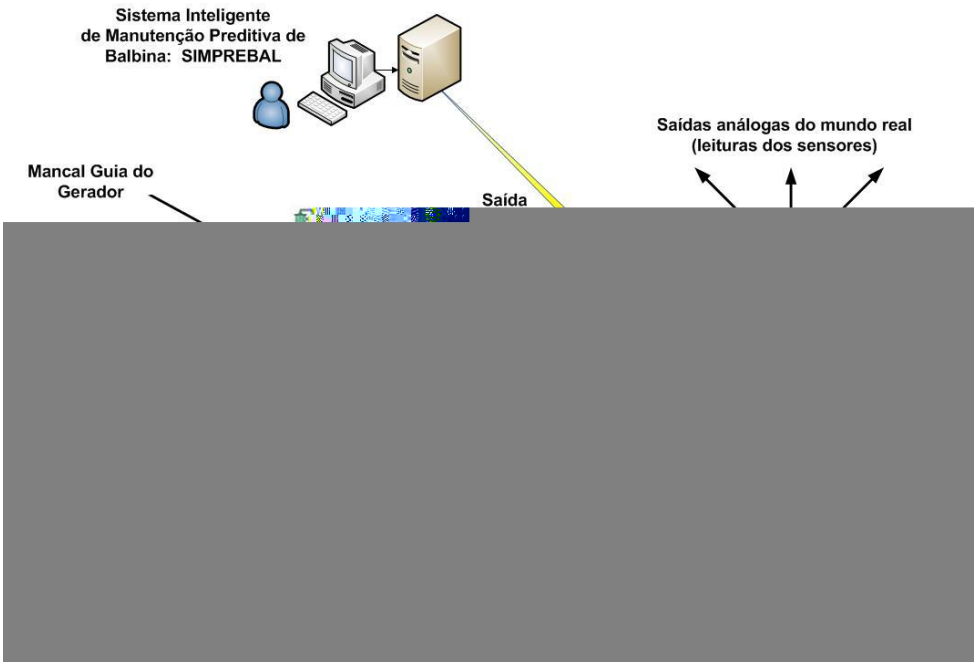

Figura 3.2 – Processamento digital: conversão analógico-digital e digital-analógico.

O SMGG apresenta as seguintes variáveis de entrada para controlar a temperatura das sapatas como mostrado na Tabela 3.1:

Tabela 3.1 – Mapeamento das variáveis físicas em estudo

MÁQUINA	FUNÇÃO	VARIÁVEL FÍSICA
Sapata 01	Permite a formação e manutenção de um filme de óleo (efeito hidrodinâmico) entre o metal patente e o eixo da turbina durante o funcionamento da unidade geradora	Temperatura do metal número 01
Sapata 02		Temperatura do metal número 02
Sapata 03		Temperatura do metal número 03

[1] Sapata é considera como máquina ou equipamento nesta dissertação e foi definido na Tabela 3.1.
[2] Cuba é considerada como máquina ou equipamento nesta dissertação e tem como função armazenar óleo de modo a imergir as partes ativas do SMGG.

3.3 *FUZZIFICATION* DAS ENTRADAS DO SISTEMA USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

Nesta seção considerando as variáveis em estudo obtidas na seção anterior, são geradas tabelas das faixas de operação de cada variável, leituras do sensor, propriedades do sensor e uma análise estatística (gráfico de tendências e histórico de anomalias dos valores de cada variável) para analisar o comportamento de cada variável e por último codificar as leituras e faixas de operação dessas variáveis usando funções de pertinência *fuzzy* (processo de *fuzziness*).

Para o caso do sistema SIMPREBAL o conjunto de variáveis do sistema estão completamente estabelecidos e cada uma deles tem várias faixas estabelecidas com significado próprio segundo as especificações técnicas dos instrumentos e valores prefixados pelos operários da planta da usina hidrelétrica de Balbina.

Especificamente no Capítulo 5, serão especificadas completamente as três variáveis em estudo (variáveis de temperatura dos metais) mostrando os parâmetros que as identificam, os valores históricos das leituras dos sensores no qual se identificou anomalias ou eventos, entre outros parâmetros. Mas para fins ilustrativos deste passo metodológico é apresentada a Tabela 3.1 a qual mostra as três variáveis para o SMGG e suas faixas de operação.

A Figura 3.3 apresenta o comportamento da variável temperatura do metal número 1. Por outro lado, a Figura 3.4 apresenta um gráfico das faixas de valores de cada sensor em estudo.

Tabela 3.2 – Descrição das variáveis do sistema em estudo

TAG	DESCRIÇÃO	PV	ALERTA	ALARME	TRIP	FAIXA
38GMM1	Temperatura do metal número 01	56°C	75°C	80°C	85°C	0-200°C
38GMM2	Temperatura do metal número 02	48°C	75°C	80°C	85°C	0-200°C
38GMM3	Temperatura do metal número 03	59°C	75°C	80°C	85°C	0-200°C

e tendências para a temperatura do metal 01

ura 3.4 – Rep gráfica da faixa de valores das variáveis físicas

a análise estatística es na Figura 3.3 pode ser calculado o valor máximo °C), o valor mínimo (4 5°C) e o valor meio (55.583°C) para a temperatura do 1. Estes valores são im rtantes para estabelecer os limites das funções de nência *fuzzy* e serão analisados no Capítulo 5.

om relação à Figura 3.4, algumas observações são descritas a seguir:

- As faixas de valores não têm nenhum valor de transição. Uma solução a isto é desenvolvida neste Capítulo usando funções de pertinência *fuzzy* triangulares. Com isto, pretende-se dar flexibilidade ao processamento de monitoramento de condição das regras não *fuzzy* do sistema SIMPREBAL para uma integração futura.

- Qualquer valor contido nessas faixas possui o máximo nível de pertinência a essa faixa. Para solucionar esta pertinência total, se propõe usar funções de pertinência *fuzzy* triangulares para atingir uma pertinência parcial de cada valor de temperatura do sensor em alguma faixa específica. Com isto, pretende-se incrementar a confiabilidade de monitoramento de condição de máquinas usando uma base de regras *fuzzy*, para serem implementadas futuramente na camada de monitoração de condição do SIMPREBAL.

As funções triangulares *fuzzy* quantificam a incerteza contida nos sinais analógicos e por outro lado, funções *singletons*[3] *fuzzy* quantificam uma magnitude no qual a incerteza é desprezível no nível dos conjuntos *fuzzy* quando é requerido codificar (*fuzzificate*) um valor advindo da leitura de um sensor. Ver Equações A.1 e A.4 no apêndice A.

Para finalizar com o tema da codificação (*fuzzification*) das leituras e faixas dos sensores de temperatura de entrada, a seguir será estabelecido o tipo de análise, que será aplicada à variável física de **temperatura do metal número 1**. Este tipo de análise é aplicável para qualquer variável física sob monitoramento.

Quando se codifica a leitura de um sensor usando funções triangulares *fuzzy* é necessário ter em conta algumas propriedades do transmissor. Na tabela 3.3 são mostradas as características mais importantes do transmissor de temperatura que mensura a variável do metal número 1.

Tabela 3.3 – Lista das características mais relevantes do transmissor de temperatura TT302

Característica	Detalhes técnicos
Saída	Digital *Foundation Fieldbus,* modo de tensão 31,25 kbits/s com alimentação pelo barramento.
Faixa Máxima	-40°C a 200°C.
Precisão	+/-0,02%

[3] Funções *singletons* são basicamente funções triangulares com base igual a zero. Uma função *singleton* é caracterizada por possuir só um valor no seu domínio, no qual é atingido o máximo valor de pertinência.

Usando os valores da Tabela 3.3 pode-se criar a Figura 3.5 que mostra a fase de codificação (*fuzzification*), por exemplo, para uma leitura do sensor de 42°C.

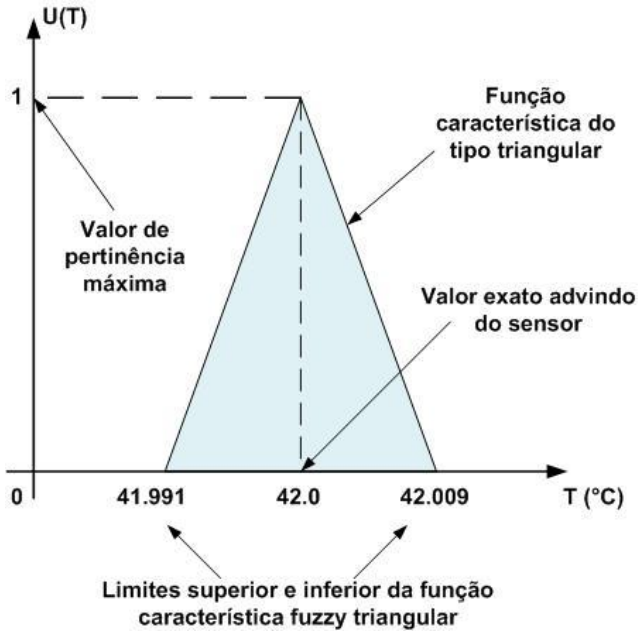

Figura 3.5 – Codificação da leitura do sensor usando uma função triangular

Na Figura 3.5, o limite superior (42.009°C) e inferior (41.991°C), da função triangular que tem como valor central a 42°C, são calculados com a precisão do instrumento (+/- 0.02%). Mas como a precisão do transmissor é bastante boa, convém codificar as variáveis físicas usando funções de pertinência *singletons fuzzy* como mostrado na Figura 3.6.

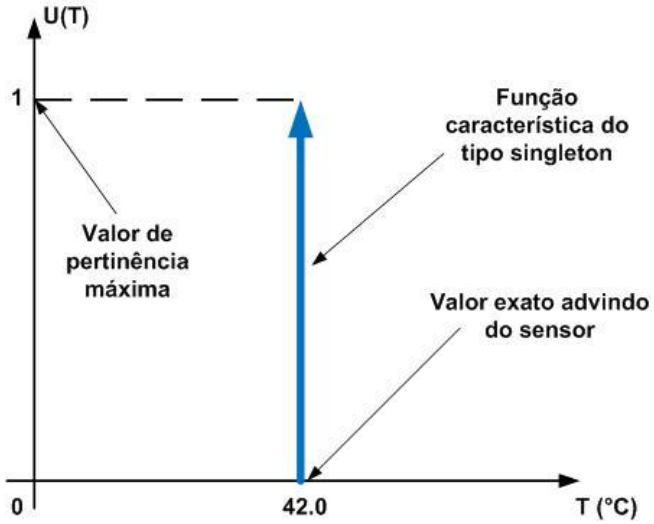

Figura 3.6 – Codificação da leitura do sensor usando uma função *singleton*

Usando funções *singletons* para codificar as variáveis de entrada têm vantagens de processamento do que usando as funções triangulares, pois é só necessário computar em um só ponto para a função *singleton*, a diferença das triangulares, onde é necessário computar um intervalo de pontos.

Por outro lado para codificar as faixas de operação dos sensores, são usadas funções triangulares e trapezoidais *fuzzy* para quantificar a pertinência parcial do valor de temperatura em qualquer faixa de operação do sensor.

Na Figura 3.7 é apresentada a codificação das faixas de operação para a temperatura do metal número 1 pertencente à sapata número 1 do SMGG.

Algumas observações com relação à Figura 3.7 são apresentadas a seguir:

- A variável de temperatura do metal número 1, possui agora faixas de transição linearmente crescentes e decrescentes, as quais representam graficamente a incerteza o faixa de transição que possui cada faixa de operação da variável física.
- As funções de pertinência *fuzzy* triangulares por serem funções lineares são mais fáceis de computar do que as funções gaussianas, pois as funções gaussianas são definidas por funções exponenciais (ver Equações A.1 e A.2 no Apêndice A).
- As funções de pertinência *fuzzy* para a variável de temperatura do metal 1, foram estabelecidas segundo o histórico de leituras do sensor na base de dados do SIMPREBAL e entrevistas com os operários da usina de Balbina.
- Os valores e as formas dadas para construir as funções de pertinência *fuzzy* serão calculados no Capítulo 4 e avaliados no Capítulo 5.

O seguinte e último passo é conceber o método para gerar as regras de produção usando o tipo de funções de pertinência *fuzzy* selecionados na secção 3.3. A base de regras concebidas na secção 3.4 serão implementadas no Capítulo 6 usando a interface JESS de forma não integrada à base de conhecimento do sistema SIMPREBAL. Por último, para avaliar sua integração futura, na camada de monitoramento de condição do SIMPREBAL, o Capítulo 5 oferece um estudo de caso no SMGG, com o histórico de dados nos meses de setembro, outubro e novembro do ano 2009, da usina de Balbina.

Figura 3.7 – Funções de pertinência *fuzzy*: BAIXA, NORMAL, ALTA, MUITO ALTA e EXTREMAMENTE ALTA.

3.4 CONCEPÇÃO DA BASE DE REGRAS DE PRODUÇÃO *FUZZY*

Nesta última fase é feito a análise para a concepção da estrutura das regras de produção usando as funções de pertinência *fuzzy* obtidas na subsecção anterior. Esta fase possui mais observações e artifícios computacionais do que conceituais.

A base de regras usando funções de pertinência *fuzzy* concebidas nesta secção serão implementadas no Capítulo 4, na interface JESS usando a biblioteca *FuzzyJess*. Esta biblioteca integra conceitos de lógica *fuzzy* ao sistema experto para Java (JESS). No Apêndice B são detalhadas as funções de usuário básicas para a criação de regras *fuzzy* em JESS.

Antes de estabelecer a estrutura das regras em formato pseudocódigo é necessário efetuar uma análise prévia da estrutura das regras originais do sistema SIMPREBAL.

3.4.1 Estrutura da base de regras original do sistema SIMPREBAL

A base de regras original desenvolvida na concepção do sistema inteligente de manutenção preditiva, para as camadas de processamento de sinal, monitoramento de condição e avaliação da saúde, possuem a seguinte estrutura em formato pseudocódigo para o SMGG pertencente à unidade geradora número 2:

Camada de processamento de sinal:

SE
O sinal OPC do metal N°1 na sapata N°1 tem uma *qualidade* igual a 3
ENTÃO
O sinal OPC do metal N°1na sapata N°1 possui uma condição de sinal BOA

SE
O sinal OPC do metal N°1 na sapata N°1 possui uma condição de sinal BOA e
O sinal OPC do metal N°1 na sapata N°1 possui um *status* igual a 2
ENTÃO
O sinal *fieldbus* do metal N°1 na sapata N°1 possui uma condição BOA

Camada de monitoração de condição:

SE
A temperatura do metal N°1 na sapata N°1 é maior o igual que 75°C e
O sinal *fieldbus* do metal N°1 na sapata N°1 tem uma condição BOA
ENTÃO
A temperatura do metal N°1 na sapata N°1 possui uma condição ALERTA.

Camada de avaliação de saúde:

SE
A temperatura do metal N°1 na sapata N°1 possui um condição de ALERTA ou
A temperatura do metal N°2 na sapata N°2 possui um condição de ALERTA ou
A temperatura do metal N°3 na sapatas N°3 possui um condição de ALERTA

ENTÃO (ações operacionais de manutenção através de ordens de serviço)

-Possíveis vibrações excessivas do eixo da turbina.

-As sapatas do mancal poderiam estar desalinhadas.

-Enviar emails para o grupo de mecânicos.

-No sinótico do SIMPREBAL da UGH02 piscar na cor amarela

Detalhamentos e significados dos sinais OPC e *fieldbus* para a camada de processamento do sinal, podem ser encontrados em trabalhos de Amaya (2008) e Souza (2008). Aqui não serão especificadas porque ficam fora do escopo desta dissertação.

Como se pode ver nas três camadas apresentadas, essas são as estruturas básicas no qual está formada toda a base de conhecimento do sistema SIMPREBAL. Uma das lacunas para este conjunto de regras é que, se não é estabelecida uma consistente condição para a variável física (temperatura do metal N°1, N°2, N°3), ela gerará um diagnóstico errado e, por conseguinte um conjunto de ações de manutenção não adequadas.

Por exemplo, no caso que a variável de temperatura do metal N°1 na camada de monitoração da condição esteja marcando 74.775°C em algum instante de tempo, o sistema SIMPREBAL, segundo a sua base de conhecimento, não vai identificar esse valor de temperatura como uma situação de ALERTA e, por conseguinte vai ser considerada como uma situação NORMAL, gerando assim falsos eventos ou anomalias. Este tipo de comportamento deve-se ao fato de que as faixas de operação de ALERTA, ALARME e TRIP não possuem nenhum tipo de flexibilidade com relação a valores muito próximos a cada faixa de operação de suas variáveis físicas (não existem valores de transição).

Além disso, a estrutura de regras propostas na camada de avaliação da saúde não reflete consistentemente a verdadeira natureza de inter-relacionamento entre as suas variáveis dentro de algum equipamento ou subsistema a controlar. Por exemplo, o inter-relacionamento entre as variáveis de temperatura do metal N°1, N°2, e N°3. Além disso, só é estabelecido um conjunto de possíveis ações de manutenção preventiva, mas não possui nenhuma ação de controle que poderia ser implementada no futuro, como uma ação corretiva para o caso em situações onde é requerida uma ação rápida e efetiva.

O anteriormente exposto é efetivamente desempenhado pela lógica *fuzzy*. Para o caso da **camada de monitoração de condição**, o problema pode ser abordado através do uso de **funções de pertinência *fuzzy*** e estabelecendo **indicadores de degradação de sinal.**

Os **indicadores de degradação de sinal**[4] são definidos nesta dissertação como uma medida de mensurar a degradação da leitura de um sensor que monitora as condições de uma máquina, com o intuito de diagnosticar e prognósticas falhas ou defeitos futuros. A ideia anteriormente exposta, adere-se à definição de prognóstico inteligente, o qual foi definido por Holsapple (2003), no Capítulo 2, página 13.

Os **indicadores de degradação de sinal** (Y1, Y2 e Y3) usados nesta dissertação são obtidos de forma gráfica (ver Figura 3.8) como resultado de projetar no eixo vertical a intersecção da leitura instantânea codificada do sensor (X°C) com a função de pertinência *fuzzy* que a intersecta (BAIXA e NORMAL).

Com relação à Figura 3.8, algumas observações são apresentadas a seguir:

- X°C é o valor da leitura do sensor em algum instante de tempo codificado usando uma função de pertinência *singleton fuzzy*.
- Os valores de Xi, Xi+1 e Xi+2 são calculados tendo em conta o histórico de dados dessa variável e entrevistas com os operários que monitoram essa variável.
- Y1 é um valor no intervalo [0, 1], como resultado de projetar no eixo vertical a intersecção de X°C com a função de pertinência *fuzzy* BAIXA.
- Y3 é um valor no intervalo [0, 1], como resultado de projetar no eixo vertical a intersecção de X°C com a função de pertinência *fuzzy* NORMAL.
- Y2 é um valor no intervalo [0, 1], como resultado de projetar no eixo vertical a intersecção das funções de pertinência *fuzzy* BAIXA e NORMAL.
- Para calcular o correto estado de funcionamento de máquinas são comparados continuamente os valores de Y1 e Y3, de maneira que:
  Se: Y1>Y3 então a condição é BAIXA.
  Se: Y1=Y3 então a condição é BAIXA.
  Se: Y1<Y3 então a condição é NORMAL.

[4] Indicador de degradação de sinal possui o mesmo significado que o nível de pertinência de um valor a um conjunto *fuzzy*, e foi usada tal terminação pelo motivo que o trabalho está inserido no contexto de manutenção preditiva de máquinas.

- Para seguir a pista relativa à degradação da saúde da máquina que o sensor de temperatura tem sob controle, pode-se estabelecer um gráfico de Y1 através do tempo para saber como está evoluindo a degradação desse sinal. No capítulo 5 são mostradas algumas figuras simulando o histórico de dados da variável de temperatura do metal N°2.

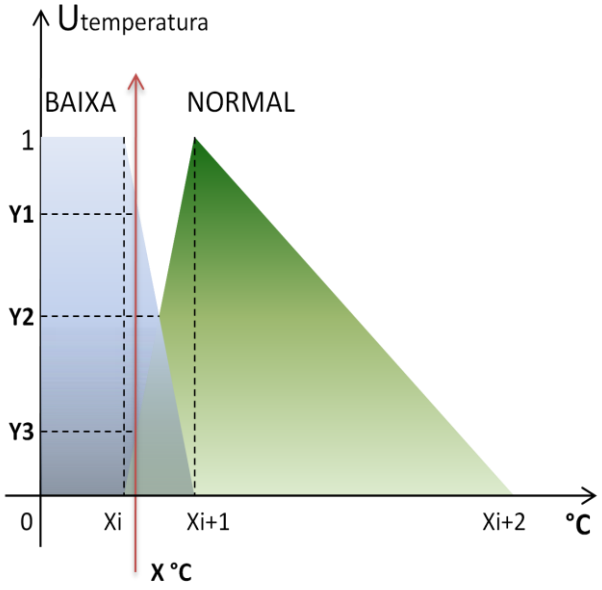

Figura 3.8 – Gráfico para o cálculo do **indicador de degradação de sinal**

Por outro lado, o problema na camada de avaliação da saúde poderia ser abordado através do uso de termos linguisticos (conjunto *fuzzy*) que mapeiem a verdadeira natureza de inter-relacionamento entre as suas variáveis e ao mesmo tempo possam ser desempenhadas algumas ações de controle no sistema (através de desempenhar inferência *fuzzy*).

Tendo em conta todas as observações anteriormente expostas, pode-se conceber a nova base de regras de produção usando funções de pertinência *fuzzy*.

3.4.2 Estrutura da base de regras *fuzzy* para futura integração na camada de monitoração de condição do sistema SIMPREBAL

Para fins explicativos far-se-á a análise na variável de temperatura do metal N°2 (nas suas funções de pertinência *fuzzy*: BAIXA e NORMAL), este raciocínio é satisfatoriamente aplicável para as outras duas variáveis de temperatura (temperatura do metal N°1 e temperatura do metal N°3) em todas suas funções de pertinência *fuzzy*.

Na Figura 3.9 é apresentado o gráfico principal para a criação das regras de monitoração da condição para esta variável em particular. É considerada uma leitura arbitrária de temperatura (X°C) que é codificada com uma função *singleton* a qual corta as funções de pertinência *fuzzy* BAIXA e NORMAL em Y1 e Y3 respectivamente.

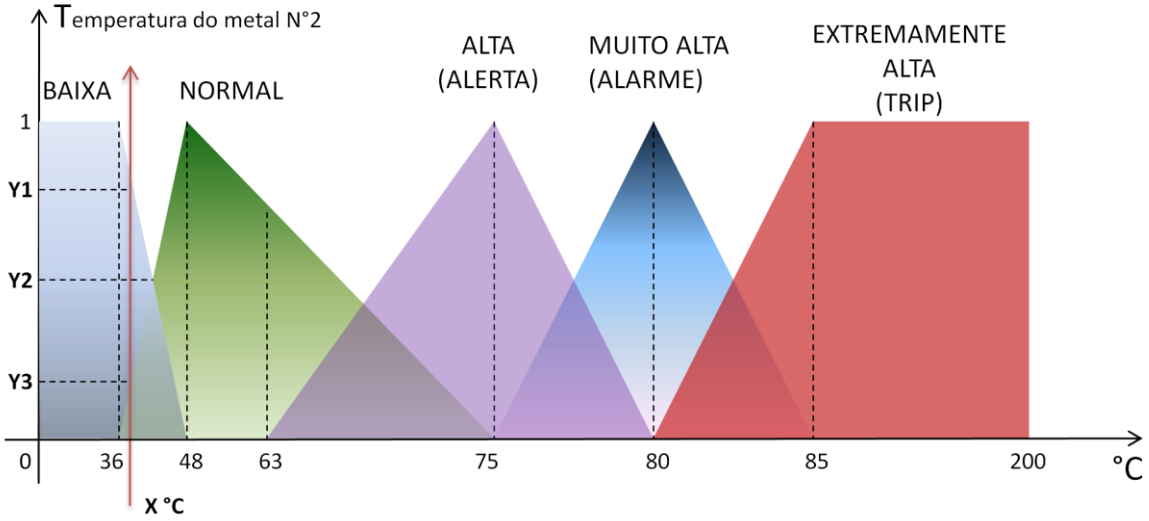

Figura 3.9 – Gráfico para o análise da concepção das regras *fuzzy*

A seguir é desempenhada a análise em formato pseudocódigo, assume-se que os valores (36, 48, ..., 85°C) foram calculados previamente através de uma análise estatística de tendências das leituras do sensor, estes valores serão calculados no Capítulo 6:

R1: **SE** {A INTERSECÇÃO DE X°C COM O TERMO FUZZY BAIXA > 0}
ENTÃO
{Obter indicador de degradação do termo *fuzzy* BAIXA (Y1)}
{Obter indicador de degradação do termo *fuzzy* NORMAL (Y3)}
{**SE** Y1 > Y3
ENTÃO
Condição BAIXA com indicador de degradação Y1}
{**SE** Y1 = Y3
ENTÃO
Condição BAIXA com indicador de degradação Y1}
{**SE** Y1 < Y3
ENTÃO Condição NORMAL com indicador de degradação Y3}

R2: **SE** {A INTERSECÇÃO DE X°C COM O TERMO FUZZY NORMAL > 0}
ENTÃO
{Obter indicador de degradação do termo *fuzzy* NORMAL (Y1)}
{Obter indicador de degradação do termo *fuzzy* ALTA (Y3)}
{SE Y3>0
ENTÃO
{**SE** Y1 > Y3
ENTÃO
Condição NORMAL com indicador de degradação Y1}
{SE Y1 = Y3
ENTÃO
Condição NORMAL com indicador de degradação Y1}
{SE Y1 < Y3
ENTÃO
Condição ALTA com indicador de degradação Y3}
}

...

No capítulo 4, serão implementadas estas estruturas de regras *fuzzy* (incluindo a fase de *fuzzification*) e será avaliada, no Capítulo 5, uma comparação de desempenho com a base de regras original do sistema SIMPREBAL.

3.5 CONSIDERAÇÕES E SÍNTESE DO CAPÍTULO

A seguir são sintetizadas as observações mais relevantes deste capítulo:

Foi proposta e desenvolvida uma metodologia genérica para a concepção de uma base de regras usando funções de pertinência *fuzzy* implementando a fase de codificação da lógica *fuzzy* para monitoramento inteligente de máquinas.

Foi proposto o raciocínio para a fase de codificação (*fuzzification*) das leituras (usando funções de pertinência *fuzzy* singleton) e das faixas dos sensores (usando funções de pertinência trapezoidais e triangulares *fuzzy*) das variáveis físicas de temperatura.

Neste capítulo foi definido e inserido o conceito de "**indicador de degradação de sinal**" que será usado ao longo dos Capítulos 4 e Capítulo 5.

Foi proposto o raciocínio para conceber a nova base de regras *fuzzy*. A base de regras fuzzy será implementada no Capítulo 4 e no Capítulo 5 é feito um estudo prévio sobre a viabilidade de uma futura integração na camada de monitoramento de condição do sistema SIMPREBAL.

4 IMPLEMENTAÇÃO COMPUTACIONAL

"A imaginação é mais importante que o conhecimento"

Albert Einstein.

Neste capítulo, primeiro é estabelecido o raciocínio para obter os valores que constroem as funções de pertinência triangulares e trapezoidais *fuzzy* mencionadas no Capítulo 3, com o intuito de definir as faixas de operação das variáveis físicas que serão usadas para implementar a fase de codificação (*fuzzification*).

Depois de estabelecer as funções de pertinência *fuzzy*, são implementadas as regras de produção usando essas funções de pertinência. A base de regras *fuzzy* é criada na interface JESS usando a biblioteca *FuzzyJess*.

No Capítulo 5 será feito um prévio estudo sobre a integração das regras *fuzzy* geradas neste capítulo, para avaliar uma futura integração na camada de monitoramento de condição do sistema SIMPREBAL.

4.1 MÉTODO DE ANÁLISE PARA A CRIAÇÃO DAS FUNÇÕES DE PERTINÊNCIA *FUZZY*

Como uma primeira instância esta seção objetiva a análise de tendências do histórico de dados para as variáveis físicas em estudo (temperatura do metal 01, 02 e 03 nas sapatas do mancal guia superior que conformam o SMGG) obtidos através da base de dados do sistema SIMPREBAL nos meses de setembro, outubro e novembro do ano 2009 da UGH02 com o intuito de estabelecer uma série de condições para os parâmetros que conformarão e delimitarão a criação de cada função de pertinência *fuzzy* em cada variável em particular.

Depois de estabelecer as funções de pertinência *fuzzy*, será criado um conjunto de regras de produção usando essas funções de pertinência *fuzzy* para monitoramento inteligente de condição de cada variável em estudo. Com esse conjunto de regras *fuzzy* pretende-se elaborar um estudo prévio no Capítulo 5 sobre a sua viabilidade de integração na camada de monitoramento de condição do sistema SIMPREBAL.

4.1.1 Análise estatística para a variável de temperatura do metal número 1

A seguir é mostrada a Tabela 4.1, que apresenta os valores de configuração da variável de temperatura do metal 01 (TAG: 38GMM1) para o seu monitoramento de condição.

Tabela 4.1 – Valores de configuração para a variável de temperatura do metal número 01

TAG	DESCRIÇÃO	PV	ALERTA	ALARME	TRIP	FAIXA
38GMM1	Temperatura do metal número 01	56°C	75°C	80°C	85°C	0-200°C

O valor de ALERTA foi estabelecido pela equipe do SIMPREBAL como uma media de ação proativa, os valores de ALARME e TRIP foram estabelecidos pelos funcionários da usina de Balbina. O PV (variável de processo) é o **valor médio** redondeado da Tabela 4.3

Analisando o histórico de dados para esta variável de temperatura nos meses de setembro, outubro e novembro do ano 2009, não houve eventos produzidos nesses meses como mostrado na Tabela 4.2.

