MARCEL STEFAN WAGNER

Sistema cognitivo com tomada de decisão baseada em Lógica Fuzzy para aplicação em ambientes de redes de sensores sem fio com múltiplos saltos

São Paulo
2016

MARCEL STEFAN WAGNER

Sistema cognitivo com tomada de decisão baseada em Lógica Fuzzy para aplicação em ambientes de redes de sensores sem fio com múltiplos saltos

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Ciências.

São Paulo
2016

MARCEL STEFAN WAGNER

Sistema cognitivo com tomada de decisão baseada em Lógica Fuzzy para aplicação em ambientes de redes de sensores sem fio com múltiplos saltos

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Ciências.

Área de concentração: Engenharia Elétrica – Sistemas Eletrônicos.

Orientador: Prof. Dr. Miguel Arjona Ramírez

Coorientador: Prof. Dr. Wagner Luiz Zucchi

São Paulo
2016

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Wagner, Marcel Stefan
Estudo e desenvolvimento de sistema cognitivo com tomada de decisão baseada em Lógica Fuzzy para aplicação em ambientes de redes de sensores sem fio com múltiplos saltos / M. S. Wagner -- versão corr. -- São Paulo, 2016.
237 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1.Wireless 2.Cognição 3.Lógica Fuzzy 4.Sistema Adaptativo 5.Cognitive Networks I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.

Dedico este trabalho à minha esposa Bruna
e aos meus pais Eldon e Jane.

AGRADECIMENTOS

Ao Nosso Senhor, por me oferecer tantas alegrias e apontar sempre um caminho para seguir, mesmo nos momentos mais difíceis. Por fortalecer meu espírito e me enriquecer de bons sentimentos e de perseverança.

Ao meu orientador Prof. Dr. Miguel Arjona Ramírez e ao meu coorientador Prof. Dr. Wagner Luiz Zucchi, pela oportunidade fornecida de cursar o Programa de Doutorado da USP, além de orientação, compreensão, ensinamentos e busca por caminhos alternativos de incentivo à publicação. Que este obrigado, que já não é o primeiro, não seja o último, mas sim a continuação de uma parceria duradoura.

À minha querida esposa Bruna por sempre estar ao meu lado, oferecendo carinho, dedicação e compreensão, além de me apoiar em decisões complexas e desafios. Passei por muitos momentos difíceis, mas com esta verdadeira companheira eu sei que nunca estarei só na jornada da vida.

Aos meus pais Eldon e Jane, por me proporcionarem as condições para crescer, estudar e realizar meus sonhos. À minha mãe pela dedicação nas horas mais complicadas, auxiliando também nas decisões e conquistas. Ao meu pai, por proporcionar a entrada na vida acadêmica. Ao meu irmão Jean, pelo apoio, incentivo e auxílio na busca por alternativas nas mais diversas situações.

Aos meus amigos Ton, Celso e Cadu pela amizade e por sempre estarem presentes me ajudando a manter vivo o espírito acadêmico. Pelas dicas e conselhos muito bem vindos, inclusive nas horas mais complicadas. Ao Rafael Morgado, Guilherme Alaia e Fábio Japonês pelo apoio moral.

Ao Prof. Dr. Marco Antonio Garms, pelo auxílio no entendimento de Lógica *Fuzzy* e à Sra. Ana Maria de Castro Badiali pela revisão deste trabalho.

Por fim, mas não menos importante, agradeço aos meus amigos, parentes e padrinhos, que sempre estão presentes na minha vida oferecendo alegria e otimismo, e às pessoas que indiretamente me ajudaram de alguma forma para o sucesso e direcionamento desta obra.

“*O sucesso nasce do querer, da determinação e persistência em se chegar a um objetivo. Mesmo não atingindo o alvo, quem busca e vence obstáculos, no mínimo fará coisas admiráveis.*”

José de Alencar

RESUMO

Esta Tese estuda a implementação de um novo mecanismo de análise e atuação em Redes de Sensores Sem Fio (RSSF) com múltiplos saltos baseado em características de cognição aplicadas aos nós que compõem a rede. Para tanto, é proposto um algoritmo de detecção de variabilidade dos nós sensores, envolvendo movimentação do nó, alcance do sinal da antena do sensor, quantidade de nós que fazem parte da rede e o número de conexões possíveis com nós vizinhos.

Além do algoritmo de detecção de variabilidade, propõe-se um sistema *multi-layer* denominado *Adaptive Cognitive System* (ACS) com base na arquitetura de *Cognitive Networks* (CN), que abrange: coleta, tratamento e tomada de decisão. O tratamento se refere à parte cognitiva do sistema, contemplando a criação do *Cognitive Processor Module* (CPMod), que por sua vez, abrange a semântica da rede, aplicação de Lógica *Fuzzy* e interação com um simulador de *Wireless Sensor Networks* (WSN) e a tomada de decisão é realizada pelo CPMod com base no resultado de análises executadas em *rounds* e histórico da rede com o uso de funções de pertinência de fuzzificação e defuzzificação, regras *Fuzzy* e inferência sobre informações coletadas da rede.

Observou-se com os testes realizados na rede, utilizando-se o algoritmo de detecção, que a variabilidade dos nós sensores afeta diretamente o desempenho da rede, devido à necessidade de reestabelecimento de *links* e rotas entre os nós. Através de testes realizados via *software* na WSN, identificou-se que com o uso do ACS houve melhora significativa no desempenho em relação ao atraso fim-a-fim, latência, quantidade de pacotes descartados e de energia consumida pelos nós na rede. O ACS demonstrou potencial para a solução de problemas relacionados com as métricas destacadas, realizando ajustes em múltiplas camadas de rede do padrão IEEE 802.15.4 para até 200 nós na rede.

Palavras-chave: *Wireless*; Cognição; Lógica *Fuzzy*; Sistema Adaptativo; Redes Cognitivas.

ABSTRACT

This Dissertation examines the implementation of a mechanism to analyze and act on multi-hop Wireless Sensor Networks (WSN) with the use of cognitive features applied to the network nodes. For this purpose, a variation detection algorithm was proposed for monitoring sensor nodes, involving the node's mobility features, signal range of the sensor antenna, the number of nodes in the network and the number of possible connections to neighboring nodes.

In addition to the detection algorithm, a multi-layer system is proposed, named Adaptive Cognitive System (ACS). It is based on Cognitive Networks (CN) architecture, including data gathering, information treatment and decision making. The main part of the system is the Cognitive Processor Module (CPMod), which extracts the information about the WSN. In turn the Fuzzy Logic block works in tandem with the semantic engine to feed the codes to CPMod in the decision making process. The codes are the result of analysis performed on rounds using fuzzification and defuzzification membership functions, fuzzy rules and inference over information collected from the network.

It was observed in tests performed in the WSN, using the detection algorithm, that the variability in sensor nodes directly affects the network performance due to the effort spent in rerounting links and paths. Through WSN testing performed via software, it was found that using the ACS implies in significant improvement in performance over the end-to-end delay, network latency, dropped packets and amount of energy consumed by nodes on the network. The ACS potential is proven for solving problems related to the previously mentioned metrics, performing adjustments on multiple network layers standardized by IEEE 802.15.4 up to 200 nodes in the network.

Keywords: Wireless; Cognition; Fuzzy Logic; Adaptive System; Cognitive Networks.

LISTA DE FIGURAS

Figura 1. Estrutura básica interna em blocos de um nó sensor. 37
Figura 2. Estrutura básica de conexão interna de um *wireless sensor node*. 38
Figura 3. Exemplo de troca de mensagens no AODV. 45
Figura 4. Exemplo de *Route Request* (RREQ) no AODV. 45
Figura 5. Exemplo de *Route Reply* (RREP) no AODV. 46
Figura 6. Fluxo de mensagens no AODV. 46
Figura 7. Exemplo de *Route Error* (RERR) no AODV. 47
Figura 8. Envio de dados no DSDV. 47
Figura 9. Comparativo entre protocolos com *flooding*. (a) protocolos *flooding* comuns; (b) utilizando MPR no protocolo OLSR. 51
Figura 10. Exemplo de *Source Routing* no DSR. 53
Figura 11. Exemplo de encaminhamento de dados no *Route Angle*. 56
Figura 12. Escolha de rota no ABRP. 57
Figura 13. Envio de dados do nó S para o D no SFN-DCRP *broadcast*. 59
Figura 14. Modelo representativo abstrato entre HW / SW e Cérebro / Mente. 62
Figura 15. Tomada de decisão e estados intencionais. 63
Figura 16. Exemplo de clusterização no ambiente de ciência cognitiva. 64
Figura 17. Clusterização na formatação de tabela correlacional utilizada para formar sequências causais. 64
Figura 18. Arquitetura baseada em 4 estados mentais. 65
Figura 19. Diagrama de interação de agentes e fontes de informação. 66
Figura 20. Exemplo de Sistema Dinâmico de controle de processos. 66
Figura 21. *Cognitive framework* e assuntos críticos de projetos. 69
Figura 22. Arquitetura interna de um CR. 71
Figura 23. Processo cognitivo do CogSeNet. 74
Figura 24. Exemplo de reconexão entre S e D pelo protocolo ABAM. 78
Figura 25. Estrutura básica do módulo cognitivo do RVM. 79
Figura 26. Arquitetura do DCN. 82
Figura 27. Representação de 2 nós no TCNet. 84
Figura 28. Representação gráfica do número médio de pacotes na fila de processamento pela utilização do canal de transmissão. 86
Figura 29. Interface principal do *Prowler*. 100
Figura 30. Interface do *Prowler* adaptado. 101
Figura 31. Pilha com blocos genéricos internos dos nós sensores. 104
Figura 32. Característica de movimentação aleatória do nó na WSN. 106

Figura 33. Característica de movimentação circular do nó na WSN. 107
Figura 34. Característica de cobertura do sinal do nó na WSN. 108
Figura 35. Função Densidade de Probabilidade do tipo Rayleigh para canal de comunicação. 110
Figura 36. Característica de possibilidade de conexões do nó na WSN. 112
Figura 37. Características gerais da WSN. 113
Figura 38. Pseudo-código da variabilidade do sistema. 115
Figura 39. Diagrama em blocos de interconexão do processo cognitivo. 116
Figura 40. Relação do limiar de decisão (DL) com o DMp. 120
Figura 41. Diagrama em blocos da arquitetura do ACS. 121
Figura 42. Arquitetura do ACS. (a) modelo com 4 camadas; (b) modelo genérico tridimensional com 5 camadas. 122
Figura 43. Rede de Petri do funcionamento geral do sistema. 123
Figura 44. Estrutura geral do nó com o CPMod. 124
Figura 45. Rede de Petri do CPMod. 126
Figura 46. Rede de Petri sobre procedimento de adequação da rede. 129
Figura 47. Fluxograma sobre a Lógica *Fuzzy*. 130
Figura 48. Diagrama em blocos da Lógica *Fuzzy*. 131
Figura 49. Tela inicial de *Fuzzy* no MATLAB. 132
Figura 50. Tela de gráficos de entrada de fuzzificação no MATLAB. 134
Figura 51. Tela de regras *Fuzzy* aplicadas ao módulo Mamdani no MATLAB. 135
Figura 52. Tela de gráficos de saída da defuzzificação no MATLAB. 136
Figura 53. Tela do *Prowler* adaptado. 137
Figura 54. Função Densidade de Probabilidade do tipo Rayleigh para o canal de comunicação da rede simulada no *Prowler*. 138
Figura 55. Tela de dados básicos para simulação do *Prowler*. 138
Figura 56. Tela de configuração de parâmetros para simulação do *Prowler*. 139
Figura 57. Modelo do CPMod desenvolvido no *Simulink* do MATLAB. 141
Figura 58. Modelo de correção aplicado ao CPMod que foi desenvolvido no *Simulink* do MATLAB. 142
Figura 59. Blocos de entrada de dados provenientes do *Prowler* e de DMp com base no DL. 143
Figura 60. Blocos de semântica e de histórico da rede. 143
Figura 61. Bloco de Lógica *Fuzzy*. 144
Figura 62. Bloco de memória episódica do início de verificação da rede. 144
Figura 63. Cenários de 50, 100, 150 e 200 nós. 145
Figura 64. Primeira rodada de testes finalizada conforme a quantidade de nós. 146

Figura 65. Uso da inferência para auxiliar o CPMod na adoção de tomada de decisão..... 147

LISTA DE GRÁFICOS

Gráfico 1. Relação de causa e efeito sobre o CPMod. 131
Gráfico 2. Relação de capacidade de armazenamento na memória episódica. . 150
Gráfico 3. Taxa de erro relativa aos eventos da rede. 151
Gráfico 4. Grau de variabilidade dos nós sensores. 151
Gráfico 5. Variabilidade para um número maior de nós. 152
Gráfico 6. Entrada e saída do módulo *Fuzzy* da primeira rodada de testes. 153
Gráfico 7. Dados sobre o sinal gerado entre os nós na primeira rodada. 154
Gráfico 8. Entrada e saída do módulo *Fuzzy* do primeiro *round* de testes. 155
Gráfico 9. Dados sobre o sinal gerado entre os nós no primeiro *round*. 156
Gráfico 10. Entrada e saída do módulo *Fuzzy* do segundo *round* de testes. 157
Gráfico 11. Dados sobre o sinal gerado entre os nós no segundo *round*. 158
Gráfico 12. Entrada e saída do módulo *Fuzzy* do terceiro *round* de testes. 159
Gráfico 13. Dados sobre o sinal gerado entre os nós no terceiro *round*. 160
Gráfico 14. Entrada e saída do módulo *Fuzzy* do quarto *round* de testes. 161
Gráfico 15. Entrada do bloco *Fuzzy* utilizando o ARP. 162
Gráfico 16. Saída do bloco *Fuzzy* utilizando o ARP. 162
Gráfico 17. Entrada do bloco *Fuzzy* utilizando o ABAM. 163
Gráfico 18. Saída do bloco *Fuzzy* utilizando o ABAM. 163
Gráfico 19. Relação de eventos na rede e salvamento de casos sobre o ABAM. 164
Gráfico 20. Saída do bloco *Fuzzy* com troca de protocolos de roteamento. 164
Gráfico 21. Entrada do bloco *Fuzzy* com troca de protocolos de roteamento. 165
Gráfico 22. Implicação no caso de retorno do ABAM para o ARP. 165
Gráfico 23. Entrada do bloco *Fuzzy* para simulação com uso inicial do ARP para 210 nós. 166
Gráfico 24. Densidade de alcance dos *motes* MICAz. 167
Gráfico 25. Tendência da quantidade de conexões estimadas por *mote*. 168
Gráfico 26. Entrada do bloco *Fuzzy* para 50 nós sensores. 187
Gráfico 27. Saída do bloco *Fuzzy* para 50 nós sensores. 187
Gráfico 28. Entrada do bloco *Fuzzy* para 100 nós sensores. 188
Gráfico 29. Saída do bloco *Fuzzy* para 100 nós sensores. 188
Gráfico 30. Entrada do bloco *Fuzzy* para 150 nós sensores. 189
Gráfico 31. Saída do bloco *Fuzzy* para 150 nós sensores. 189
Gráfico 32. Entrada do bloco *Fuzzy* para 200 nós sensores. 190
Gráfico 33. Saída do bloco *Fuzzy* para 200 nós sensores. 190

LISTA DE TABELAS

Tabela 1 – Comparativo das tecnologias de comunicação sem fio. 25
Tabela 2 – Comparação entre variadas tecnologias de transmissão sem fio..... 39
Tabela 3 – Comparação entre diferentes simuladores de WSN..... 97
Tabela 4 – Características dos sensores MICA *motes*..... 103
Tabela 5 – Componentes do *Memory vector*..... 117
Tabela 6 – Tomada de decisão conforme o DL..... 118
Tabela 7 – Configurações básicas do *Prowler*. 140
Tabela 8 – Resumo das atribuições do vetor de controle..... 167

LISTA DE ABREVIATURAS E SIGLAS

1xEV-DO	*Single-carrier Evolution-Data Optimized*
1xRTT	*Single-carrier Radio Transmission Technology*
3D	3 dimensões
3G	3ª Geração de redes celulares
AACR	*Aware, Adaptive and Cognitive Radio*
ABAM	*Associativity-based Ad hoc Multicast*
ABC	*Always Best Connected*
ABI	*Always Better Informed*
ABRP	*Angle based Routing Protocol*
ACK	*Acknoledgment*
ACO	*Ant Colony Optimization*
ACS	*Adaptive Cognitive System*
ADC	*Analog-to-Digital Converter*
AI	*Artificial Intelligence*
Anatel	Agência Nacional de Telecomunicações
AODV	*Ad hoc On-Demand Distance Vector*
AP	*Access Point*
ARP	*Angle Routing Protocol / Address Resolution Protocol*
BISC	*Berkeley Initiative in Soft Computing*
BS	*Base Station*
CCIR	*Comité Consultatif International pour la Radio*
CDMA	*Code Division Multiple Access*
CDMA2000	*Code Division Multiple Access 2000*
CIoT	*Cognitive Internet of Things*
CoA	*Center of Area*
CogSeNet	*Cognitive Wireless Sensor Network*
CoM	*Center of Maximum*
CoMP	*Coordinated MultiPoint*
CP	*Cognitive Process*
CPMod	*Cognitive Processor Module*
CR	*Cognitive Radio*
CRP	*Cooperative Routing Protocol*
CSL	*Cognition Selfish Layer*
CSMA	*Carrier Sense Multiple Access*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
CSN	*Cognitive Sensor Networks*