Tabela 4.2 – Eventos produzidos pela variável temperatura do metal 01

VARIÁVEL	Número de ALERTAS	Número de ALARMES	Número de TRIPS	Médio de valores no qual se atingiu ALERTA	Médio de valores no qual se atingiu ALARME	Médio de valores no qual se atingiu TRIP
Temperatura do metal número 01	0	0	0	-	-	-

A Tabela 4.1 e a Tabela 4.2 são consistentes com a tendência dos valores para esta variável nos meses de setembro, outubro e novembro do ano 2009 como é mostrada na Figura 4.1.

Analisando o histórico de dados para esta variável de temperatura (38GMM2) nos meses de setembro, outubro e novembro do ano 2009, os eventos produzidos nesses meses são mostrados na Tabela 4.5.

Tabela 4.5 – Eventos produzidos pela variável temperatura do metal 02

VARIÁVEL	Número de ALERTAS	Número de ALARMES	Número de TRIPS	Médio de valores no qual se atingiu ALERTA	Médio de valores no qual se atingiu ALARME	Médio de valores no qual se atingiu TRIP
Temperatura do metal número 02	45	183	117	78.994°C	82.383°C	87.362°c

A Tabela 4.4 e a Tabela 4.5 são consistentes com a tendência dos valores para esta variável nos meses de setembro, outubro e novembro do ano 2009 como é mostrada na Figura 4.2.

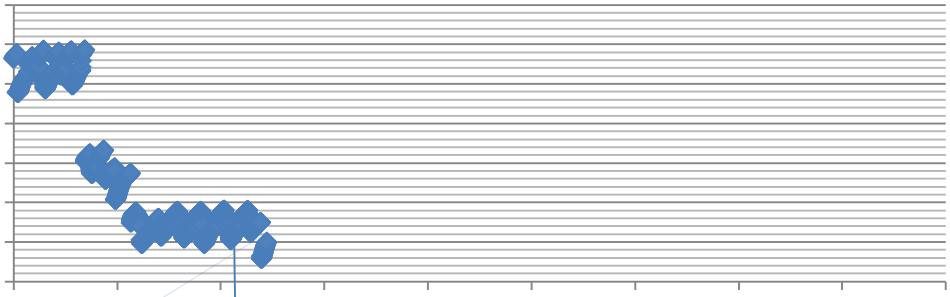

Figura 4.2 – Gráfico de tendências para a temperatura do metal 02

Segundo o histórico de dados para esta variável e os dados fornecidos pela Figura 4.2 foram calculados os seguintes valores como são mostrados na Tabela 4.6 (estes dados foram previamente filtrados para valores menores a 75°C):

Tabela 4.6 – Valores máximo, mínimo e médio atingido pela temperatura do metal 02

VARIÁVEL	VALOR MÁXIMO	VALOR MÍNIMO	VALOR MÉDIO
Temperatura do metal número 02	63.345°C	35.784°C	48.403°C

Os valores contidos na Tabela 4.4, Tabela 4.5 e Tabela 4.6 são consistentes.

4.1.3 Análise estatística para a variável de temperatura do metal 3

A seguir é mostrada a Tabela 4.7, que apresenta os valores de configuração da variável de temperatura do metal 03 (TAG: 38GMM3) para o seu monitoramento de condição.

Tabela 4.7 – Valores de configuração para a variável de temperatura do metal número 03

TAGS	DESCRIÇÃO	PV	ALERTA	ALARME	TRIP	FAIXA
38GMM3	Temperatura do metal número 03	59°C	75°C	80°C	85°C	0-200°C

O valor de ALERTA foi estabelecido pela equipe do SIMPREBAL como uma media de ação proativa, os valores de ALARME e TRIP foram estabelecidos pelos funcionários da usina de Balbina. O PV (variável de processo) é o **valor médio** redondeado da Tabela 4.9.

Analisando o histórico de dados para esta variável de temperatura (38GMM3) nos meses de setembro, outubro e novembro do ano 2009, não houve eventos produzidos nesses meses como mostrado na Tabela 4.8.

Tabela 4.8 – Eventos produzidos pela variável temperatura do metal 03

VARIÁVEL	Número de ALERTAS	Número de ALARMES	Número de TRIPS	Médio de valores no qual se atingiu ALERTA	Médio de valores no qual se atingiu ALARME	Médio de valores no qual se atingiu TRIP
Temperatura do metal número 03	0	0	0	-	-	-

A Tabela 4.7 e a Tabela 4.8 são consistentes com a tendência dos valores para esta variável nos meses de setembro, outubro e novembro do ano 2009 como é mostrada na Figura 4.3.

Segundo o histórico de dados para esta variável e os dados fornecidos pela Figura 4.3 foram calculados os seguintes valores como são mostrado na Tabela 4.9.

Os valores contidos na Tabela 4.7, Tabela 4.8 e Tabela 4.9 são consistentes.

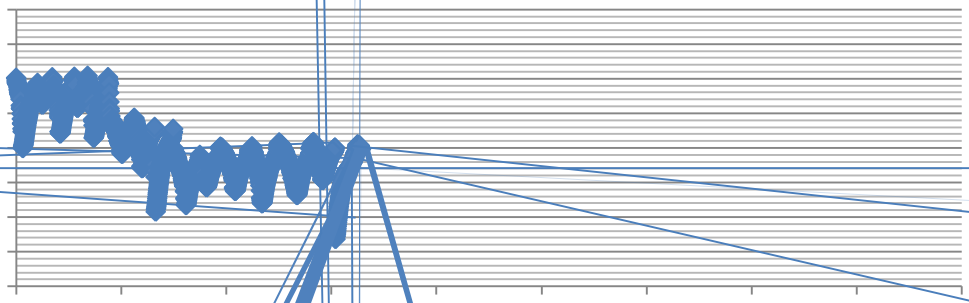

Figura 4.3 – Gráfico de tendências para a temperatura do metal 03

Tabela 4.9 – Valores máximo, mínimo e médio atingido pela temperatura do metal 03

VARIÁVEL	VALOR MÁXIMO	VALOR MÍNIMO	VALOR MÉDIO
Temperatura do metal número 03	69.794°C	41.814°C	58.917°C

Depois de estabelecer os valores de configuração para cada variável em estudo, são construídas as funções de pertinência *fuzzy* que conformarão suas faixas de operação.

4.1.4 Construção das funções de pertinência *fuzzy* para a variável de temperatura do metal número 1

Usando a Tabela 4.1, a Tabela 4.2 e a Tabela 4.3 é possível codificar de forma consistente a faixa de operação da variável de temperatura do metal número 1, usando funções triangulares (ver Figura A.1) em vez de usar funções gaussianas (ver Figura A.2) para evitar muito consumo de processamento computacional ao executar as regras de produção *fuzzy*.

A justificativa para esta escolha é que funções triangulares *fuzzy* são formadas por funções lineares e, por conseguinte, levam menos tempo de processamento (ver Equação A.1). Por outro lado, as funções gaussianas *fuzzy* são formadas por funções exponenciais e, por conseguinte, consomem um tempo maior de processamento (ver Equação A.2).

Na Figura 4.4 é mostrado o conjunto de funções de pertinência *fuzzy* para definir completamente os níveis de operação (BAIXA, NORMAL, ALTA/ALERTA, MUITO ALTA/ALARME e EXTREMAMENTE ALTA/TRIP) desta variável em particular.

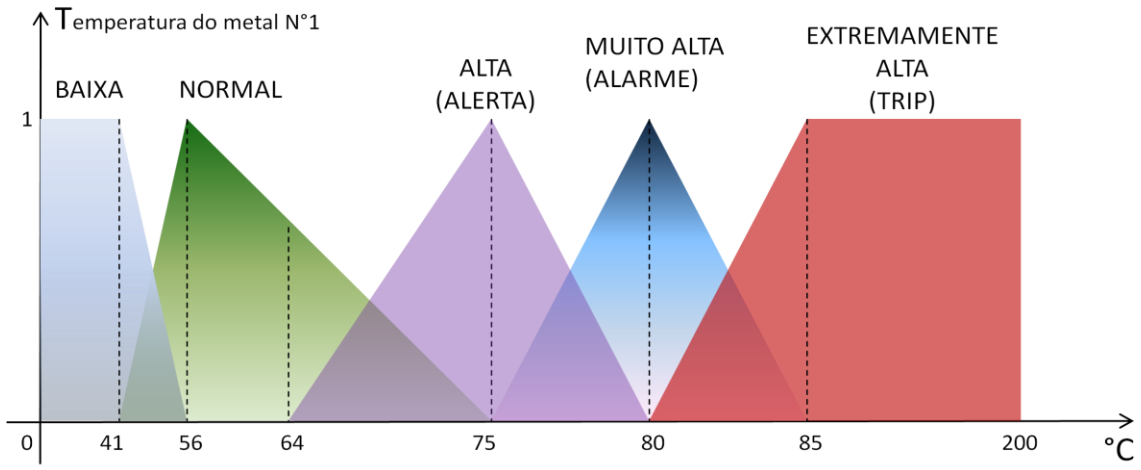

Figura 4.4 – Codificação da faixa de operação da temperatura do metal 1

As funções de pertinência *fuzzy* ou termos *fuzzy* na Figura 4.4, são consistentes com as Tabelas 4.1, 4.2 e 4.3, bem com o conhecimento heurístico fornecido pelos operários da usina de Balbina na última visita. Os valores que definem as formas das funções de pertinência *fuzzy* são definidos na Tabela 4.10.

Tabela 4.10 – Valores de configuração para os termos *fuzzy* do metal número 1

VARIÁVEL	VALOR MÍNIMO	VALOR MÉDIO	VALOR MÁXIMO	VALOR DE ALERTA	VALOR DE ALARME	VALOR DE TRIP
Temperatura do metal número 01	41°C	56°C	64°C	75°C	80°C	85°C

4.1.5 Construção das funções de pertinência *fuzzy* para a variável de temperatura do metal número 2

Usando a Tabela 4.4, a Tabela 4.5 e a Tabela 4.6 é possível codificar de forma consistente a faixa de operação da variável de temperatura do metal número 2, usando funções triangulares (ver Figura A.1) em vez de usar funções gaussianas (ver Figura A.2) para evitar muito consumo de processamento computacional ao executar as regras de produção *fuzzy*. A justificativa para esta escolha é a mesma feita no item 4.1.4.

Segundo a análise anteriormente exposta, esta variável de temperatura do metal N°2 apresentou eventos ou anomalias e, por esse motivo, é necessário elaborar uma prévia análise como é mostrado na Tabela 4.11.

Tabela 4.11 – Valores máximos e mínimos para os eventos ALERTA, ALARME e TRIP

VARIÁVEL	VALOR MÍNIMO QUE ATINGIU ALERTA	VALOR MÁXIMO QUE ATINGIU ALERTA	VALOR MÍNIMO QUE ATINGIU ALARME	VALOR MÁXIMO QUE ATINGIU ALARME	VALOR MÍNIMO QUE ATINGIU TRIP	VALOR MÁXIMO QUE ATINGIU TRIP
Temperatura do metal número 02	77.762°C	79.994°C	80.052°C	84.930°C	85.131°C	88.742°C

A Tabela 4.11 mostra os valores obtidos através do histórico de dados do SIMPREBAL, nos meses de setembro, outubro e novembro do ano 2009, para analisar quais foram os valores das leituras do metal número 2 no qual foi atingido o valor máximo e mínimo para os eventos ALERTA, ALARME e TRIP, estes valores também serão usados para validar se as funções de pertinência *fuzzy* triangulares estão configuradas corretamente. No Capítulo 5 será analisada, com a nova base de regras *fuzzy* aqui desenvolvida, a confiabilidade desses eventos produzidos pela temperatura do metal número 2.

Depois da análise anterior, na Figura 4.5 é mostrado o conjunto de funções de pertinência *fuzzy* para definir completamente os níveis de operação (BAIXA, NORMAL, ALTA/ALERTA, MUITO ALTA/ALARME e EXTREMAMENTE ALTA/TRIP) desta variável em particular.

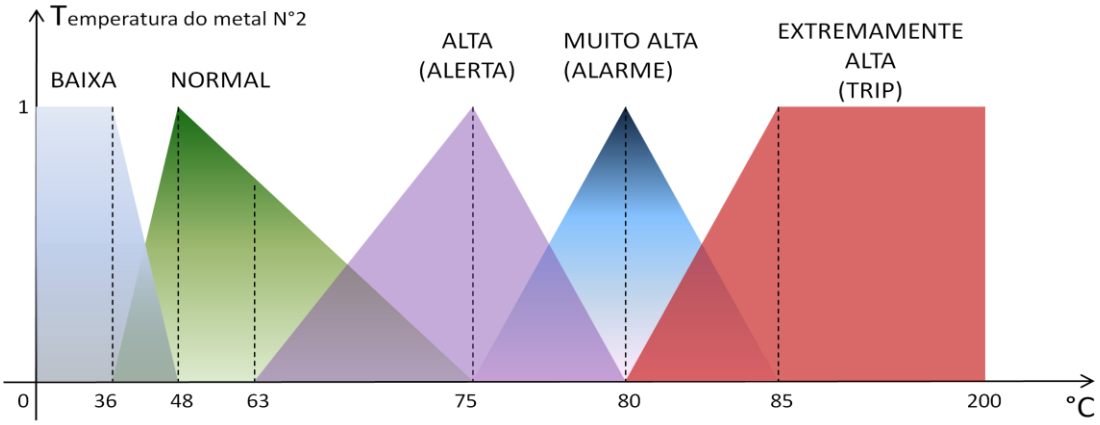

Figura 4.5 – Codificação da faixa de operação da temperatura do metal 2

Para verificar a consistência da funções de pertinência *fuzzy* da Figura 4.5 basta usar os valores da Tabela 4.11. Pode-se comprovar que todos esses valores caem dentro de cada faixa estabelecida com o seu respectivo **indicador de degradação de sinal**.

Por exemplo, os valores mínimo (80.052°C, com um indicador de degradação de sinal igual a Y1) e máximo (84.930°C, com um indicador de degradação de sinal igual a Y2) de ALARME caem na faixa de valores de 80 a 85°C. O conceito de **indicador de degradação de sinal** foi definido no Capítulo 3, e oferece uma noção sobre a evolução de algum defeito ou falha no estado de condição da máquina na faixa de 80 a 85°C.

As funções de pertinência *fuzzy* ou termos *fuzzy* da Figura 4.5, são consistentes com a Tabela 4.4, Tabela 4.5 e Tabela 4.6, bem com o conhecimento heurístico fornecidos pelos operários da usina de Balbina na última visita. Os valores que definem as formas das funções de pertinência *fuzzy* são definidos na Tabela 4.12.

Tabela 4.12 – Valores de configuração para os termos *fuzzy* do metal número 2

VARIÁVEL	VALOR MÍNIMO	VALOR MÉDIO	VALOR MÁXIMO	VALOR DE ALERTA	VALOR DE ALARME	VALOR DE TRIP
Temperatura do metal número 02	36°C	48°C	63°C	75°C	80°C	85°C

4.1.6 construção das funções de pertinência *fuzzy* para a variável temperatura do metal número 3

Usando a Tabela 4.7, a Tabela 4.8 e a Tabela 4.9 é possível codificar de forma consistente a faixa de operação da variável de temperatura do metal número 3, usando funções triangulares (ver Figura A.1) em vez de usar funções gaussianas (ver Figura A.2) para evitar muito consumo de processamento computacional ao executar as regras de produção *fuzzy*. A justificativa para esta escolha é a mesma feita no item 4.1.4.

Na Figura 4.6 é mostrado o conjunto de funções de pertinência *fuzzy* para definir completamente os níveis de operação (BAIXA, NORMAL, ALTA/ALERTA, MUITO ALTA/ALARME e EXTREMAMENTE ALTA/TRIP) desta variável em particular.

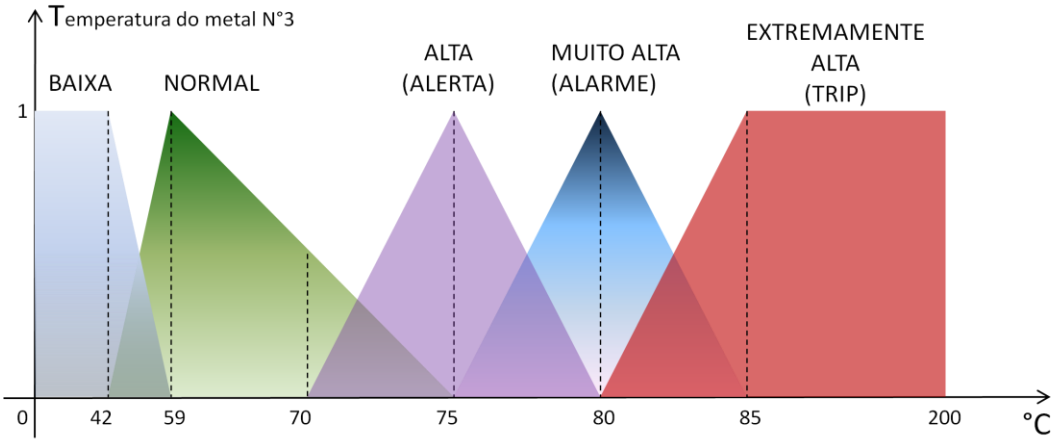

Figura 4.6 – Codificação da faixa de operação da temperatura do metal 3

As funções de pertinência *fuzzy* ou termos *fuzzy* na Figura 4.6, são consistentes com as Tabelas 4.7, 4.8 e 4.9, bem com o conhecimento heurístico fornecido pelos operários da usina de Balbina na última visita. Os valores que definem as formas das funções de pertinência *fuzzy* são definidos na Tabela 4.13.

Tabela 4.13 – Valores de configuração para os termos *fuzzy* do metal número 3

VARIÁVEL	VALOR MÍNIMO	VALOR MÉDIO	VALOR MÁXIMO	VALOR DE ALERTA	VALOR DE ALARME	VALOR DE TRIP
Temperatura do metal número 03	42°C	59°C	70°C	75°C	80°C	85°C

4.2 IMPLEMENTAÇÃO DAS REGRAS DE PRODUÇÃO USANDO FUNÇÕES DE PERTINÊNCIA *FUZZY*

Depois de serem definidas as faixas de operação das variáveis de temperatura do metal 1, 2 e 3, a seguir são implementadas suas regras de produção usando as funções de pertinência *fuzzy* estabelecidas na secção 4.1.

O objetivo de implementar uma base de regras *fuzzy* é elaborar um estudo prévio no Capítulo 5 sobre a viabilidade de uma implementação futura na camada de monitoramento de condição do sistema SIMPREBAL, usando o histórico de dados das variáveis de temperatura do metal 1, 2 e 3 do ano 2009.

4.2.1 Criação da base de regras *fuzzy* no ECLIPE

Assumindo que foi instalado o software ECLIPSE, o *plug-in* JESS para ECLIPSE, e foi exportada a biblioteca *FuzzyJess* no *path* da interface ECLIPE. O primeiro passo, para construir a base de regras *fuzzy* é iniciar o ECLIPE (ver Figura 4.7) onde é mostrada uma janela para configuração da pasta de projetos, no qual será criado o jogo de regras *fuzzy* para as variáveis físicas em estudo.

Figura 4.7 – Inicialização da interface ECLIPSE

Uma vez iniciado o ECLIPSE, é criado um novo projeto para guardar e editar as regras *fuzzy* como é mostrado na Figura 4.8. Depois é criado um arquivo, com extensão (.clp), para a edição das regras *fuzzy* (ver Figura 4.9). Este passo é muito importante, por que se for criado um arquivo sem essa extensão, simplesmente não será reconhecido pelo ECLIPSE, como uma base de regras *fuzzy*.

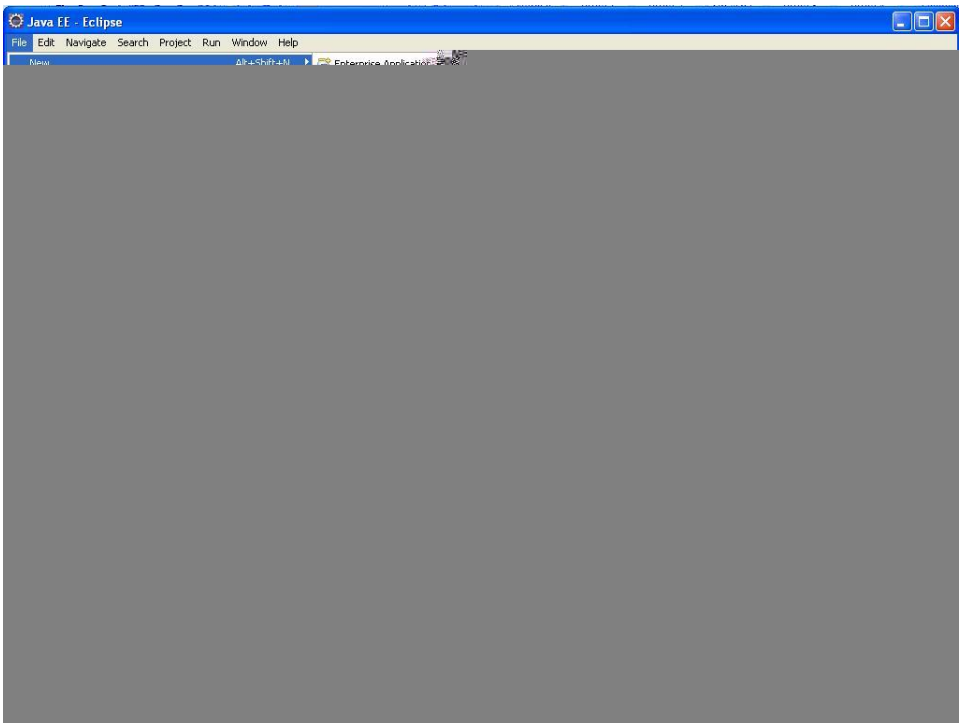

Figura 4.8 – Criação de um novo projeto no ECLIPSE

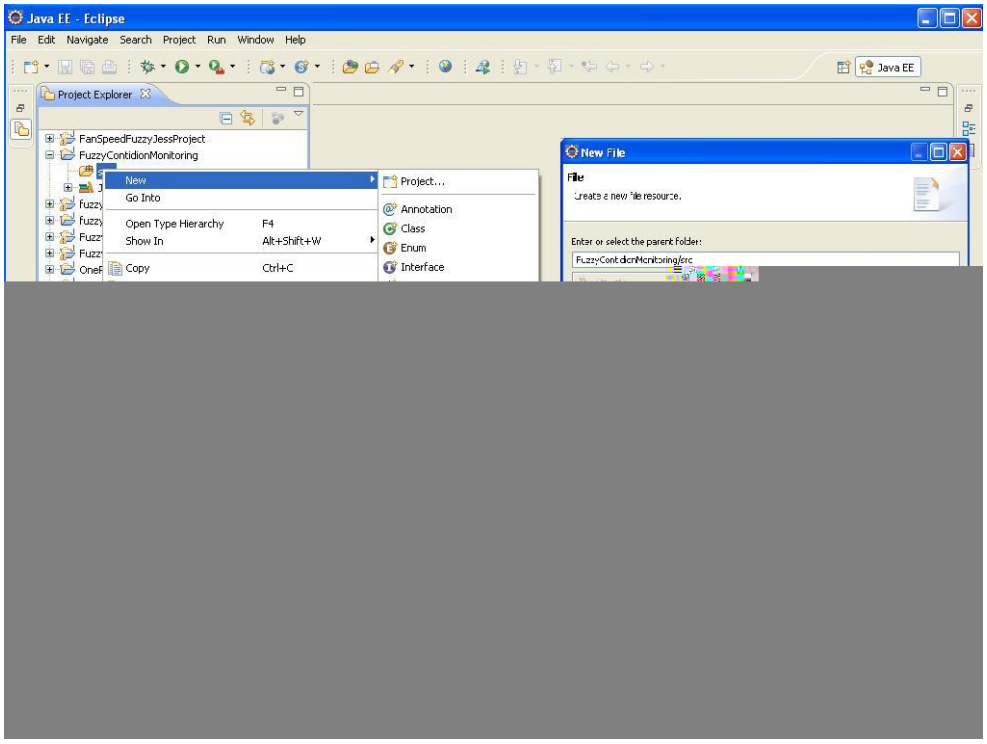

Figura 4.9 – Criação de um arquivo para guardar as regras *fuzzy* no ECLIPE

Depois de ser criado o arquivo, é muito importante importar o jogo de classes, métodos e construtores *fuzzy* (**fuzzyJ110a.jar**) no projeto para construção das regras como é mostrada na Figura 4.10.

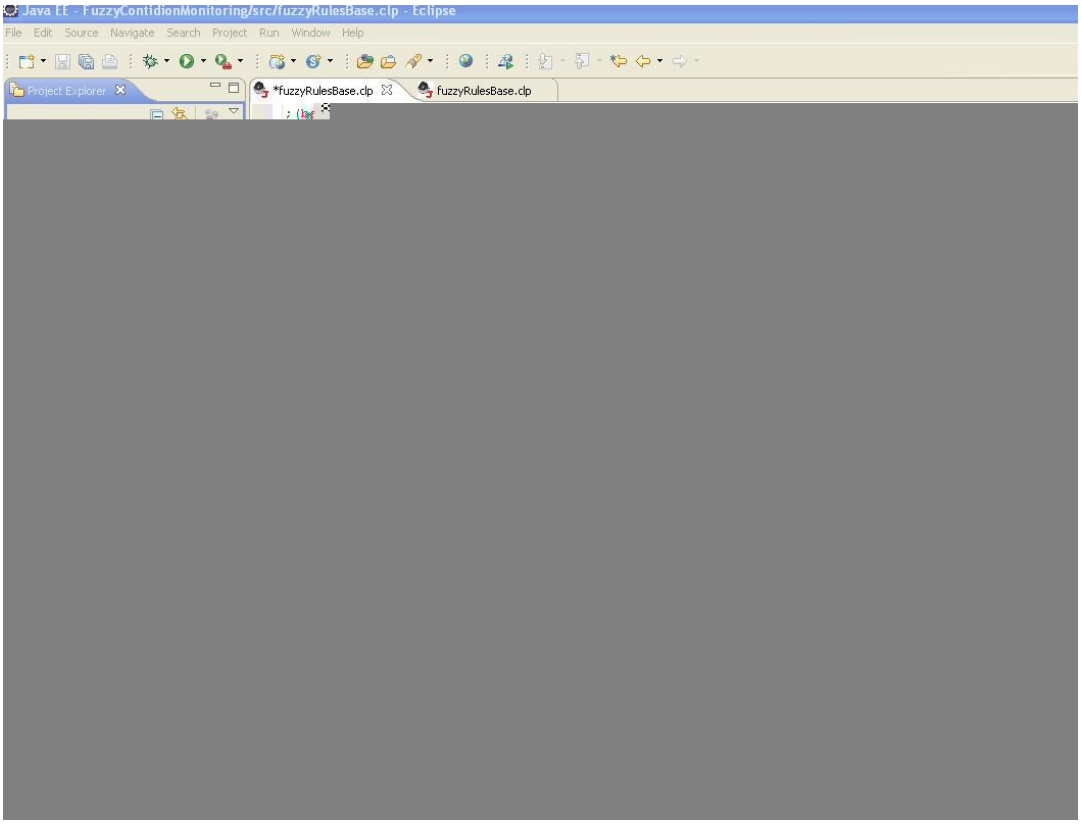

Figura 4.10 – Importação da biblioteca *FuzzyJess* no ECLIPSE

Depois de criar um arquivo vazio para a edição de regras, devem ser primeiro importadas as classes da biblioteca *FuzzyJess* no projeto para poder usá-las (ver Figura 4.11).

```
(import nrc.fuzzy.*)
(import nrc.fuzzy.jess.*)
(load-package nrc.fuzzy.jess.FuzzyFunctions)
```

Figura 4.11 – Importação da biblioteca *FuzzyJess* no arquivo de regras

Depois de ter importado a biblioteca *FuzzyJess*, pode ser criada a base de regras *fuzzy* para cada uma das variáveis em estudo. Mas como a variável de temperatura do metal número 2 foi a única que apresentou eventos, a seguir será implementada sua base de regras *fuzzy*.

Com a base de regras *fuzzy* para a temperatura do metal número 2, um estudo prévio é feito, no Capítulo 5, sobre a sua viabilidade de integração na camada de monitoramento de condição do sistema SIMPREBAL. A base de regras para as variáveis de temperatura do metal 1e 3, serão desenvolvidas no Apêndice C.

4.2.1.1 Criação da base de regras *fuzzy* para a temperatura do metal número 2

Primeiro são criadas um conjunto de variáveis globais representando as funções de pertinência *fuzzy* que serão usadas para determinar os **indicadores de degradação de sinal** desta variável de temperatura para algum instante de tempo (ver Figura 4.12).

```
(defglobal ?*rlf* = (new RightLinearFunction))
(defglobal ?*llf* = (new LeftLinearFunction))
(defglobal ?*baixa_m2* = (new RFuzzySet 36.0 48.0 ?*rlf*))
(defglobal ?*normal_m2* = (new TriangleFuzzySet 36.0 48.0 75.0))
(defglobal ?*alta_m2* = (new TriangleFuzzySet 63.0 75.0 80.0))
(defglobal ?*muitoalta_m2* = (new TriangleFuzzySet 75.0 80.0 85.0))
(defglobal ?*extremamentealta_m2* = (new LFuzzySet 80.0 85.0 ?*llf*))
```

Figura 4.12 – Conjunto de variáveis globais *fuzzy*

Depois duas variáveis globais são criadas para guardar os valores dos **indicadores de degradação de sinal** dessa variável como mostrada na Figura 4.13.

```
(defglobal ?*c1_m2* = 0)
(defglobal ?*c2_m2* = 0)
```

Figura 4.13 – Criação de duas variáveis globais para guardar os valores dos indicadores

Subseqüentemente, é criada uma variável *fuzzy* que represente à variável temperatura do metal 2 no ECLIPE, com as funções de pertinência *fuzzy* estabelecidas na secção 4.1, implementando a fase de codificação (*fuzzification*) como é mostrada na Figura 4.14.

```
(defglobal ?*TempFvar2* = (new FuzzyVariable "TemperaturaMetal2" 0.0 200.0 "Deg C"))
(defrule init2
 "An initialization rule that adds the terms to the FuzzyVariables"
;;(declare (salience 100))
 =>
;; the nrc FuzzyJess functions are loaded
;(import nrc.fuzzy.*)
;(load-package nrc.fuzzy.jess.FuzzyFunctions)
(bind ?rlf (new RightLinearFunction))
(bind ?llf (new LeftLinearFunction))
;; terms for the metal2 temperature Fuzzy Variable
(?*TempFvar2* addTerm "baixa" (new RFuzzySet 36.0 48.0 ?rlf))
(?*TempFvar2* addTerm "normal" (new TriangleFuzzySet 36.0 48.0 75.0))
(?*TempFvar2* addTerm "alta" (new TriangleFuzzySet 63.0 75.0 80.0))
(?*TempFvar2* addTerm "muitoalta" (new TriangleFuzzySet 75.0 80.0 85.0))
(?*TempFvar2* addTerm "extremamentealta" (new LFuzzySet 80.0 85.0 ?llf))
;; assert the first the fuzzy input -- temperature at 78.3497°C
(bind ?t2 78.3497)
(assert (crispTemp ?t2))
(assert (tempMetal2 (new FuzzyValue ?*TempFvar2* (new TriangleFuzzySet ?t2 ?t2 ?t2))))
)
```

Figura 4.14 – Criação de uma regra de inicialização para o metal número 2

Antes de mostrar a estrutura da regra *fuzzy* para o metal número 2, primeiro será mostrada a estrutura da regra original do sistema SIMPREBAL para o metal número 2 (ver Figura 4.15).

```
(defrule UGH3-MGG-MonitoracaoCondicao-7
    (Tag (label == "g3.mgg.t.oleo.mguia.sup2" && value != "" && value < "75"))
    =>
    (assert (condition-NORMAL ?label))
)

(defrule UGH3-MGG-MonitoracaoCondicao-10
    (Tag (label == "g3.mgg.t.oleo.mguia.sup2" && value != "" && value > "75" && value < "80"))
    =>
    (assert (condition-ALERTA ?label))
)

(defrule UGH3-MGG-MonitoracaoCondicao-8
    (Tag (label == "g3.mgg.t.oleo.mguia.sup2" && value != "" && value > "80" && value < "85"))
    =>
    (assert (condition-ALARME ?label))
)

(defrule UGH3-MGG-MonitoracaoCondicao-14
    (Tag (label == "g3.mgg.t.oleo.mguia.sup2" && value != "" && value > "85"))
    =>
    (assert (condition-TRIP ?label))
)
```

Figura 4.15 – Estrutura da base de regras original do SIMPREBAL na camada de monitoramento de condição

Para terminar a análise sobre a criação da base de regras *fuzzy*, uma análise quantitativa e qualitativa, entre a base de regras *fuzzy* e a base de regras originais do SIMPREBAL, é discutida a seguir:

- ✓ Os valores de leitura advindos dos sensores são codificados usando funções triangulares com base zero ou funções *singleton* (ver Figura 4.14) e é feito o casamento dessa leitura codificada com a função de pertinência que representa a faixa de operação através da função *fuzzy-match*. Para especificação desta função de usuário e outras ver o Apêndice B.

- ✓ Como se analisou no Capítulo 3, as faixas de monitoramento de condição da Figura 4.15 não são flexíveis e isso pode gerar muitos eventos de condição falsos. Por exemplo, para a temperatura de 79.356°C o SIMPREBAL deveria originar um ALARME, mas com sua base de regras gera um evento ALERTA por que, 79.356°C é menor do que 78°C.

- ✓ Para solucionar, o problema anteriormente exposto, se propõe uma estrutura de regras mais flexíveis usando a lógica *fuzzy* (ver Figuras 4.16, 4.17, 4.18 e 4.19), propondo computar essa estrutura de regras com palavras em vez de faixas muito rigorosas para incrementar a confiabilidade de monitoramento de condição de máquinas.

- ✓ A base de regras *fuzzy* proposta computa, por exemplo, a palavra "NORMAL" em vez de uma faixa rigorosa, como de 0 a 75°C. Com isto consegue-se solucionar o problema de atribuir um estado correto de condição de máquinas para valores de leitura do sensor muito próximo aos limites estabelecidos pelo equipe do SIMPREBAL e os funcionários de Balbina.

- ✓ Com relação aos indicadores de degradação de sinal, foram concebidos no Capítulo 3 com o intuito de seguir a pista a degradação de sinal do sensor com o intuito de incrementar a confiabilidade dos estados de monitoramento de máquinas. Alguns gráficos demonstrando, o anteriormente exposto, será visto no Capítulo 5.

- ✓ Segundo a estrutura de regras *fuzzy* concebidas, o cálculo dos indicadores de degradação de sinal é feito pela função de usuário *getMembership*, que obtém um nível de pertinência (valores entre 0 a 1) a partir de cada valor de litura do sensor em algum instante de tempo.

- ✓ Para a implementação da base de regras *fuzzy* proposta nesta dissertação, foram implementadas funções triangulares e trapezoidais *fuzzy* (para as faixas de operação das variáveis físicas) e funções *singletons fuzzy* para codificar as leituras dos sensores.

- ✓ Segundo a análise feito neste capítulo, pode ser integrada a base de regras *fuzzy* na camada de monitoramento de condição do sistema SIMPREBAL, pois para criar a base de regras originais do sistema, foram também usadas ferramentas computacionais desenvolvidas em Java e JESS. A base de regras *fuzzy*, além de usar as mesmas ferramentas, usou a biblioteca *FuzzyJess*, que é uma extensão de JESS, para poder usar conceitos de lógica *fuzzy*, o qual também é compatível.

- ✓ Algumas considerações extras, com relação às classes: **JessProxy.class** e **ServerMain.class,** implementadas no lado do servidor do sistema SIMPREBAL, devem ser levadas em conta para integrar a base de regras *fuzzy* na camada de monitoramento de condição. Essas classes deverão ser modificadas, o qual não foi feito completamente neste trabalho de dissertação dada à complexidade do código Java no lado do servidor, mas algumas sugestões são dadas no Capítulo 6.

```
(defrule conditionmonitoringO2_m2
    (tempMetal2 ?t&:(fuzzy-match ?t "normal"))
=>
 (bind ?*c1_m2* (?*normal_m2* getMembership ?*t2*))
 (bind ?*c2_m2* (?*alerta_m2* getMembership ?*t2*))

 (if(> ?*c2_m2* 0.0)
   then
     (if(> ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de " ?*t2* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(= ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de " ?*t2* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(< ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de: " ?*t2* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal de: " ?*c2_m2* " [0-1]" crlf)
     )
   )
 )
```

Figura 4.16 – Base de regras *fuzzy* para a faixa NORMAL

```
(defrule conditionmonitoringO3_m2
    (tempMetal2 ?t&:(fuzzy-match ?t "alerta"))
=>
 (bind ?*c1_m2* (?*alerta_m2* getMembership ?*t2*))
 (bind ?*c2_m2* (?*alarme_m2* getMembership ?*t2*))

  (if(> ?*c2_m2* 0.0)
   then
     (if(> ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de " ?*t2* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(= ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de " ?*t2* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(< ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal O2 de: " ?*t2* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal de: " ?*c2_m2* " [0-1]" crlf)
     )
   )
 )
```

Figura 4.17 – Base de regras *fuzzy* para a faixa ALERTA

```
(defrule conditionmonitoring04_m2
    (tempMetal2 ?t&:(fuzzy-match ?t "alarme"))
=>
 (bind ?*c1_m2* (?*alarme_m2* getMembership ?*t2*))
 (bind ?*c2_m2* (?*trip_m2* getMembership ?*t2*))

 (if(> ?*c2_m2* 0.0)
   then
     (if(> ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal 02 de " ?*t2* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(= ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal 02 de " ?*t2* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal de: " ?*c1_m2* " [0-1]" crlf)
     )
     (if(< ?*c1_m2* ?*c2_m2*)
     then
     (printout t "Para uma temperatura do metal 02 de: " ?*t2* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal de: " ?*c2_m2* " [0-1]" crlf)
     )
   )
 )
```

Figura 4.18 – Base de regras *fuzzy* para a faixa ALARME

```
(defrule conditionmonitoring05_m2
    (tempMetal2 ?t&:(fuzzy-match ?t "trip"))
=>
 (bind ?*c1_m2* (?*alarme_m2* getMembership ?*t2*))
 (bind ?*c2_m2* (?*trip_m2* getMembership ?*t2*))

 (if(= ?*c1_m2* 0.0)
   then

     (printout t "Para uma temperatura do metal 02 de: " ?*t2* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal de: " ?*c2_m2* " [0-1]" crlf)
  )
)
(reset)
(run)
```

Figura 4.19 – Base de regras *fuzzy* para a faixa TRIP

Depois de ter criado a base de regras *fuzzy* para a variável temperatura do metal número 2, serão simuladas, para avaliar o desempenho, com entradas que são leituras de temperaturas advindas do sensor de temperatura do metal 2.

No Capítulo 5, é usado o histórico de leituras da variável de temperatura do metal número 2 do ano 2009, para validar o desempenho das regras *fuzzy* criadas aqui.

Mas para fins de ilustrativos será simulada a base de regras *fuzzy* com algumas temperaturas para dispará-las e verificar suas saídas. Mas antes disso, o ECLIPE deve ser configurado como é mostrado na Figura 4.20.

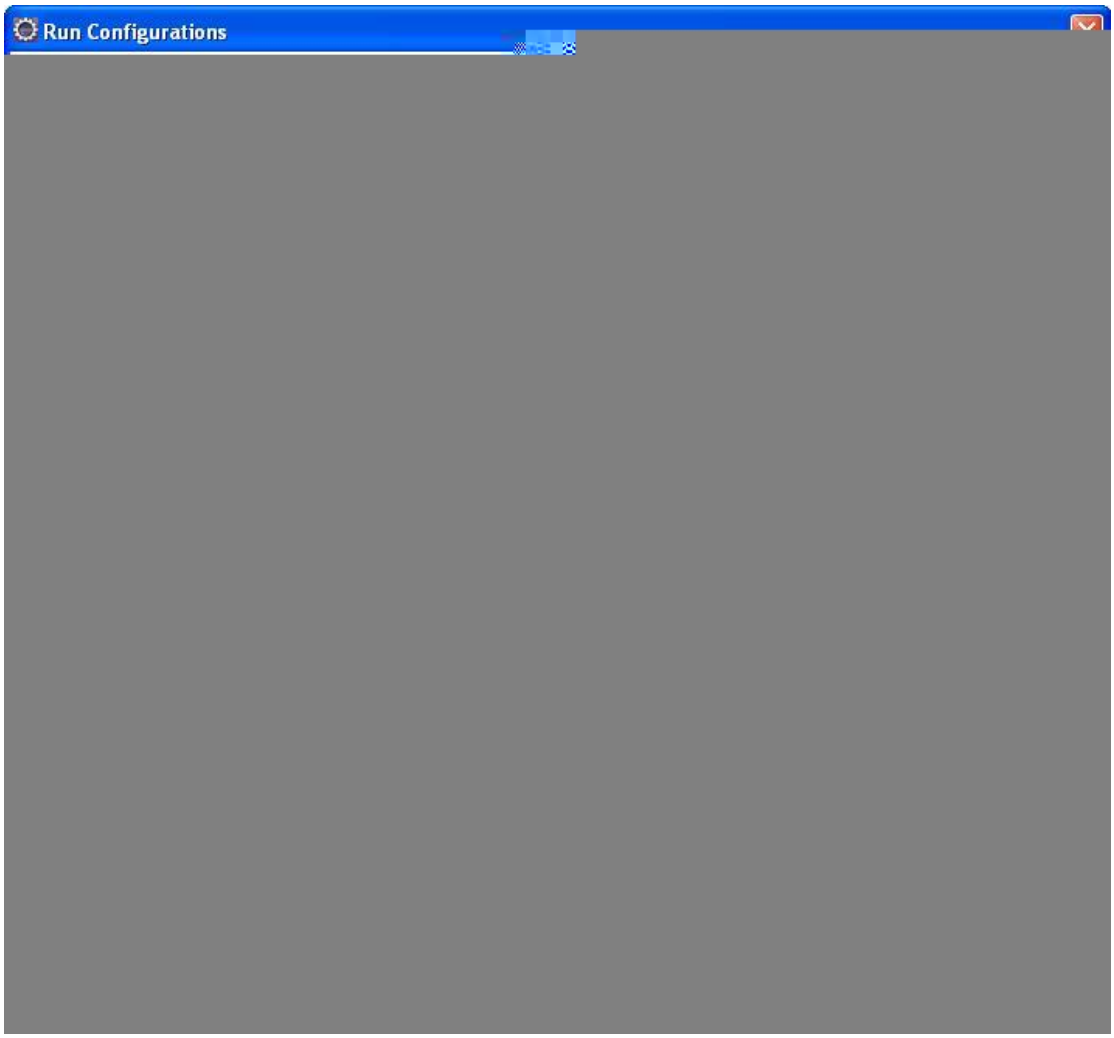

Figura 4.20 – Configuração do Eclipse para simular as regras fuzzy

Além de configurar corretamente, a pasta e o arquivo de simulação da Figura 4.20, é necessário configurar a classe principal do JESS com **FuzzyMain**. Agora será simulada a base de regras para as temperaturas de: 64°C (ver figura 4.21), 74°C (ver Figura 4.22), 78°C (ver Figura 4.23) e 83°C (ver Figura 4.24).

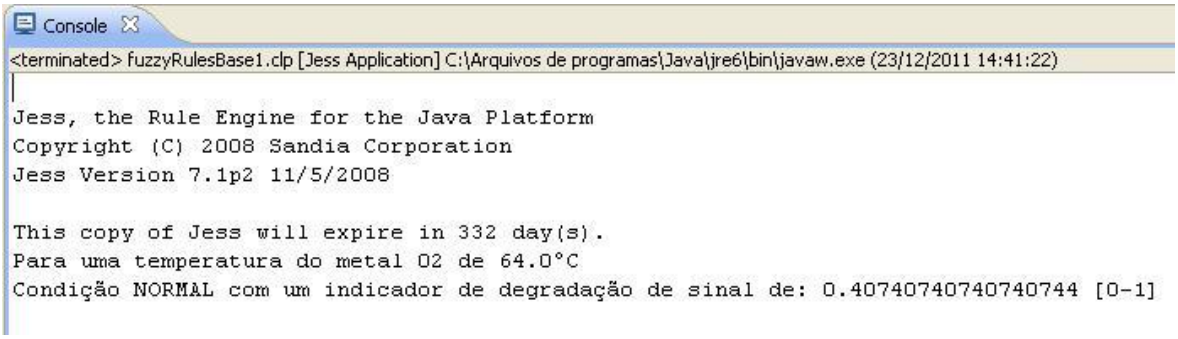

Figura 4.21 – Resultado da simulação para uma temperatura de 64°C

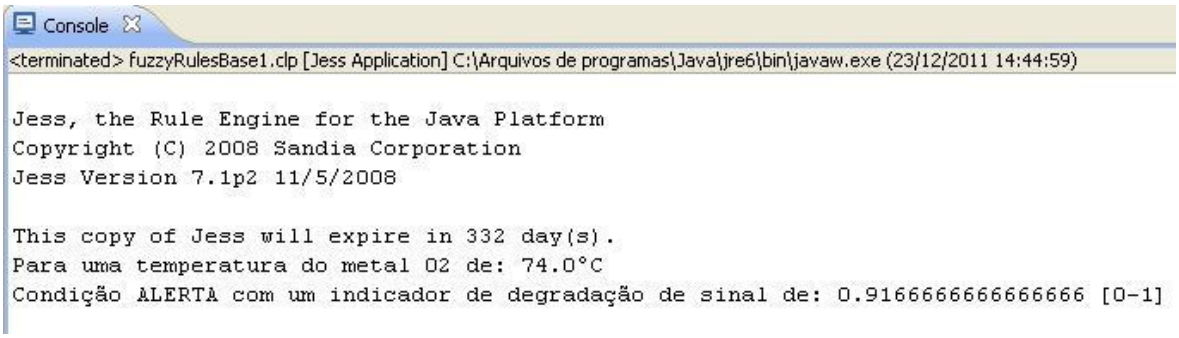

Figura 4.22 – Resultado da simulação para uma temperatura de 74°C

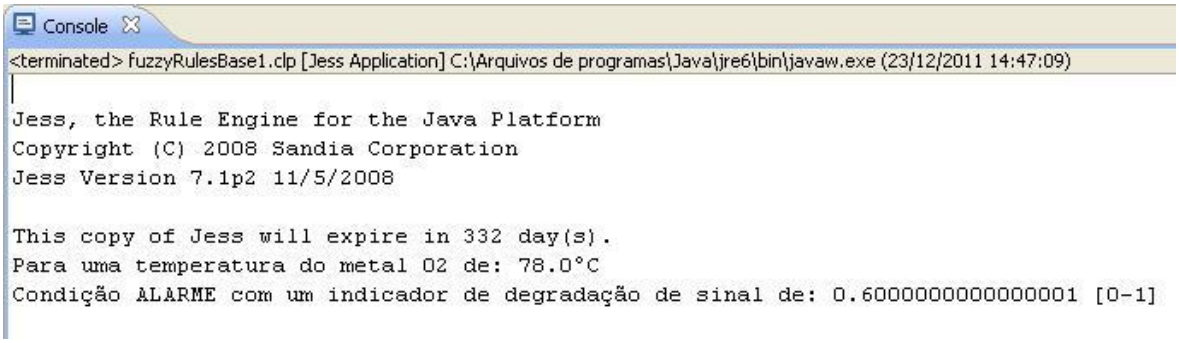

Figura 4.23 – Resultado da simulação para uma temperatura de 78°C

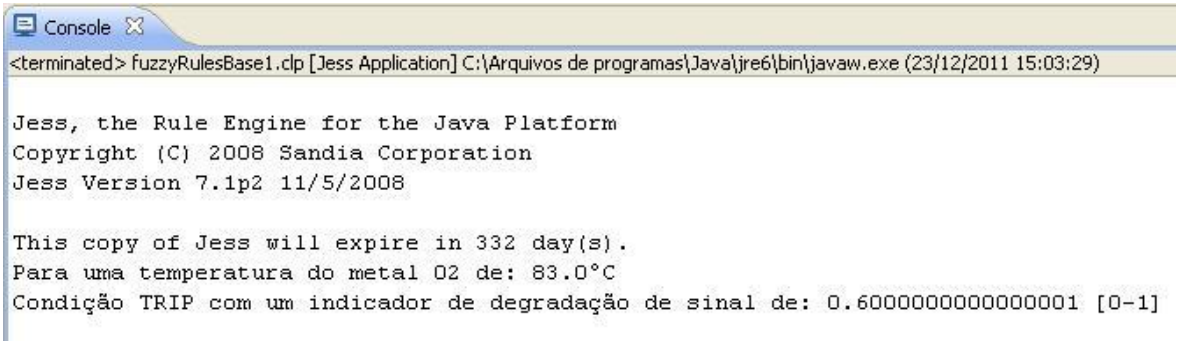

Figura 4.24 – Resultado da simulação para uma temperatura de 83°C

Como pode ser visto nas figuras anteriores, para uma dada temperatura é calculado seu indicador de degradação de sinal, como uma medida de mensurar a evolução de uma falha ou defeito nesse sinal.

No Capítulo 5 serão esboçados estes **indicadores de degradação de sinal** para o histórico da variável do metal 2 nos meses de setembro, outubro e novembro do ano 2009 para mensurar graficamente a evolução dos defeitos nesses sinais.

5 ESTUDO DE CASO: SISTEMA DE MANCAIS

"Nenhum homem realmente produtivo pensa como se estivesse escrevendo uma dissertação"

Albert Einstein

Neste penúltimo capítulo é apresentado um estudo preliminar sobre a viabilidade de uma futura integração, da base de regras *fuzzy* criada no Capítulo 4, na camada de monitoramento de condição do sistema SIMPREBAL. A análise será focada no sistema de mancais (subsistema mancal guia do gerado – SMGG) da UGH 02 pertencente à usina hidrelétrica de Balbina.

O objetivo principal deste capítulo é estabelecer um estudo comparativo entre a base regras original do SIMPREBAL e nova base de regras *fuzzy* proposta para sua camada de monitoração da condição, com o intuito de avaliar sua viabilidade de integração e confiabilidade de detecção do eventos de ALERTAS, ALARMES ou TRIPS no SMGG usando o histórico de leituras do metal número 2 nos meses de setembro, outubro e novembro do ano 2009.

5.1 APRESENTAÇÃO DO CENÁRIO

Nesta seção será apresentada uma descrição geral da usina. Em seguida será descrito o sistema de mancais incluindo o seu subsistema (SMGG), a instrumentação e equipamentos associados. O objetivo desta seção é fornecer uma visão do ambiente em que está inserida a atual pesquisa de forma a elucidar melhor o estudo de caso.

5.1.1 Descrição da usina

Segundo a NBR-5460 (1992) uma usina hidrelétrica pode ser definida como: "Complexo arquitetônico elétrico-mecânico, no qual a geração de energia elétrica é realizada através do aproveitamento de recursos hídricos. A usina hidrelétrica de Balbina (ver Figura 5.1)

tem por finalidade produzir energia elétrica através do aproveitamento da energia potencial hidráulica da água do rio Uatumã (afluente da margem norte do Rio Amazonas) e possui características construtivas como: barragem, reservatório, estruturas de concreto (tomada d'água e vertedouro), casa de força, onde estão instalados os principais equipamentos para o controle de qualidade e operação dos seus três sistemas principais: sistema de mancais (SM), sistema de turbina (ST) e sistema do gerador (SG) como mostrado na Figura 5.2.

Figura 5.1 – Vista frontal da usina hidrelétrica de Balbina

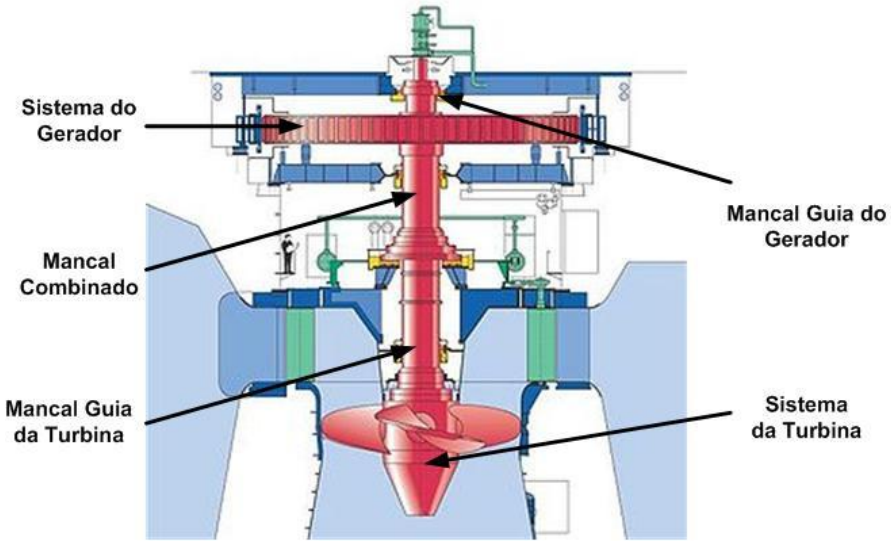

Figura 5.2 – Sistemas que compõem cada unidade geradora hidráulica

A usina possui uma capacidade máxima de geração de 250MW (5 geradores de 50MW cada). Os geradores são de baixa rotação (aproximadamente 105,88rpm), capacidade nominal de 55,5MVA e tensão nominal de 13,8KV e são numeradas de 1 a 5, como observado na Figura 5.3, sendo que o gerador 1 está mais próximo da margem do rio e o gerador 5 está mais próximo do leito. As unidades geradoras 1, 3 e 5 são auto-alimentadas

(parte da energia gerada por elas é utilizada para fornecer a corrente de excitação do estator e alimentar as bombas e demais componentes elétricos da unidade), os geradores 2 e 4 possuem alimentação externa. Para o funcionamento correto da usina um dos geradores (1, 3 ou 5) deve estar sempre ligado.

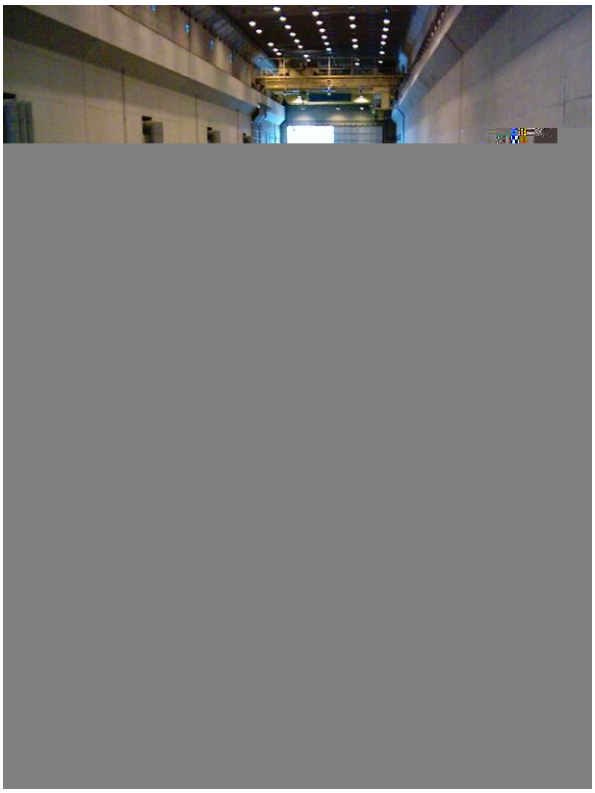

Figura 5.3 – Vista superior das 5 unidades geradoras hidráulicas

As turbinas hidrelétricas existentes em Balbina são do tipo Kaplan de eixo vertical (ver Figura 5.2) e são especificadas para uma queda líquida de aproximadamente 21,85m. Os equipamentos na usina de Balbina estão divididos em sistema (SM, ST e SG).

Quanto à instrumentação até o ano 2009, cada unidade geradora era monitorada por 3 DFI, cada DFI possui 4 canais de comunicação (sendo um de reserva) nas quais estão conectadas 8 transmissores que por sua vez estão ligados 8 sensores por canal.

Tanto as DFI quanto os transmissores são instrumentos fabricados pela empresa Smar, possuem capacidade de processamento local e comunicam-se por meio de uma rede *Foundation Filedbus.* Os sensores são basicamente de temperatura, pressão e densidade. Uma distribuição genérica para cada unidade é mostrada na Figura 5.4.

Na Figura 5.4 o servidor OPC armazena os itens (valores de leitura e outros parâmetros) da instrumentação. Pode-se obter informações dos instrumentos através da conexão com o servidor OPC em tempo real ou através do banco de dados. No banco de dados é armazenado o histórico de anomalias, tomadas de decisão, leituras das variáveis do processo, cadastro de usuários e dados dos equipamentos.

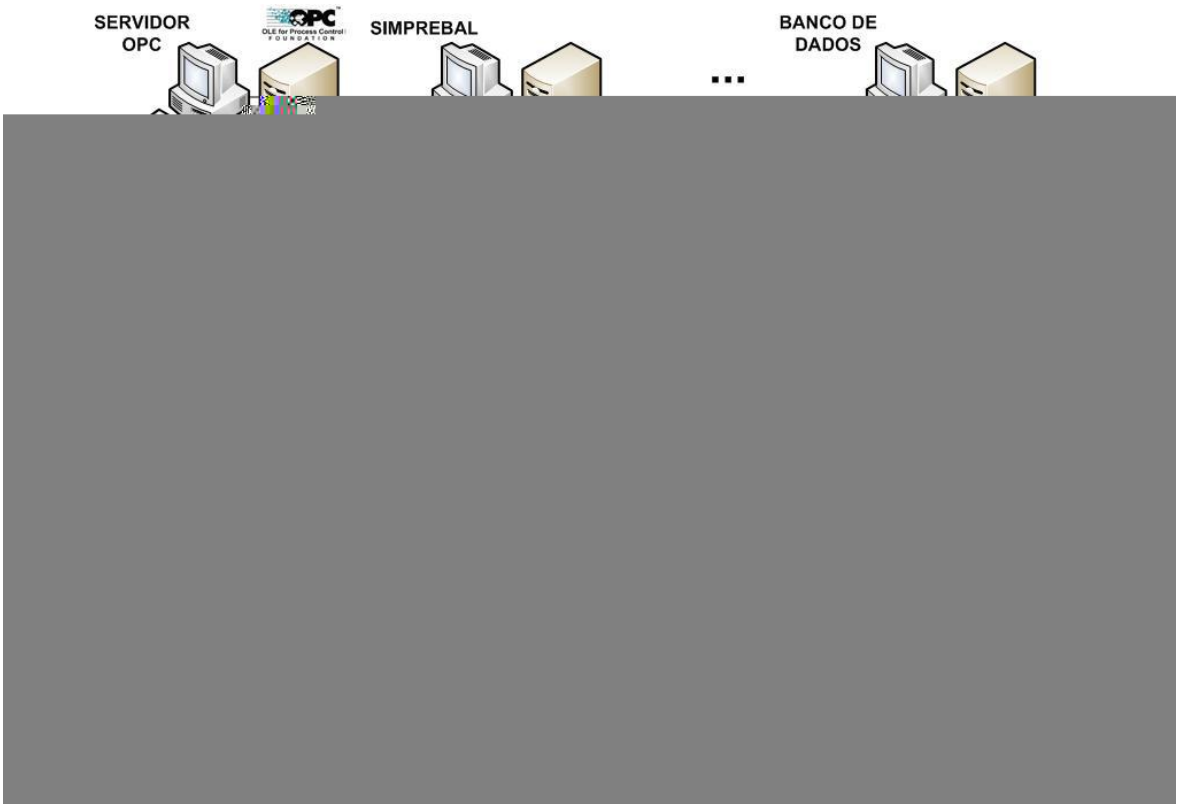

Figura 5.4 – Configuração de instrumentos e equipamentos para uma unidade

Na seguinte subseção será apresentada uma breve descrição das funções de cada equipamento e sua instrumentação para o subsistema em estudo, uma descrição detalhada pode ser encontrada em Souza (2008), Amaya (2008) e Tonaco (2008).