CSPV	*Channel Signal Propagation Velocity*
CT	*Channel Throughput*
CWPDA	*Cognitive Wireless Personal Digital Assistant*
DAG	*Directed Acyclic Graph*
DAO	*Destination Advertisement Object*
DAO-ACK	*Destination Advertisement Object Acknowledgement*
DCN	*Dynamic Cognitive Networks*
DD	*Directed Diffusion*
DDIS	*Destination Discovery*
DIO	DODAG *Information Object*
DIS	DOGAG *Information Solicitation*
DL	*Decision Limit*
DM	*Decision Making*
DMp	*Decision Making process*
DNA	*DeoxyriboNucleic Acid*
DODAG	*Destination Oriented Directed Acyclic Graph*
DREAM	*Distance Routing Effect Algorithm for Mobility*
DREQ	*Destination Request*
DSDV	*Destination-Sequenced Distance Vector*
DSP	*Digital Signal Processor*
DSR	*Dynamic Source Routing*
DTAR	*Discovery's Target*
DV	*Distance Vector*
ETAMR	*Efficient Traffic Aware Multipath Routing*
EWSNSIM	*Eyes WSN Simulator*
FCC	*Federal Communications Commission*
FCM	*Fuzzy Cognitive Maps*
FDP	Função Densidade de Probabilidade
FIS	*Fuzzy Inference System*
FL	*Fuzzy Logic*
FSM	*Finite State Machine*
Gbps	*Giga bits per second*
GHz	*Giga Hertz*
GLS	*Grid's Location Service*
GPP	*General Purpose Processor*
GPRS	*General Packet Radio Service*
GPS	*Global Positioning System*
GSM	*Global System for Mobile Communication*
GUI	*Graphical User Interface*
HW	*Hardware*

IA	Inteligência Artificial
iCPMod	*initial Cognitive Processor Module*
iCR	*ideal Cognitive Radio*
ID	*Identification*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IoE	*Internet of Everything*
IoT	*Internet of Things*
IP	*Internet Protocol*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
IR	*InfraRed*
ITS	*Intelligent Transport Systems*
ITU-R	*International Telecommunication Union Radiocommunication Sector*
ITU-T	*International Telecommunication Union Telecommunication Sector*
JQ	*Join Query*
kbps	*kilo bits per second*
LAR	*Location-Aided Routing*
LLC	*Low-Power and Lossy Networks*
LQ	*Local Query*
LR	*Link Reversal*
LS	*Link State*
LTE	*Long Term Evolution*
M2M	*Machine-to-Machine*
MAC	*Medium Access Control*
MANET	*Mobile Ad Hoc Network*
Mbps	*Mega bits per second*
MBQ	*Multicast Broadcast Query*
MBWA	*Mobile Broadband Wireless Access*
MH	*Mobile Hosts*
MHz	*Mega Hertz*
MIT	*Massachussets Institute of Technology*
MM	*Mealy Machine*
MoM	*Mean of Maximum*
MOO	*Multi-Objective Optimizations*
MPR	*MultiPoint Relays*
MRHOF	*Minimum Rank with Hysteresis Objective Function*
NEST	*Network Embedded Systems Technology*
NFC	*Near Field Communication*
NN	*Neural Networks*

OFDM	*Orthogonal Frequency-Division Multiplexing*
OLSR	*Optimized Link State Routing*
OS	*Operating System*
OSI	*Open System Interconnection*
OSPF	*Open Shortest Path First*
OWL	*Ontology Web Language*
PAN	*Personal Area Network*
PC	*Personal Computer*
PL	*Packet Lenght*
PR	*Probabilistic Reasoning*
QoE	*Quality of Experience*
QoI	*Quality of Information*
QoS	*Quality of Service*
QR	*Quick Response*
RAM	*Random Access Memory*
RDF	*Resource Description Framework*
RERR	*Route Error*
RF	*Radiofrequency*
RFC	*Request For Comments*
RFID	*Radio-Frequency Identification*
RIP	*Routing Information Protocol*
RNA	Redes Neurais Artificiais
ROM	*Read-only Memory*
RPL	IPv6 *Routing Protocol for Low-Power and Lossy Networks*
RREP	*Route Reply*
RREQ	*Route Request*
RSSF	Rede de Sensores Sem Fio
RT	*Routing Table*
RTT	*Round Trip Time*
RVM	*Relevance Vector Machine*
RXML	*Radio eXtensible Markup Language*
SAN	*Software Adaptable Network*
SC	*Soft Computing*
SCM	Serviço de Comunicação Multimídia
SDP	*Signal Distance Propagation*
SDR	*Software-Defined Radio*
SFN	*Single Frequency Network*
SFN-DCRP	*Single Frequency Network based Distributed Cooperative Routing Protocol*
SIR	*Source Initiated Routing*
SNIR	*Signal-to-Noise-plus-Interference ratio*

SOL	*Secure Operations Language*
STP	*Spanning Tree Protocol*
SVM	*Support Vector Machine*
SW	*Software*
TC	*Trust Congestion*
TC-ACO	*Trust Congestion Ant Colony Optimization*
TCNet	*Trellis Coded Network*
TCP	*Transmission Control Protocol*
TIREM	*Terrain-Integrated Rough-Earth Model*
TM	*Trust Metrics*
TORA	*Temporally Ordered Routing Algorithm*
TOSSIM	*TinyOS Simulator*
TTL	*Time-to-Live*
TV	*Television*
UDP	*User Datagram Protocol*
UMTS	*Universal Mobile Telecommunication System*
USB	*Universal Serial Bus*
UWB	*Ultra Wide Band*
XML	*eXtensible Markup Language*
XOR	*eXclusive-OR*
WCDMA	*Wideband Code Division Multiple Access*
Wi-Fi	*Wireless-Fidelity*
WiMax	*Worldwide Interoperability for Microwave Access*
WLAN	*Wireless Local Area Network*
WMAN	*Wireless Metropolitan Area Network*
WPAN	*Wireless Personal Area Network*
WSN	*Wireless Sensor Networks*
WWAN	*Wireless Wide Area Network*

LISTA DE UNIDADES E SÍMBOLOS

%	Porcentagem
≈	aproximadamente
+	mais
±	mais ou menos
∧	lógica AND
∨	lógica OR
¬	lógica NOT
|	tal que
|a|	módulo de a
≤	menor igual
≥	maior igual
≪	muito menor
<	menor
>	maior
⊂	está contido
a ↔ b	a se e somente se b
a > b	a maior do que b
®	*Registered* (marca registrada)
ℝ	conjunto dos números reais
ℰ	erro quadrático médio
∀	todo ou qualquer
Σ	somatório
Π	produtório
∫	integral
∈	pertence
∉	não pertence
α	proporção de posição
β	proporção de alcance
γ	proporção de conexões possíveis
σ	proporção da quantidade de nós
ρ	número de bits transmitidos pelo canal
λ	taxa média de chegada de pacotes na fila
θ	ângulo de inclinação para deslocamento do nó
θ_i	quantidade i de proporções de características de variabilidade
ϕ_i	quantidade i de características de variablididade
ζS_I	valor estimado do sinal de entrada
ξS_O	valor real do sinal de saída

μ	micro
A	Ampère
b	*bits*
B	*Bytes*
C	Cérebro
D	*Destination* (nó de destino)
F_V	*Feature variability*
G	Giga
Hz	Hertz
J	Joules
k	quilo
L_V	*Link variation*
m	metros / mili
M	Mente / variável / Mega
N_V	*Node variation*
pkts	*packets*
P_V	*Position variation*
R_V	*Range variation*
R_X	*receiver* (nó receptor)
s	segundos
S	*Source* (nó fonte)
™	*Team* (marca comercial)
T_X	*transmitter* (nó transmissor)
W	Watts
ºC	graus celsius
$\bar{X}$	média aritmética simples

SUMÁRIO

1. APRESENTAÇÃO 24
1.1 Justificativa 25
1.2 Contexto 27
1.3 Objetivos gerais 28
1.4 Objetivos específicos 28
1.5 Motivação 29
1.6 Estado da arte 30
1.7 Contribuição 32
1.8 Estrutura do trabalho 35
2. FUNDAMENTAÇÃO TEÓRICA 36
2.1. RSSF 36
2.2. Transmissão *wireless* 39
2.2.1. *ZigBee* 40
2.2.2. *Bluetooth* 40
2.2.3. Wi-Fi 41
2.2.4. *Ultra Wide Band* 43
2.3. Protocolos de roteamento em WSN 44
2.3.1. *Ad hoc On-Demand Distance Vector* 44
2.3.2. *Destination-Sequenced Distance Vector* 47
2.3.3. *Optimized Link State Routing* 50
2.3.4. *Dynamic Source Routing* 52
2.3.5. *Temporally Ordered Routing Algorithm* 54
2.3.6. *Angle Routing Protocol* 55
2.3.7. *Grid's Location Service* 57
2.3.8. SFN *based Distributed Cooperative Routing Protocol* 58
2.4. *Internet of Things* 59
2.4.1. RPL – IPv6 *Routing Protocol for Low-Power and Lossy Networks* 60
2.5. Conceitos de cognição 61
2.5.1. Princípios básicos de ciência cognitiva 61
2.5.2. Tomada de decisão 62
2.5.3. Clusterização 63
2.5.4. Modelo BDI 65
2.5.5. Processos em Teoria dos Sistemas Dinâmicos 66
2.6. Redes cognitivas 67
2.6.1. *Cognitive Network* 67
2.6.2. *Cognitive Radio* 70

2.6.3. *Cognitive Sensor Network* 72
2.7. Protocolos de roteamento em CSN 74
2.7.1. *Efficient Traffic Aware Multipath Routing* 74
2.7.2. *Trust Congestion Ant Colony Optimization* 75
2.7.3. *Directed Diffusion* 76
2.7.4. *Associativity-based Ad hoc Multicast* 77
2.7.5. *Relevance Vector Machine* 79
2.8. Outros aspectos gerais em redes *wireless* e cognitivas 79
2.8.1. Segurança 80
2.8.2. *Dynamic Cognitive Networks* 81
2.8.3. *Trellis Coded Network* 83
2.8.4. Latência e Atraso fim-a-fim 84
2.8.5. Desvanecimento 87
2.9. Lógica *Fuzzy* 88
2.9.1. Inferência e modelos Mamdani e Sugeno 90
2.9.2. Fuzzificação 91
2.9.3. Defuzzificação 92
3. METODOLOGIA 94
3.1. Método científico 98
3.1.1. Método para o desenvolvimento do algoritmo de variabilidade 98
3.1.2. Método para o desenvolvimento do ACS e CPMod em *software* 99
3.1.3. Nós sensores 102
4. DESENVOLVIMENTO 105
4.1. Desenvolvimento do algoritmo de detecção de variabilidade 105
4.1.1. Configuração inicial do processo cognitivo 116
4.1.2. *Memory vector* 117
4.1.3. *Error vector* 118
4.1.4. *Decision Making vector* 118
4.1.5. *Control vector* 120
4.2. Modelagem do ACS 121
4.3. CPMod 124
4.4. Desenvolvimento em *Software* 126
4.4.1. Configuração geral do sistema 127
4.4.2. Implementação da Lógica *Fuzzy* 131
4.4.3. Configuração do *Prowler* 136
4.4.4. Implementação do ACS e CPMod no *software* de simulação 141
4.5. Simulação em *Software* 145
4.5.1. Configuração da camada MAC do *Prowler* 148

5. RESULTADOS OBTIDOS 150
5.1. Resultados específicos do algoritmo de variabilidade 150
5.2. Resultados das simulações no MATLAB 152
6. CONCLUSÃO 169
REFERÊNCIAS 174
APÊNDICES 185
APÊNDICE I 186
Texto auxiliar sobre o *Prowler* 186
APÊNDICE II 187
Gráficos das simulações 187
APÊNDICE III 191
Software do *Prowler* adaptado 191
APÊNDICE IV 203
Regras *Fuzzy* 203
APÊNDICE V 205
Módulo do CPMod 205
APÊNDICE VI 207
Módulo de correção do CPMod 207
ANEXOS 209
ANEXO I 210
Artigo apresentado no IEEE CSNT 2015 210
ANEXO III 217
Artigo publicado no IJCSNT 2016 217
ANEXO III 234
Datasheet do MICAz (família MICA *motes*) 234

1. APRESENTAÇÃO

O presente trabalho trata sobre o estudo de *Wireless Sensor Networks* (WSN) e *Cognitive Sensor Networks* (CSN) visando identificar possíveis melhorias em aspectos de desempenho da rede. Para tanto, serão realizadas análises de tráfego e de padrões de comportamento da rede conforme a topologia e a distribuição de dados entre os nós. A representação dos cenários nos quais os nós se encontram descreve a topologia de ambientes de *Internet of Things* (IoT) nos quais os nós se comportam como sensores e atuadores no ambiente, sendo que os cenários são baseados na arquitetura CSN.

A relação e interação entre dispositivos detectores e atuadores forma uma rede de sensores. Junto com este sistema interconectado, encontra-se o paradigma de mobilidade e de conexão *wireless*, ou seja, os estudos sobre Redes de Sensores Sem Fio (RSSF) aumentaram significativamente na última década, para detectar formas inovadoras de realizar e manter conexões entre os nós sensores mesmo que estes tenham mobilidade e velocidades variadas.

Assim como a área de redes de sensores teve avanços significativos, a ciência cognitiva proporcionou grandes avanços nas últimas décadas, evoluindo conceitos e descobertas relacionadas ao ser humano e às relações de atitudes entre os indivíduos, desenvolvendo sistemas mecânicos dinâmicos e em forma de *software*, que utilizam diferentes técnicas para inserção de “inteligência”, abrangendo a neurociência e incluindo Redes Neurais Artificiais (RNA) e Inteligência Artificial (IA).

Conforme Johnson (2013), as tecnologias de comunicação sem fio que podem ser utilizadas para uma rede de sensores são variadas, como por exemplo, *General Packet Radio Service* (GPRS), *Wireless-Fidelity* (Wi-Fi), *Bluetooth*, e *ZigBee*.

A Tabela 1 mostra um comparativo entre as tecnologias destacadas. Pode-se observar, que conforme a aplicação escolhida, a inserção de nós na rede deve seguir diferentes parâmetros, uma vez que as tecnologias de comunicação sem fio variam nas métricas de sucesso, conforme a característica *Success metrics*.

Tabela 1 – Comparativo das tecnologias de comunicação sem fio.

Parâmetros	Tecnologias e Padrões			
	***ZigBee* 802.15.4**	***Bluetooth* 802.15.1**	***Wi-Fi* 802.11b**	**GPRS/GSM 1xRTT/CDMA**
Recursos do Sistema	4 - 32 kB	250 kB +	1 MB ±	16 MB +
Tempo de vida da bateria (dias)	100 – 1000 +	1 – 7	0.1 – 5	1 – 7
Quantidade de nós por rede	255 – 65000 +	7	30	1 – 1000
Largura de banda (kbps)	20 – 250	720	11000 +	64 – 128
Alcance (m)	1 – 75 +	1 – 10 +	1 – 100	1000 +
Foco da aplicação	Monitoramento e controle	Substituição de cabos	Web, e-mail e vídeo	Envio de voz e dados sem fio
Métricas de sucesso	Confiabilidade, Segurança da rede, Simplicidade do protocolo, Baixo consumo de energia	Baixo custo para incremento de nós, Facilidade de uso / conveniência, Taxa de envio de dados moderada	Alta taxa de entrega de dados entre os nós, Flexibilidade (uso comercial ou residencial), Conectividade com *Hot Spot*	Grande area de cobertura, Planos de precificação baseados em dados, Rede utilizada em áreas urbanas

Fonte: Adaptado de Johnson (2013).

A interconexão de redes cognitivas com as redes tradicionais *wireless* e cabeadas, acrescenta mais flexibilidade de inserção de novas aplicações e formas de transferência de dados, contribuindo desta forma para a universalidade das comunicações e o aprimoramento de tecnologias da informação, baseando-se para isto em métricas, políticas de rede e semântica do processo cognitivo.

1.1 Justificativa

Este trabalho abrange aspectos relevantes no que se refere à evolução de redes, mais especificamente às redes de sensores. As WSN são redes que podem ser utilizadas nos mais variados ambientes com quantidades que podem variar de um a milhares de nós sensores. Servem também como base para implementação de cenários IoT, dentre as inúmeras formas de configuração possíveis. As aplicações podem variar desde uma utilização residencial, industrial e de controle, até o monitoramento de ambientes hostis como aéreos e submersos.