5.1.2 Descrição funcional do SMGG

Para o estudo de caso desta dissertação é conveniente especificar o funcionamento básico dos subsistemas pertencentes ao sistema de mancais e especificar os equipamentos e a instrumentação que compõem cada subsistema em estudo.

Basicamente a função dos mancais é diminuir o atrito e, portanto, aumentar o rendimento do sistema mecânico, entre partes que se movem entre si, também são responsáveis por transferir os esforços radiais e axiais do eixo da turbina ao concreto evitando assim uma vibração excessiva do mesmo e o desgaste prematuro de todo o grupo.

Os equipamentos que conformam cada mancal em cada unidade geradora na usina de Balbina são basicamente cubas e sapatas. A cuba é responsável pelo armazenamento de óleo que lubrifica as partes ativas do mancal, por outro lado, as sapatas são estruturas metálicas responsáveis pela formação de um filme de óleo, por efeito hidrodinâmico, entre o metal patente e o eixo da turbina durante o funcionamento da unidade geradora como é mostrada na Figura 5.5.

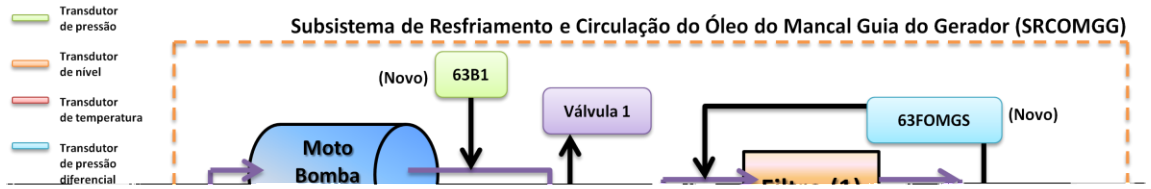

Figura 5.5 – Instrumentação e equipamentos dos subsistemas MGG e RCOMGG

O mancal guia da turbina é refrigerado pela própria água que passa nas pás do distribuidor, por outro lado, tanto o subsistema do mancal combinado e o SMGG são refrigerados por subsistemas auxiliares. Na Figura 5.5 é mostrado o SRCOMGG (subsistema auxiliar) para SMGG.

Cada subsistema em cada unidade geradora está constituído por motobombas (uma em reserva) para a circulação de óleo, por filtros (uma em reserva) e trocadores de calor. Até o ano 2009 os mancais eram monitorados basicamente por sensores de temperatura do óleo e do metal das sapatas e só o sistema auxiliar do mancal combinado era monitorado por um medidor de densidade. No começo do ano 2010 foi projetada na usina de Balbina a instalação de nova instrumentação Smar para incrementar a confiabilidade dos SM, ST e SG, mas o servidor SIMPREBAL já não estava mais em funcionamento.

Pelo anteriormente exposto, resolveu-se usar o histórico de leituras da variável temperatura do metal número 2 para este estudo prévio de avaliação das regras *fuzzy*, pois foi a única variável que apresentou eventos no SMGG, nos meses de setembro, outubro e novembro do ano 2009. A seguir é desenvolvida a análise de validação das regras *fuzzy*.

5.2 ANÁLISE COMPARATIVO ENTRE A BASE DE REGRAS ORIGINAL E A BASE DE REGRAS *FUZZY*

Para validar a futura integração e confiabilidade da nova estrutura de regras *fuzzy* proposta para a camada de monitoramento de condição, foi analisado o histórico de leituras do SIMPREBAL para a variável de temperatura do metal 02 nos meses de setembro, outubro e novembro do ano 2009, o qual gerou um conjunto total de eventos ou anomalias na UGH02 (ALERTAS, ALARMES e TRIPS) como é mostrado na Figura 5.6.

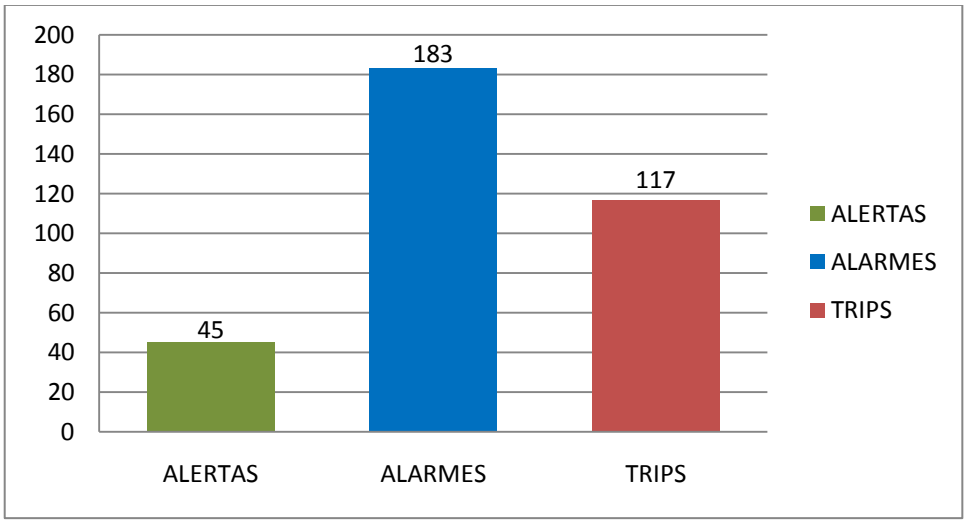

Figura 5.6 – Número de eventos gerados pela temperatura do metal 02 nos meses de setembro, outubro e novembro do ano 2009

Para avaliar a confiabilidade do número de eventos detectados pelo sistema SIMPREBAL como foi mostrado na Figura 5.6, é necessário saber quais foram o conjunto de valores de temperatura do metal 2, na qual o sistema os considerou como eventos de ALERTAS, ALARMES ou TRIPS. A Tabela 5.1 apresenta uma pequena amostra para cada tipo de evento feito com o histórico de dados do SIMPREBAL no 2009 para essa variável.

Tabela 5.1 – Conjunto de valores das leituras do metal número 2 que originaram eventos

CONJUNTO DE VALORES QUE ORIGINARAM ALERTAS (°C)	CONJUNTO DE VALORES QUE ORIGINARAM ALARMES (°C)	CONJUNTO DE VALORES QUE ORIGINARAM TRIPS (°C)
77,96954	80,09567	86,46695
78,01779	80,21246	86,644745
78,06354	80,33688	86,75253
77,94235	80,44943	86,83093
77,8309	80,56961	86,92966
77,762634	80,66385	87,019226
...	...	...

Rapidamente podem ser identificados na Tabela 5.1 quais foram os eventos verdadeiros e quais foram os falsos. Por exemplo, ao executar a base de regras *fuzzy* para uma entrada de temperatura de 78,06354°C (ver Tabela 5.1), é fácil perceber que esse valor pertence mais (por o seu indicador de degradação, ver Figura 5.7) ao termo *fuzzy* ALARME do que ao termo *fuzzy* ALERTA, portanto o sistema SIMPREBAL deveria detectá-lo como um evento ALARME e não ALERTA, mas o sistema SIMPREBAL com sua base de regras original o detectou como ALERTA gerando um evento falso. A mesma análise poderia ser feita para todas as temperaturas. Este tipo de análise foi feita conjuntamente com os operadores da usina para validar sua efetividade.

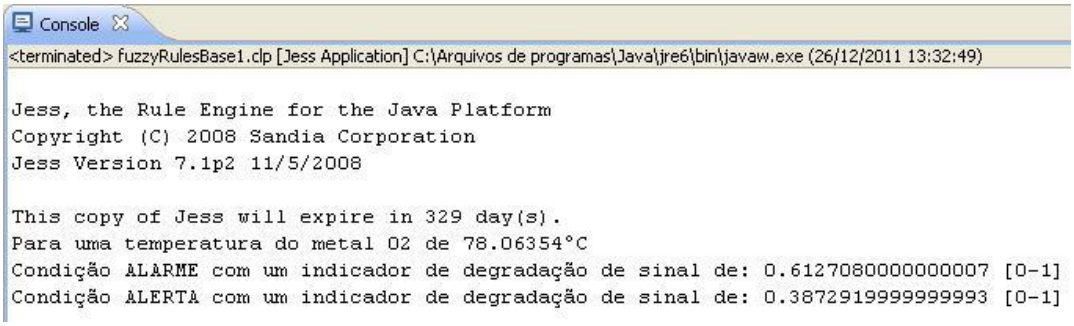

Figura 5.7 – Comparação entre indicadores de degradação: ALARME e TRIP

No Capítulo 4 foi descrita a implementação e discussão da base de regras *fuzzy* proposta, aplicando o mesmo tipo de análise exposto no parágrafo anterior. Neste capítulo será analisada a seguir, a veracidade dos eventos (conjunto de ALERTAS, ALARMES e TRIPS) gerados pela base de regras original do SIMPREBAL com o intuito de discernir quantos eventos foram estabelecidos corretamente.

A análise de confiabilidade do "conjunto de valores de temperatura que originou o evento ALERTA no SIMPREBAL" da Tabela 5.1 é apresentada na Tabela 5.2. Os indicadores de degradação tanto para o evento ALERTA quanto ALARME, foram obtidos executando a base de regras *fuzzy* para cada leitura de temperatura, na qual se originou ALERTA no sistema SIMPREBAL (ver Capítulo 4).

Tabela 5.2 – Conjunto de indicadores de degradação de sinal para os eventos ALERTA e ALARME do metal número 2

CONJUNTO DE LEITURAS QUE OGINARAM ALERTAS NO SIMPREBAL (°C)	INDICADOR DE DEGRADAÇÃO DO EVENTO ALERTA	INDICADOR DE DEGRADAÇÃO DO EVENTO ALARME
77,96954	0,406092	0,593908
78,01779	0,396442	0,603558
78,06354	0,387292	0,612708
77,94235	0,41153	0,58847
77,8309	0,43382	0,56618
77,762634	0,4474732	0,5525268
77,812225	0,437555	0,562445
77,8869	0,42262	0,57738
77,96185	0,40763	0,59237
...	...	...

Da Tabela 5.2 pode-se elaborar um gráfico de tendências dos indicadores de degradação (ver Figura 5.8). Como se pode ver na Tabela 5.2 e a Figura 5.8 ao incrementar-se a temperatura do metal número 2 o indicador de degradação do evento ALERTA (linha azul) está decrescendo enquanto o indicador de degradação do evento ALARME (linha vermelha) está crescendo. Depois de algum instante de tempo tem-se um indicador de degradação de sinal do evento ALARME maior do que o indicador proporcionado por ALERTA.

A base de regras original do SIMPREBAL carece da ideia anteriormente exposta, portanto os eventos que deveriam ser considerados, depois de um instante de tempo como ALARMES, o SIMPREBAL os considerou como ALERTA criando eventos falsos na base de dados de anomalias.

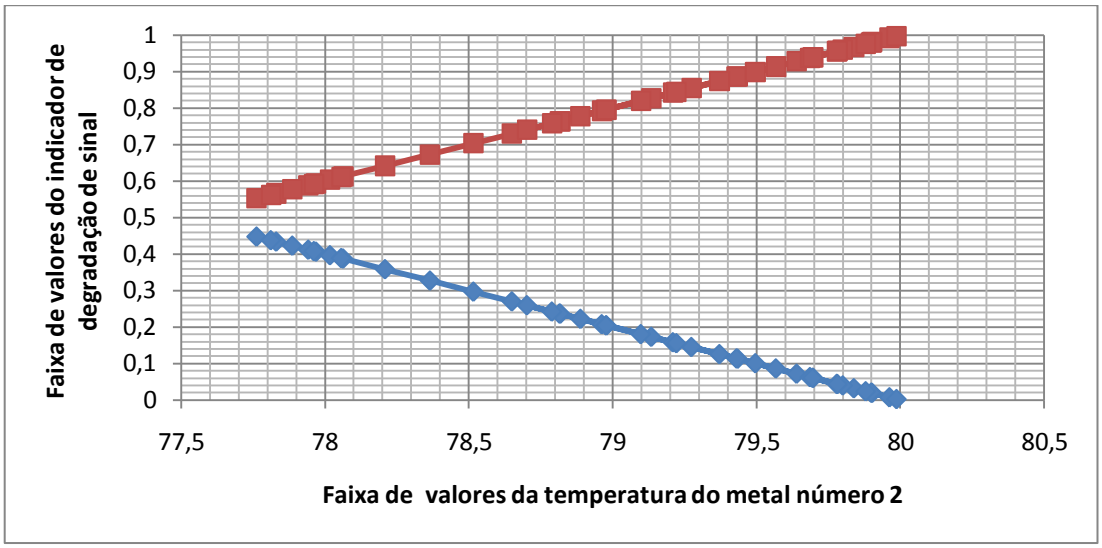

Figura 5.8 – Gráfico de tendência dos indicadores de degradação de sinal com relação aos valores do sensor para os ventos ALERTA e ALARME

Para fins ilustrativos, pode-se gerar outro gráfico de tendências dos indicadores de degradação de sinal tanto para o evento ALERTA (ver Figura 5.9) quanto para o evento ALARME (ver figura 5.10) em um intervalo de tempo a partir da Figura 5.8.

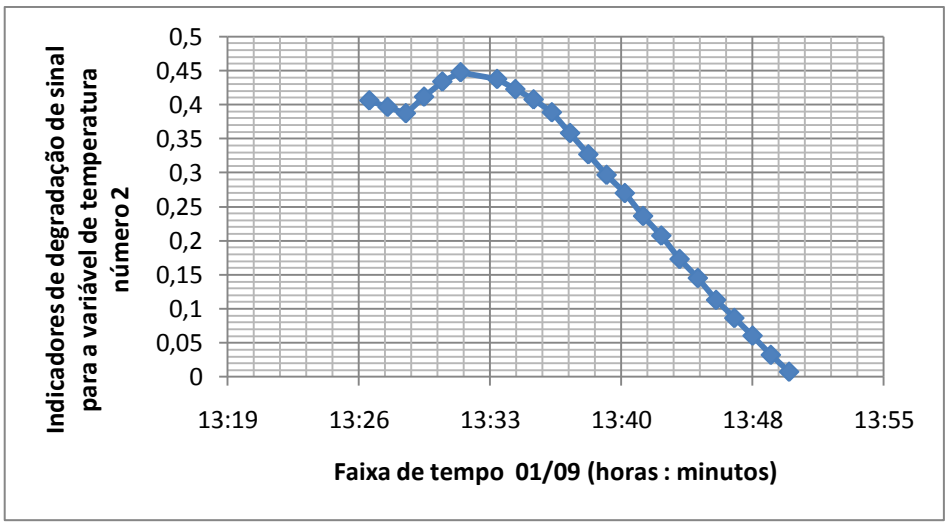

Figura 5.9 – Gráfico de tendência do indicador de degradação do sinal para o evento ALERTA do metal 2

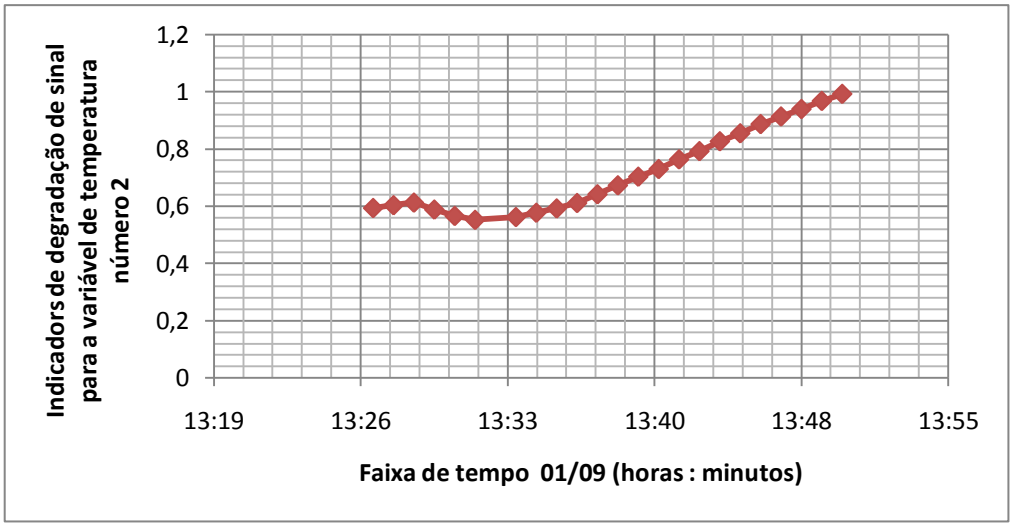

Figura 5.10 – Gráfico de tendência do indicador de degradação do sinal para o evento ALARME do metal 2 (crescente)

As Figuras 5.8, 5.9, 5.10, 5.11, 5.12. 5.13, 5.14 e 5.15 foram elaboradas para contribuir com o monitoramento de condição do sinal da variável física em estudo (variável de temperatura do metal número 2).

Uma vez implementada a base de regras *fuzzy* no sistema SIMPREBAL e rodando no servidor da usina hidrelétrica de Balbina, o operador de processo observando as Figuras 5.9, 5.10, 5.12, 5.13 e 5.15 (que seriam gerados em tempo real na janela principal do sistema SIMPREBAL) teria uma melhor noção de como está evoluindo o defeito nesse valor de sinal através do seu indicador de degradação e por tanto tomar uma ação preventiva adequada.

Analogamente ao anteriormente exposto, a análise de confiabilidade do "conjunto de valores de temperatura que originou o evento ALARME no SIMPREBAL" da Tabela 5.1 é apresentada na Tabela 5.3.

Tabela 5.3 – Conjunto de indicadores de degradação de sinal para os eventos ALARME e TRIP do metal número 2

CONJUNTO DE LEITURAS QUE ORIGINARAM ALARMES NO SIMPREBAL (°C)	INDICADOR DE DEGRADAÇÃO DO EVENTO ALARME	INDICADOR DE DEGRADAÇÃO DO EVENTO TRIP
80,09567	0,980866	0,019134
80,21246	0,957508	0,042492

80,33688	0,932624	0,067376
80,44943	0,910114	0,089886
80,56961	0,886078	0,113922
80,66385	0,86723	0,13277
....	...	...

Da Tabela 5.2 pode-se elaborar um gráfico de tendências dos indicadores de degradação de sinal para os eventos ALARME e TRIP (ver Figura 5.11). Como se pode ver na Figura 5.11 ao incrementar-se a temperatura do metal número 2 o indicador de degradação do evento ALARME (linha vermelha) está decrescendo enquanto o indicador de degradação do evento TRIP (linha azul) está crescendo.

Depois de algum instante de tempo tem-se um indicador de degradação de sinal do evento TRIP maior do que o indicador proporcionado por ALARME. Na Figura 5.11 pode-se enxergar que o gráfico resultante tem a mesma forma que os conjuntos *fuzzy* na faixa de 80 a 85°C, isso quer dizer que tem o mesmo comportamento quando foram projetados, pois à medida que a temperatura aumenta os indicadores de degradação do evento ALARME (linha vermelha) decrescem os indicadores do evento TRIP (linha azul) cresce.

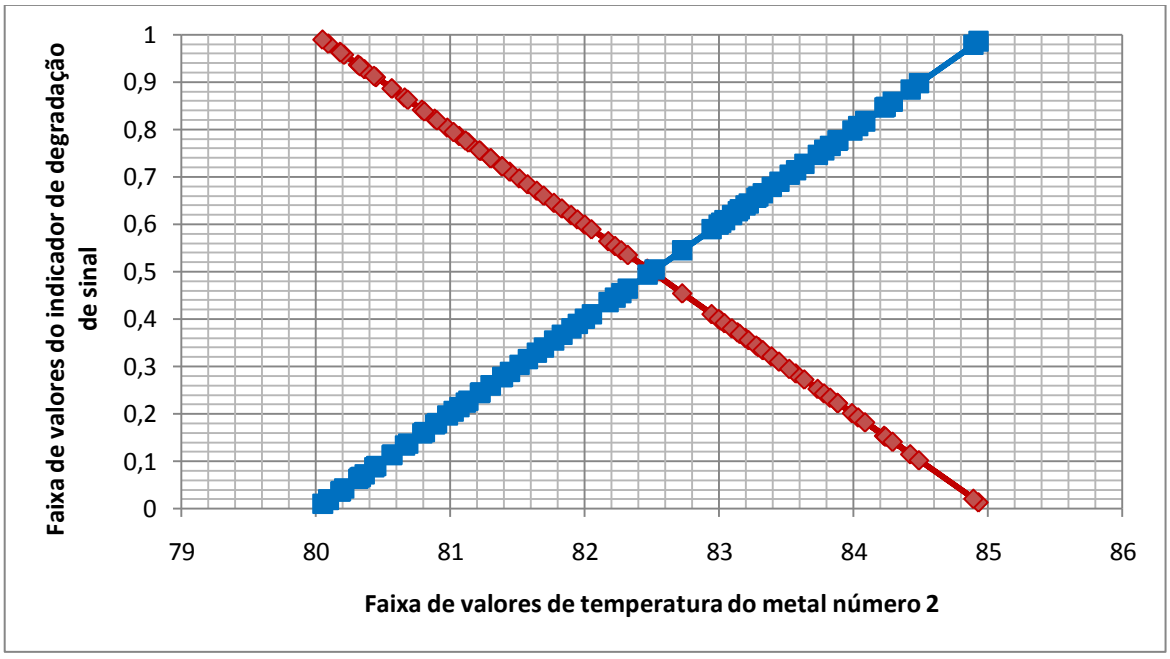

Figura 5.11 – Gráfico de tendência dos indicadores de degradação de sinal com relação aos valores do sensor para os ventos ALARME e TRIP

Portanto, da Tabela 5.3 e a Figura 5.11, também existiram eventos TRIP que foram reconhecidos como eventos ALARME devido ao fato que o sistema SIMPREBAL não possui a flexibilidade que possuem a base de regras *fuzzy*.

Seguindo a mesma análise, pode-se gerar outro gráfico de tendências dos indicadores de degradação de sinal tanto para o evento ALARME (ver Figura 5.12) quanto para o evento TRIP (ver figura 5.13) em um intervalo de tempo a partir da Figura 5.11.

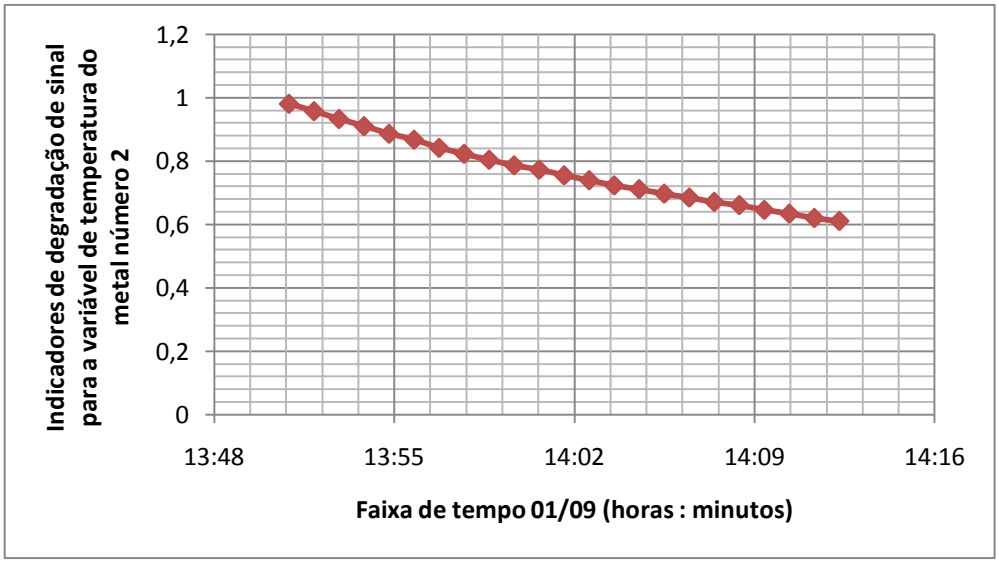

Figura 5.12 – Gráfico de tendência do indicador de degradação do sinal para o evento ALARME do metal 2 (decrescente)

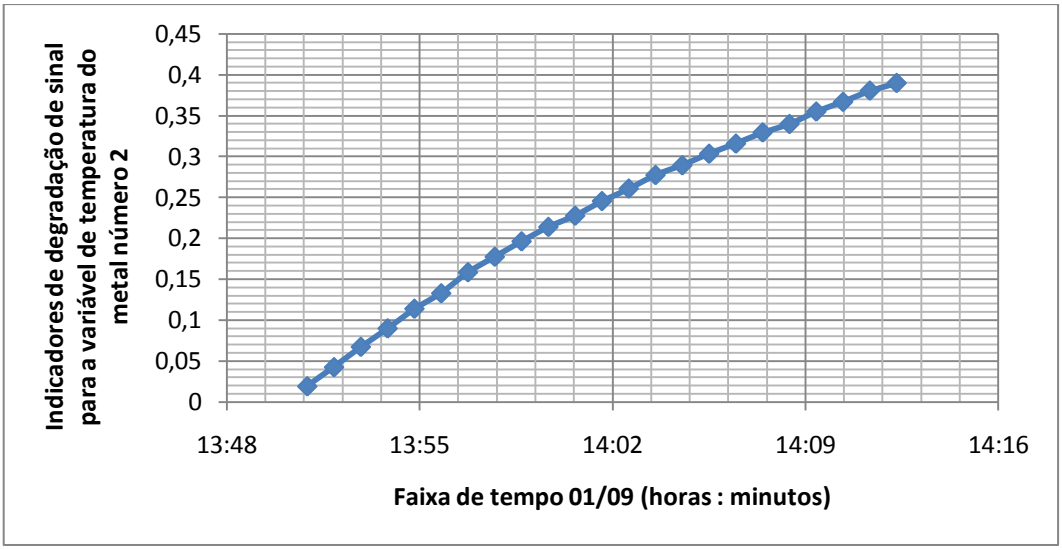

Figura 5.13 – Gráfico de tendência do indicador de degradação do sinal para o evento TRIP do metal 2 (crescente)

Finalmente a análise de confiabilidade do "conjunto de valores de temperatura que originou o evento TRIP no SIMPREBAL" da Tabela 5.1 é apresentada na Tabela 5.4.

Tabela 5.4 – Conjunto de indicadores de degradação de sinal para o evento TRIP do metal número 2

CONJUNTO DE LEITURAS QUE ORIGINARAM TRIPS NO SIMPREBAL (°C)	INDICADOR DE DEGRADAÇÃO DO EVENTO ALARME	INDICADOR DE DEGRADAÇÃO DO EVENTO TRIP
86,46695	0	1
86,644745	0	1
86,75253	0	1
86,83093	0	1
86,92966	0	1
87,019226	0	1
87,12897	0	1
87,20502	0	1
87,30246	0	1
...	...	...

Uma particularidade da Tabela 5.4 é que todos os indicadores de degradação para a faixa ALARME são zeros e para a faixa TRIP são uns, pois todos esses valores de leitura sobrepassam o valor de 85°C (limite superior de ALARME), por isso todos esses valores originam eventos TRIP, portanto, todos eles possuem um indicador máximo (unidade).

Todos os valores da Tabela 5.4 podem ser esboçados como é mostrado na Figura 5.14. Da mesma forma que a Figura 5.11, a Figura 5.14 mostra a função de pertinência *fuzzy* para a faixa TRIP como foi projetado desde um início.

Para encerrar esta validação da base de regras *fuzzy* proposta para a camada de monitoração da condição do sistema SIMPREBAL, é mostrada na Figura 5.16, o número de ALARMES, ALERTAS e TRIPS totais geradas com a nova base de regras *fuzzy* nos mesmos meses (setembro, outubro e novembro do ano 2009) para uma análise comparativa com a Figura 5.6 gerada pela base de regras original para esta camada.

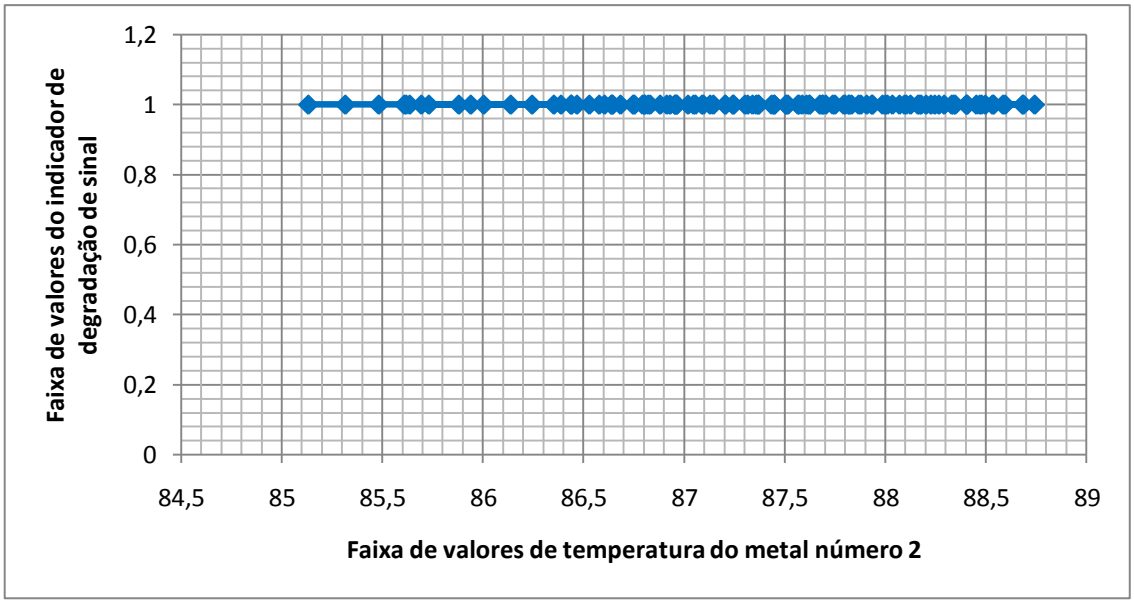

Figura 5.14 – Gráfico de tendência do indicador de degradação de sinal com relação aos valores do sensor para o evento TRIP

Finalmente como foram feitos para as Figuras 5.8 e 5.11, pode-se criar um gráfico para monitoramento da evolução do indicador de degradação de sinal a partir da Figura 5.12 como é mostrado na Figura 5.15.

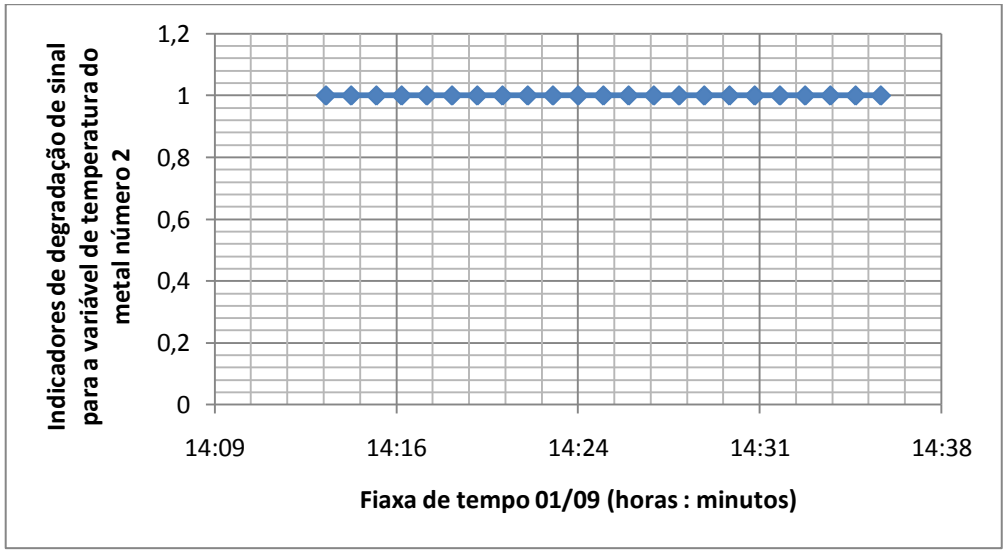

Figura 5.15 – Gráfico de tendência do indicador de degradação do sinal para o evento TRIP do metal 2 (constante)

As Figuras 5.9, 5.10, 5.12, 5.13 e 5.15 são intencionadas para monitoramento dos indicadores de degradação sinal para a variável em estudo em tempo real.

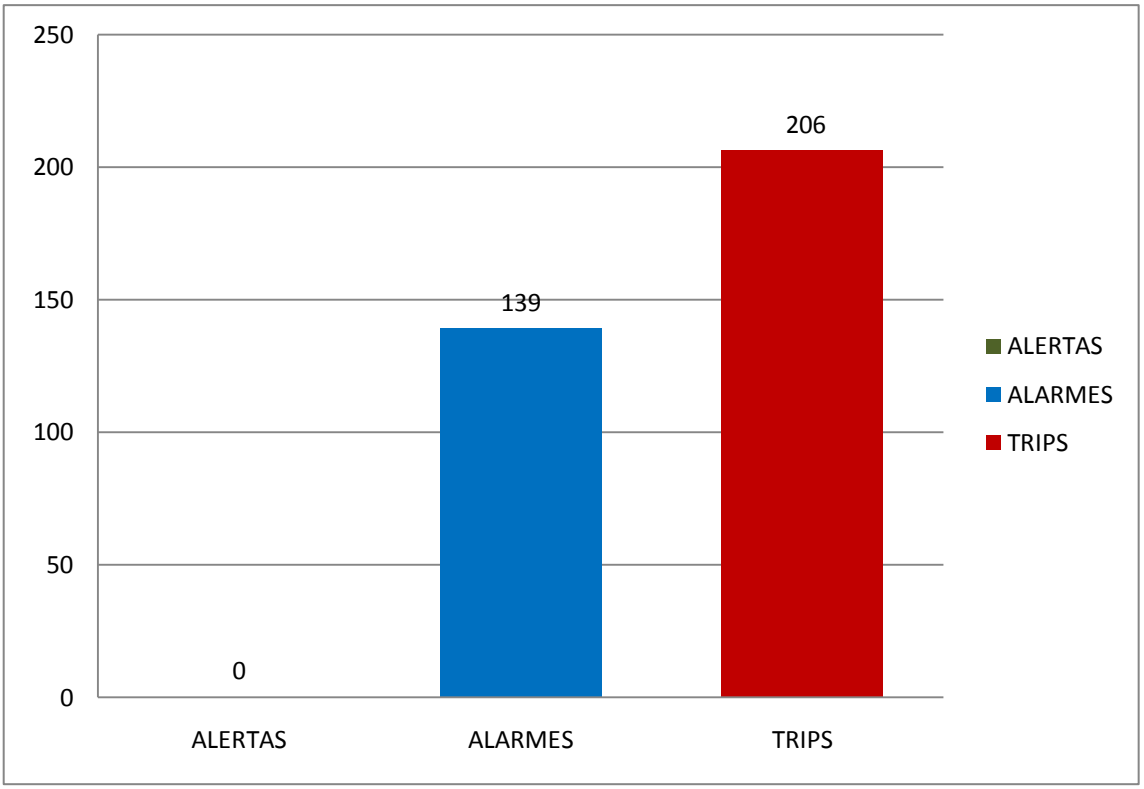

Figura 5.16 - Número de eventos geradas pela temperatura do metal 02 com a base de regras *fuzzy* nos meses de setembro, outubro e novembro do ano 2009

Algumas observações entre a Figura 5.6 e a Figura 5.16 são dadas a seguir:

- ✓ Houve um re-ordenamento do número de ALERTAS, ALARMES e TRIP. Isto é devido ao fato que a base de regras *fuzzy* possui indicadores de degradação de sinal para mensurar os eventos que podem acontecer. Isto traz flexibilidade à base de regras para um monitoramento de condição mais efetivo.

- ✓ Não houve nem um evento ALERTA, por que as leituras de temperatura que originavam esse evento estavam muito próximos ao limite para gerar eventos ALARMES, portanto o número de ALERTAS distribuiu-se na faixa de ALARME. Na faixa de ALARME aconteceu a mesma cosia, parte delas distribui-se como eventos TRIPS.

- ✓ Com a base de regras *fuzzy* proposta, superou-se o problema da faixa muito rigorosa para catalogar eventos (ALERTAS, ALARMES e TRIPS) na camada de monitoramento de condição do SIMPREBAL.

- ✓ O bom desempenho das regras deve-se ao fato de possuir funções de pertinência fuzzy triangulares representando suas faixas de operação. A base de regras *fuzzy* é integrável à camada de monitoramento de condição por que foi criada usando as mesmas ferramentas computacionais: JESS, *FuzzyJ Toolkit* e *FuzzyJess*. Deve-se levar em conta outras considerações com relação ao código Java do lado do servidor no sistema SIMPREBAL que serão comentadas no Capítulo 6.

- ✓ Através do conceito de indicador de degradação de sinal, pode-se seguir a pista à evolução do defeito para tomar ações preventivas oportunas. Pode-se esboçar os indicadores de degradação de sinal para qualquer variável de qualquer subsistema de Balbina, criando-se previamente uma base de regras *fuzzy* consistente aplicando a metodologia proposta nesta dissertação.

6 CONCLUSÕES, CONTRIBUIÇÕES E SUGESTÕES PARA TRABALHOS FUTUROS

Como parte final desta pesquisa de mestrado são apresentadas as contribuições e conclusões associadas à metodologia proposta para o desenvolvimento de um sistema de monitoramento inteligente de condição usando funções de pertinência *fuzzy* para manutenção preditiva de ativos. Subsequentemente são apresentadas sugestões para uma futura integração da base de regras *fuzzy* na camada de monitoramento de condição do sistema SIMPREBAL.

6.1 CONCLUSÕES

Como principais conclusões deste trabalho de pesquisa pode-se citar as seguintes:

- A metodologia proposta propõe o desenvolvimento de uma base de regras de produção usando funções de pertinência *fuzzy* para o monitoramento inteligente de condição de máquinas. Isto traz uma maior confiabilidade na catalogação de eventos de ALERTAS, ALARMES e TRIPS do sistema SIMPREBAL, o qual foi validado nos Capítulo 4 e 5.

- Como parte do desenvolvimento computacional das regras *fuzzy*, foi inserido e definido o conceito do indicador de degradação de sinal. Os indicadores de degradação de sinal foram projetados para monitorar a evolução do defeito contida no sinal. Isto consegue mensurar o defeito que pode estar contida em alguma variável de estudo e, por conseguinte, poder tomar ações preventivas de forma eficiente, antes de acontecer alguma falha na máquina.

- A construção de um conjunto de regras alternativo do tipo SE-ENTÃO *fuzzy* oferece um processamento mais eficaz e flexível do que a base de regras original do sistema SIMPREBAL, pois possui a capacidade de lidar com a incerteza inerente a qualquer representação do conhecimento. Mas como uma contraparte à habilidade de lidar com a incerteza contida nas informações, é originado um aumento na complexidade de criação das regras *fuzzy*.

- A nova base de regras *fuzzy* foi exitosamente implementada e simulada na interface ECLIPSE usando a base de dados das leituras da variável de temperatura do metal 2 do sistema SIMPREBAL, para avaliar o desempenho e sua viabilidade de integração na sua camada de monitoramento de condição (ver Capítulos 4 e 5).

- O análise e resultados encontrados nos Capítulo 4 e 5 sugerem que a base de regras *fuzzy* proposta para a camada de monitoramento de condição do sistema SIMPREBAL é integrável.

6.2 CONTRIBUIÇÕES DO TRABALHO

São mencionadas a seguir as contribuições mais relevantes desta dissertação:

- A metodologia concebida está baseada no uso de funções de pertinência *fuzzy* para incrementar a confiabilidade do monitoramento de condição de máquinas. Isto é outro tipo de aplicação da lógica *fuzzy*. Geralmente a metodologia da lógica *fuzzy*, implica codificação, inferência e descodificação.

- A raciocínio para a criação da base de regras *fuzzy* originado nesta pesquisa de mestrado poderá ser amplamente usado, com uma prévia adaptação dos recursos computacionais segundo o cenário de desenvolvimento, numa gama diversificada de aplicações industriais para monitoramento de condição inteligente de máquinas.

- Através dos indicadores de degradação de sinal, pode ser monitorado a evolução do defeito em alguma variável física e assim, poder gerar um mais eficaz monitoramento de condição, melhores diagnósticos e sugestões de ações operacionais de manutenção.

- Criou-se uma base de regras *fuzzy* projetada para a camada de monitoramento de condição do sistema SIMPREBAL com o intuito de melhorar a confiabilidade dos estados das máquinas que tem sob controle.

6.3 IMPLEMENTAÇÂO COMPUTACIONAL

Com relação à implementação computacional da base de regras *fuzzy*, foi feita de forma não integrada ao sistema SIMPREBAL, pois está dissertação visa a um estudo prévio sobre

sua viabilidade de integração e desempenho na camada de monitoramento de condição do sistema SIMPREBAL usando variáveis simuladas.

Como produto do anteriormente exposto, foi implementando uma base de regras *fuzzy* usando as ferramentas computacionais: JESS, Java, FuzzyJ Toolkit e a biblioteca *FuzzyJess*, a qual possui as seguintes características:

- Compatibilidade computacional na camada de monitoramento de condição do sistema SIMPREBAL, pois foram usadas as mesmas ferramentas computacionais anteriormente mencionadas.
- Um estrutura de regras que computa variáveis linguisticas em vez de intervalos muito rigorosos. Isto traz flexibilidade no processamento para catalogar algum tipo de evento que possa originar qualquer leitura advinda de alguma variável de processo.
- Processamento de indicadores de degradação de sinal. Os indicadores foram projetador para monitorar a degradação do sinal de alguma variável de processo com o intuito de mensurar a evolução de algum defeito contida nesse sinal.

6.4 SUGESTÕES PARA TRABALHOS FUTUROS

Esta seção objetiva oferecer um conjunto de sugestões para a integração da base de regras *fuzzy* na camada de monitoramento de condição do sistema SIMPREBAL o qual não foi feito dada à complexidade do código Java no lado do servidor. Além disso, este estudo prévio foi direcionado para implementar e validar, de forma não integrada ao SIMPREBAL, a base de regras *fuzzy* desenvolvida a partir da metodologia proposta com o intuito de avaliar seu desempenho e viabilidade de integração futura nessa camada.

A seguir um conjunto de sugestões para integrar a base de regras *fuzzy* na camada de monitoramento de condição do SIMPREBAL, como resultado do análise ao longo deste trabalho de pesquisa é mencionada:

- Desenvolver um estudo detalhado nas classes: **JessProxy.java** e **ServerMain.java**, pois na classe JessProxy.java encontra-se a configuração para executar a base de regras

da camada de monitoramento de condição, essa classe faz o ponte entre o código Java e o código da base de regras em JESS no lado do servidor. Por outro lado, a classe ServerMain.java é onde se executa o ciclo de inferência do sistema inteligente de manutenção preventiva levando em conta a configuração estabelecida na outra classe.

- Com relação à classe **JessProxy.java,** podem ser levadas em conta algumas sugestões para integração de processamento nessa classe usando a biblioteca *FuzzyJess*:

  - ✓ Importar as bibliotecas: **nrc.fuzzy.*** e **nrc.fuzzy.jess.***
  - ✓ Criar objetos privados do tipo **FuzzyRete** em vez de **Rete** para computar a base de regras *fuzzy* proposta.
  - ✓ Modificar o nome da base de regras *fuzzy* proposta nas linhas de código para executá-la.

- Criar uma classe extra no lado do servidor para a fase de codificação da base de regras com o intuito de codificar os valores das leituras dos sensores advindos do servidor OPC.

- Integrar os indicadores de degradação de sinal no lado do servidor para gerar gráficos em tempo real da evolução de um possível defeito no sinal de qualquer variável com o intuito de melhorar as tendências em diagnósticos e tomadas de decisão.

6.5 CONSIDERAÇÕES FINAIS

Foi usada a lógica *fuzzy* para incrementar a flexibilidade do monitoramento de condição de máquina, implementando a fase de codificação (*fuzzification*), com o objetivo de incrementar a flexibilidade das faixas de operação das variáveis físicas da usina hidrelétrica de Balbina. Para a integração da base de regras *fuzzy* ao sistema SIMPREBAL é necessário levar em conta algumas modificações no código Java do sistema SIMPREBAL para que possa ser executada satisfatoriamente no servidor da usina e avaliar as tendências das variáveis físicas monitoradas através dos indicadores de degradação de sinal propostas nesta dissertação.

REFERÊNCIAS BIBLIOGRÁFICAS

Abraham, A., Baets, B., Koppen, M., Nickolay, B., (2006), "Applied soft computing techniques: the challenge of complexity", SPRINGER, pp. 825.

Alape, L.F. Moreno, I.P., Álvares, A.J., Amaya, E.J. (2011a). A Methodology Based In Case-Based Reasoning to Build a Knowledge-Base Applied to Failure Diagnosis System of Hidrogenerators Machinery, Proceedings of COBEM 2011, 21st Brazilian Congress of Mechanical Engineering, October 24 – 28, Natal, RN, Brazil.

Alape, L.F. Moreno, I.P., Álvares, A.J., Amaya, E.J., (2011b). Diseño y Construcción de una Base de Conocimiento para un Sistema Inteligente de Mantenimiento Aplicado a Equipos de Plantas Hidroeléctricas Basado en el Enfoque RBC, X Congresso Ibero-Americano em Engenharia Mecânica, CIBEM10, Porto, Portugal.

Álvares, A.J., (2006), "Sistema *I-Kernel*: um kernel inteligente para o SIMPREBAL, Sistema de Manutenção Preditiva de Balbina", Relatório técnico de pesquisa, Relatório de produtos gerados – Etapa 3, Ano 1, UnB, Brasília, DF.

Álvares, A.J., Amaya, E.J., Souza, R.Q., Tonaco, R.P., Lima, A.A., (2009), "Sistema Inteligente de Manutenção Baseada em Condição para Usina Hidrelétrica de Balbina", Anais do V congresso de inovação Tecnológica em Energia Elétrica – V CITENEL, Belém/PA, 22 a 24 de junho.

Álvares, A.J., Amaya, E.J., (2010), "SIMPREBAL: An expert system for real-time fault diagnosis of hydrogenerators machinery", Emerging Technologies and Factory Automation (ETFA), IEEE Conference, 13 – 16 September.

Álvares, A.J., Amaya, E.J., Souza, R.Q., Tonaco, R.P., Lima, A.A., (2009), "Sistema inteligente de manutenção baseada em condição para usina hidrelétrica de Balbina", Anais do V congresso de Inovação Tecnológica em Energia Elétrica – V CITENEL, Belém/PA, 22 a 24 de Junho.

Álvares, A.J., Amaya, E.J., Tonaco, R.P., (2007a), "Sistema de manutenção baseada em condição para usina hidrelétrica de Balbina", Congresso de Computação Aplicada CAIP'2007.

Álvares, A.J., Gudwin, R.R., Souza, R.Q., Amaya, E.J., Tonaco, R.P., (2007b), "Na intelligent kernel for maintenance system of a hydroelectric power plant", In: COBEM 2007 – 19th International Congress of Mechanical Engineering: TT1681, Anais, Brasília, Brazil.

Álvares, A.J., Amaya, E.J., (2007), "Different control strategies used in didactic plant PD-3 of SMAR through OPC technology", 19th International Congress of Mechanical Engineering, November 5 – 9, Brasília, DF.

Álvares, A.J., Tonaco, R.P., Fernandes, L.P., (2007c), "Análise FMEA para aplicação da metodologia de manutenção centrada em confiabilidade: Estudo de caso em turbinas hidráulicas", Congresso de Computação Aplicada CAIP'2007.

Amaya, E. J., (2008), "Aplicação de Técnicas de Inteligência Artificial no Desenvolvimento de um Sistema de Manutenção Baseada em Condição". Dissertação de Mestrado em Sistema Mecatrônicos, Publicação ENM.DM-21A/08, Departamento de Engenharia Mecânica, Universidade de Brasília, DF, 172p.

Bagadia, K., (2006), "Computerized maintenance management: systems made easy", How to Evaluate, Select and Manage CMMS, McGraw-Hill Companies, pp. 255.

Bond, R.R., (2010), "Vibration-based condition monitoring", Industrial, Aerospace and Automotive Applications, WILEY, pp. 284.

BSI., (2011), "Web site: http://www.standardcentre.co.uk", British Standard Institution.

Carpenter, G.A., Grossberg, S., (2003), "Adaptive resonance theory", In M.A. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, Second Edition, pp. 87-90.

Chang, Ch. L., Wei, Ch. Ch., Lee, Y. H., (1999), "Failure Mode and Effect Analysis Using Fuzzy Method and Grey Theory", Cybernetic, Vol. 28, N° 9, 1072p.

Cho, H.J., Park, J.K., (1997), "An expert system for fault selection diagnosis of power systems", IEEE Trans. Power Syst., vol. 12, pp. 342 – 348, Feb.

Ciarapica, F.E., Giacchetta, G., (2006), "Managing the condition-based maintenance of a combined-cycle power plant: An approach using soft computing techniques", Journal of Loss Prevention in the Process Industries 19, ELSEVIER.

Cigolini, D.R., Desmukh, A.V., Fedele, L., McComb, S.A., (2009), "Recent advances in maintenance and infrastructure management", SPRINGER, pp. 289.

Cox, E., (1994), "The fuzzy systems handbook", Academic Press, Inc.

Dhillon, B.S., (2007), "Human reliability and error in transportation systems", SPRINGER, pp. 177.

Dote, Y., Ovaska, S.J., (2001), "Industrial Application of Soft Computing: A Review", Proceedings of the IEEE, vol. 89, No. 9, Manuscript received January 15, revised April 9.

Fast, M., Palmé, T., (2009), "Application of Artificial Network to the Condition Monitoring and Diagnosis of a Combined Heat and Power Plant", ELSEVIER, Energy 35 (2010), pp. 1114 – 1120.

Fujimoto, R.Y., Padovese, L.R., (2001), "Rolling Bearing Fault Diagnostic System Using Fuzzy Logic", IEEE International Fuzzy Systems Conference.

Friedman-Hill, E., (2003), "Jess in Action: Rule-Based System in Java". 1era Ed, Manning Editor, Greenwich, CT.

Friedman-Hill, E., (2008), "Jess® The Rule Engine for the Java™ Platform Version 7.1p2", Sandia National Laboratories, 5 November.

Garcia, C.M., Sanz-Bobi, A.M., Pico, J., (2006), "SIMAP: Intelligent System for Predictive Maintenance, Application to the health condition monitoring of a windturbine gearbox", Computers in Industry, ELSEVIER.

Garga, K.A., McClintic, T.K., Campbell, L.R., Yang, Ch-Ch., Lebold, S.M., Hay, A.T., Byington, S.C., (2001), "Hybrid Reasoning for Prognostic Learning in CBM Systems", Applied Research Laboratory, The Pennsylvania State University.

Grabys, B., Leiviska, K., Strackeljan, J., (2005), "Do smart adaptive systems exist: A best practice for selection and combination of intelligent methods", SPRINGER, pp. 341.

Gudwin, R.R., (2006), "Sistema I-Kernel: Um Kernel Inteligente para o SIMPREBAL – Sistema de Manutenção Preditiva de Balbina", Especificação do Sistema, Relatório Técnico, Campinas, Dezembro.

Han, Y., Song, Y.H., (2003), "Condition Monitoring Techniques for Electrical Equipment – A Literature Survey", IEE Transactions On Power Delivery, Vol. 18, N°. 1, January.

Holsapple, C.W., (2003), "Handbook on knowledge management", International Handbooks on Information Systems, Springer, pp. 679.

Isermann, R., (2006), "Fault-Diagnosis Systems – An Introduction from Fault Detection to Fault Tolerance", Springer, 478p.

Javadpour, R., Knapp, M.G., (2003), "A fuzzy neural network approach to machine condition monitoring", Computers and Industrial Engineering.

Jain, C.L., Sato-Llic, M., Virvou, M., Tsihrintzis, Balas, E.V., (2008), "Computational intelligence paradigms: innovative applications", SPRINGER, pp. 251.

Jing, D., Ping, Z., Xingshan, L., Jinsong, Y., (2007), "A Review on Reasoning Techniques implementing Integrated Health Management", The Eighth International Conference on Electronic Measurement and Instruments, ICEMI'2007, Beijing 100083 China.

Kosko, B., (1997), “Fuzzy Engineering”, Prentice Hall, Inc.

Kothamasu, R., Huang, H. S., VerDuin, H. W., (2006), “System health monitoring and prognostic – A review of current paradigms and practices”, Springer-Verlag London Limited, DOI: 10.1007/s00170-004-2131-6.

Kothamasu, R., Huang, S.H., (2007), “Adaptive Mamdani fuzzy model for condition-based maintenance”, Fuzzy sets and Systems 158, ELSEVIER, pp 2715 – 2733.

Kumar, V.E., Chaturvedi, S.K., Deshpandé, A.W., (2009), “Maintenance of industrial equipment: Degree of certainty with fuzzy modeling using predictive maintenance”, International Journal of Quality & Reliability Management, Vol.26 Iss:2,196-221 pp.

Lee, J., Ni, J., Djurdjanovic, D., Qiu, H., Liao, H., (2006), “Intelligent prognostics tools and e-maintenance”, Computers in Industry 57, 476-489 pp.

Ma, Z., (2006), “Fuzzy database modeling of imprecise and uncertain engineering information”, Studies In Fuzziness and Soft Computing, Springer, pp. 178.

McCarthy, J., (1979), “History of Lisp”, Artificial Intelligence Laboratory, Stanford University, 12 February, *http://www-formal.stanford.edu/jmc/history/lisp/lisp.html.*

Mechefske, C.K., (1998), “Objective machinery fault diagnosis using fuzzy logic”, Mechanical Systems and Signal Processing.

MIMOSA., (2008), “Web site: http://www.mimosa.org”, Machinery Information Management Open Systems Alliance.

Mitra, S., Hayashi, Y., (2000), “Neuro-fuzzy rule generation: Survey in soft computing framework”, IEEE Transaction on Neural Networks, Vol. 11, No. 3, May.

Momoh, J.A., Tomsovic, K., (1995), “Overview and literature survey of fuzzy set theory in power systems”, IEEE Transaction on Power Systems, Vol. 10, No 3, August.

Moreno, I.P., Álvares, A.J., Alape, L.F. (2011a). “Una Abordaje Metodológica para Manutención Predictiva Basada em Lógica Fuzzy Aplicada a Plantas de Poder Hidroeléctrica”, X Congresso Ibero-Americano em Engenharia Mecânica, CIBEM10, Porto, Portugal.

Moreno, I.P., Álvares, A.J., Alape, L.F. (2011b). “Methodology for the building of a fuzzy expert system for predictive maintenance of hydroelectric power plants”, Proceedings of COBEM 2011, 21st Brazilian Congress of Mechanical Engineering, October 24 – 28, Natal, RN, Brazil.

Moubray, J., (1997), “Reliability-Centered Maintenance”, 2Ed, Industrial Press Inc, Woodbine, New Jersey, 412p.

Nadakatti, M., Ramachandra, A., Kumar, S.A.N., (2008), "Artificial Intelligence-Based Condition Monitoring for Plant Maintenance", Assembly Automation, Emerald Group Publishing Limited [ISSN 0144-5154].

NASA., (1996), "NASA clips rule-based language". http://www.siliconvalleyone.com/clips.htm.

NBR-5460 (1992), "Eletrotécnica e eletrônica, Sistemas elétricos de potência, Terminologia", Associação Brasileira de Normas Técnicas, Rio de Janeiro, Brasil.

NBR-5462., (1994), "Confiabilidade e Mantenabilidade", Terminologia, Associação Brasileira de Normas Técnicas, Rio de Janeiro, Brasil.

Niu, G., Yang, Bo-Suk., Pecht, M., (2010), "Development of an optimized condition-based maintenance system by data fusion reliability-centered maintenance", Reliability Engineering and System Safety 95, pp. 786 – 796.

Orchard, R., (2006), "NRC FuzzyJ Toolkit for the Java™ Platform User' Guide Version 1.10a", Integrated Reasoning, Institute for Information Technology, National Research Council Canada, September.

Pandian, A., Ali, A., (2009), "A Review of Recent Trends in Machine Diagnosis and Prognosis Algorithms", IEEE press.

Pawlak, Z., (1982), "Rough Sets", International Journal of Computer and Information Sciences, Vol. 11, No. 5.

Pedrycz, W., Gomide, F., (2007), "Fuzzy Systems Engineering: Toward Human-Centric Computing", Wiley Interscience, IEEE, 526p.

Prazzo, C.E., Antunes, M.B., Turra, A.E., Pereira, J.A., (2010), "The Use of Fuzzy Logic to Classify Heating Severity of Components Applied in a Thermography Based Maintenance", XVIII Congresso Brasileiro de Automática, 12 a 16 de setembro, Bonito, MS.

Rigoni, E., (2009), "Metodologia para implementação da manutenção centrada na confiabilidade: uma abordagem fundamentada em Sistemas Baseados em Conhecimento e Lógica *Fuzzy*", UFSC, Florianópolis, Fevereiro, pp. 342.

Ross, J.T., (2010), "Fuzzy logic with engineering applications", Third Edition, WILEY, pp. 530.

Runkler, T., Glesner, M.A., (1993), "A set of axioms for defuzzification strategies toward a theory of rational defuzzification operators", Proceedings of the 2nd IEEE International Conference on Fuzzy Systems, San Francisco, IEEE Press, pp.1161-1166.

Russell, E., Yuhui, S., (2007), "Computational intelligence", Concepts and Implementations, ELSEVIER, pp. 469.

Rutkowski, L., Cpalka, K., (2003), "Flexible neuro-fuzzy systems", IEEE Transactions on Neural Networks, Vol. 14, No. 3.