O uso de WSN possibilita um grande avanço no que tange ao mecanismo de coleta de dados, bem como aos sistemas de controle. Alcaraz (2010) define como claro o uso das WSN como parte integrante conectada à rede Internet, portanto, tornando-se parte da IoT. Porém, menciona também, que existem problemas de segurança que devem ser levados em consideração quando se realiza esta integração, tais como mecanismos e serviços de segurança, aceitação dos usuários e gestão da privacidade dos dados que são roteados na rede.

Wang (2010), em seu trabalho sobre RSSF comenta que esta tecnologia surgiu como uma técnica de rede promissora, graças aos recentes avanços em eletrônica, na área de redes e de tecnologias de informação. No entanto, ainda existe uma grande lacuna de pesquisa na qual são realizados esforços adicionais necessários para que se torne uma tecnologia madura. Apresentam-se três fatores que dificultam o desenvolvimento de redes de sensores: falta de análise de tráfego e modelagem da rede para RSSF; otimização da rede; e desenvolvimento de técnicas de detecção de anomalias. Dentre esses fatores, o entendimento sobre a dinâmica de tráfego dentro da RSSF fornece uma base para posteriores trabalhos vinculados à otimização e detecção de anomalias na rede.

A linha de pesquisa desta Tese não se limita apenas às RSSF comuns e já conhecidas, mas traz principalmente a inclusão de inteligência na rede com a inserção de análise de dados coletados e tomada de decisão, o que caracteriza uma análise de rede do tipo CSN. Estas redes se baseiam em RSSF com inserção de aspectos de *Artificial Intelligence* (AI). Dentre estes aspectos, serão analisados: aprendizado (de rotas e/ou dados de ambientes), tomada de decisão (que pode ocorrer sobre nós individuais ou apenas em um nó central controlador) e histórico do ambiente da rede (referente aos fatos ocorridos na rede, porém com restrição, devido à limitação de memória para armazenamento de informações pelos nós).

Como contribuições específicas desta Tese, tem-se a inserção de inteligência baseada no aprendizado com histórico da rede e tomada de decisão e, também, o estudo de convergência. A convergência foi estudada para redes de 50 a 200 nós, sendo verificadas limitações ocasionadas pela densidade de nós na área de cobertura da rede e à mobilidade dos nós.

Diante dessa gama de motivações e da necessidade de aplicações que possam garantir mais eficiência da rede, deseja-se estudar o comportamento de nós sensores em CSN para diversas situações: variação na quantidade de nós sensores nos cenários de simulação, alteração de parâmetros dos sensores e da rede conforme análise realizada pelo sistema desenvolvido nesta Tese, verificação do desempenho da rede de acordo com os *rounds* e modificação da posição dos nós com representação da mobilidade dos sensores. Estes estudos sobre redes de última geração foram realizados em ambiente de simulação no *software* MATLAB, com os resultados dos testes nos cenários mostrando a eficiência do sistema para maior quantidade de *rounds*.

1.2 Contexto

A tecnologia voltada para detecção e atuação de mecanismos em benefício do ser humano vem progredindo rapidamente, o que possibilita utilizar dispositivos sensores em ambientes antes não explorados, como por exemplo, em uso doméstico e em áreas rurais, proporcionando conforto e praticidade aos usuários destes sistemas, além de benefícios relacionados à aquisição, armazenamento e manutenção de dados oriundos destas redes de sensores (UFPA, 2013).

A RSSF representa uma das tecnologias envolvidas no paradigma computacional da IoT (IOT BRASIL, 2014). A padronização da RSSF é feita pelo *Institute of Electrical and Electronics Engineers* (IEEE) com abordagem de características gerais e técnicas (IEEE, 2013) e pelo *Internet Engineering Task Force* (IETF). A seguir, apresenta-se uma reflexão sobre o avanço das tecnologias de redes de sensores e da importância destas para diversas áreas do conhecimento.

> *"Na última década, houve um grande avanço tecnológico nas áreas de sensores, circuitos integrados e comunicação sem fio, que levou a criação de redes de sensores sem fio. Este tipo de rede pode ser aplicada no monitoramento, rastreamento, coordenação e processamento em diferentes contextos. Por exemplo, pode-se interconectar sensores para fazer o monitoramento e controle das condições ambientais numa floresta, oceano ou um planeta. A interconexão de sensores através de redes sem fio, com a finalidade de executar uma tarefa de sensoriamento maior, irá revolucionar a coleta e processamento de informações."* (LOUREIRO, 2003)

Diante da ideia exposta, este trabalho traz não somente a RSSF de forma

comum, mas com a introdução de recursos cognitivos que auxiliarão os nós da rede a tomarem decisões acerca do funcionamento, conforme processos de análises implementados no sistema desenvolvido.

1.3 Objetivos gerais

Esta Tese visa ao estudo de RSSF no contexto de ciência cognitiva, o que torna a rede “inteligente” e intuitivamente remete a uma melhora no desempenho (no que se refere ao uso de recursos da rede).

No contexto de criação do algoritmo desta Tese, e respectiva análise de características que são identificadas nos sensores que compõem a rede, foi definida a variabilidade vinculada aos nós da rede, que contempla a movimentação do nó (com a leitura de posição do nó), a variação de alcance entre os nós (que abrange o ganho das antenas transmissoras), a variação da quantidade de conexões possíveis e atuais dos nós (*links* efetuados entre os nós) e a variação na quantidade de nós na rede (para determinar a dinâmica na quantidade de possíveis conexões).

Para a verificação da variabilidade dos nós sensores, propõe-se o desenvolvimento de um algoritmo baseado nas características básicas descritas dos nós em ambiente de WSN. Este algoritmo deve prover a detecção de variabilidade nos nós que compõem a rede.

Visando a inserção de cognição em RSSF, propõe-se também a criação de um mecanismo de análise e atuação sobre características *multi-layer* dos nós, com base em informações coletadas da rede. Este mecanismo deve ter característica cognitiva e prover tomada de decisão.

1.4 Objetivos específicos

Para o estudo de inteligência em RSSF, é necessário criar um sistema que realize a inserção de mecanismos ativos no ambiente de rede. Para tanto, escolheu-se a análise por uso de simulação de rede em *software*, pois nesta formatação é possível a verificação e análise de um grande número de nós e facilidade para variação na quantidade de sensores.

Pretende-se então, realizar a escolha do uso de um *software* de simulação para WSN, que contemple a inserção e utilização de sequências de análises intermediárias com o apoio de mecanismos de decisão. Portanto foi escolhido o *software* MATLAB (OPENCADD, 2015) para as simulações e desenvolvimento do sistema cognitivo, com uso de Lógica *Fuzzy* e de implementação de *rounds* via *Simulink*.

Para a verificação da variabilidade dos nós, pretende-se desenvolver um algoritmo que contemple quatro características relativas ao comportamento ou propriedade dos nós na rede, que são movimentação do nó, ganho das antenas transmissoras, número de conexões com nós adjacentes e quantidade de nós na rede. Estas características foram escolhidas com base na análise que está detalhada no Capítulo 4.

1.5 Motivação

Com a crescente evolução dos mecanismos de coleta de dados e a importância de se existir um sistema de controle para que haja otimização e monitoramento de fenômenos da rede, este trabalho visa à exploração das características correspondentes ao tráfego entre os nós sensores que fazem parte da rede, com análise de dados coletados e de transferência de informações. Junto a este conjunto de redes de sensores, existe uma evolução que engloba a área de ciência cognitiva, com as CSN que além de possuírem as características de coleta e encaminhamento de informações, possuem aspectos cognitivos.

Na literatura atual existe uma carência por mecanismos que realizam controle e redimensionamento da rede para que seja possível estabelecer ou alterar características que melhorem a *Quality of Service* (QoS). Neste contexto, esta pesquisa visa a contribuir para que existam alternativas que possam ser adotadas nos ambientes de redes do tipo RSSF e CSN.

A ideia básica para o sistema desenvolvido nesta Tese surgiu através do estudo aprofundado em redes de sensores e em ciência cognitiva, visualizando-se a necessidade de formas mais avançadas de otimização de rede com a junção das duas áreas. Verificou-se que o foco do *Cognitive Radio* abrange as camadas mais

baixas (enlace e física) e que existe uma lacuna para alcançar o desempenho esperado de QoS em redes de sensores, sendo que este tema vem sendo tratado em muitas pesquisas focadas em camadas específicas, como a de Rede ou Roteamento.

1.6 Estado da arte

Existem muitos estudos sobre WSN, CSN, IoT e redes em geral, que buscam a melhoria de aspectos de desempenho e alternativas de roteamento de redes e como esta Tese trata sobre a criação de um dispositivo que faz a leitura de aspectos de desempenho de RSSF e que atua sobre a mesma com a finalidade de realizar ajustes *multi-layers* em aspectos relativos às métricas de rede, segue a relação de estudos da área de pesquisa desta Tese e vinculados aos assuntos mencionados.

Além disso, esta Tese está relacionada ao desenvolvimento de um algoritmo de variabilidade dos nós sensores, à criação de uma arquitetura adaptativa denominada ACS e à implementação de um módulo de controle CPMod, juntamente com o DL e V_C em *software*.

Segundo a IDTechEx (2013), os telefones celulares estão conectando as pessoas a outras pessoas e também a outros tipos de dispositivos, com crescimento de forma linear. A automação *Machine-to-Machine* (M2M) com sensores sem fio está sendo inserida gradativamente, mas geralmente isoladas em diversos sistemas distintos. A evolução deste cenário encontra-se na IoT, que tenta conectar esses sistemas isolados e logo em seguida aumentá-los com a ligação de mais “coisas”. Neste panorama os dados são enviados pela rede, possibilitando que esta tome suas próprias decisões automatizadas, enquanto que os dados sobre as preferências do usuário são transportados automaticamente e de forma discreta e transparente, mesmo com a movimentação dos dispositivos. Portanto, nessa visão, qualquer ambiente em que se estiver será adaptado às necessidades do usuário da rede, sem que seja exigida qualquer atuação externa sobre preferências ou ainda, as decisões sobre alterações serão bem mais simples de executar.

Vijay (2011) acredita que as WSN são a base que permite uso de tecnologia com inteligência nos ambientes em geral. Para tanto, proporcionam um meio de

comunicação capaz de permitir a criação de uma rede com manipulação de aplicações que evoluem a partir de necessidades do usuário. Dotadas de agentes cognitivos capazes de tomar decisões proativas baseadas em aprendizado, raciocínio e compartilhamento de informação, quando introduzidas em redes de sensores, podem ajudar a alcançar as metas fim-a-fim da rede, mesmo na presença de restrições e de objetivos complexos para otimização.

O *Cognitive Radio* (CR) nas camadas física e de enlace dos agentes móveis pode habilitar o uso oportunista do ambiente heterogêneo sem fio, com a sensibilidade sobre o meio físico de transmissão de informações e uso inteligente dos canais disponíveis na rede (MITOLA III, 2000).

Os cenários IoT podem ser configurados juntamente com CSN, formando uma rede *Cognitive Internet of Things* (CIoT) conforme explica Zhang (2012). Segundo Zhang, IoT é uma rede ubíqua heterogênea, misturada e incerta, na qual têm-se a perspectiva de ampliação para serviços modernos e mais inteligentes.

Zhang realizou uma investigação profunda sobre as discrepâncias entre o serviço, a oferta e a exigência de aplicação, acreditando-se que a atual IoT não tem inteligência suficiente para suprir todas as necessidades, pois não alcança o desempenho esperado das crescentes aplicações. A CIoT pode perceber as condições atuais da rede, analisar o conhecimento percebido, tomar decisões inteligentes e executar ações adaptativas, que objetivam a maximizar o desempenho da rede. A topologia foi modelada como rede CIoT relacionada com o processo de cognição, que tem base na Teoria dos Jogos, considerada como um dos modelos de decisão que visa a potencializar o pensamento estratégico, sendo denominada também como a ciência da estratégia. Concluiu-se que estas novas concepções podem dotar a IoT com inteligência e melhorar completamente o desempenho do sistema.

Esta Tese apresenta um mecanismo *multi-layer*, ou seja, trata-se de um sistema capaz de efetuar medidas na rede relacionadas ao desempenho e ainda operar modificações em diferentes camadas dos nós e da rede. Zhang aplicou o conceito de Teoria dos Jogos, enquanto que neste trabalho será usada a cognição com aplicação de Lógica *Fuzzy* para a tomada de decisões. O mecanismo

desenvolvido aqui se aplica às redes *wireless* com múltiplos sensores, *multi-hop* e com configurações de topologia não definidas, ou seja, com liberdade de mobilidade dos nós. Portanto, pode ser aplicada em quaisquer ambientes de sensores móveis, alcançando até ambientes IoT.

1.7 Contribuição

Dentre os diferentes campos de pesquisa em que esta Tese está situada, uma das contribuições refere-se à inovação de características de análise da rede, com uso de Lógica *Fuzzy* em *rounds* de análise de uma WSN com aplicação de aspectos cognitivos específicos: memória (lembrança), histórico e tomada de decisão.

Decidiu-se utilizar o *software* MATLAB com auxílio do *Prowler* (BERKELEY, 2016) adaptado para a realização de simulações dos cenários de rede, incluindo topologia, arquitetura e roteamento de dados, juntamente com o uso do *toolbox* de Lógica *Fuzzy*, no qual foram inseridas as regras, inferência, fuzzificação e defuzzificação, além de desenvolvimento do sistema de análises de *rounds* via uso do *Simulink* e de alterações de programação nos arquivos base do *Prowler*. Para as análises, escolheu-se o método de coleta de dados em *rounds*, ou seja, a cada iteração executada na rede, existe a análise dos dados para verificação junto à semântica adotada, e caso os dados estejam de acordo com a semântica, o resultado é armazenado em um arquivo para posterior uso na execução do processo de lembrança. Caso contrário, novas análises de *rounds* são realizadas.

No cumprimento dos objetivos propostos, desenvolveu-se o *Adaptive Cognitive System* (ACS), que é uma arquitetura de sistema *multi-layer* que trabalha com o monitoramento e atuação sobre diferentes camadas da WSN, abordando as seguintes *layers*: física, enlace e rede. Este sistema foi elaborado com característica de atuar diretamente nas camadas executando alterações em aspectos dos nós da rede, conforme resultados dos *rounds*, visando contemplar as diretrizes estipuladas na semântica e política da rede, buscando sempre adequação de acordo com a semântica.

No ACS, caso a “solução ótima” não seja encontrada, ou seja, que os valores

encontrados pelo CPMod após a análise sobre o limite de decisão e a Lógica *Fuzzy* estejam em desacordo com os valores estipulados na semântica, o resultado a ser adotado será baseado na formatação mais próxima ao contido na semântica. Para tanto, o sistema possui 4 interfaces de entrada e adequação, que estão presentes no módulo do processo cognitivo. Como módulo principal para o ACS, desenvolveu-se o *Cognitive Processor Module* (CPMod), que agrega interfaces de entrada e saída, consultas e alterações nos blocos de armazenamento e de Lógica *Fuzzy*, o conjunto de dados da semântica e interações com o *software* de simulação.

Para representar as características do nó na WSN, desenvolveu-se um algoritmo que detecta a variabilidade do nó sensor conforme 4 condições, que contemplam variação de posição, alcance da antena, quantidade de *links* estabelecidos pelos nós e quantidade de nós na rede.

Foram estabelecidos também os intervalos relacionados aos limites de decisão da Tabela 6, baseando-se em um diagrama composto por 5 níveis ou estados de abstração, que proporcionaram base para a implementação das regras do bloco *Fuzzy* que foi implementado em *software*. Além disso, foram desenvolvidas as funções de pertinência da Lógica *Fuzzy* para a fuzzificação e defuzzificação, além da criação das regras *Fuzzy* a serem aplicadas na inferência dos dados de entrada do bloco *Fuzzy*.

Em relação à QoS, a contribuição está relacionada com a obtenção de melhora no desempenho da rede, do ponto de vista do usuário da WSN, pois foram realizados testes sobre energia dos nós, quantidade de memória, fila de processamento e protocolos de roteamento, para análises de serviços abrangendo *throughput*, latência, quantidade de pacotes enviados, taxa de perda de pacotes, taxa de sucesso de pacotes e quantidade de energia.

Estipulou-se uma variação de 50 a 200 nós na rede, contemplando WSN com característica que abrange rede em larga escala (pode-se considerar acima de 100 nós). Como as redes simuladas possuem muitos nós, executa-se a transmissão dos dados e encaminhamento dos pacotes via múltiplos saltos. Atribuiu-se a Função Densidade de Probabilidade (FDP) do tipo Rayleigh para o canal de transmissão e recepção de sinais de dados na rede simulada, aplicando o efeito de

desvanecimento ao meio de transmissão dos nós.