Rutkowski, L., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, M.J., (2010), "Artificial intelligence and soft computing", 10th International Conference, ICAISC 2010, Zakopane, Poland, Part I, pp. 579.

Siemens., (2011), Web Site: http://www.industry.siemens.com.br/automation/br/, SIMATIC HMI, a Principal Solução de Interface Homem-Máquina.

Siler, W., Buckley J.J., (2005), "Fuzzy expert systems and fuzzy reasoning", Wiley-Interscience, pp. 422.

Smar., (2001), "Equipamento de Campo Série 302 Foundation. Manual de Instalação, Operação e Manutenção".

Smar., (2004), "Manual de operações: planta didática III", Departamento de Engenharia de Operações, *http://www.smar.com.br*.

Smar., (2005), "Manual de instruções dos blocos funcionais Fieldbus Foundation".

Smar., (2011), Web Site: http://www.smar.com/brasil2/system302/, Plataforma de Controle e Automação de Processos.

Souza, R. Q., (2008), "Metodologia e Desenvolvimento de um Sistema de Manutenção Preditiva Visando à Melhora da Confiabilidade de Ativos de Usinas Hidrelétricas". Dissertação de Mestrado em Sistemas Mecatrônicos, Publicação ENM.DM-23A/08, Departamento de Engenharia Mecânica, Universidade de Brasília, Brasília, DF, 226p.

Tasker, P., McFarlane, D., Wild, Peter, J.W., Parry, G., Ng, I., (2011), "Complex engineering service systems: concepts and research", SPRINGER, pp. 466.

Tavner, P., Ran, L., Penman, J., Sedding, H., (2008), "Condition monitoring of rotating electrical machines", British Library Cataloguing in Publication data, pp. 269.

Telang, A., (2010), "Comprehensive maintenance management: policies, strategies and options", ISBN-978-81-203-3953-8, pp. 170.

Tonaco, R.P., Fernandes, L., Álvares, A.J., (2007), "Sistema especialista, baseado em regras a partir da FMEA, para avaliação de saúde de equipamentos de malhas industriais", 8° Congresso Iberoamericano de Engenharia Mecânica, Cusco, 23 a 25 de Outubro.

Tonaco, R. P., (2008), "Metodologia para Desenvolvimento de Base de Conhecimento Aplicada à Manutenção Baseada em Condição de Usinas Hidrelétricas", Publicação ENM.DM 22/08, Departamento de Engenharia Mecânica e Mecatrônica, Universidade de Brasília, Brasília, DF, 167p.

Valavanis, P.K., (2009), "Applications of intelligent control to engineering systems", Intelligent Systems, Control and Automation: Science and Engineering, Springer, pp. 393.

Wang, K., (2003), "Intelligent condition monitoring and diagnosis system: A computational intelligence approach", Frontiers in Artificial Intelligence and Applications, IOS Press, pp. 118.

Wang, W.Q., Golnaraghi, F.M., Ismail, F., (2004), "Prognosis of machine health condition using neuro-fuzzy systems", Mechanical Systems and Signal Processing.

Wen, J., Cheng, S., Malik, O.P., (1988), "A synchronous generator fuzzy excitation controller optimally designed with a genetic algorithm", IEEE Trans. Power Syst., vol 13, pp. 884 – 889, Aug.

Yam, R.C.M., Tse, P.W., Li, L., Tu, P., (2001), "Intelligent predictive decision support system for condition-based maintenance", The International Journal of Advanced Manufacturing Technology, pp 383 – 391.

Yu, J., Zhao, H., (2005), "Maintenance plan based on RCM", IEEE/PES Transmission and Distribution, Conference & Exhibition: Asia and Pacific, Dalian, china.

Zadeh, L. A., (1965), "Fuzzy Sets", Information and Control, Vol. 8, pp. 338 – 353.

Zadeh, L.A., (1984), "Coping with the imprecision of the real world", Reports and Articles, Communications of the ACM, April, Volume 27, Number 4.

APÊNDICES

APÊNDICE A – BREVE INTRODUÇÃO À LÓGICA *FUZZY*

Neste apêndice apresentar-se-á uma breve introdução à lógica *fuzzy*, o qual não pretende ser um compêndio exaustivo. Para um estudo mais detalhado da matemática envolvida neste trabalho, podem ser pesquisadas as seguintes referências: Pedrycz e Gomide (2007), Cox (1994) e Kosko (1997) contidas nesta dissertação.

A.1 FUNÇÃO CARACTERÍSTICA TRIANGULAR

As funções características do tipo triangular são totalmente definidas por segmentos lineares, como é mostrada na Equação A.1:

$$A(x,a,m,b) = \begin{cases} 0, & Se\ "x" \leq a \\ \frac{x-a}{m-a}, & Se\ "x"\ \in < a,m > \\ \frac{b-x}{b-m}, & Se\ "x"\ \in [m,b > \\ 0, & Se\ "x"\ \geq b \end{cases} \quad \text{(A.1)}$$

Pode-se usar uma notação simplificada para representar a Equação A.1, como: $A(x,a,m,b) = \max\{\min[(x-a)/(m-a),\ (b-x)/(b-m)], 0\}$. Uma representação gráfica da Equação A.1 é mostrada na Figura A.1.

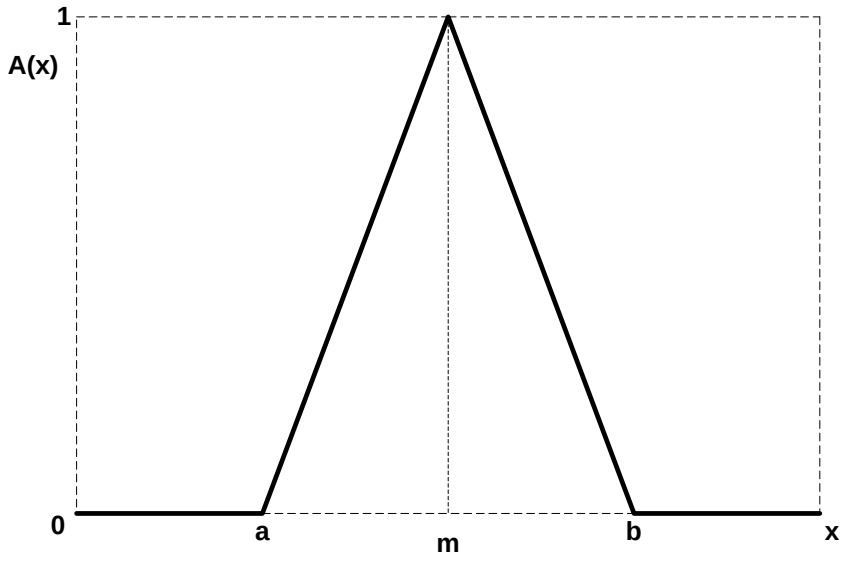

Figura A.1 – Função característica triangular

Na Figura A.1, “m” denota um valor modal típico de um conjunto *fuzzy*, “*a*” e “*b*” são o limite inferior e superior respectivamente e poderiam ser pensados como elementos extremos do universo de discurso que delimita os elementos pertencentes a “*A(x)*” com graus de pertinência não nulos.

As funções triangulares *fuzzy* são a forma mais simples de modelar o grau de pertinência dos elementos de um conjunto *fuzzy*, e é completamente definida com só três parâmetros para evitar muito consumo de processamento computacional. Como exposto anteriormente, deve-se notar que, a função triangular *fuzzy* deve refletir, com relação à largura, parâmetros a, b e m, o significado físico para a variável a ser codificada (Pedrycz e Gomide 2007, Orchard 2006, Negnevitsky 2005).

A.2 FUNÇÃO CARACTERÍSTICA GAUSSIANA

As funções características do tipo gaussiana podem ser definidas matematicamente pela Equação A.2.

$$A(x, m, \sigma) = exp\left(-\frac{(x-m)^2}{\sigma^2}\right) \tag{A.2}$$

A Equação A.2 pode ser expressa de forma gráfica como é mostrada na Figura A.2. Uma função característica gaussiana possui dois parâmetros importantes. O valor modal “m” que representa um elemento típico de A(x), enquanto, “σ” representa a largura da função.

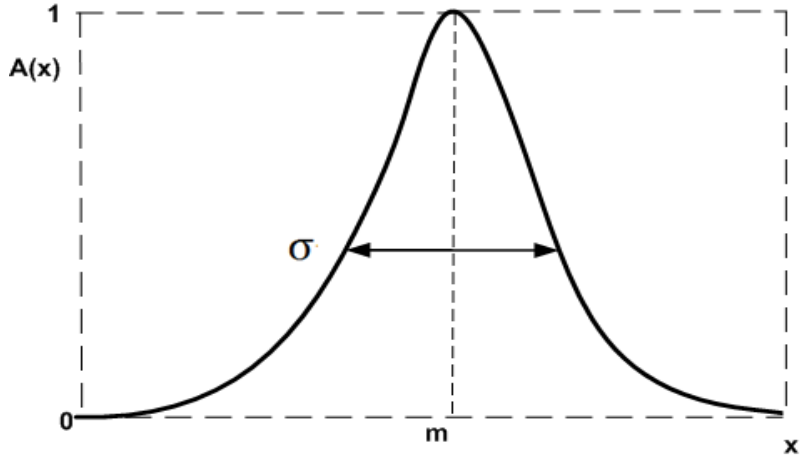

Figura A.2 – Função característica gaussiana

A.3 FUNÇÃO CARACTERÍSTICA TRAPEZOIDAL

As funções do tipo trapezoidal *fuzzy* são completamente especificadas por quatro parâmetros genéricos “a”, “m”, “n” e “b”, onde cada um define uma das quatro partes lineares da função como mostrado na Figura A.3 e definida matematicamente pela Equação A.3.

$$A(x) = \begin{cases} 0, & Se\ "x" < a \\ \frac{x-a}{m-a}, & Se\ "x"\ \in [a, m > \\ 1, & Se\ "x"\ \in [m,\ n > \\ \frac{b-x}{b-n}, & Se\ "x"\ \in [n, b] \\ 0, & Se\ "x"\ > b \end{cases} \quad \text{(A.3)}$$

Analogamente como aplicado para a função triangular podemos sintetizar a Equação A.3 como: $A(x, a, m, n, b) = \max\{\min[(x-a)/(m-a),\ 1,\ (b-x)/(b-n)], 0\}$.

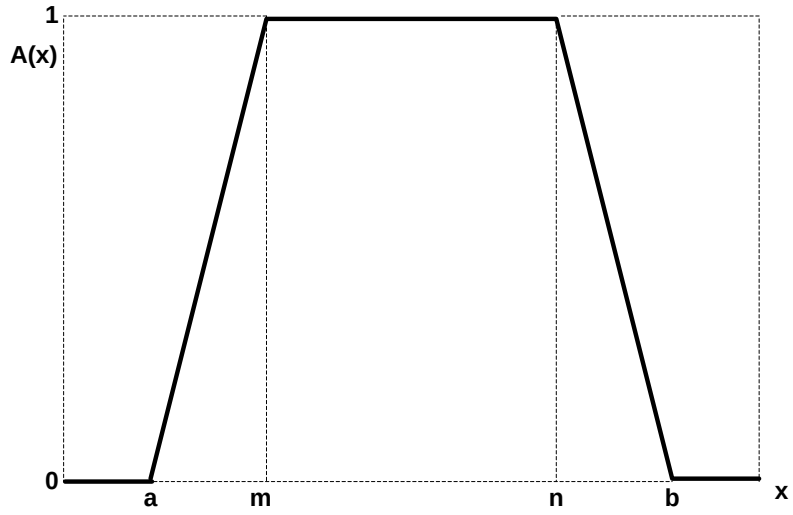

Figura A.3 – Função Característica trapezoidal

A.4 FUNÇÃO CARACTERÍSTICA *SINGLETON*

A função do tipo *singleton* usada neste trabalho, caracterizam-se por possuir um único valor no seu domínio atingindo o máximo valor de pertinência (o valor de um). Matematicamente pode ser representado pela Equação A.4 e graficamente especificado pela Figura A.4.

$$A(x) = \begin{cases} 1, & Se \text{ "x"} = a \\ 0, & Se \text{ "x"} \neq a \end{cases} \quad \text{(A.4)}$$

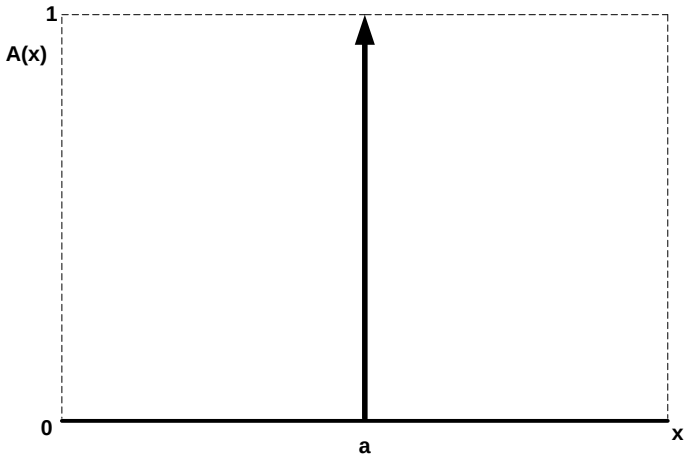

Figura A.4 – Função característica *singleton*

A.5 CASAMENTO DE REGRAS *FUZZY*

Modelos baseados em regras de produção jogam um rol muito importante em modelamento *fuzzy*. Regras *fuzzy* capturam relações entre variáveis *fuzzy* e provêem um mecanismo para relacionar expressões lingüísticas de sistemas com suas necessidades computacionais.

Regras *fuzzy* podem ser formalizadas via uma coleção de relações *fuzzy*, eles naturalmente provêem uma forma para construir modelos de sistemas envolvendo domínio de conhecimento, experiência e dados experimentais. Quando se interpreta regras *fuzzy* como relações *fuzzy* permite-se o uso de cálculos de relações de processos de informações e pode-se computar eficientemente regras *fuzzy* em inúmeras aplicações (Pedrycz e Gomide, 2007).

Basicamente o conhecimento humano pode ser interpretado como uma coleção de proposições em uma linguagem em particular. A representação do conhecimento humano tem de ser de alguma forma manipulável para poder ser computada usando algum formalismo matemático, como por exemplo, a teoria dos conjuntos *fuzzy*. A coleção de proposições deveria conter restrições *fuzzy* ou nebulosas (relações *fuzzy*).

Por outro lado, representar esse conjunto de proposições como representações manipuláveis envolve:

- ✓ Identificação das variáveis cujos valores são restritos pela proposição;
- ✓ Identificação das restrições induzidas pelas proposições;
- ✓ Caracterização de cada restrição através de relações *fuzzy*.

A continuação, algumas representações com relação à sintaxe de regras, composição de regras, semântica, implicações, representação gráfica e computação matemática *fuzzy* serão abordados rapidamente com o intuito de brindar a bagagem matemática sólida para o entendimento desta seção.

Uma proposição básica possui a seguinte estrutura:

Se (**ATRIBUTO**) do (**OBJETO**) é **VALOR**

Cuja forma canônica é representada como:

p: **X** é A

Um exemplo com relação à Temperatura de Ar Frio do Radiador Número 01, com Tag: G126GA12, Canal 01 da UGH01, pertencente ao Sistema Gerador Elétrico, Subsistema de Resfriamento do Gerador (SRG), é apresentado para exemplificar a idéia anteriormente exposta.

Exemplo:

A Temperatura de ar frio do radiador número 01 é HIGH

Temperatura (de ar frio do radiador número 01) é HIGH

p: T é HIGH

Onde:

Variável com valor restrito: **Temperatura**

Restrição induzida: **HIGH**

Caracterização: **Conjunto *fuzzy* HIGH**

Por outro lado, existem proposições compostas (como é o caso do sistema SIMPREBAL), elas são construídas a partir de conjunções e/ou disjunções. Por exemplo, para o caso dos 08 radiadores de temperatura de ar frio que mapeiam o SRG, podem formar uma proposição composta na sua forma canônica como segue:

$$\text{p: } T_1 \text{é } A_1 \;\; e \;\; T_2 \text{é } A_2 \;\; \ldots \;\; e \;\; T_8$$

$B_1, B_2,$ ... $B_m,$ representam conjuntos *fuzzy* em: $Y_1, Y_2,$ ... Y_m

$P\ e\ Q$ são relações em: $X_1 \times X_2 \times$ ... $\times X_n \times Y_1 \times Y_2 \times$ ... $\times Y_m$

Subseqüentemente:

$$P(x_1, \dots, x_n, y_1, \dots, y_m) = f\big(P_a(x_1, \dots, x_n), P_c(y_1, \dots, y_m)\big) \ ;\ p\text{: } (X_1, X_2, \dots, Y_1, \dots, Y_m) \text{ é } P$$

$$Q(x_1, \dots, x_n, y_1, \dots, y_m) = f\big(Q_a(x_1, \dots, x_n), Q_c(y_1, \dots, y_m)\big) \ ;\ q\text{: } (X_1, X_2, \dots, Y_1, \dots, Y_m) \text{ é } Q$$

Onde:

$P\ e\ Q$ são relações em: $X_1 \times X_2 \times$ ... $\times X_n \times Y_1 \times Y_2 \times$ ... $\times Y_m$ induzidas por p e q.

P_a, Q_a, P_c, Q_c são as relações induzidas pelos antecedentes e conseqüentes.

A função "*f*" define o significado da regra através de:

- Implicação *fuzzy*.
- Conjunção (t-normas).
- Disjunção (s-normas).

Um exemplo gráfico de:

P: Se X é *A* Então Y é *B*

É representado pela Figura A.5:

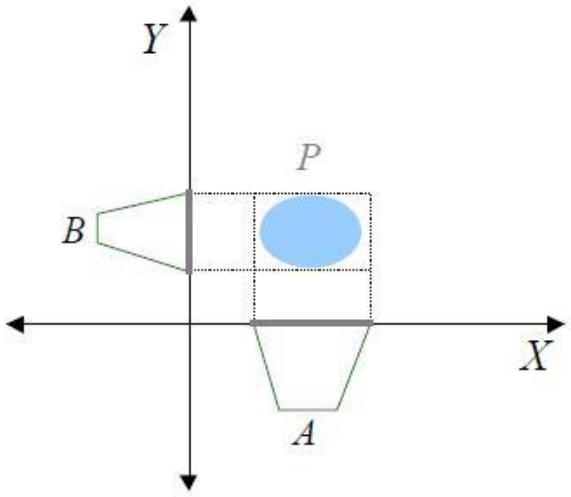

Figura A.5 – Mapeamento do conjunto *fuzzy A* sobre *P*, originando *B*

Generalizando o anterior, para o caso de:

R: Se X é A_1 Então Y é B_1

Se X é A_2 Então Y é B_2

...

Se X é A_N Então Y é B_N

Esse sistema de regras pode ser representado pela Figura A.6:

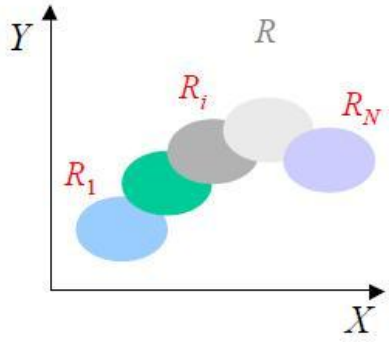

Figura A.6 – Uma representação gráfica do sistema de regras conformado por *R*

Seguindo o estudo sobre a semântica das regras do tipo SE-ENTÃO *fuzzy*. Considerando que o máximo valor de pertinência para qualquer conjunto *fuzzy* é a unidade, são definidas as seguintes relações (Pedrycz e Gomide, 2007):

1) **Conjunção *fuzzy*:**

$$f_t: [0,1]^2 \rightarrow [0,1]; \quad f_t\big(A(x), B(y)\big) = A(x)tB(y), \forall(x,y)\ X \times Y$$

- $f_c\big(A(x), B(y)\big) = A(x) \wedge B(y)$ $\quad$ (***Mamdani***)
- $f_p\big(A(x), B(y)\big) = A(x).B(y)$ $\quad$ (***Larsen***)

2) **Disjunção *fuzzy*:**

$$f_s: [0,1]^2 \rightarrow [0,1]; \quad f_s\big(A(x), B(y)\big) = A(x)sB(y), \forall(x,y)\ X \times Y$$

3) **Implicação *fuzzy*:**

$$f_i: [0,1]^2 \rightarrow [0,1] \quad tal\ que, \quad \forall(x,y)\ X \times Y:$$

- Monotônica no segundo: $B(y_1) \leq B(y_2) \rightarrow f_i\big(A(x), B(y_1)\big) \leq f_i\big(A(x), B(y_2)\big)$
- Dominância da falsidade: $f_i(0, B(y)) = 1$
- Neutralidade da verdade: $f_i\big(1, B(y)\big) = B(y)$

Em geral, com relação à implicação *fuzzy*, pode-se acrescentar as seguintes propriedades:

- Monotônica no primeiro: $A(x_1) \leq A(x_2) \rightarrow f_i\big(A(x_1), B(y)\big) \geq f_i\big(A(x_2), B(y)\big)$

- Troca: $f_i\left(A(\right.$

- P: Se X é A Então Y é B

  (X,Y) é AxB

  AxB = A t B

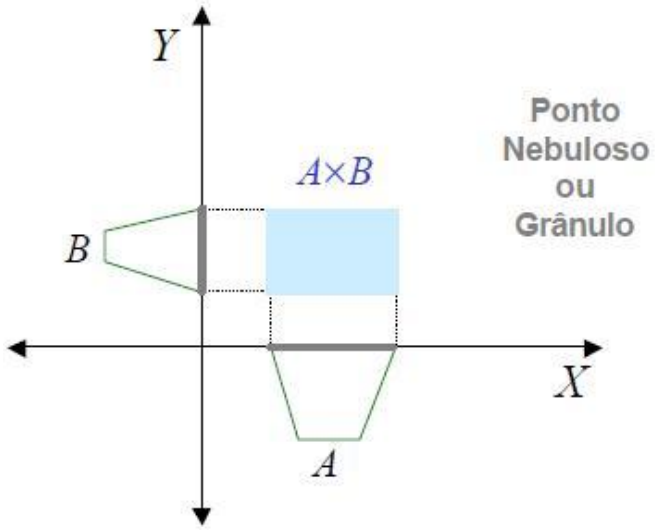

Figura A.7 – Representação *fuzzy* para uma simples regra com relação $A \times B$

- R: Se X é A_1 Então Y é B_1

  Se X é A_2 Então Y é B_2

  ...

  Se X é A_N Então Y é B_N

Onde também cumpre-se: (X,Y) é $(\sum_{i=1}^{N} A_i \times B_i) \equiv (X,Y)$ é $\sum_{i=1}^{N} R_i \equiv (X,Y)$ é R

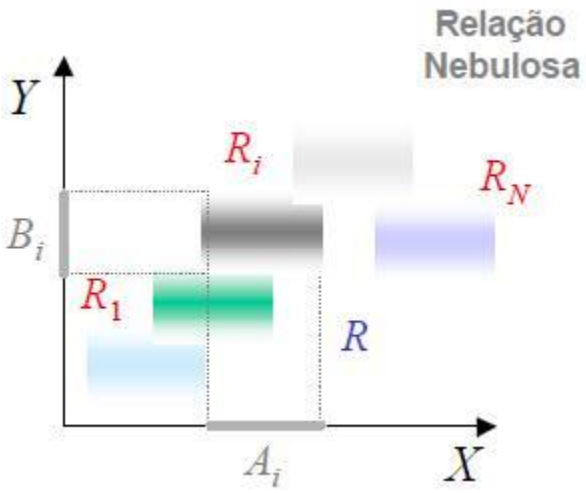

Figura A.8 – Representação *fuzzy* para um sistema de regras com relação $A_i \times B_i$

Subseqüentemente algumas representações de grafos de uma função comum em contraste com um grafo *fuzzy* de uma função, são apresentadas na Figura A.9 e na Figura A.10:

- Função: $y = f(x); \quad f: X \to Y$
  Grafo: $F = \{(x, y)/y = f(x),\ x \in X,\ y \in Y\}$

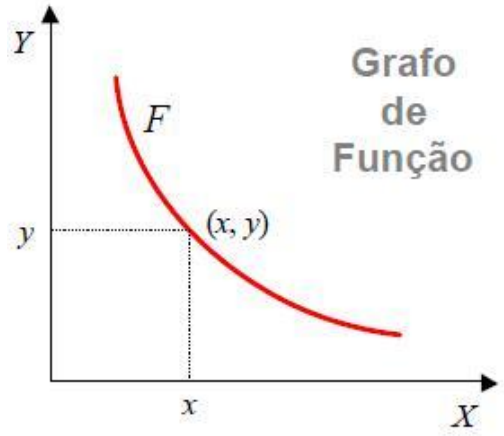

Figura A.9 – Representação de uma função bidimensional

- Função: $y = f(x); \quad f: X \to Y$
  Grafo *fuzzy*: $F^* = representação\ aproximada, granular\ da\ função\ f$
  Onde: $F^*(x, y) = S_{i=1}^{N}[A_i(x) t B_i(y)], \quad \forall (x, y) \in X \times Y$

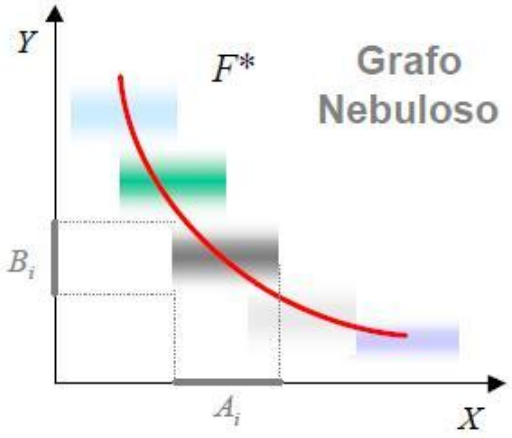

Figura A.10 – Representação granular ou grafo *fuzzy* da função f

A inferência e raciocínio aproximado podem ser expressos graficamente, de uma maneira ilustrativa, com segue na Figura A.11, Figura A.12, Figura A.13 e Figura A.14:

- $x = a$

  $y = f(x)$

  $\rightarrow y = b$

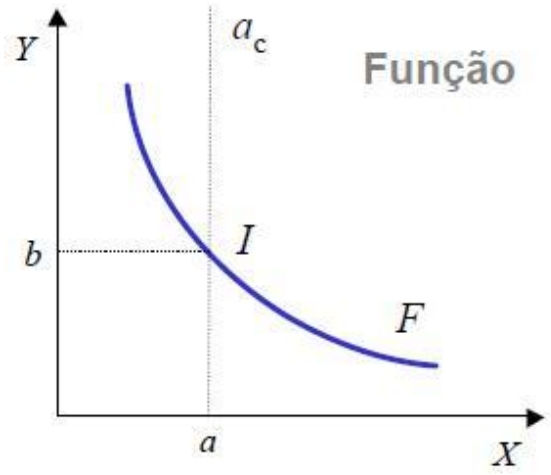

Figura A.11 – Projeção gráfica *fuzzy* de um ponto sobre *F*

- x é A

  (x, y) é F

  $\rightarrow y$ é B

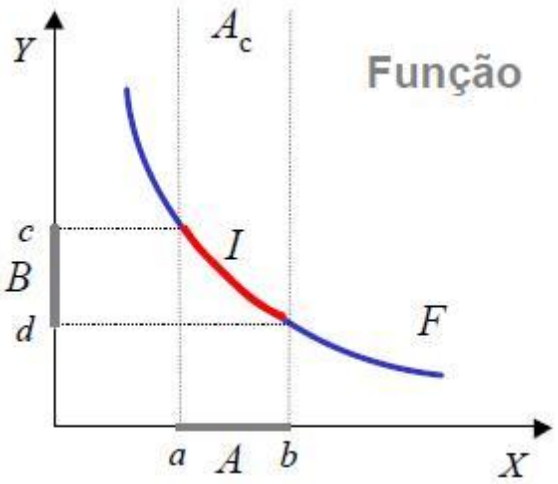

Figura A.12 – Projeção gráfica *fuzzy* de um segmento de linha sobre *F*

- x é A
  (x, y) é F
  $\rightarrow y$ é B

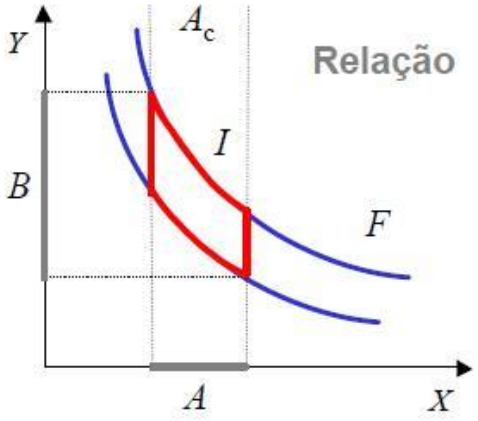

Figura A.13 – Representação gráfica *fuzzy* da projeção de *A* sobre *F* originando *B*

- X é A
  (X, Y) é F^*
  $\rightarrow Y$ é B

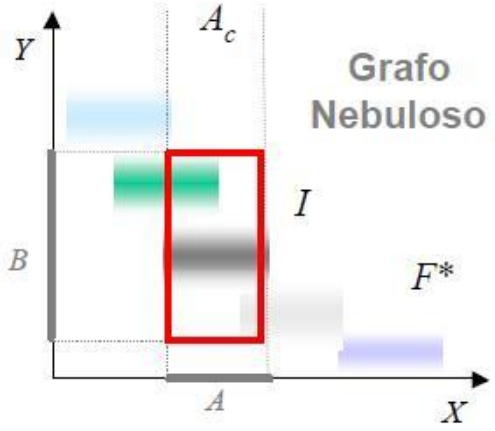

Figura A.14 – Representação gráfica *fuzzy* da projeção de *A* sobre F^* originando *B*

Como abordou-se linhas atrás, de uma maneira sintetizada, foram estabelecidas as relações, equações e alguns grafos de raciocínio aproximado que governam o fundamento básico matemático dos conjuntos *fuzzy* e lógica *fuzzy*. Cabe ressaltar que, o objetivo desta seção não é aprofundar completamente em tais conceitos e definições complexas e estritamente

matemáticas sobre a teoria dos conjuntos *fuzzy*, senão, oferecer uma visão geral para a compreensão desta dissertação de mestrado. Com o intuito de aprofundar sobre tais conceitos e possuir um maior domínio do tema podem ser consultadas as seguintes fontes de referência (Kosko 1997, Cox 1994, Tsoukalas e Uhrig 1997).