Como contribuições específicas e pontuais, destacam-se:

- Escolha e avaliação de um modelo de canal e características de propagação via uso de *Radiofrequency* (RF) no *Prowler*;
- Desenvolvimento de interface de simulação no *Simulink* do MATLAB e integração do mesmo com o *Prowler*;
- Elaboração e verificação do bloco de Lógica *Fuzzy*, para integração com o CPMod;
- Avaliação e verificação do CPMod, incluindo os resultados dos *rounds* de análises realizados para cada cenário;
- Avaliação e dimensionamento dos cenários de testes, abrangendo a dinâmica da topologia de redes *Ad Hoc*;
- Avaliação e comparação dos efeitos resultantes do processo cognitivo para cenários antes e após a execução do processo;
- Avaliação e verificação do efeito do sistema desenvolvido com relação à camada de enlace, *Medium Access Control* (MAC) *layer*, na presença de *Carrier Sense Multiple Access* (CSMA).

Foi apresentada parte deste trabalho referente ao desenvolvimento matemático no IEEE *Fifth International Conference on Communication Systems and Network Technologies* (CSNT) 2015, de 4 a 6 de abril de 2015, Gwalior, Índia, através do *paper Adaptive Cognitive System Applied on Wireless Sensor Networks Nodes Decisions* (WAGNER, 2015).

Parte deste trabalho que está relacionada com a implementação do mecanismo desenvolvido em Lógica *Fuzzy* foi publicada no *International Journal of Communication System and Network Technology* (IJCSNT), vol. 5, *issue* 01, em fevereiro de 2016, com publicação on-line, com o *paper* intitulado *Adaptive Cognitive System Applied to WSN Decisions at Nodes with a Fuzzy Logic Approach* (WAGNER, 2016).

1.8 Estrutura do trabalho

- Capítulo 1 – Apresentação: apresenta justificativa, contexto, objetivos gerais e específicos do trabalho, bem como motivação, estado da arte e contribuição, através de uma visão geral do contexto da Tese com enfoque no panorama tecnológico e na relevância do assunto desenvolvido;
- Capítulo 2 – Fundamentação Teórica: apresenta os conceitos referentes à evolução dos sistemas, incluindo respectivas arquiteturas e funcionamentos, que formam a base para a tecnologia de rede estudada neste trabalho, além de conceitos de cognição e de redes cognitivas, incluindo aspectos e aplicações;
- Capítulo 3 – Metodologia: apresenta a metodologia utilizada, bem como o método científico, parâmetros gerais do trabalho e escolha do *software* utilizado para a simulação dos cenários propostos;
- Capítulo 4 – Desenvolvimento: apresenta abordagem matemática e testes no *software* de simulação escolhido (MATLAB) com respectivas descrições dos cenários e análise dos dados coletados, bem como o funcionamento e a descrição das construções de simulações realizadas neste trabalho;
- Capítulo 5 – Resultados Obtidos: apresentação e análise dos dados coletados através das simulações do capítulo anterior, bem como das métricas coletadas da rede e de formatação dos nós, contemplando gráficos e tabelas;
- Capítulo 6 – Conclusão: apresenta as considerações finais deste trabalho, indicando as principais contribuições, o cumprimento dos objetivos propostos e sugestões de continuação de pesquisa;
- Referências: apresenta as referências utilizadas como auxílio para a elaboração desta Tese;
- Apêndices: apresentam uma breve explicação sobre o *software Prowler*, os dados relativos ao desenvolvimento em *software*, gráficos e tabelas auxiliares que foram obtidos durante as análises e simulações;
- Anexos: apresentam o artigo do CSNT 2015 e do IJCSNT 2016, bem como os documentos que auxiliam no entendimento desta Tese.

2. FUNDAMENTAÇÃO TEÓRICA

Apresenta uma descrição sucinta sobre alguns tópicos, que abrange áreas distintas como ciência cognitiva e rede de sensores. Estas redes podem ser cabeadas ou sem fio, e como no escopo trata-se sobre nós sensores com mobilidade, serão expostas especificamente as tecnologias de sensores que tenham característica *wireless*. Serão explicados também alguns conceitos de cognição para melhor entendimento do funcionamento do sistema desenvolvido nesta Tese.

2.1. RSSF

Uma Rede de Sensores Sem Fio (RSSF) consiste basicamente em sensores autônomos distribuídos em um ambiente qualquer para monitorar condições físicas ou ambientais, tais como temperatura, pressão, movimento, entre outros. Outra atividade do nó pode ser a retransmissão de informações oriundas de outros nós.

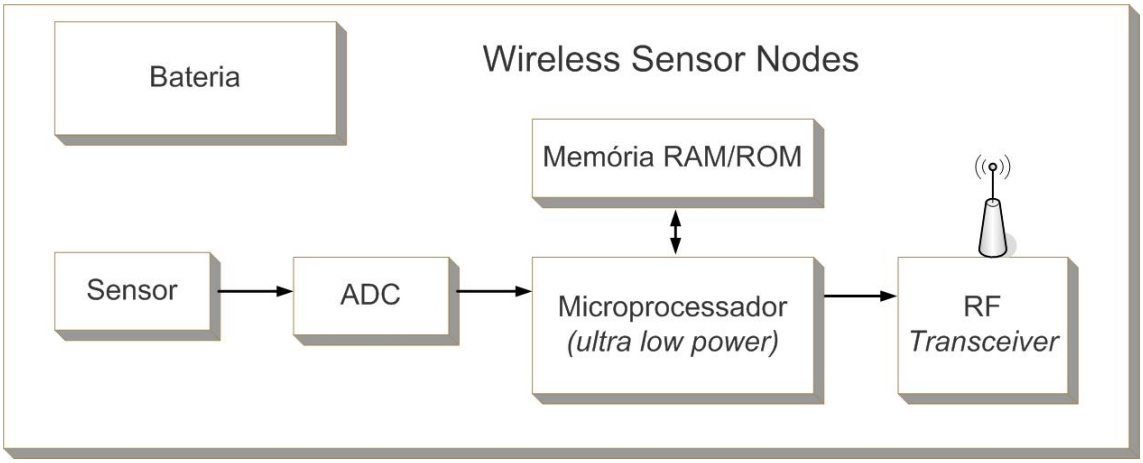
Geralmente os dados coletados pelos sensores são encaminhados através da rede para um local central, que é responsável pelo processamento das informações coletadas. As redes mais modernas são bidirecionais, ou seja, é possível também executar o controle da atividade dos sensores.

Inicialmente, o desenvolvimento de redes de sensores foi motivado por aplicações na área militar para coleta de dados, assim como em diversos outros ramos da tecnologia. Atualmente, estas redes também são utilizadas para monitoramento de: processos industriais, controle e monitoramento hospitalar, e outras aplicações nos mais diversos ramos de atividades da sociedade.

Cada um dos nós é composto normalmente por partes como: transmissor de rádio com uma antena interna (ou conexão para uma antena externa), microcontrolador, circuito eletrônico para fazer a interface com os sensores e fonte de energia (geralmente uma bateria ou uma forma integrada de captação de energia alternativa). Um nó sensor pode ter diferentes dimensões, e pode ser do tipo *Sink*, ou seja, um nó centralizador de informações, com dimensões desde o tamanho de uma caixa de sapatos até um grão de areia, pois com o avanço da nanotecnologia, alguns modelos passaram a ser criados com dimensões microscópicas, mantendo a

mesma ou proporcionando aumento da capacidade de processamento. Para este trabalho será utilizado o nó do tipo MICAz (de tamanho $5{,}8 \times 3{,}2 \times 0{,}7$ cm^3), conforme ANEXO III.

Figura 1. Estrutura básica interna em blocos de um nó sensor.



Fonte: Adaptado de Wang (2013).

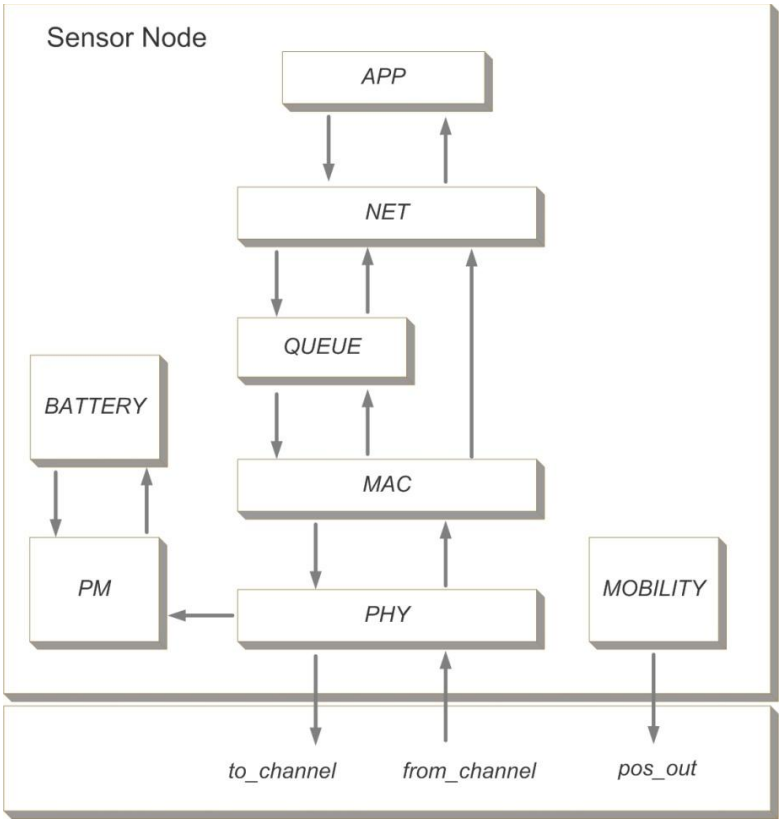
Na Figura 1, os módulos em blocos referem-se ao sensor que faz leitura do meio em que se aplica, *Analog-to-Digital Converter* (ADC) ao conversor de sinais analógicos para digitais, microprocessador (de baixa potência para baixo consumo de energia), memórias *Random Access Memory* (RAM) e *Read-only Memory* (ROM), e RF *transceiver* correspondente ao transmissor em radiofrequência na banda escolhida para a rede.

Conforme Sohraby (2007), o custo dos sensores varia de acordo com a aplicação e tamanho, ou seja, depende do ambiente no qual será utilizado, contemplando características como parâmetros de precisão e condições climáticas (se está vulnerável à chuva, se o ambiente é submerso, com alto grau de temperatura, etc.). Como limitações para a aplicação dos nós sensores em diferentes ambientes, podem-se destacar o tamanho, as restrições de custo e de recursos, como energia, memória, velocidade computacional e largura de banda de comunicação.

A topologia das redes de sensores pode variar de uma rede simples com poucos nós estáticos, até uma rede avançada sem fio do tipo *mesh* com característica *multi-hop* com muitos nós sensores e mobilidade. A técnica de propagação entre os sensores da rede, que influencia no número de saltos até o

destino, pode ser de encaminhamento de pacotes ou de *broadcast* dependendo do objetivo da aplicação (DARGIE, 2010).

Figura 2. Estrutura básica de conexão interna de um *wireless sensor node*.



Fonte: Adaptado de Wang (2013).

Na Figura 2, observa-se a existência de um fluxo interno, no qual *APP* corresponde à aplicação, *NET* à camada de rede (*network*), *QUEUE* à fila de dados (caso seja necessária para desafogar o processamento interno do nó), *MAC* à camada de acesso à rede e *PHY* à camada física. Ainda na estrutura interna do sensor, tem-se a parte relativa à *BATTERY*, que corresponde ao bloco de armazenamento de energia e *PM* ao gerenciamento de energia. A parte relativa à mobilidade do nó fica atribuída ao módulo *MOBILITY*. A estrutura principal de comunicação com o meio fica atribuída à camada PHY, que faz interface com o canal de comunicação (*to_channel* / *from_channel*) e o módulo de mobilidade gera como saída um vetor de posição (*pos_out*).

É através da camada física, que ocorre a interação do nó com a rede, na transmissão e recebimento de dados. A camada física da Figura 2 tem relação direta com o bloco RF *transceiver* da Figura 1.

2.2. Transmissão *wireless*

Para visualizar melhor as diferenças entre as diversas tecnologias de transmissão *wireless*, tem-se a Tabela 2, que demonstra a aplicação, área de cobertura e frequência de operação de cada uma das tecnologias existentes no mercado de telecomunicações. As tecnologias mais próximas das estudadas neste trabalho encontram-se entre o *Ultra Wide Band* (UWB), *Bluetooth*, *Wireless-Fidelity* (Wi-Fi) e *ZigBee*, devido à área de cobertura.

Tabela 2 – Comparação entre variadas tecnologias de transmissão sem fio.

Tecnologia	Padrão	Aplicação	Área de Cobertura (m)	Frequência (GHz)
UWB	802.15.3a	WPAN	10	3,1 – 10,6
Bluetooth	802.15	WPAN	1 (classe3) / 10 (classe2) / 100 (classe1)	2,4
Wi-Fi	802.11a	WLAN	25 – 100 (*indoor*)	5
Wi-Fi	802.11b	WLAN	100 – 150 (*indoor*)	2,4
Wi-Fi	802.11g	WLAN	100 – 150 (*indoor*)	2,4
Wi-Fi	802.11n	WLAN	70 (*indoor*) / 250 (*outdoor*)	2,4 ou 5
ZigBee	802.15.4	WPAN / WLAN	10 – 100	0,8, 0,9 ou 2,4
WiMax	802.16d	WMAN	6400 a 9600	11
WiMax	802.16e	*Mobile*/WMAN	1600 a 4800	2 a 6
WCDMA	3G	WWAN	1600 a 8000	1,8, 1,9 ou 2,1
CDMA2000	3G	WWAN	1600 a 8000	0,4, 0,8 ou 0,9
1xEV-DO	3G	WWAN	1600 a 8000	1,7, 1,8, 1,9 ou 2,1
MBWA	802.20	WWAN	4000 a 12000	3,5

Fonte: Adaptado com atualização de Wagner (2009).

As tecnologias apontadas na Tabela 2 para WMAN, WLAN e WPAN se encaixam dentro das normas relativas ao Serviço de Comunicação Multimídia (SCM) por oferecerem transmissão, emissão e recepção de informações multimídia, permitindo inclusive o provimento de conexão à internet, utilizando meios de comunicação *wireless*, a usuários que estejam dentro da área de prestação do serviço (ANATEL, 2013).

Para a simulação da WSN desta Tese foi escolhido o padrão 802.15.4 correspondente ao *ZigBee*, com nós do tipo MICAz (ANEXO III), devido ao alcance ou ganho da antena dos nós, facilidade para a implementação em *software* de simulação e frequência que permite operação multi-banda.

2.2.1. *ZigBee*

O termo *ZigBee* corresponde a um conjunto de especificações que destinam-se a comunicação sem fio entre dispositivos eletrônicos e possui características de baixa potência de operação, taxa de transmissão de dados razoável e custo reduzido para implementação. Este conjunto de especificações pode ser correlacionado com as camadas OSI (*Open System Interconnection*), conforme estabelece o padrão IEEE 802.15.4 (IEEE, 2011). A rede *ZigBee* foi idealizada para interligar pequenas unidades que coletam dados locais, juntamente com o respectivo controle, utilizando parte do espectro de radiofrequências no qual não existe a necessidade de licenciamento para uso de RF no Brasil, porém, os dispositivos devem ser homologados pela Agência Nacional de Telecomunicações (Anatel) (BRASIL, 2008).

A tecnologia utilizada pode ser comparada às usadas em redes *Wi-Fi* e *Bluetooth*, diferenciando-se destas pela característica de possuir menor consumo de energia para um alcance reduzido de cerca de 100 m. No *ZigBee*, todos os nós podem funcionar como retransmissores da informação. Por exemplo, uma topologia *Mesh* de unidades de rede *ZigBee* pode ser feita em um ambiente doméstico ou não, sem a necessidade de utilização de conexões elétricas entre os nós.

Atualmente é possível encontrar módulos *ZigBee* com taxas de aproximadamente 250 kbps que alcançam aproximadamente 70 m em visada direta. O sistema pode funcionar com ou sem criptografia dos dados gerados e trafegados na rede e, no caso de uso de criptografia, podem ser configurados o nome de usuário, o endereço na rede, os endereços que receberão os dados e as mensagens numa determinada comunicação. Outros módulos podem ser utilizados para vídeo, imagens, ou mesmo para comunicação com microcontroladores e com um *Personal Computer* (PC) (JAMAJ, 2013).

2.2.2. *Bluetooth*

Bluetooth é uma tecnologia de comunicação sem fio para pequenas distâncias, destinado a substituir os cabos que conectam muitos tipos de dispositivos locais, como telefones celulares e fones de ouvido, monitores auditivos e

equipamentos médicos, entre outros. A relação de eficiência de energia do *Bluetooth* o torna ideal para aplicação em dispositivos que necessitam funcionar com uma pequena bateria por longos períodos.

Nas primeiras realizações, há 15 anos, a IoT teorizou que objetos e as pessoas poderiam se conectar via comunicação sem fio. O *Bluetooth* clássico forneceu os meios para esta comunicação e fabricantes de dispositivos focaram na construção de dispositivos como PCs, *Smartphones*, *Tablets* e integração com navegadores de mídia em carros para tirar proveito dessas conexões. De acordo com a ABI *Research*, o projeto de introduzir 30 bilhões de dispositivos entrará no sistema IoT em 2020 (BLUETOOTH, 2015).

Para os consumidores, isso se resume à Internet intrinsecamente conectada aos dispositivos eletrônicos e utensílios domésticos. Por exemplo, em uma corrida portando um monitor de frequência cardíaca, este dispositivo pode interagir diretamente com um relógio inteligente, ou mesmo quando se escova os dentes utilizando uma escova com sensor embutido enquanto se ouve música através do chuveiro, ou ainda pode-se monitorar os filhos enquanto jogam basquete com uma bola "inteligente" e sapatos com sensores. Pode-se também dispor a tecnologia para um ambiente residencial para desbloquear portas, acender luzes e executar comando para acionar uma *smart* TV com um *Smartphone*.

Bluetooth é uma tecnologia amigável, ou seja, de fácil utilização, cujo aplicativo é suportado por todos os principais sistemas operacionais disponíveis, inclusive em dispositivos móveis. Os custos para implantação da tecnologia são baixos e o *Bluetooth* oferece uma arquitetura de desenvolvimento flexível e prática para criar aplicações embarcadas em *Smartphones*, *Tablets* ou outros dispositivos semelhantes (BLUETOOTH, 2015).

2.2.3. Wi-Fi

Conhecido como *Wireless-Fidelity* (Wi-Fi), o padrão opera em faixas de frequências que não necessitam de licença para instalação e/ou operação, tornando-se atrativo para a inserção no mercado de telecomunicações. No entanto, para uso comercial no Brasil, é necessário que o equipamento seja homologado pela Anatel

(BRASIL, 2008). As frequências são livres de licença, ou seja, o usuário dos serviços não paga taxa para utilização, porém são permitidos apenas equipamentos que tenham sido analisados, avaliados e com certificado de homologação, sendo que esses equipamentos recebem um selo de identificação fornecido pela Anatel.

Para ter acesso à Internet através de uma rede Wi-Fi, o dispositivo deve estar dentro do raio de atuação (área de cobertura) de um ponto de acesso (tecnicamente conhecido por *hotspot*) ou local público onde estiver disponível a rede sem fio, realizando a conexão por um computador, dispositivo portátil, *Tablet*, PC, PDA ou *Smartphone*, com capacidade de comunicação sem fio, deixando o usuário conectado à Internet através de uma conexão Wi-Fi (WI-FI ALLIANCE, 2015).

Este esquema de conexão oferece várias vantagens, tais como: permitir ao usuário utilizar a rede em qualquer ponto dentro dos limites de alcance da transmissão; possibilidade da inserção rápida de outros computadores e dispositivos na rede; evitar que paredes ou estruturas prediais sejam furadas ou adaptadas para a passagem de fios; e mobilidade em pequenas distâncias, dentro da área de abrangência.

A flexibilidade do Wi-Fi é tão grande que se tornou viável a implementação de redes que fazem uso desta tecnologia nos mais variados lugares, principalmente pelo fato de diminuição de custos. Desta forma, é comum encontrar redes Wi-Fi em locais como: hotéis, aeroportos, rodoviárias, bares, restaurantes, shoppings, escolas, universidades, escritórios, hospitais, etc. Para utilizar estas redes, basta o usuário ter um dispositivo compatível e caso a rede Wi-Fi possua *login* e senha, efetuar a inserção destes dados (INFOWESTER, 2014).

A evolução do Wi-Fi desde 1999 tem sido marcada por avanços contínuos e com um compromisso de interoperabilidade com o legado de versões anteriores. Os primeiros produtos Wi-Fi possuíam velocidade de 11 Mbps de dados, enquanto que dispositivos Wi-Fi mais avançados, atualmente podem chegar a velocidades de aproximadamente 1 Gbps de taxa de dados ao se comunicar com outra tecnologia de geração atual.

Apenas alguns anos após a sua inserção em *laptops* e pontos de acesso, e

impulsionado por tecnologias desenvolvidas pela Wi-Fi *Alliance* abordando facilidade de uso e QoS, muitos avanços em velocidade e conectividade entre dispositivos foram introduzidos com Wi-Fi *Direct*®, exposição *easy-to-use* com *Miracast*® e acesso contínuo a *hotspots* Wi-Fi com *Passpoint*™ (WI-FI ALLIANCE, 2015).

No Brasil e em outros países esta tecnologia vem sendo disponibilizada em áreas públicas com acesso liberado para quaisquer pessoas que queiram utilizar-se desta conexão para acesso à Internet ou troca de dados entre dispositivos móveis.

2.2.4. *Ultra Wide Band*

Ultra Wide Band (UWB) baseia-se em uma forma de transmissão que ocupa uma largura de banda muito larga, que normalmente estende-se em muitos GHz. Este aspecto permite encaminhar dados em velocidades que chegam a taxas de dezenas de Gbps. Estas taxas muito altas de transmissão de dados são compatíveis com as tecnologias atualmente implantadas, tais como 802.11a, b, g, WiMax (*Worldwide Interoperability for Microwave Access*) e outras. A utilização em certo número de áreas facilita a implementação do UWB, enquanto que o *Bluetooth*, USB (*Universal Serial Bus*) sem fio e outras tecnologias estão procurando desenvolver soluções, uma vez que estas operam em áreas isoladas e de pequeno alcance (RADIO-ELECTRONICS, 2015).

Ao contrário do espalhamento espectral, o UWB transmite num modo que não interfere com a transmissão de banda estreita convencional e a portadora de transmissão está presente na mesma banda de frequência, ou seja, envia a informação espalhada por uma grande largura de banda. Definições regulatórias por parte da *Federal Communications Commission* (FCC) dos Estados Unidos, pretendem fornecer uma utilização eficiente da largura de banda de RF ao permitir alta taxa de dados para *Personal Area Network* (PAN) em conectividade sem fio e em aplicações de longo alcance e baixa taxa de dados, como em sistemas de radar e de monitoração.

A FCC e a *International Telecommunication Union Radiocommunication Sector* (ITU-R) definiram a UWB como uma antena para transmissão de largura de banda que emite sinal acima de 500 MHz ou em 20% da largura de banda

fracionada, deste modo, cada pulso transmitido ocupa a largura de banda UWB (ou um agregado de pelo menos 500 MHz de portadora de banda estreita) com uso de *Orthogonal Frequency-Division Multiplexing* (OFDM) para realizar o acesso ao meio (RADIO-ELECTRONICS, 2015).

2.3. Protocolos de roteamento em WSN

Os protocolos de roteamento de pacotes mais populares em redes convencionais são classificados principalmente como *Link State* (LS) e *Distance Vector* (DV). A seguir, alguns tipos de protocolos de roteamento utilizados em redes de sensores para diversas arquiteturas e topologias, abrangendo a funcionalidade e características. Como o intuito do trabalho não é focar em roteamento, são apresentados alguns protocolos utilizados em redes de sensores que podem ser inseridos ou não no sistema desenvolvido.

2.3.1. *Ad hoc On-Demand Distance Vector*

O *Ad hoc On-Demand Distance Vector* (AODV) é um protocolo de roteamento reativo destinado à utilização em nós móveis de uma rede do tipo *Ad Hoc*, ou seja, uma rede com movimentação aleatória dos nós. O protocolo oferece rápida adaptação às condições de vínculo dinâmico dos nós e caracteriza-se por: baixo processamento; evitar sobrecarga de memória; propiciar baixa utilização da rede; e também, pela determinação de rotas *unicast* para os nós de destino dentro da rede (PERKINS, 1999).

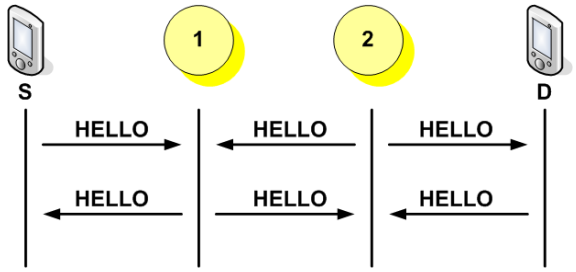
O algoritmo utiliza números de sequência de destino para assegurar certo grau de liberdade de encaminhamento entre os nós, auxiliando os múltiplos saltos e tentando evitar *loops*, mesmo que haja algum tipo de anomalia durante o percurso dos dados até o destino.

Mensagens de controle de roteamento evitam problemas tais como "contagem até ao infinito" que ocorrem em protocolos vetor distância clássicos. Considerações e demais informações sobre o protocolo podem ser obtidas no *Request For Comments* 3561 (RFC 3561, 2003) do IETF.

A Figura 3 mostra o fluxo de mensagens sendo trocadas entre um nó fonte S

até um nó de destino D, passando por 2 nós sensores intermediários, no qual o nó S envia uma mensagem *broadcast Hello* a todos os nós da rede para conexão com um nó D. Periodicamente envia mensagens *Hello* via *broadcast*, que servem para a manutenção das rotas já estabelecidas, visando identificar possíveis mudanças na rota ou mesmo falhas nos nós pertencentes à conexão com o destino (VIEIRA, 2011).

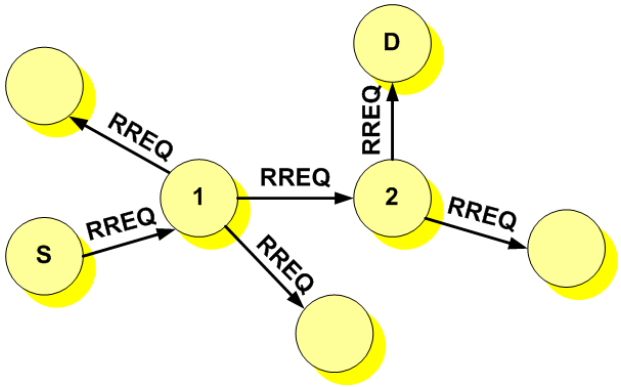
Figura 3. Exemplo de troca de mensagens no AODV.



Fonte: Adaptado de Qualität (2015).

Nas Figuras 3 e 4, pode-se ver a atuação do AODV dentro do conceito de *Mobile Ad Hoc Network* (MANET), em que os nós possuem mobilidade (SARKAR, 2007). A mensagem enviada de S para D, na Figura 4, é do tipo *Route Request* (RREQ) e representa uma requisição para estabelecer uma rota entre o nó fonte até o nó destino em sequência ao *Hello* enviado anteriormente.

Figura 4. Exemplo de *Route Request* (RREQ) no AODV.

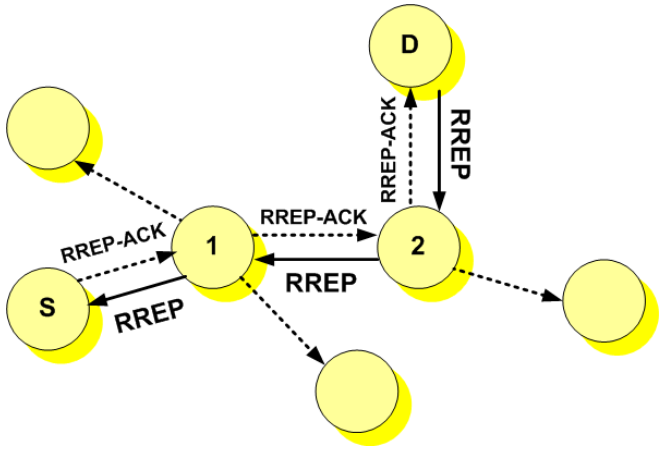


Fonte: Adaptado de AODV (2013).

O RREQ é enviado a todos os nós da rede para a verificação de conexão de distância entre os nós, estabelecendo uma rota com o nó D em que houver menor número de saltos, que por sua vez, envia em resposta ao RREQ recebido, um *Route Reply* (RREP) em direção ao nó S, conforme Figura 5, estabelecendo desta forma a conexão.

Os nós presentes na rota estipulada até o nó D e que realizam o envio do RREP em direção ao nó S, enviam uma resposta RREP-ACK, que simboliza a confirmação da conexão estabelecida entre o nó fonte e destino, passando por estes nós intermediários.

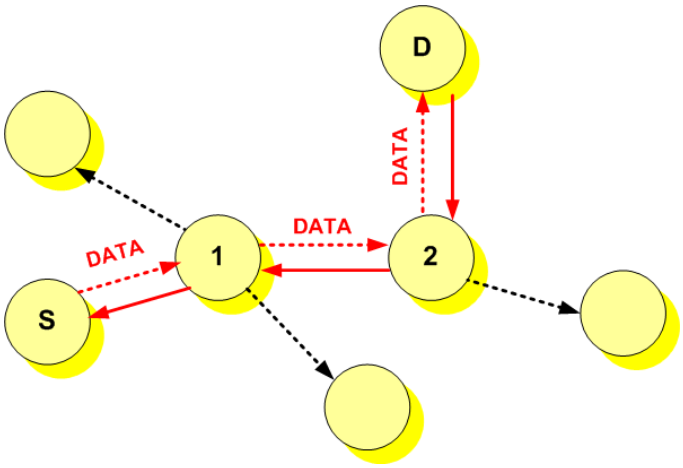
Figura 5. Exemplo de *Route Reply* (RREP) no AODV.



Fonte: Adaptado de AODV (2013).

A Figura 6 mostra o fluxo de mensagens trocadas (destacadas em vermelho) no protocolo de roteamento AODV, com estabelecimento de conexão entre o nó S e o D com passagem por nós intermediários.

Figura 6. Fluxo de mensagens no AODV.

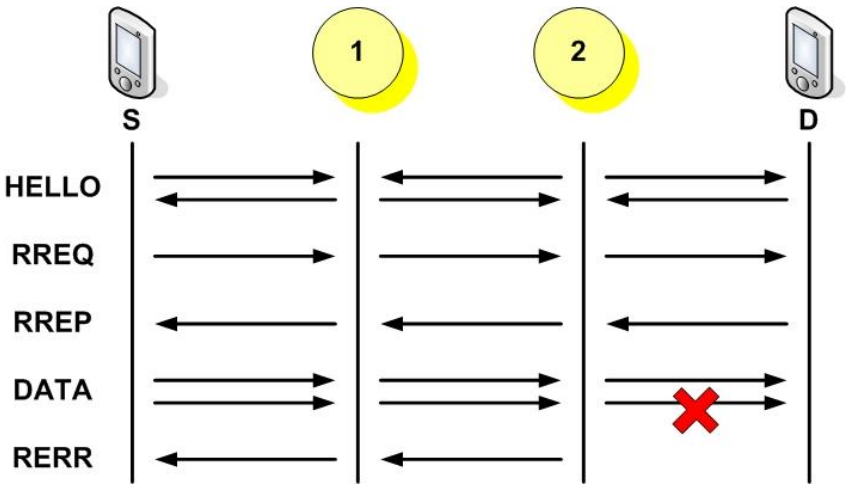


Fonte: Adaptado de AODV (2013).

Quando há erro de roteamento em nós intermediários, devido ao não recebimento do RREP no tempo esperado (em ms) ou por falha de encaminhamento de dados por quaisquer dos nós da rota estabelecida, uma mensagem de erro *Route Error* (RERR) é gerada e devolvida ao nó S, conforme Figura 7.

Caso o nó S queira enviar mais informações para serem trafegadas entre os nós da rede, ou mesmo continuar a enviar os dados que foram impedidos de chegar ao nó D por motivo de falha em algum nó ou mesmo por descarte devido a uma fila de processamento excedente, uma nova rota deve ser estabelecida, iniciando-se desta forma, todo o processo descrito anteriormente.

Figura 7. Exemplo de *Route Error* (RERR) no AODV.

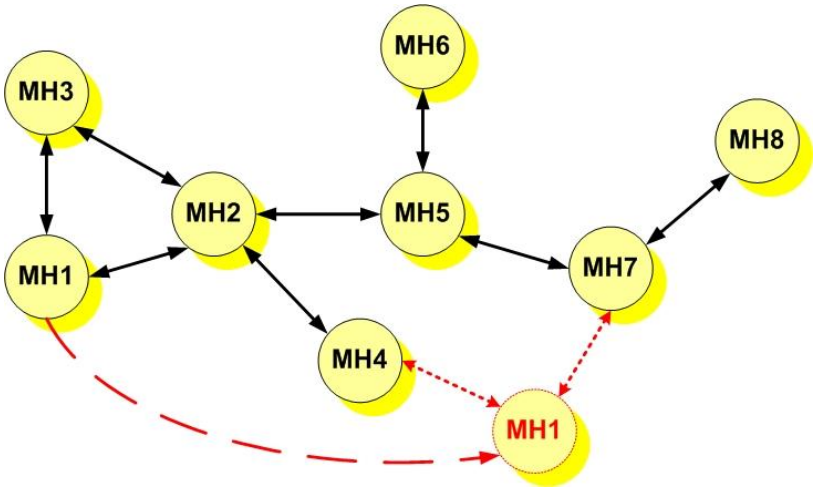


Fonte: Adaptado de Qualität (2015).