A continuação será abordada principalmente as conjunções *fuzzy* que foram mencionadas linhas atrás, tais como Larsen e Mamdani. Tanto Larsen quanto Mamdani oferecem ferramentas computacionais para o raciocínio com relação à computação de regras *fuzzy* em algum ambiente de programação, como é o caso da ferramenta computacional *FuzzyJess,* que foi utilizado para o desenvolvimento computacional desta metodologia.

A.5.1 Desempenhando inferência *fuzzy*

Em palavras gerais, inferência *fuzzy* pode ser definida como um processo de mapeamento desde uma entrada dada para uma saída, usando a teoria dos conjuntos *fuzzy*.

O processo de inferência, em termos gerais, é desempenhado em quatro passos: codificação (*fuzzification*) de todas as varáveis de entrada, avaliação da regra (computação entre todos os antecedentes e entradas), contribuição global de todas as regras executadas independentemente e finalmente descodificação (*defuzzification*).

Como foi exposto anteriormente, inferência *fuzzy* é desempenhada em quatro passos, mas a aplicação de tal técnica subjaz em como os conjuntos *fuzzy* de saída são originados através do casamento de todos os pares (entrada/conseqüente) de todas as regras executadas uma por vez.

Dependendo do *software* usado para a implementação do sistema experto *fuzzy*, deve-se ter em conta as suas propriedades computacionais como, por exemplo, tipos de conjuntos *fuzzy* que podem ser implementados (funções triangulares, trapezoidais, gaussianas, etc), tipos de inferência *fuzzy* que podem ser implementados (inferência estilo Mamdani, Larsen, Tsukamoto, SAM, TSK, etc), tipos de técnicas de descodificação que podem ser implementadas (método do COG, COA, meia de pesos, meia de máximos) e deve-se fazer uma análise exaustiva com relação as suas limitações computacionais de implementação.

Para entender completamente o tipo de raciocínio que é executado quando é usado qualquer destas duas técnicas *fuzzy* (Larsen ou Mamdani), é necessário exemplificar tais métodos para um melhor entendimento. Pata atingir tal objetivo vai ser considerado um conjunto de "n" regras *fuzzy*, cada uma delas contendo "m" antecedentes e "um conseqüente" e para computar ambas técnicas, serão inseridas à base de regras "m" entradas codificadas. Para simplificar um pouco o cálculo, cada regra possuirá uma só saída (um conseqüente). Onde:

- Número de regras *fuzzy*: "n", onde R_i é usada para denotar a "i-éssima regra do sistema".
- O conjunto de regras a serem computadas possuem a seguinte estrutura:

$$R_1: \quad Se\ X_1 \text{ é } A_1\ e\ X_2 \text{ é } A_2\ e\ X_3 \text{ é } A_3 \ldots X_j \text{ é } A_j \ldots X_m \text{ é } A_m\ \ Então\ \ Y \text{ é } W_1$$

$$R_2: \quad Se\ X_1 \text{ é } B_1\ e\ X_2 \text{ é } B_2\ e\ X_3 \text{ é } B_3 \ldots X_j \text{ é } B_j \ldots X_m \text{ é } B_m\ \ Então\ \ Y \text{ é } W_2$$

...

$$R_i: \quad Se\ X_1 \text{ é } F_1\ e\ X_2 \text{ é } F_2\ e\ X_3 \text{ é } F_3 \ldots X_j \text{ é } F_j \ldots X_m \text{ é } F_m\ \ Então\ \ Y \text{ é } W_i$$

...

$$R_n: \quad Se\ X_1 \text{ é } G_1\ e\ X_2 \text{ é } G_2\ e\ X_3 \text{ é } G$$

A.5.1.1 Inferência *fuzzy* estilo Mamdani

A Figura A.15 mostra como é computada o conjunto de regras anteriormente exposto.

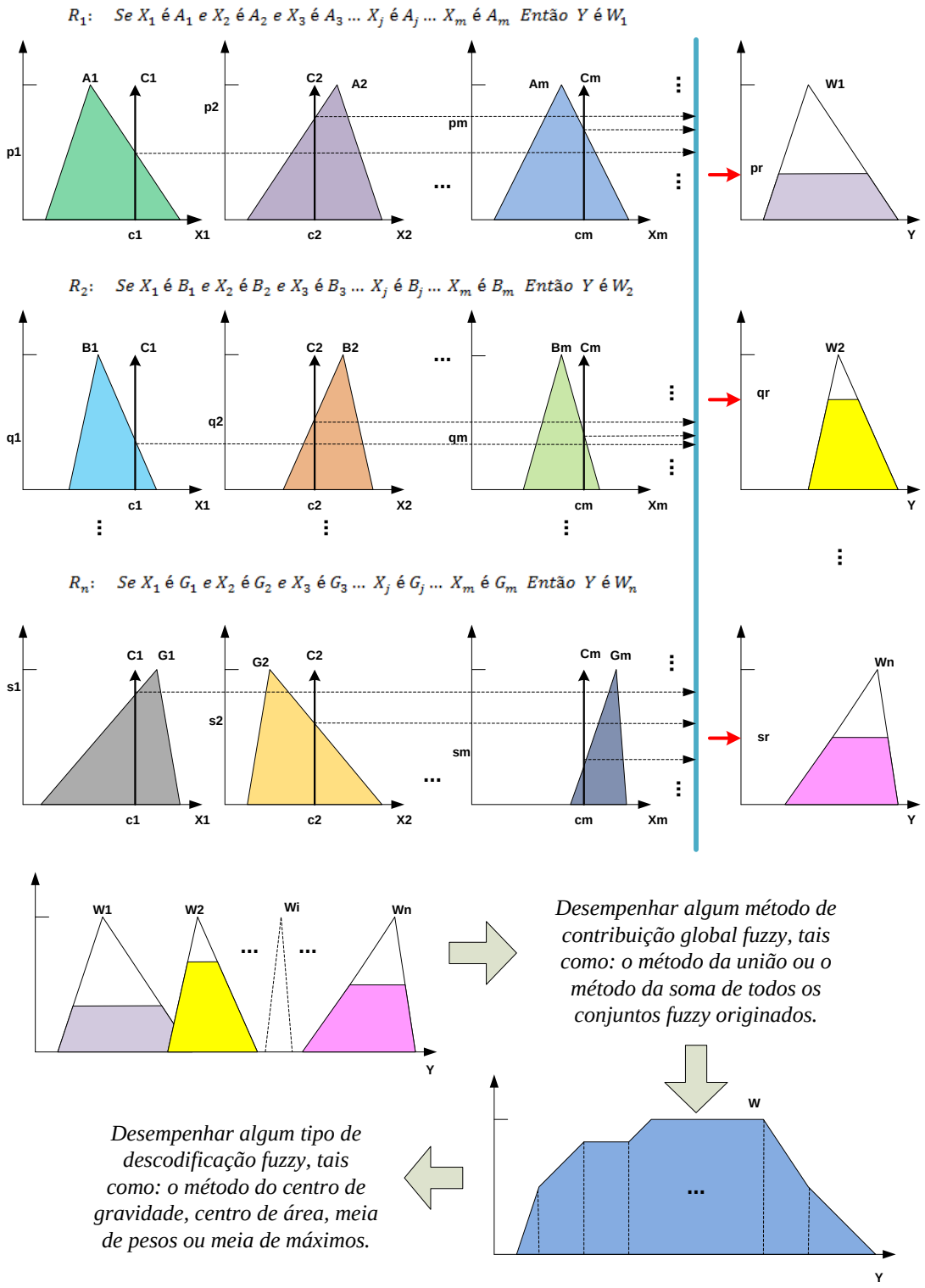

Figura A.15 – Processo de computação de um sistema de "m" regras *fuzzy* aplicando o método de inferência estilo Mamdani

Um conjunto de observações é necessário para o entendimento das partes menos obvias da Figura A.15:

(a) Com relação ao valor "p_r", este valor é escolhido em cada de uma maneira independente em cada regra, que dizer que nenhuma regra "R_i" será afetada por outras regras como "R_{i-1}" ou "R_{i+1}".

O processo de seleção desse valor em cada regra é desempenhado através de selecionar o valor "mínimo" de cada par de interseções (antecedente/conseqüente) como produto do casamento de todos os antecedentes com os seus respectivos valores de entrada. Por tanto, " p_r" $= min\{p_1, p_2, \ldots, p_j, \ldots p_m\}$. Graficamente este processo poderia complicar-se se as entradas são conjuntos *fuzzy* ao invés de funções *singletons* como mostrado na Figura A.16.

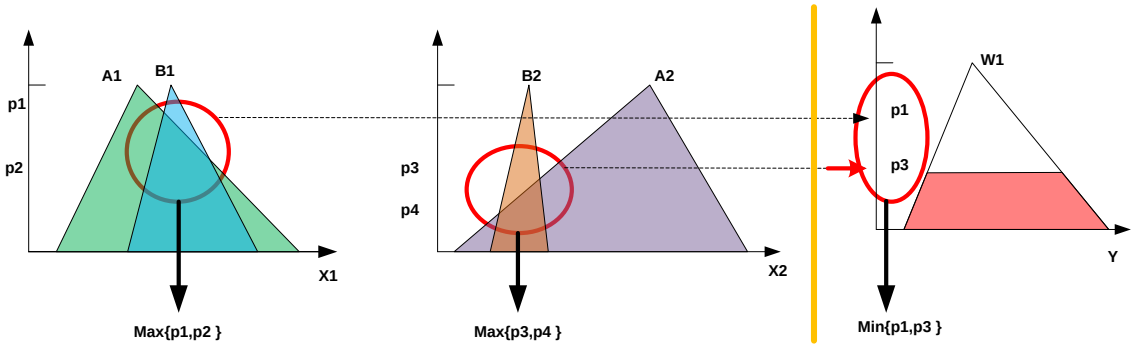

Figura A.16 – Computação de regras estilo Mamdani quando as entradas são conjuntos *fuzzy*

Para simplificar o raciocínio, na Figura A.15, é apresentada uma só regra com dois antecedentes (conjuntos *fuzzy*: "A_1" e "A_2"), um conseqüente (conjunto *fuzzy*: "W_1") e duas entradas (conjuntos *fuzzy*: "B_1" e "B_2"). Uma observação neste ponto subjaz em que "p_3" pode ser calculado de várias maneiras e, dependendo do *software* que esteja usando-se, para esta pesquisa NRC FuzzyJ Toolkit, o qual implementa a operação "mínimo", "produto" e alternativamente a operação "compensatório" (este último não será tratado nesta dissertação).

Habitualmente e na maioria de aplicações de sistemas *fuzzy* é usada a operação "mínimo" para desempenhar inferência estilo Mamdani e "produto" para desempenar inferência estilo Larsen, e "compensatório" para sistemas *fuzzy* no qual é desejado um valor maior ao obtido aplicando qualquer das operações "mínimo" ou "máximo".

(b) Depois de ser obtido a resultante para cada regra ("$p_r, q_r, \dots, s_r$") estes valores são aplicado à conclusão ou conseqüente de cada regra. Este processo, no caso de Mamdani, estes valores "cortam" aos seus respectivos conseqüentes nesse valor em particular e é desse jeito que se geram as saídas para cara regra independentemente ("$W_1, W_2, \dots, W_n$").

(c) Com relação à contribuição global do sistema de regras, esta pode ser realizada de duas formas: "união"ou "suma". O NRC FuzzyJ Toolkit implementa estas duas formas. É claro que dependendo da aplicação uma é mais apropriada do que a outra. Quando é desempenhado "união *fuzzy*" cada valor de pertinência desse conjunto é originado como resultado de escolher o "máximo valor de pertinência" nesse ponto "y" de ambos conjuntos *fuzzy* consecutivos que intersectam-se.

Para o caso da "soma *fuzzy*" esta origina os seus valores de pertinência como resultado de "somar ambos valores de pertinência" em ambos conjuntos *fuzzy* consecutivos que intersectam-se no correspondente valor de "y".

(d) Com relação à descodificação do conjunto *fuzzy* resultante, existem várias técnicas que o NRC FuzzyJ Toolkit implementa (COG, COA, etc). Dependendo das características geométricas do conjunto *fuzzy* resultante uma técnica é mais adequada que a outra.

A.5.1.2 Inferência *fuzzy* estilo Larsen

Seguindo o mesmo raciocínio aplicado no item A.5.1.1, é apresentada a Figura A.17, para ser analisado este método de uma maneira mais intuitiva e dinâmica.

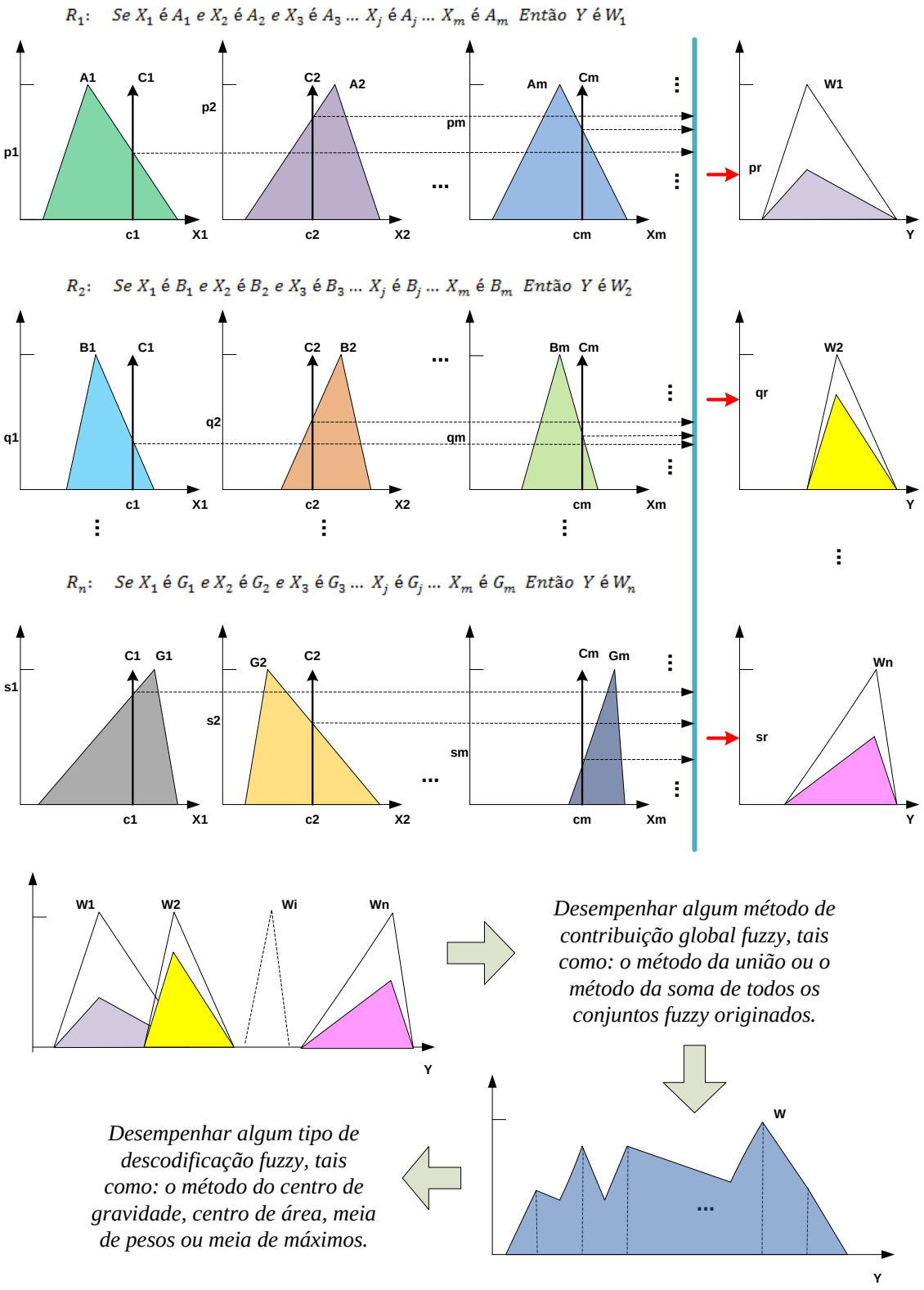

Figura A.17 – Processo de computação de um sistema de "m" regras *fuzzy* aplicando o método de inferência estilo Larsen

Um conjunto de observações é necessária para o entendimento das partes menos obvias da Figura A.17:

(a) Com relação ao valor "p_r", este valor é escolhido em cada de uma maneira independente em cada regra, que dizer que nenhuma regra "R_i" será afetada por outras regras como "R_{i-1}" ou "R_{i+1}". O processo de seleção desse valor em cada regra é desempenhada através de selecionar o "produto" de cada par de interseções (antecedente/conseqüente) como conseqüência do casamento de todos os antecedentes com os seus respectivos valores de entrada. Por tanto, "$\mathrm{p_r}$" $= produto\{p_1 \times p_2 \times \ldots \times p_j \times \ldots \times p_m\}$.

Analogamente à Figura A.16 este processo poderia complicar-se se as entradas são conjuntos *fuzzy* ao invés de funções *singletons.* No NRC FuzzyJ Toolkit esta operação pode ser modificada, que dizer, Mamdani pode desempenhar a operação "produto" para o cálculo de "p_r" e Larsen pode desempenhar a operação "mínimo" para o cálculo de "p_r".

(b) Depois de ser obtido a resultante para cada regra ("$p_r, q_r, \ldots, s_r$") estes valores são aplicado à conclusão ou conseqüente de cada regra. Este processo, no caso de Larsen, estes valores "escalam" aos seus respectivos conseqüentes nesse valor em particular e é desse jeito que se geram as saídas para cara regra independentemente ("$W_1, W_2, \ldots, W_n$"). O termo escalar, na FST, simplesmente significa redimensionar esse conjunto *fuzzy* no valor desejado "p_r" (esse valor pode variar de 0 a 1), em outras significa ajustar a geometria desse conjunto *fuzzy* ao valor "p_r". O valor máximo de pertinência desse conjunto escalado é o valor "p_r".

(c) Com relação à contribuição global e à fase de descodificação, aplica-se o mesmo raciocínio para o método de inferência Larsen. Uma observação com relação à "soma de valores de pertinência" para desempenhar contribuição global através da soma, é que os valores resultante de pertinência podem ser maiores que 1, isto é permitido e é uma acepção.

Em algumas abordagens de sistemas *fuzzy* isto é desejado, como por exemplo quando se deseja implementar um conjunto de regras *fuzzy* do tipo SAM; quando é aplicado o método SAM, é desejado que seja desempenhado inferência fuzzy estilo Mamdani, combinação das (entradas/antecedentes) usando o operador "produto", contribuição global usando o operador "soma" e finalmente para obter as saídas exatas aplicar o método de descodificação do COG. Outro exemplo onde é necessário desempenhar operador "soma" é feito quando aplicado a inferência estilo Tsukamoto.

A.6 DESCODIFICAÇÃO DE REGRAS *FUZZY*

O processo de descodificação (*defuzzification*) é complementário ao processo de codificação (*fuzzification*). Dado alguns conjuntos *fuzzy*, agora o objetivo é desenvolver uma representação numérica. Esta transformação é qualificada como um mecanismo de descodificação. Existem dois caminhos que são abordados (Pedrycz e Gomide, 2007):

(a) Decodificação realizada baseando-se de um único conjunto *fuzzy*. Esta via parece ser a mais vigorosamente discutida com várias técnicas.
(b) Descodificação realizada baseando-se em uma família de conjuntos *fuzzy* e níveis de suas ativações.

Um deve estar completamente consciente de que o processo de descodificação pode ser efetuado de diferentes formas. Isto é porque as funções características são simplesmente funções contínuas com um número infinito de valores característicos ou quando lidamos com vetores espaciais finitos de graus característicos, digamos $[0,1]^n$. O associamento de um único valor numérico com um vetor de números não pode ser feito de uma única forma (Pedrycz e Gomide, 2007).

As seguintes formulas foram adaptadas por Pedrycz e Gomide (2007) de Runkler e Glesner (1993) e Wierman (1997).

Considere certo conjunto *fuzzy* com função característica *B(x)*. Denota-se a transformação de B para algum valor numérico representativo por $\hat{x} = D(B)$. Os métodos mais comuns encontrados sistemas *fuzzy* incluem os seguintes:

A.6.1 Técnica da média de máximos

Com esta técnica, determinam-se os argumentos de $\boldsymbol{X}$ de maneira que a função característica atinge seus valores máximos. Denotam-se eles por $\tilde{x}_1 + \tilde{x}_2 + \cdots + \tilde{x}_p$. O resultado da decodificação é considerado como a média desses valores, que é representada pela Equação A.5:

$$\hat{x} = \frac{\tilde{x}_1 + \tilde{x}_2 + \cdots + \tilde{x}_p}{p} \tag{A.5}$$

No caso de encontrar só um valor máximo de *B*, então este método seria uma escolha altamente intuitiva. Existem várias possíveis variações desta técnica, tal como, a primeira do máximo e a última do máximo, onde seleciona-se o valor modal particular da função característica. Esta técnica possui alguns inconvenientes no caso que existisse diferentes valores de "*x*" no qual se atinge o valor máximo para a função característica ou lugares onde o máximo valor de pertinência é atingido num simples valor de "*x*". O anteriormente dito pode ser contemplado na Figura A.18.

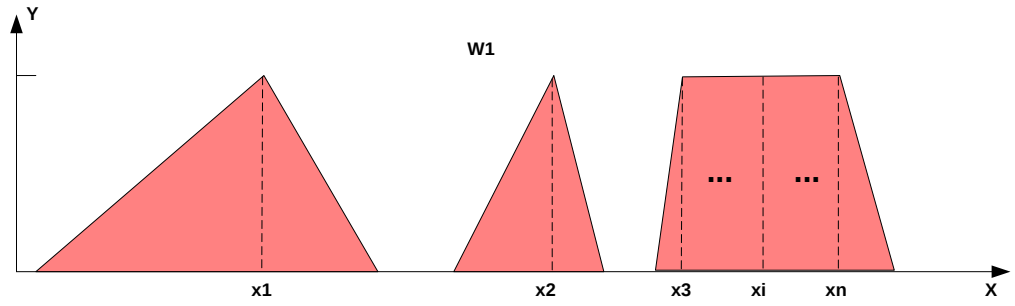

Figura A.18 – W_1 apresenta o conjunto *fuzzy* resultante que possui vários pontos nos quais atinge o seu máximo valor de pertinência

Com relação à Figura A.18, o conjunto *fuzzy* "W_1" possui os seus valores máximo de pertinência em "X_1", "X_2" e na faixa de valores: "$X_3, \ldots, X_i, \ldots, X_n$", isto contrasta com a idéia acima exposta.

A.6.2 Técnica do centro de área (COA)

Neste ponto, encontra-se uma posição de $\hat{x}$ tal que resulta em as áreas iguais embaixo da função característica posicionado na parte esquerda e direita desde esse valor representativo. Em outras palavras, esse valor é governado pela Equação A.6:

$$\int_{-\infty}^{\hat{x}} B(x)dx = \int_{\hat{x}}^{\infty} B(x)dx \tag{A.6}$$

Assume-se que a função característica pode ser integrada.

Esta técnica também possui uma acepção o qual pode ser visto na Figura A.19.

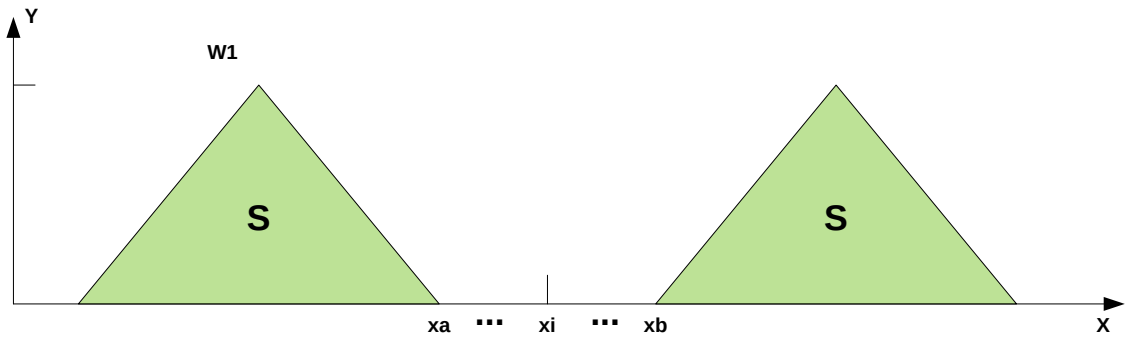

Figura A.19 - W_1 apresenta o conjunto *fuzzy* resultante formando duas áreas simétricas os quais limitam pontos infinitos simétricos com relação a "S"

Com relação à Figura A.19, entre as duas áreas "S" existem infinitos valores no quais ambos lados possuem a mesma área, quer dizer, qualquer ponto maior que "x_a" e menos que "x_b" pode ser resultado ao aplicar este tipo de técnica, mas, pelo geral é procurado pela técnica o valor intermédio entre esse conjunto infinito de valores, a dizer, $x_i = \frac{x_a + x_b}{2}$.

Aqui seguindo um simples raciocínio, basta com encontrar um caso que não se ajuste à implementação computacional ou aos resultados esperados pelo sistema para não ser considerado o método em estudo.

A.6.3 Técnica do centro de gravidade (COG)

Aqui o resultado da decodificação é obtido como segue (ver Equação A.7):

$$\hat{x} = \frac{\int B(x)x dx}{\int B(x) dx} \tag{A.7}$$

Assume-se que a integral de acima existe.

Graficamente pode ser apresentado a técnica do COG como é mostrado na Figura A.20, com o objetivo de clarificar como é computado a descodificação do conjunto *fuzzy* resultante para encontrar o valor numérico mais representativo.

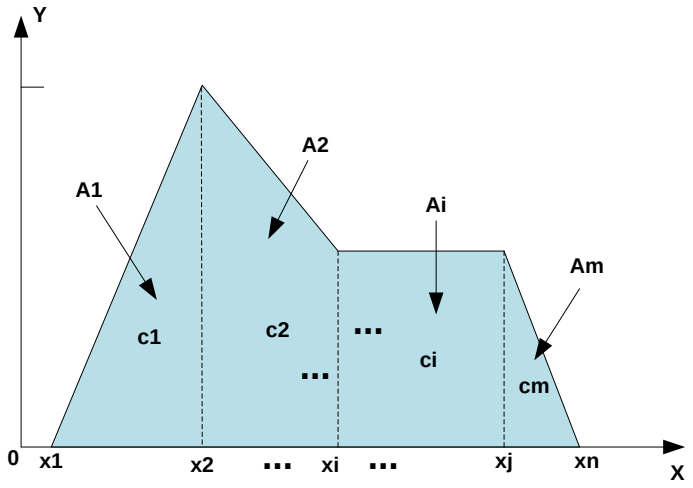

Figura A.20 – Conjunto *fuzzy* resultante genérico

Como mostrado na Figura A.20, o conjunto *fuzzy* resultante é dividido em outros conjuntos de formas conhecidas, tais como, triângulos retângulos, trapézios, quadrados, etc. O objetivo desta divisão é calcular as suas áreas ("A_1", "A_2", ... , "A_i", ... ,"A_m") e os seus centros de gravidade ("c_1", "c_2", ... , "c_i", ... , "c_m") como um primeiro passo, depois disso, é aplicado a Equação A.7 para encontrar o valor numérico representativo desse conjunto *fuzzy*. As Equações A.7 e A.8 são idênticas.

$$\hat{x} = \frac{\sum_{i=1}^{m} c_i . A_i}{\sum_{i=1}^{n} Ai} \quad \text{(A.8)}$$

Onde:

"m" é o número de subdivisões ou áreas embaixo do conjunto *fuzzy*.

"$\hat{x}$" é o valor numérico como produto da descodificação.

Está técnica só possui um pequeno inconveniente só quando é usado para descodificar funções *singleton*, já que, tais funções não possuem uma área por baixo e por tanto no cálculo do valor numérico originária uma divisão entre zero o qual é indeterminado. Na maioria de aplicações de sistemas *fuzzy*, está técnica é a mais adequada para encontrar o valor numérico mais apropriado para representar esse conjunto *fuzzy* a descodificar.

Dado um grande número de alternativas em processos de descodificação, poderia ser de utilidade estabelecer alguns critérios que podem ser aceitos e que qualquer estratégia de descodificação poderia satisfazer sem problemas.

O exemplo de algumas estratégias axiomáticas tem sido oferecido por Runkler e Glesner (1993) adaptado por Pedrycz e Gomide (2007); nessas estratégias axiomáticas os autores propuseram uma série de requerimentos que são organizados em vários grupos, a dizer (a) restrições básicas com relação a assuntos relacionados com as formas das funções características (conjuntos e *singletons*) e monoticidade, (b) requerimentos motivados graficamente incluindo simetria, translação, escalamento e *offset*, (c) restrições motivadas pelo uso de operações lógicas e modificadores lingüísticos (dilatação e concentração) e (d) requerimentos específicos a algum domínio de aplicação.