2.3.2. *Destination-Sequenced Distance Vector*

Trata-se de um protocolo de roteamento do tipo proativo denominado *Destination-Sequenced Distance Vector* (DSDV). Neste protocolo, cada nó envia periodicamente atualizações a outros nós da rede, incluindo o seu próprio número de sequência, tabela de roteamento e respectivas atualizações.

Figura 8. Envio de dados no DSDV.



Fonte: Adaptado de Perkins (1994).

Na Figura 8, existe uma distribuição de 8 nós sensores, denominados de *Mobile Hosts* (MH), na qual ocorre deslocamento do nó MH_1 para outra posição na mesma rede. Com a mudança na posição do nó, realiza-se então uma nova revisão da tabela de roteamento, atualizando os números de sequência dos nós e conexões com nós vizinhos. Na atuação em redes móveis *Ad Hoc*, o protocolo proporciona auto-organização e pode atuar sobre topologias dinâmicas, além de operar em sistemas fins ou intermediários. A métrica principal utilizada pelo protocolo é a contagem de saltos, na qual o objetivo é encontrar o caminho com menor contagem de saltos até o destino (NARRA, 2011).

Os nós trocam mensagens de controle para estabelecer caminhos com múltiplos saltos, da mesma forma que ocorre no algoritmo distribuído de Bellman-Ford. Este mecanismo é distribuído, pois cada nó recebe informação de um ou mais nós vizinhos conectados diretamente e opera sobre a equação que relaciona os menores custos:

$$d_x\,(y) = \min_{v}\{c(x,v) + d_v(y)\} \qquad 2.1$$

Onde: $d_x(y)$ é o custo do caminho de menor custo entre x e y e min_v é o valor mínimo calculado sobre todos os vizinhos de x. Na equação 2.1, $c(x,v)$ representa o custo do caminho entre o nó x e o nó v (se não houver conexão entre estes nós o custo passa a ser infinito) e $d_v(y)$ é o custo do caminho de menor custo entre v e y (CAMPOS, 2008).

Os nós também podem enviar aos outros nós da rede atualizações de tabela de roteamento para casos importantes em que haja alteração de *links* conhecidos das tabelas de roteamento atuais. Quando existem duas rotas para um destino, ou seja, se o nó recebeu informação de opção de caminho por dois nós vizinhos, existe a escolha pelo nó fonte para o destino com o maior número de sequência e, caso este parâmetro seja igual, a escolha ocorre pela menor métrica, que pode ser a contagem de saltos até o destino. No funcionamento do DSDV, cada nó i mantém uma tabela que contém um conjunto de distâncias ou custos $\{d_{ij}(x)\}$, onde j é um nó vizinho de i. Supondo-se que um nó i trate um nó k vizinho como o próximo salto para o envio de pacote de dados destinado para um nó x, tem-se que:

$$d_{\min}\left\{d_{ij}(x)\right\}_{ik} + \forall d_{\min}\left\{d_{ij}(x)\right\}_{kx} = \left\{d_{ij}(x)\right\}_{ix} \qquad 2.2$$

A equação 2.2 relaciona a relação de distância entre os nós de origem e destino, contabilizando o custo da rota adotada entre o nó *i* até o nó *x*. A tabela de roteamento dá a melhor distância para cada destino, indicando ainda qual a rota para se chegar ao mesmo.

Para manter a distância atualizada, cada nó troca informações com todos os seus vizinhos periodicamente. Se a distância mínima a qualquer nó vizinho sofrer mudanças, o processo será repetido até que todos os nós tenham atualizado as informações de roteamento. No entanto, o algoritmo de roteamento de vetor-distância pode causar *loops* de curta e longa duração devido à atualização de tabela de roteamento com informações desatualizadas.

O problema de *loop* pode ser eliminado utilizando o método de coordenação *inter-nodal*, que exige que os nós roteadores se coordenem mutuamente por mensagens de confirmação em ambiente relativamente estável. Contudo, as redes *Ad Hoc* mudam rapidamente em ambientes móveis e, portanto, este mecanismo *inter-nodal* é de difícil implementação em redes móveis *Ad Hoc* (CANDYCAT, 2016).

O uso do protocolo de roteamento vetor-distância *Routing Information Protocol* (RIP) tem sido empregado com sucesso dentro de redes locais, no entanto, ele introduz um sério problema para redes com muitas mudanças na topologia, tais como falhas de *links* ou de nós roteadores, que é a contagem até o infinito. Para melhorar a convergência e aplicação do RIP, muitas pesquisas foram realizadas, porém, infelizmente, algumas destas resolvem problemas e acabam introduzindo outros ao mesmo tempo. Portanto, o RIP não é um protocolo a ser aplicado em redes móveis *Ad Hoc* (CAMPOS, 2008).

Pelas razões mencionadas, o DSDV é um dos protocolos de roteamento propostos para operar diretamente em redes *Ad Hoc*, pois se destina a superar o problema *looping* e a manter a simplicidade do RIP. Ele adiciona ainda um novo atributo, que é número de sequência para cada entrada da tabela de roteamento. Usando o número de sequência, os nós móveis podem distinguir as informações de rotas estáveis, obsoletas e novas, evitando deste modo a formação de *loops* de

roteamento.

2.3.3. *Optimized Link State Routing*

O protocolo *Optimized Link State Routing* (OLSR) é um protocolo proativo de estado de enlace para redes móveis *Ad Hoc*, no qual existe uma troca de informação sobre a topologia da rede entre os nós vizinhos, o qual envia periodicamente *broadcasts* de mensagens *Hello*, a fim de descobrir nós e *links* na rede. Este protocolo é padronizado através do *Request For Comments* 3626 (RFC 3626, 2003) do IETF.

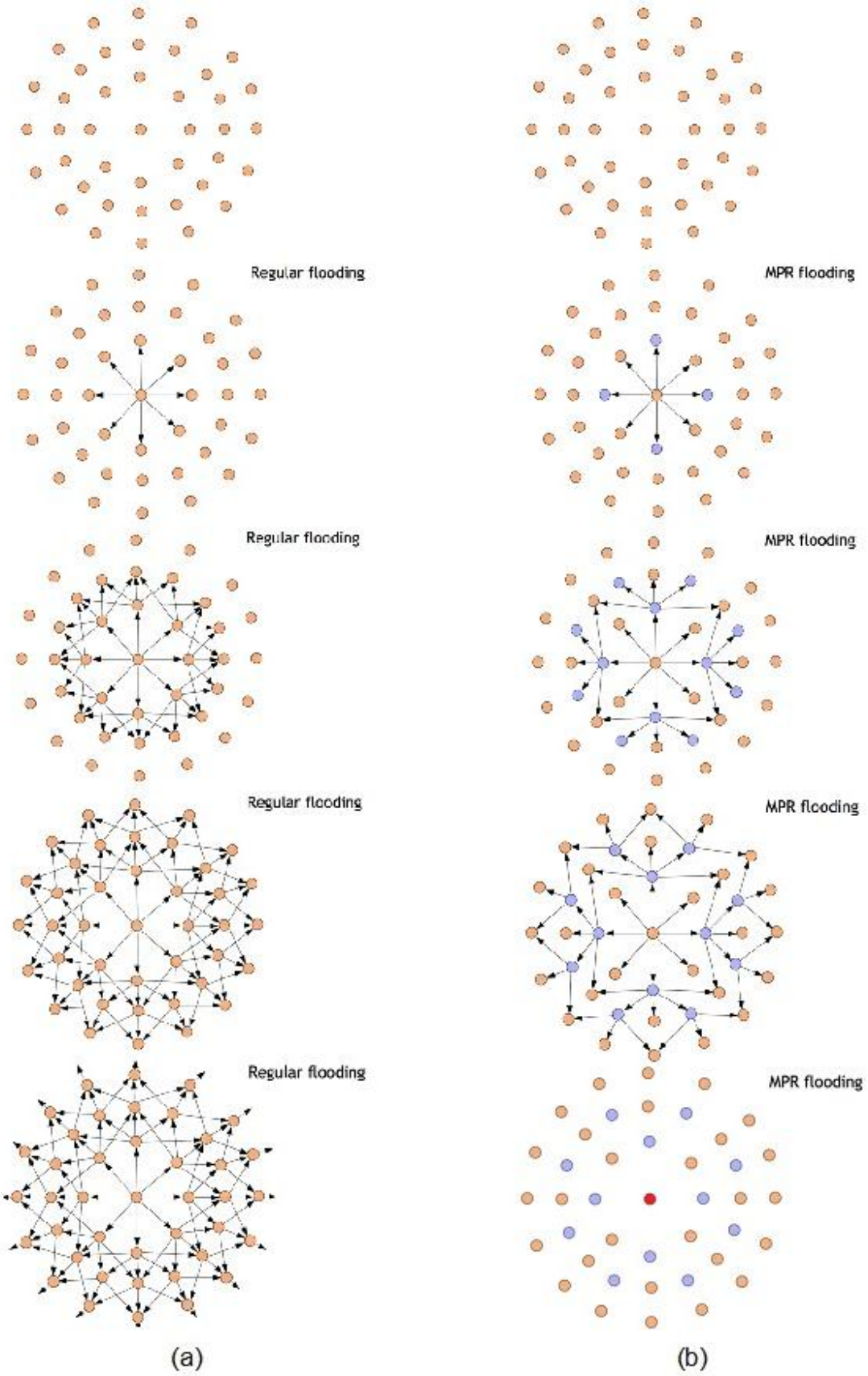
O OLSR executa retransmissão de informações multiponto para reduzir transmissões duplicadas, na qual todos os nós mantêm seletores *MultiPoint Relays* (MPR) e somente os nós selecionados como MPR poderão gerar mensagens de estado do enlace. As MPRs representam nós selecionados para encaminhamento de mensagens *broadcast* durante o processo de atualização da rede. Esta técnica reduz substancialmente a elevada quantidade de mensagens quando comparada aos clássicos mecanismos de *flooding*, onde todos os nós retransmitem cada mensagem assim que recebem.

No encaminhamento de tráfego, um nó somente poderá retransmitir um pacote se tiver sido escolhido como MPR pelo último nó que retransmitiu o pacote e ainda, se este pacote tiver *Time-to-Live* (TTL) maior do que zero. O protocolo é hierárquico, pois alguns nós tem diferenciação com relação aos outros que compõem a rede e roda sobre *User Datagram Protocol* (UDP) utilizando endereços *Internet Protocol* (IP) como únicos identificadores de cada nó.

O protocolo suporta o IPv4 e o IPv6, com diferença no tamanho mínimo de cada pacote que pode ser encaminhado na rede. Todo nó calcula seu conjunto de MPRs através do seu conjunto de nós vizinhos simétricos, que são escolhidos de tal forma que todos os vizinhos de distância 2 saltos podem ser atingidos através de um MPR (UFPE, 2016).

Para visualizar melhor a sequência de descobrimento e determinação de nós MPR, a Figura 9 mostra um comparativo entre um protocolo de *flooding* regular e o relacionamento e criação de MPRs pelo protocolo OLSR.

Figura 9. Comparativo entre protocolos com *flooding*. (a) protocolos *flooding* comuns; (b) utilizando MPR no protocolo OLSR.



Fonte: Adaptado de UFPE (2016).

A Figura 9a apresenta a forma de *flooding* convencional, na qual todos os nós encaminham dados para todos os outros nós vizinhos. A Figura 9b apresenta o processo de *flooding* dentro do OLSR, no qual são determinados os nós MPRs que terão condições de encaminhar pacotes de dados aos demais nós. Na última parte da Figura 9b, visualiza-se o nó central (vermelho) que iniciou o processo de *flooding* e os demais nós comuns (laranja) e MPRs (azul).

O conceito de MPR utilizado no OLSR tem o objetivo de reduzir o número de retransmissões duplicadas durante o processo de *flooding* de um pacote encaminhado pela rede.

2.3.4. *Dynamic Source Routing*

O *Dynamic Source Routing* (DSR) é um protocolo reativo para redes móveis *Ad Hoc* utilizando IPv4, sendo padronizado pelo IETF no *Request For Comments* 4728 (RFC 4728, 2007). O DSR é um protocolo de roteamento para uso em redes sem fio com múltiplos saltos. O protocolo permite que a rede seja completamente auto-organizável e autoconfigurável sem a necessidade de qualquer infraestrutura de rede específica.

O protocolo é composto por mecanismos de descoberta e manutenção de rota, que funcionam em conjunto para permitir que os nós possam descobrir e manter rotas para destinos arbitrários na rede *Ad Hoc*. Todos os aspectos do protocolo operam inteiramente sob demanda com reação às mudanças nas rotas atualmente em uso.

O protocolo permite várias rotas para qualquer destino e, que cada nó transmissor possa selecionar e controlar as rotas utilizadas no encaminhamento de seus pacotes, adequando o equilíbrio de carga ou aumentando a robustez. Outras vantagens do protocolo DSR incluem: estar livre de *loops* de roteamento, operar em redes contendo *links* unidirecionais, utilizar apenas *soft state* em roteamento e muito rápida recuperação quando ocorrem mudanças de rotas na rede (RFC 4728, 2007).

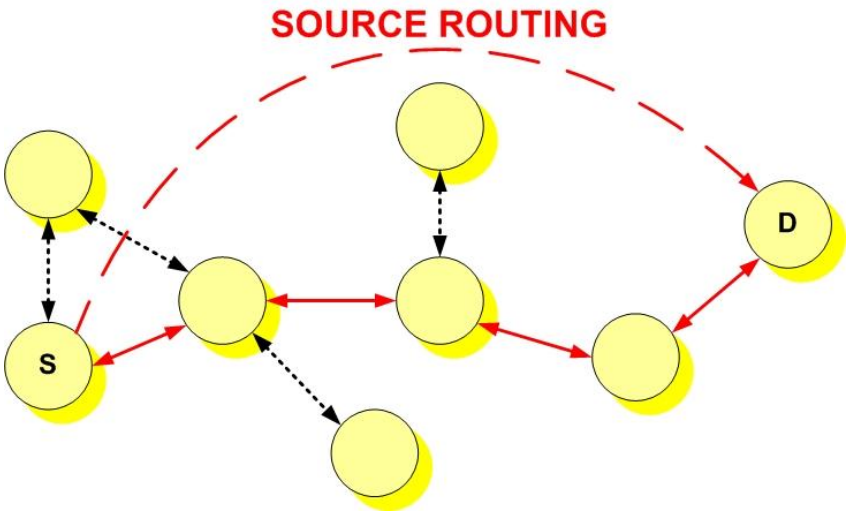
O DSR é bastante semelhante ao AODV, uma vez que uma rota só é descoberta quando for necessário, contemplando um roteamento reativo. No entanto, o DSR usa o *Source Routing* em vez de confiar nas tabelas de roteamento

de nós intermediários. O *Source Routing* permite que o nó de origem especifique parcialmente ou completamente a rota na qual os pacotes serão enviados através da rede (Figura 10). Este procedimento possui muitas vantagens, como por exemplo: permite a detecção de erros simples, torna o traçado da rota desnecessário, e habilita um nó a descobrir todos os caminhos possíveis para um destino, incluindo a oportunidade de otimizar as rotas baseando-se na informação disponível. Permite ainda, que os nós de origem reajam diretamente às mudanças na topologia da rede, visando a preservar recursos, desta forma, evitando congestionamento e forçando os pacotes a trafegar por um determinado caminho (DSR, 2015).

Os endereços dos nós pertencentes à rota, a partir da origem até o destino são acumulados durante o processo de descoberta de rota. Cada nó que encaminha um RREQ adiciona o seu endereço à lista que está contida no cabeçalho dos pacotes de roteamento.

Na recepção pelo nó de destino, um RREP é gerado em direção ao nó de origem contendo uma cópia de toda a lista de nós intermediários que foram adicionando os seus endereços durante o trajeto do pacote. O RREP viaja pelo caminho contrário com destino ao nó fonte, ao longo da rota, atualizando as informações em todos os nós envolvidos na rota para o encaminhamento do pacote.

Figura 10. Exemplo de *Source Routing* no DSR.



Fonte: Adaptado de DSR (2015).

Todos os pacotes de dados terão a lista de nós a percorrer mantendo os endereços do percurso. Como a rota pode conter muitos saltos ou mesmo possuir

endereços muito grandes, como por exemplo, quando apresenta nós com IPv6, a sobrecarga na rede é maior em comparação ao AODV.

Para melhorar o desempenho, elaborou-se no DSR uma extensão ID de fluxo (que é opcional), a qual permite a criação de um fluxo de dados via uso de um circuito virtual comutado para o encaminhamento de pacotes. Como no AODV, os nós que não são o destino e que apresentam uma rota com melhores métricas disponíveis, podem responder à RREQ, otimizando o processo de roteamento.