APÊNDICE B – FERRAMENTAS COMPUTACIONAIS

Neste apêndice se apresentará uma guia rápida sobre os conceitos, algoritmos, sintaxe, motor de inferência e métodos de raciocínio relacionados com as ferramentas computacionais: *NRC FuzzyJ Toolkit*, *FuzzyJess* e Jess, usadas para a construção da base de regras de produção usando funções de pertinência *fuzzy*. Para um estudo mais detalhado das diferentes sintaxes dos construtores, classes, métodos, deve-se procurar o API para cada das ferramentas computacionais mencionadas, os quais podem ser encontrados nas seguintes referências, contidas neste trabalho de pesquisa: Friedman-Hill (2008) e Orchard, (2006).

B.1 O JESS *(Java Expert System Shell)*

O Jess procura de um jeito amigável, a integração e criação de sistemas especialistas (sistemas expertos) em aplicações Java dada a crescente utilização desta linguagem especialmente em aplicações *web*.

A criação de sistemas expertos (SE), com Jess, é feita com uma sintaxe muito similar à do Sistema de Produção Integrado da Linguagem C (CLIPS por suas siglas em inglês), porém a integração do CLIPS (NASA, 1996) com a linguagem C é substituída pelo uso de Java, o que pode ocasionar algumas dificuldades na migração de aplicações de um para outro *shell*.

O Jess (Friedman-Hill, 2003), possui a desvantagem inerente, em aplicações Java, de ser pesado em termos de execução computacional, o que não chega a ser um problema crítico, quando executado em estações preparadas para suportar aplicações Java.

O Jess utiliza o algoritmo *Rete* (Charles, 1982) na sua máquina de inferência. Este algoritmo é caracterizado por organizar as regras numa árvore, de modo que sejam divididas em padrões, assim, regras semelhantes percorrerão o mesmo ramo da árvore enquanto possuírem cláusulas iguais. Tal organização é mais versátil em termos de memória, mas proporciona grandes melhoras na velocidade de verificação das regras. O motor de inferências do Jess permite tanto o encadeamento regressivo quanto progressivo.

A linguagem Jess é uma linguagem de programação de propósito geral e pode acessar diretamente a todas as classes e bibliotecas de Java. Por esta razão, Jess é muito usado como ambiente de desenvolvimento rápido de aplicações e *scripts*. Enquanto o código Java tem que ser compilado antes de rodar, uma linha de código do Jess é executada imediatamente após ser digitado. Isto permite que os usuários experimentem as API *(Application Programming Interface)* de Java interativamente e desenvolvam programas.

B.1.1 Comandos principais do Jess

Abaixo apresentamos alguns comandos principais do Jess, para inicialização execução e programação:

- Iniciar o Jess: No *prompt* do MS-DOS, diretório Jess71p2, digitar o seguinte comando: *Java -classpath jess.jar jess.Main*
- Ler um *script*: (batch NomeDoPrograma.clp)
- Carregar na memória: (reset)
- Executar as regras: (run)
- Comentários: Símbolo de ponto e vírgula ";"
- Encerrar o Jess: (exit)

B.1.2 Linguagem de regras Jess

A maneira mais simplificada de desenvolvimento na plataforma Jess é realizada pela escrita de regras na *Jess Rule Engine*, qual é baseada na linguagem Lisp (McCarthy, 1979). Isso permite ao Jess ter uma alta expressividade, com a possibilidade de escrita de relações lógicas complexas com pouco código.

B.1.3 Premissas básicas

A apresentação da linguagem Jess segue uma notação informal para apresentação da gramática. Na grande maioria das vezes *strings* são apresentadas entre os símbolos < *string* > com alguns dados. "Coisas" são inseridas entre [*coisas*] e são opcionais. Seguindo o princípio de expressões regulares, tudo que parece com + na descrição da linguagem pode

aparecer uma ou mais vezes. Da mesma forma, a presença de * indica que algo pode aparecer uma ou mais vezes no código. No geral, a entrada do Jess é de formato livre.

B.1.4 Símbolos

Os símbolos são o núcleo da linguagem Jess e se parecem com identificadores de linguagens de programação. Um símbolo pode conter qualquer caractere e não pode começar com dígito podendo inclusive iniciar com marcadores de pontuação, por exemplo, "#".

Os símbolos também são sensíveis ao contexto: teste, Teste e TESTE são símbolos diferentes. Existem três símbolos especiais que o Jess interpreta com significados internamente definidos: *nil* representa *null* da linguagem Java, *TRUE* e *FALSE* representam valores booleanos. Exemplos de símbolos poderiam ser: *foo, first-value, context#type*.

B.1.5 Números

Os valores numéricos em Jess seguem o mesmo padrão da linguagem Java. Exemplos de números poderiam ser: 3, 4, 562L, 6.0E3.

B.1.6 *Strings*

As cadeias de caracteres em Jess são formadas por caracteres entre o operador "*string*". Alguns exemplos de *strings*: *"foo", "Hello, World"*.

B.1.7 Comentários

Os comentários são formados pelo operador ";", de comentário de linha, e entre operadores / * ... * / representado comentários de bloco.

B.1.8 Listas

A linguagem Jess segue um formato funcional. Neste caso, tudo acaba sendo uma chamada de função. A estruturação do código é formada por tuplas com valores informados em

formato pré-ordem. Assim, o primeiro elemento da tupla (cabeça da tupla) é a função e os demais itens são os dados, parâmetros ou outras funções.

Exemplos de listas poderiam ser: (+ 3 4), (a b c), ( ), *(deftemplate myTemplate (slot name) (slot value))*. As chamadas de funções são realizadas pelo motor de inferência naturalemente, o qual entende que o elemento "cabeça" é uma função e a invoca passando os demais parâmetros.

B.1.9 Variáveis

As variáveis em Jess são declaradas no seu primeiro uso por meio da função *bind*, que cria e atribui um valor a uma variável. A declaração de uma variável é realizada sempre que um símbolo inicia com o operador "?".

Um exemplo de declaração de variável: (*bind* ?x 2). Existe ainda a possibilidade de declaração de variável global. Neste caso deve-se utilizar a seguinte sintaxe: *"(defglobal [?<global-name> = <value>]+)*". Onde: *?<global-name>* segue o formato: ?*<*var-name*>* e *<var-name>* é o símbolo que designa o nome da variável.

B.1.10 Controle de fluxo

Como nas linguagens de programação usuais, Jess também apresenta controle de fluxo de execução. O controle de fluxo é dado por funções que são: *while*, *for*, *foreach*, *break*, *if* e *try*. Embora que o Jess ofereça estas funcionalidades, o interessante é evitar sua utilização para poder obter um grau mais elevado de não determinismo (heurístico).

B.1.11 Definindo funções

A ferramenta Jess permite que o usuário defina suas próprias funções. A sintaxe é a seguinte:

(deffunction <function-name> [<doc-comment>] (<parameter>) <expr>* [<return-specifier>]).*

Onde:

- <*functio-name*>: Deve ser um símbolo.
- <*parameter*>: Cada parâmetro deve ser uma variável.
- <*doc-comment*>: É uma string opcional que descreve o propósito da função.
- <*expr*>: São expressões e podem existir várias.
- <*return-specifier*>: É um valor opcional de retorno da função.

Um exemplo simples seria algo como: *(deffunction max (?a ?b) (if (?a ?b) then (return ?a) else (return ?b)))*

B.1.12 Memória de trabalho

O funcionamento do Jess é baseado em fatos e regras. Fatos são coleções de conhecimento armazenados na memória de trabalho. A inferência é realizada somente quando são inseridos, apagados, ou alterados as dados da memória de trabalho.

Existem dois tipos de fatos em Jess: os fatos "puros", criados e definidos pela própria feramente e os fatos "mascarados" (*shadow*), fornecidos pela plataforma Java ou criados pelo usuário como classes Java. Esses são mapeados para Jess atuando como "pontes" permitindo ao Jess ter conhecimento de informações que não se encontram na memória de trabalho.

Cada fato obrigatoriamente tem um *template*, o qual é definido como uma tupla de informações. A sintaxe é exibida a seguir:

```
(deftemplate template-name
        [extends template-name]
        ["Documentation comment"]
        [(declare       (slot-specific TRUE | FALSE)
                        (backchain-reactive TRUE | FALSE)
                        (from-class class name)
                        (include-variables TRUE | FALSE)
                        (ordered TRUE | FALSE))]
```

(slot | multislot slot-name

([(type ANY | INTEGER | FLOAT |

NUMBER | SYMBOL | STRING |

LEXEME | OBJECT | LONG)]

[(default default value)]

[(default-dynamic expression)]

[(allowed-values expression+)]))*

Onde:

Toda declaração de *template* tem um nome, uma documentação opcional, uma cláusula opcional "*extends*", uma lista opcional de declarações, uma lista de zero ou mais *slots* de descrições.

O <*template-name*> é a "cabeça" da tupla e identifica o tipo do fato na memória de trabalho. Cada *template* pode ter um <*template-father*> informado pela cláusula <*extends*>.

Os *templates* possuem pelo menos um *slot* de descrição contendo informações para um fato. Cada *slot* pode opcionalmente ter um qualificador de tipo ou valor padrão. Um qualificador de *slot* pode ter os tipos de dados aceitos pelo Jess: *ANY, INTEGER, FLOAT, NUMBER, SYMBOL, STRING, LEXEME e OBJECT*. Um exemplo simples desta definição se dá a continuação:

(deftemplate carro

(slot fabricante)

(slot modelo)

(slot cor (default prata))

(slot ano (type INTEGER)))

B.1.13 Fatos

Os fatos podem ser carregados/removidos da memória de trabalho com os comandos: *assert*, e *retract*. Uma modificação em um slot pode ser realizada pelo comando *modify*. Um pequeno exemplo de esta definição pode ser visualizado abaixo:

(deffacts carros

(carro (fabricante GM) (modelo Corsa) (cor azul))
(carro (fabricante BMW) (modelo X5) (cor vermelho))
(carro (fabricante VW) (modelo Fusca) (cor verde))
(carro (fabricante VW) (modelo Gol)))

B.1.14 Definindo regras

As regras são utilizadas pelo Jess para reagir às alterações na memória de trabalho. Essas reações são mecanismos de ação que levam à adição, remoção ou alterações de fatos na mesma memória de trabalho. Este processo constante converge para o objetivo do programa escrito em Jess.

A sintaxe de definição de regras é a seguinte:

(defrule rule-name

["rule-description"]
<match-definition>
=>
<action-definition>)

Onde:

- <*rule-name*>: Nome da regra.
- <*rule-description*>: Documentação opcional da regra.
- <*match-definition*>: Definição de casamento de fatos.

- <*action-definition*>: Definição de medidas a serem tomadas caso aconteça um casamento de fatos.

Um pequeno exemplo desta definição pode ser visualizado abaixo:

```
(defrule encontra-verde-ou-prata
        ?c <- (carro (cor ?x&:(or (= ?x verde) (= ?x prata))))
        =>
        (printout t "Carro encontrado >  Fabricante:" ?c.fabricante
                ", Modelo:" ?c.modelo ", Cor:" ?c.cor crlf))
```

B.2 O PACOTE DE FERRAMENTAS *FUZZY*

O *NRC FuzzyJ Toolkit*, é um API Java para representação e manipulação de informação *fuzzy* (permite a exploração de idéias e razoamento *fuzzy* em um palco Java), e está composto por dois pacotes principais: **nrc.fuzzy.*** e o pacote **nrc.fuzzy.jess.* (FuzzyJess)**. O primeiro pacote (nrc.fuzzy.*), permite a construção de sistemas fuzzy em um ambiente puramente Java, enquanto que, o segundo pacote (FuzzyJess), permite a construção de sistema híbridos, já que, abastece algumas funções de usuário para integração com o Jess (Orchard, 2006).

B.2.1 Variáveis *fuzzy*

As variáveis *fuzzy* definem os componentes básicos para descrever um conceito *fuzzy*. Cada variável *fuzzy* possui um NOME (Exemplos, temperatura, pressão, fluxo, etc.), UNIDADES (Exemplos, °C, Kpa, l/seg etc.), um Universo de Discurso (*UOD*) – que representa o rango de valores permitidos da variável lingüística (Exemplos, 0-100°C, 0-500Kpa, 0-50l/seg) e finalmente possui um conjunto de términos *fuzzy* usados para descrever conceitos *fuzzy* particulares para esta variável (Exemplo, um nome de término *HOT* junto com um conjunto *fuzzy* que representa geometricamente esse término).

No API do pacote nrc.fuzzy.*, pode-se encontrar as diferentes estruturas básicas dos construtores para a criação de variáveis fuzzy, um exemplo básico de como criar uma variável seria algo como:

...

FuzzyVariable temp = new FuzzyVariable(“Temperature”, 0, 100, “°C”);

temp.addTerm(“hot”, new TriangleFuzzySet(2,2.5,3));

…

Onde:

- **temp**, seria a variável *fuzzy* que representa o conceito *fuzzy* Temperatura. Para criar variáveis fuzzy, tem-se criar um objeto da classe *FuzzyVariable*.
- **0-100**, é o UOD da variável *fuzzy* temperatura.
- **°C**, são as unidades da variável *fuzzy* temperatura.
- **addTerm,** é o método que se usa para criar términos *fuzzy* em uma variável *fuzzy*. Internamente os términos *fuzzy* são guardados como valores *fuzzy*.
- **new,** é a palavra chave que se usa para a criação de objetos de diferentes classes. Neste simples exemplo, se criou uma objeto da classe *TriangleFuzzySet* (que serve para criar o conjuntos *fuzzy* que representa o conceito *fuzzy* “*hot*”).

B.2.2 Conjuntos *fuzzy*

Um conjunto *fuzzy* é uma representação física de uma função característica (geralmente uma função matemática) de algum conceito *fuzzy*.

Um conjunto *fuzzy* no pacote nrc.fuzzy.*, é usado para definir um conjunto de valores x e y que determinam a forma de um conjunto *fuzzy* (é simplesmente um objeto que serve para representar e guardar a forma de um conjunto *fuzzy*).

Os conjuntos *fuzzy* geralmente estão conteúdos ou são iguais aos valores *fuzzy*.
Um exemplo básico de como criar conjuntos fuzzy no pacaote nrc.fuzzy.*, seria algo como:

...

pressure.addTerm(“low”, new ZFuzzySet(2.0, 5.0));

pressure.addTerm(“medium”, new PIFuzzySet(5.0, 2.5));

pressure.addTerm(“high”, new SFuzzySet(5.0, 8.0));

Os términos "*low*", "*medium*" e "*high*", são representados matematicamente pelos objetos das classes *ZFuzzySet*, *PIFuzzySet* e *SFuzzySet* respectivamente. Esses objetos dessas classes representam totalmente os conjuntos *fuzzy* que dão alguma interpretação a esses términos *fuzzy*.

B.2.3 Valores *fuzzy*

Um valor *fuzzy* é uma associação de uma variável *fuzzy* e uma expressão lingüística para descrever algum conceito *fuzzy*. Com a classe *FuzzyValue*, pode-se criar objetos específicos de um conceito *fuzzy* para uma variável *fuzzy* (Exemplo, a temperatura é muito quente).
Um exemplo simples de como criar valores fuzzy, seria algo como:

```
...
FuzzyValue fval1 = new FuzzyValue(temp, "hot");
FuzzyValue fval2 = new FuzzyValue(pressure, "low or medium");
FuzzyValue fval3 = new FuzzyValue(temp, "very medium");
...
```

No exemplo anterior criaram-se três valores *fuzzy* cada um com seus construtores. No caso do valor *fuzzy fval1*, que representa o conceito (temperatura é quente), tem associado implicitamente o seu respectivo conjunto que define completamente o conceito "*hot*", analogamente para os dois restantes valores *fuzzy*. É importante notar que neste exemplo se usou um tipo de construtor para definir três valores *fuzzy*, existem mais tipos de construtores no API do pacote nrc.fuzzy.*, cada um com seus números de parâmetros, tipos de parâmetros, etc.

B.2.4 Modificadores *fuzzy (hedges)*

Os modificadores *fuzzy* (*Hedges*), são usados para dar mais realce à habilidade para construir conceitos *fuzzy*. Os modificadores mudam a forma de um conjunto *fuzzy* usando operações matemáticas em cada ponto do conjunto.

Os términos de uma variável *fuzzy* mais, o conjunto de modificadores *fuzzy* mais, os operadores *AND* e *OR* (o operador *AND* é sinônimo de interseção em conjuntos *fuzzy*, e o operador *OR* é sinônimo de união de conjuntos *fuzzy*) mais, os parêntesis esquerdo e direito para evitar problemas de interpretação por parte do compilador do pacote **nrc.fuzzy.***. O anteriormente exposto formam as bases gramaticais para escrever expressões lingüísticas *fuzzy* que descrevem conceitos *fuzzy* em um estilo como o inglês. Finalmente essas expressões lingüísticas se codificam e valores *fuzzy*. Para clarificar este ponto veja o seguinte exemplo:

(very hot or warm) and (slightly cold)

No exemplo anterior, temos três términos *fuzzy* (*HOT*, *WARM* e *COLD*), dois modificadores (*VERY* e *SLIGHTLY*), os operadores *AND* e *OR*, o jogo de parêntesis esquerdo e direito para evitar ambigüidades de processamento computacional. Finalmente o exemplo anterior se codifica em um valor *fuzzy* para fins de inferência *fuzzy*.

Um exemplo simples computacionalmente incompleto é mostrado:

...
FuzzyValue fval3 = new FuzzyValue(temp, "very medium");
…

No exemplo anterior, é aplicado o modificador "*very*", no pacote nrc.fuzzy.*, esse modificador tem a propriedade de retornar o valor *fuzzy* expandido passado como seu argumento, tendo elevado todos os valores da função característica do valor *fuzzy* por o fator 2.

B.2.5 Regras *fuzzy*

As regras *fuzzy* matem três conjuntos *fuzzy* ou valores *fuzzy* (os antecedentes, as conclusões e as entradas). Se todas as condições da regras são verdadeiras então, a regra se executará (disparará), e as conclusões serão declaradas. As regras se executam por meio da classe *FuzzyRuleExecutor*.

O *FuzzyRuleExecutor*, implementa um dos algoritmos para inferência fuzzy, tal como por exemplo: A inferência *fuzzy* estilo Mamdani, com operador composição Max-min, este é um operador válido só para regras que têm só valores *fuzzy* no lado direito e esquerdo da regra, isto sempre cumpre dentro do pacote nrc.fuzzy.*.

Para criar uma regra *fuzzy*, deve-se criar uma instancia (objeto) da classe *FuzzyRule* e depois acrescentar-lhe valores fuzzy, que o conformam (antecedentes, entradas e conclusões).
Um exemplo incompleto sobre a sintaxe de criação de uma regra *fuzzy* é mostrado embaixo:

```
...
FuzzyRule rule1 = new FuzzyRule( );
FuzzyValue fval1 = new FuzzyValue(temp, "hot");
FuzzyValue fval2 = new FuzzyValue(pressure, "low or medium");
FuzzyValue fval3 = new FuzzyValue(temp, "very medium");
rule1.addAntecedent(fval1);
rule1.addConclusion(faval2);
rule1.addInput(fval3);
FuzzyValueVector fvv = rule1.execute( );
...
```

No exemplo anterior, criou-se uma regra *fuzzy*, melhor dito, um objeto da classe *FuzzyRule*, no qual possui um antecedente, um conseqüente e uma entrada, cada uma delas representadas por suas respectivas expressões lingüísticas (conjuntos *fuzzy*), para os quais se usaram os métodos, para acrescentá-las à regra *fuzzy*: *addAntecedent*, *addConclusion* e *addInput* respectivamente e finalmente no código incompleto de acima, executou-se a regra *fuzzy* com o método *execute( )*, por defeito no pacote nrc.fuzzy.*, é aplicado o método de inferência Mamdani com operador composição Max-min, isto pode ser facilmente mudado a través de outros métodos implementados neste pacote.

É importante fazer menção que, depois de executar a regra, este método devolve um vetor de valores *fuzzy*, os quais, depois tem que ser obtidos um a um por algum método implementado neste pacote para que, conseqüentemente seja descodificado (*defuzzificated*)

com uns dos métodos, tais como, Centro de Área (COA), Centro de Gravidade (COG), entre os mais importantes.

B.2.6 O pacote *FuzzyJess*

O pacote *FuzzJess* nasceu da necessidade de criar um conjunto de classes portáveis em Java que permitam razoamento *fuzzy* e enlaçá-las com o Jess, resultando assim um sistema composto de um conjuntos de classes mais flexíveis e permitindo ao usuário uma forma mais simples de manutenção.

Em Jess pode-se referenciar facilmente classes de Java e isto permite a um trabalhar inteiramente em Jess quando apropriado.

Uma coisa a considerar em sistemas expertos com regras tipo FuzzyJess é que, geralmente sempre vão existir múltiplos valores *fuzzy* no lado esquerdo de uma regra e múltiplos valores de entradas para combinar todas elas e assim executar a regra completamente.

Existem algumas funções de usuário implementadas neste pacote, a continuação vai se detalhar alguns deles, entre os mais importantes.

B.2.6.1 A função de usuário *FuzzyMatch*

Esta função de usuário tem a seguinte sintaxe: *(fuzzy-match FuzzyValue1 FuzzyValue2)*, devolve um valor *TRUE*, se os dois valores *fuzzy* têm algum grau de casamento que é um valor de a função característica de por o menos tão grande como o valor atual do umbral (*FuzzyValue matchThreshold*, que pode ser mudado com o método estático setMatchThreshold da classe *FuzzyValue*).

É importante mencionar que, ambos os argumentos podem ser valores fuzzy (com a mesma variável fuzzy) ou um pode ser uma cadeia (*string*) que é uma expressão lingüística válida para a variável *fuzzy* associada com o valor *fuzzy*.

B.2.6.2 A função de usuário *GlobalContributionOperator*

Esta função de usuário tem a seguinte sintaxe: *(set-fuzzy-global-contribution-operator OPERATOR)*, o qual toma um argumento que especifica a forma de contribuição global *fuzzy* que deveria ser feita quando fatos idênticos com valores *fuzzy* são declarados. Os valores deste operador (*OPERATOR*), poder ser os seguintes:

- ***union***, combina os valores fuzzy usando a operação *FuzzyUnion* (este é feito por defeito).
- ***sum***, combina os valores fuzzy usando a operação *FuzzySum*.
- ***none***, não combina os valores *fuzzy*. Neste caso, se faz um troco do valor antigo com o valor novo.

B.2.6.3 A função de usuário *FuzzyRuleExecutor*

Esta função de usuário tem a seguinte sintaxe: *(set-default-fuzzy-rule-executor EXECUTOR)*, o qual toma um argumento que especifica qual executor de regra fuzzy deveria ser usado quando uma regra é executada (disparada). O (EXECUTOR) pode tomar um dos seguintes valores:

- ***mamdanimin***, usa o executor de regras Mamdani (este é feito por defeito).
- ***larsenproduct,*** usa o executor de regras Larsen.
- ***tsukamoto,*** usa o executor de regras Tsukamoto.

B.2.6.4 A função de usuário *AntecedentCombineOperator*

Esta função de usuário tem a seguinte sintaxe: *(set-default-antecedent-combine-operator OPERATOR)*, o qual toma um argumento que especifica qual operador deveria ser usado para combinar os valores de combinação (antecedente/entrada) quando uma regra *fuzzy* é executada. Os valores que pode tomar (*OPERATOR*) são os seguintes:

- ***minimun***, usa o mínimo dos valores de combinação dos pares (antecedente/entrada), isto acontece por defeito neste pacote.

- ***product***, usa o produto dos valores de combinação dos pares (antecedente/entrada).
- ***compensatoryAnd***, usa um operador especial que dá um valor maior que o oferecido por o operador *minimun*.

APÊNDICE C – BASE DE REGRAS *FUZZY* COMPLEMENTAR

O objetivo desta seção é apresentar a base de regras *fuzzy* para as variáveis de temperatura do metal número 1 e 3. Os procedimentos para criar estas regras são os mesmo que foram dados no Capítulo 4.

C.1 BASE DE REGRAS *FUZZY* PARA A VARIÁVEL DE TEMPERATURA DO METAL NÚMERO 1

A seguir, é apresentada uma seqüência de código para criar a base de regras *fuzzy* para esta variável em particular.

Regra *fuzzy* para a condição NORMAL do metal 1:

```
(defrule conditionmonitoring02_m1
    (tempMetal2 ?t&:(fuzzy-match ?t "normal"))
=>
 (bind ?*c1_m1* (?*normal_m1* getMembership ?*t1*))
 (bind ?*c2_m1* (?*alerta_m1* getMembership ?*t1*))

 (if(> ?*c2_m1* 0.0)
   then
     (if(> ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(= ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(< ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de: " ?*t1* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c2_m1* " [0-1]" crlf)
     )
   )
 )
```

Regra *fuzzy* para a condição ALERTA do metal 1:

```
(defrule conditionmonitoring03_m1
    (tempMetal2 ?t&:(fuzzy-match ?t "alerta"))
=>
 (bind ?*c1_m1* (?*alerta_m1* getMembership ?*t1*))
 (bind ?*c2_m1* (?*alarme_m1* getMembership ?*t1*))

  (if(> ?*c2_m1* 0.0)
   then
     (if(> ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(= ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(< ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de: " ?*t1* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c2_m1* " [0-1]" crlf)
     )
   )
 )
```

Regra *fuzzy* para a condição ALARME do metal 1:

```
(defrule conditionmonitoring04_m1
    (tempMetal2 ?t&:(fuzzy-match ?t "alarme"))
=>
 (bind ?*c1_m1* (?*alarme_m1* getMembership ?*t1*))
 (bind ?*c2_m1* (?*trip_m1* getMembership ?*t1*))

 (if(> ?*c2_m1* 0.0)
   then
     (if(> ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(= ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de " ?*t1* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c1_m1* " [0-1]" crlf)
     )
     (if(< ?*c1_m1* ?*c2_m1*)
     then
     (printout t "Para uma temperatura do metal 01 de: " ?*t1* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal
de: " ?*c2_m1* " [0-1]" crlf)
     )
   )
 )
```

Regra *fuzzy* para a condição TRIP do metal 1:

```
(defrule conditionmonitoring05_m1
    (tempMetal2 ?t&:(fuzzy-match ?t "trip"))
=>
 (bind ?*c1_m1* (?*alarme_m1* getMembership ?*t1*))
 (bind ?*c2_m1* (?*trip_m1* getMembership ?*t1*))

 (if(= ?*c1_m1* 0.0)
   then

     (printout t "Para uma temperatura do metal 01 de: " ?*t1* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal
de: " ?*c2_m1* " [0-1]" crlf)
  )
)
```

C.2 BASE DE REGRAS *FUZZY* PARA A VARIÁVEL DE TEMPERATURA DO METAL NÚMERO 3

A seguir, é apresentada uma seqüência de código para criar a base de regras *fuzzy* para esta variável em particular.

Regra *fuzzy* para a condição NORMAL do metal 3:

```
(defrule conditionmonitoring02_m3
    (tempMetal2 ?t&:(fuzzy-match ?t "normal"))
=>
 (bind ?*c1_m3* (?*normal_m3* getMembership ?*t3*))
 (bind ?*c2_m3* (?*alerta_m3* getMembership ?*t3*))

 (if(> ?*c2_m3* 0.0)
   then
     (if(> ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(= ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição NORMAL com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(< ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de: " ?*t3* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c2_m3* " [0-1]" crlf)
     )
   )
 )
```

Regra *fuzzy* para a condição ALERTA do metal 3:

```
(defrule conditionmonitoring03_m3
    (tempMetal2 ?t&:(fuzzy-match ?t "alerta"))
=>
 (bind ?*c1_m3* (?*alerta_m3* getMembership ?*t3*))
 (bind ?*c2_m3* (?*alarme_m3* getMembership ?*t3*))

  (if(> ?*c2_m3* 0.0)
   then
     (if(> ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(= ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição ALERTA com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(< ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c2_m3* " [0-1]" crlf)
     )
   )
 )
```

Regra *fuzzy* para a condição ALARME do metal 3:

```
(defrule conditionmonitoring04_m3
    (tempMetal2 ?t&:(fuzzy-match ?t "alarme"))
=>
 (bind ?*c1_m3* (?*alarme_m3* getMembership ?*t3*))
 (bind ?*c2_m3* (?*trip_m3* getMembership ?*t3*))

 (if(> ?*c2_m3* 0.0)
   then
     (if(> ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(= ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de " ?*t3* "°C" crlf)
     (printout t "Condição ALARME com um indicador de degradação de sinal
de: " ?*c1_m3* " [0-1]" crlf)
     )
     (if(< ?*c1_m3* ?*c2_m3*)
     then
     (printout t "Para uma temperatura do metal 03 de: " ?*t3* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal
de: " ?*c2_m3* " [0-1]" crlf)
```

```
     )
   )
 )
```

Regra *fuzzy* para a condição TRIP do metal 3:

```
(defrule conditionmonitoring05_m3
    (tempMetal2 ?t&:(fuzzy-match ?t "trip"))
=>
 (bind ?*c1_m3* (?*alarme_m3* getMembership ?*t3*))
 (bind ?*c2_m3* (?*trip_m3* getMembership ?*t3*))

 (if(= ?*c1_m3* 0.0)
   then

     (printout t "Para uma temperatura do metal 03 de: " ?*t3* "°C" crlf)
     (printout t "Condição TRIP com um indicador de degradação de sinal
de: " ?*c2_m3* " [0-1]" crlf)
  )
)
```