2.3.5. *Temporally Ordered Routing Algorithm*

O *Temporally Ordered Routing Algorithm* (TORA) é um protocolo reativo de roteamento distribuído para uso em redes com múltiplos saltos. Um conceito chave utilizado no projeto de elaboração do protocolo é a tentativa de dissociar a geração e propagação de mensagens de controle sobre a dinâmica da topologia da rede. Basicamente, o algoritmo não é classificado como LS ou DV, sendo enquadrado como membro de uma classe referida como algoritmos *Link Reversal* (LR).

O protocolo cria uma estrutura de encaminhamento de múltiplos saltos, livre de *loops* de roteamento, que é usado como a base para o encaminhamento de tráfego de pacotes para um determinado destino. O protocolo pode simultaneamente suportar tanto *Source Initiated Routing* (SIR) como roteamento sob demanda para alguns destinos novos e também para destinos já iniciados em outro intervalo, criando também um roteamento proativo para outros destinos (LIM, 2007).

O TORA proporciona múltiplas rotas para aliviar o congestionamento e manipula partições apagando rotas inválidas. Para a criação de rotas, o protocolo estabelece uma sequência de *links* diretos do nó fonte até o destino, sendo feito por um *Directed Acyclic Graph* (DAG) orientado ao destino.

A manutenção das rotas é feita pela reação às mudanças de topologia, de forma a reestabelecer rotas. O apagamento das rotas ocorre quando uma partição é detectada na rede, sendo que todas as rotas inválidas devem ser excluídas e é realizado pelo não direcionamento de rotas com anomalias (QUADRI, 2016).

2.3.6. *Angle Routing Protocol*

Angle Routing Protocol (ARP) ou *route angle*, também conhecido como *route angle*, é um protocolo de roteamento do tipo seletivo para MANET, baseado na seleção dinâmica de ângulos entre os nós de origem e destino. Este protocolo opera sobre o posicionamento dos nós em eixos horizontais e verticais (X, Y), realizando conexões entre os nós conforme os ângulos formados entre os mesmos.

Existem protocolos desenvolvidos para atuar em redes móveis *Ad Hoc*, que também utilizam abordagens geográficas, como o *Distance Routing Effect Algorithm for Mobility* (DREAM) desenvolvido por Basagni (1998) e o *Location-Aided Routing* (LAR) elaborado por Ko (2000). Estes protocolos executam o roteamento de pacotes geograficamente, mas diferem do ARP na forma como os nós adquirem as localizações geográficas dos nós de destino. Tanto o DREAM como o LAR realizam *flooding* na rede, mas no DREAM o nó fonte executa *flooding* de sua posição, fazendo com que os outros nós construam uma base de dados completa sobre sua posição, enquanto que no LAR o nó fonte envia um *flooding* para todos os nós da rede com uma consulta sobre a posição dos nós.

A escolha dos nós para o encaminhamento é feita através do uso de avaliação de ângulos entre os nós vizinhos. Cada um dos nós da rede é um *host* móvel. Estes nós mantém algumas informações sobre os nós vizinhos para futuro uso durante a descoberta de rota.

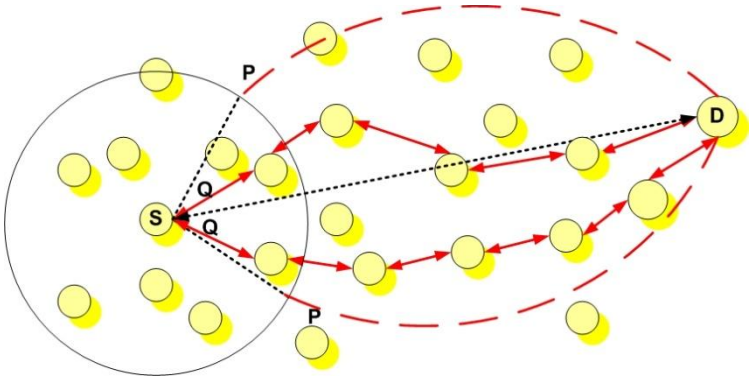
O protocolo assume que todos os nós têm informação de localização sobre os seus vizinhos em uma tabela armazenada nos mesmos. As informações geográficas sobre os nós podem ser encontradas usando *Global Positioning System* (GPS). As tabelas são atualizadas com as informações dos nós através do uso de *flooding* na rede (BANKA, 2002).

Um exemplo de encaminhamento de dados neste protocolo pode ser visualizado na Figura 11, na qual o nó de origem S seleciona uma região em que 2 nós serão selecionados para encaminhar os pacotes na rede. A seleção dos nós ocorre conforme os seguintes parâmetros: localização da fonte S e do destino D e, o ângulo estipulado Q com início na visada direta entre S e D. A junção dos ângulos

QQ, forma a região SPP, na qual, de todos os nós presentes, apenas 2 nós são selecionados para realizar a conexão com o nó D.

Esta formatação coloca à disposição duas rotas entre os nós fonte e destino, para que a probabilidade de falha no encaminhamento de dados em rota seja reduzida e a escolha do ângulo Q dinâmica.

Figura 11. Exemplo de encaminhamento de dados no *Route Angle*.



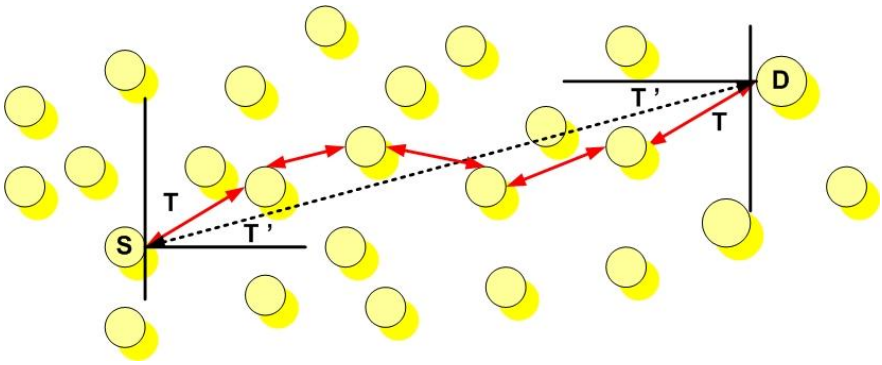
Fonte: Adaptado de Banka (2002).

No protocolo proposto por Suresh (2014), *Angle based Routing Protocol* (ABRP), quando um nó tem um pacote para ser enviado para algum destino, inicia a *Destination Discovery* (DDIS) para encontrar uma rota até o destino, sendo que este nó é conhecido como o iniciador da DDIS e o destino do pacote como *Discovery's Target* (DTAR). O iniciador transmite um pacote *Destination Request* (DREQ) como um *broadcast*, especificando o destino, identificador único do iniciador e a sua posição.

Normalmente a falha do *link* (ou mesmo rota expirada) ocorre devido à mobilidade dos nós. Quando um nó detecta falha no *link* é necessário o envio de alguma mensagem de erro de rota ao invés do nó intermediário só atuar como uma fonte e encaminhar o pacote de dados para o destino.

A Figura 12 apresenta a forma pela qual é definido o caminho de rota até o destino, passando-se por 4 nós intermediários. Uma rota é estabelecida conforme um traçado em visada direta do nó iniciador até o nó destino. Os nós intermediários são escolhidos conforme os menores ângulos formados em relação à linha de visada.

Figura 12. Escolha de rota no ABRP.



Fonte: Adaptado de Suresh (2014).

O protocolo ARP foi escolhido para ser utilizado para a realização das simulações feitas nesta Tese, por proporcionar visualização simples sobre o encaminhamento de dados da origem até o destino.

2.3.7. *Grid's Location Service*

Grid's Location Service (GLS) é um protocolo de roteamento que executa serviço de localização escalável em redes móveis *Ad Hoc*, ou seja, para redes com aumento no número de nós em roteamento geográfico. Este protocolo é atrativo para redes com larga-escala ou número grande de nós, pois os nós precisam somente de informações lógicas ao invés de terem a necessidade de saber o tamanho total da rede (LI, 2000).

O GLS é baseado na idéia de que um nó mantém a sua localização atual num certo número de *location servers* distribuídos por toda a rede. Estes servidores de localização não são especialmente projetados e cada nó age como um servidor local conforme o interesse para outros nós. O protocolo assegura que qualquer nó perto de um destino possa usar um *location server* nas proximidades para encontrar o caminho até o destino, ao mesmo tempo, limitando o número de servidores de localização para cada nó. As consultas por comprimentos de trajeto são proporcionais aos comprimentos de caminho de dados.

Conforme Li (2000), um simples encaminhamento de pacotes de forma geográfica combinado com o uso do GLS se torna favorável quando comparado ao DSR, pois em redes largas, ou seja, acima de 200 nós, existe maior entrega de pacotes consumindo menos recursos de rede.

2.3.8. SFN *based Distributed Cooperative Routing Protocol*

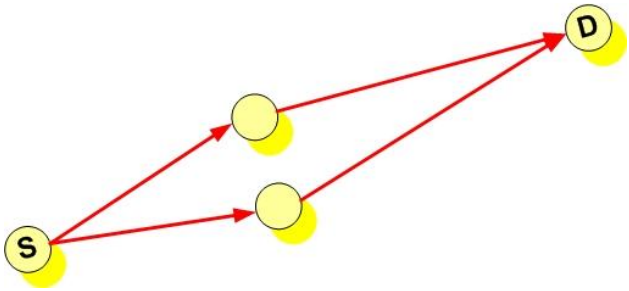
Single Frequency Network based Distributed Cooperative Routing Protocol (SFN-DCRP) é um protocolo de roteamento do tipo proativo e trabalha com o conceito de diversidade cooperativa. Este tipo de diversidade também é conhecido como macro-diversidade e está sendo usado no *soft handover* de sistemas 3G e em *Coordinated MultiPoint* (CoMP) do *Long Term Evolution* (LTE) está sendo usado no *uplink* para maior desempenho e utilização da rede (ERIKSSON, 2010).

O conceito de diversidade cooperativa pode ser aplicado às redes com múltiplos saltos, nas quais múltiplos nós cooperam para transmitir o mesmo dado para um destino e com cooperação podendo ser baseada em SFN. Um cenário possível sobre esta cooperação poderia ser quando um sensor quer enviar uma medição ou leitura para um *Access Point* (AP), porém, para a transmissão direta dos dados seria necessária alta potência na comunicação com o AP, o que provavelmente afetaria a vida útil do nó sensor. Portanto, para aumentar a vida útil dos nós na rede, pode-se reduzir a potência de transmissão utilizando nós próximos como mecanismos de cooperação. Com o uso de cooperação de vários nós com base no SFN, a energia para a cooperação é distribuída entre os diferentes nós e, portanto, alcança-se um aumento no tempo de vida da rede.

Em um cenário não-cooperativo típico, a recepção refere-se à decodificação do sinal de uma única fonte, enxergando os outros sinais como interferência. Em técnicas de cooperação os sinais de outras fontes podem contribuir para formar um sinal mais forte e, portanto, alcançando um ganho de diversidade, semelhantemente ao que ocorre na diversidade de antena, contudo sem o uso de múltiplas antenas (MUNEER, 2013).

A natureza não-determinística do canal sem fios utilizado para a comunicação, desempenha um papel muito importante na complexidade e consumo de energia dos nós. No entanto, esta característica pode ser utilizada como um benefício para a implantação de uma rede de frequência única. Numa SFN, vários emissores irradiam o mesmo sinal na mesma frequência. São utilizadas técnicas de cooperação SFN com base em uma rede sem fio com múltiplos saltos.

Figura 13. Envio de dados do nó S para o D no SFN-DCRP *broadcast*.



Fonte: Adaptado de Muneer (2013).

Na Figura 13, ocorre o encaminhamento de dados do nó fonte para o nó destino. Utilizando a diversidade cooperativa, realiza-se a transmissão dos mesmos dados por 2 caminhos diferentes, passando por 2 nós intermediários, porém, com a potência de transmissão dividida entre os nós, pois transmitem os dados na mesma frequência. O protocolo não é escalável em relação ao número de nós e é adequado para redes menos densas ou com abordagens em *clusters*, isto serve para evitar uma grande sobrecarga de atraso de inicialização no roteamento.

2.4. *Internet of Things*

Internet of Things (IoT) se refere a objetos identificáveis e suas representações virtuais em uma estrutura de Internet. O termo IoT foi proposto por Ashton (2009) e o conceito de Internet das coisas se tornou popular, proporcionando muitas pesquisas relacionadas ao tema. No início, a *Radio-Frequency Identification* (RFID) era vista como um pré-requisito para a IoT, partindo do princípio de que se todos os objetos e pessoas fossem equipados com identificadores (*Tags* RFID), eles poderiam ser gerenciados e inventariados em uma base de dados geral.

Segundo Srivastava (2013), com o avanço das tecnologias de identificação, além de usar a RFID, uma marcação dos objetos (*tagging*) pode ser feita através de tecnologias como *Near Field Communication* (NFC), padrões de códigos de barras (*barcodes*), padrões digitais de códigos *Quick Response* (QR) ou marcas d'água digitais. Disponibilizar todos os objetos do mundo com dispositivos minúsculos de identificação com certeza transformará a vida cotidiana das pessoas, uma vez que a capacidade de interagir com objetos é um grande diferencial para a interação entre pessoas e máquinas. Neste cenário, é possível alterar condições do ambiente remotamente, com base em necessidades imediatas, de acordo com uma política

adotada para a funcionalidade da rede.

De acordo com a ABI *Research* (2013), que é uma companhia de inteligência de mercado em conectividade global e de tecnologias emergentes, em 2020 mais de 30 bilhões de dispositivos *wireless* estarão conectados à IoT, caracterizando o que deve ser chamado de *Internet of Everything* (IoE). Este conceito no qual os dispositivos podem se conectar à Internet (ou *Web*) via RF de baixa potência é a área de pesquisa mais ativa da IoT atualmente.

2.4.1. RPL – IPv6 *Routing Protocol for Low-Power and Lossy Networks*

O IPv6 *Routing Protocol for Low-Power and Lossy Networks* (RPL) é um protocolo de roteamento do tipo DV, que opera com o IPv6, que foi desenvolvido para atender às redes do tipo *Low-Power and Lossy Networks* (LLC). O RPL foi padronizado pelo IETF no *Request For Comments* 6550 (RFC 6550, 2012). Este protocolo foi criado, visando o uso em redes do tipo IoT.

Para o funcionamento, o protocolo define uma árvore de roteamento, utilizando o conceito de *Directed Acyclic Graph* (DAG), onde cada nó pode realizar uma associação com outros nós, diferenciando este protocolo dos demais baseados em árvores. A organização é orientada ao destino, sendo descrita como *Destination Oriented Directed Acyclic Graph* (DODAG), na qual a rede possui um nó raiz que concentra o recebimento de dados oriundos de outros nós.

O RPL proporciona trabalhar com baixo consumo de energia, criação de árvores de rotas, sendo do tipo adaptativo, ou seja, autoconfigurável e curável, com detecção e impedimento de *loops* na rede. Não é definida qualquer métrica ou restrição para o protocolo, sendo que esta atribuição relaciona-se ao *Minimum Rank with Hysteresis Objective Function* (MRHOF).

Para o encaminhamento de dados na rede, existem diferentes tipos de mensagens de controle, visando à criação da árvore e determinação de rotas entre S e D. As mensagens de controle são: DODAG *Information Object* (DIO), DOGAG *Information Solicitation* (DIS), *Destination Advertisement Object* (DAO) e *Destination Advertisement Object Acknowledgement* (DAO-ACK).

2.5. Conceitos de cognição

Para o Wilson (1999), a ciência cognitiva é formada por categorizações que estão envolvidas em padrões complexos empíricos e teóricos. Afirma que a classificação de ciência cognitiva para o *Massachussets Institute of Technology* (MIT) é distribuída em seis diferentes áreas: filosofia, psicologia, neurociências, inteligência computacional, linguística e linguagem. Neste contexto, pode-se visualizar uma sequência que envolve cultura, cognição e evolução.

Dentre as áreas descritas pelo Wilson, o presente trabalho foca na inteligência computacional com o uso de aspectos de cognição, mais precisamente em histórico, memória e tomada de decisão. A seguir são apresentados alguns temas que abordam o conceito utilizado na confecção deste trabalho, bem como a aplicação de cognição em ambientes artificiais.

2.5.1. Princípios básicos de ciência cognitiva

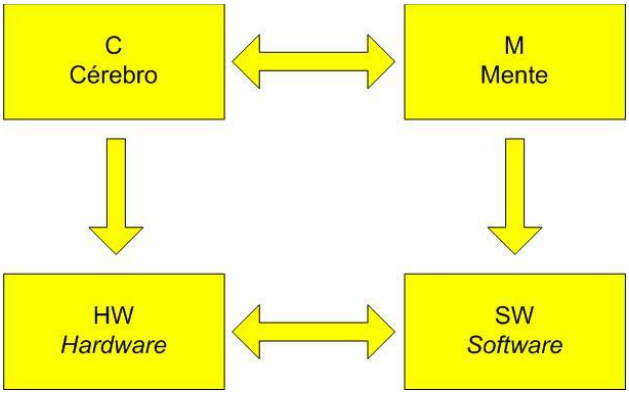
Em ciência cognitiva, o primeiro pensamento que surge e a respectiva reflexão relacionam-se com a ligação que existe entre cérebro e mente. Para a psicologia, a relação básica que existe entre cérebro e mente em termos computacionais é representada pela Figura 14. Salienta-se aqui que este tema é polêmico e que este modelo serve apenas para representar uma abstração de como *Hardware* (HW) e *Software* (SW) se comunicam quando comparados ao funcionamento do cérebro e da mente humana (LOBO NETTO, 2016).

A intenção deste trabalho não é a de elucidar ou discutir a real comparação em termos filosóficos sobre a definição do que realmente representa a mente ou o cérebro, e sim, apresentar uma forma sucinta básica sobre as relações do ser humano que podem ser refletidas para o ambiente de máquinas, visando dar base para conceitos e aplicações posteriores deste trabalho.

Na Figura 14, pode-se fazer uma equivalência entre a parte física computacional (HW) e a parte física humana (C). E entre a parte computacional (SW), ou seja, o conjunto de instruções armazenadas em disco(s) ou em *chips* internos de um computador que determinam os programas básicos, utilitários ou

aplicativos, que ele tem para serem usados e parte humana (M), ou o entendimento, intelecto, espírito, concepção e/ou imaginação do ser humano.

Figura 14. Modelo representativo abstrato entre HW / SW e Cérebro / Mente.



Fonte: Adaptado de Lobo Netto (2016).

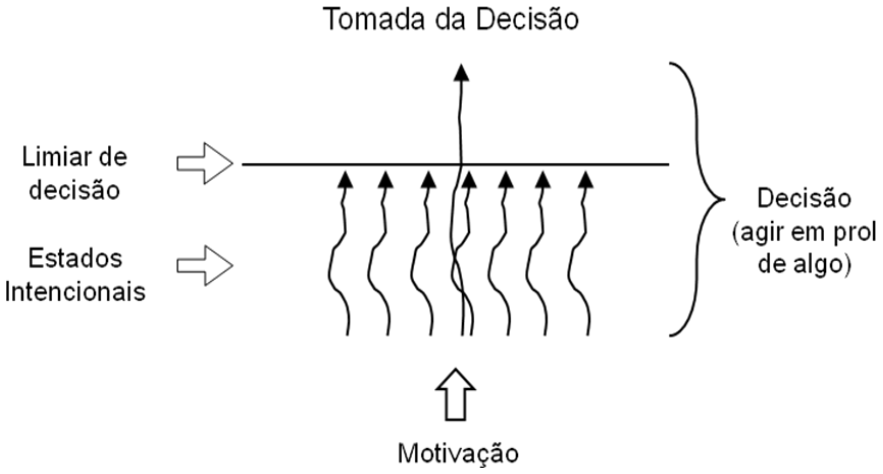
O resultado das conexões sinápticas pode ser estendido por todo o corpo humano, resultando em um vinculo da Rede Neural com a Rede Sensorial dispersa pelo corpo humano, o que proporciona uma grande capacidade de processamento e armazenamento de informação. O mesmo se aplica aos mecanismos artificiais inteligentes e sistemas de sensoriamento dispersos com processamento central (THAGARD, 2012).

2.5.2. Tomada de decisão

O processo de tomada de decisão em ciência cognitiva remete à reflexão sobre o poder da consciência e em que ponto, ou em que profundidade, é feita a tomada de decisão. Para tanto, deve-se pensar em termos de abstração de informações, ou seja, pensar na consciência como um limiar no qual existem diferentes motivações de estados intencionais que são levados a um nível elevado, que pode ser entendido como o limite ou limiar, para então transformarem-se em decisões (LOBO NETTO, 2016).

A Figura 15 representa um exemplo deste limiar de decisão, o qual ocorre sobre os estados intencionais para agir em função do benefício de algo, que no caso de redes tem a ver com índices de informações de desempenho como atraso fim-a-fim, *throughput*, atraso de processamento no nó sensor, filas de processamento, entre outros.

Figura 15. Tomada de decisão e estados intencionais.



Fonte: Autor.

Na Figura 15, o índice que ultrapassou o limiar de decisão ocasionou a tomada de decisão com base nos dados ou informações do índice destacado. Os estados intencionais são pré-ações a serem executadas ou não, que dependem do índice atingir o limiar destacado.

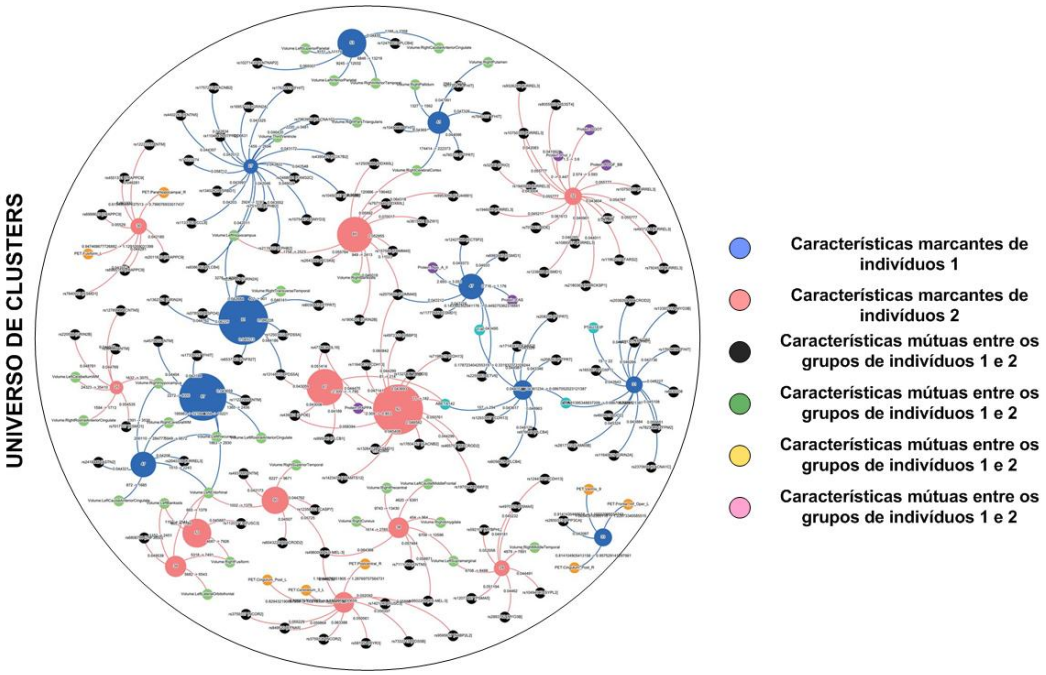
Em linguagem computacional estes estados intencionais podem representar sub-programas ou sub-rotinas vinculadas a um programa principal que executa análise em tempo real sobre dados oriundos de uma aplicação ou informações recebidas de uma rede qualquer, para então realizar a ação de dar prioridade ou à execução do sub-programa ou sub-rotina que tenha se destacado.

2.5.3. Clusterização

A clusterização é uma forma de agrupamento sobre características equivalentes de indivíduos, ou seja, que pertencem ao grupo dos indivíduos que possuam certo grau de pertinência, e a interligação ou equivalência entre estes grupos que podem estar relacionadas a uma descrição ou um objetivo comum.

A Figura 16 apresenta ligações entre diferentes *clusters*, onde os círculos maiores representam agrupamentos com maior número de indivíduos, as cores azuis e vermelhas indicam características marcantes dos indivíduos e, as cores pretas, verdes, laranjas e roxas denotam equivalências entre os grupos, como por exemplo, características mútuas, ocasionando as conexões.

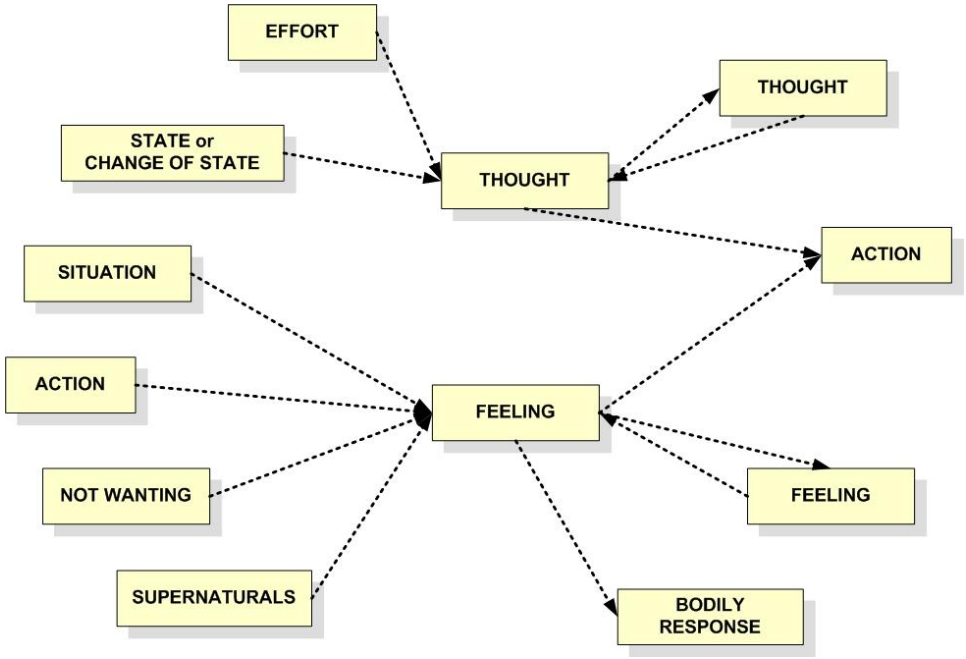
Figura 16. Exemplo de clusterização no ambiente de ciência cognitiva.



Fonte: Adaptado de FRACKOWIAK (2015).

A clusterização pode ser vista na forma de uma tabela correlacional, conforme a Figura 17, na qual é possível identificar os dados referentes a determinadas funções de pensamento e sentimento e ainda, a relação destes em agrupamentos conectados com diferentes funções de entrada e saída.

Figura 17. Clusterização na formatação de tabela correlacional utilizada para formar sequências causais.



Fonte: Adaptado de Kristiansen (2006).

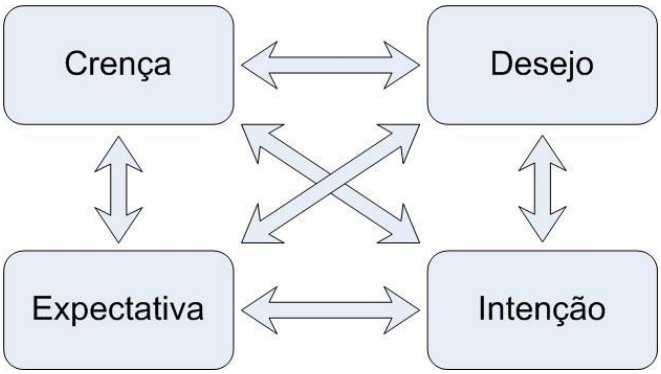
Para Palmer (2003) o termo cognição representa uma categoria analítica de perfis de pensamento, que podem ser racionais ou irracionais, conceituação, categorização e processos de atenção, que influenciam diretamente nos estados experimentais para sequências causais envolvidas com as características de pensamento e sentimento presentes na Figura 17.

2.5.4. Modelo BDI

O modelo *Belief-Desire-Intention* (BDI) é um modelo de *software* desenvolvido para a programação de agentes inteligentes baseados em estados mentais que, como o próprio nome diz, baseia-se nos conceitos de crença, desejo e intenção. Os agentes BDI utilizam estes conceitos para serem capazes de equilibrar o tempo gasto na determinação entre planos e efetuar a execução dos mesmos.

Segundo Georgeff (1999) as noções de complexidade e mudança terão grande impacto sobre a forma como são construídos os sistemas computacionais, e que os agentes de *software*, em particular os agentes BDI, fornecerão os componentes essenciais necessários para lidar com o mundo real. Expõe a necessidade de utilizar agentes em ciência da computação, mencionando que a única maneira de fazer isso é mostrar claramente como as arquiteturas de agentes podem lidar com os problemas que são intratáveis pelas abordagens convencionais.

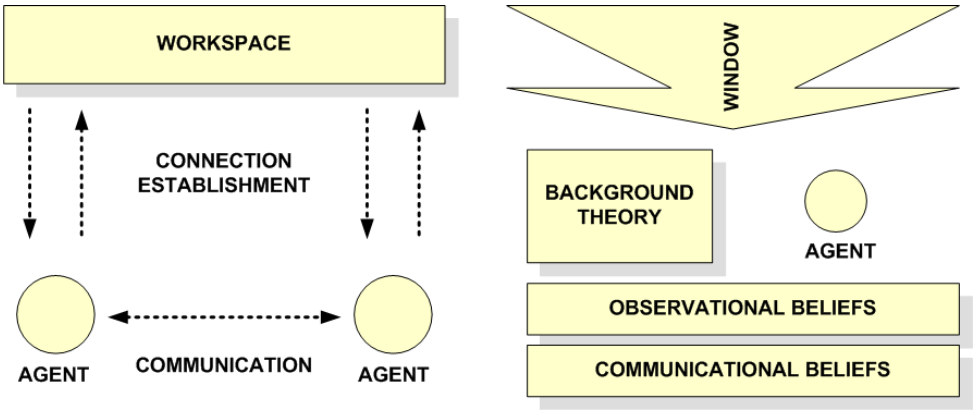
Figura 18. Arquitetura baseada em 4 estados mentais.



Fonte: Adaptado de Juchem (2001).

A Figura 18 mostra as relações que existem entre as partes internas de uma arquitetura BDI com 4 estados mentais. A Figura 19 mostra a relação de interações de agentes (na parte esquerda) e fontes de informação disponíveis aos agentes (na parte direita).

Figura 19. Diagrama de interação de agentes e fontes de informação.



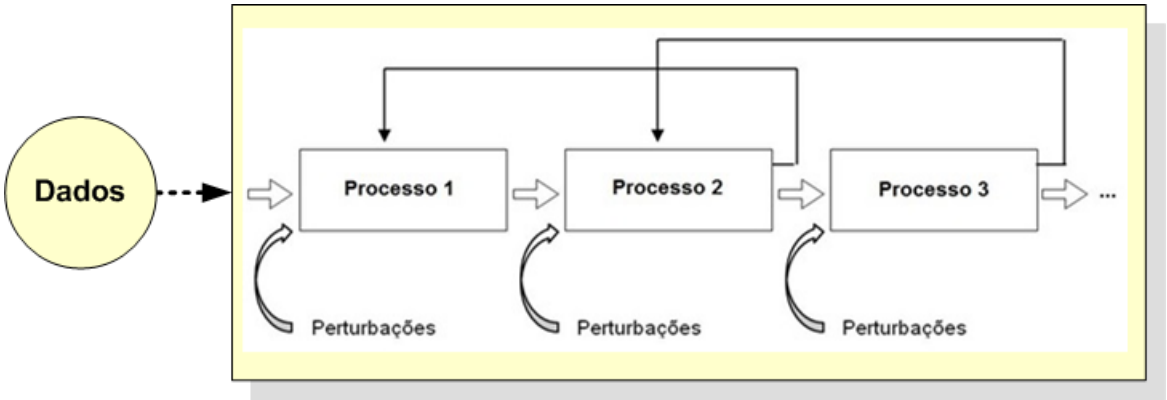
Fonte: Adaptado de Müller (1999).

2.5.5. Processos em Teoria dos Sistemas Dinâmicos

A Teoria dos Sistemas Dinâmicos (TSD) descreve o entendimento do surgimento de novas formas de comportamento sob a ação do tempo, devido à auto-organização. A TSD aponta a auto-organização como um dos elementos básicos para o desenvolvimento de um sistema. A auto-organização é proveniente de perturbações do meio que acabam rompendo formas atuais e incidindo na evolução de novos comportamentos. A variabilidade no sistema acarreta em desempenho variável e dinâmico, o que causa a necessidade por ajustes (BOOTH, 2012).

Percebe-se então que as perturbações realizam alterações na saída do sistema ou em relações intermediárias, conforme a Figura 20, ocasionando em adaptações necessárias para adequar a saída a um resultado próximo ao esperado dentro dos padrões de desempenho.

Figura 20. Exemplo de Sistema Dinâmico de controle de processos.



Fonte: Autor.

Na Figura 20, os retângulos representam processos que sofrem interferências de perturbações externas a cada cadeia executada. Esta cadeia de processos possui realimentação de informações sobre processos subsequentes, o que acarreta na dinâmica do sistema como um todo e que propicia a necessidade por ajustes intermediários conforme resultados e análises das perturbações.

2.6. Redes cognitivas

As redes cognitivas utilizam o conceito de cognição descrito anteriormente em alguns aspectos pertinentes às redes de sensores convencionais e ao meio em que a rede se encontra. São apresentados conceitos de CN e CSN que correspondem ao ramo de pesquisa desta Tese.

2.6.1. *Cognitive Network*

Cognitive Network (CN) é um novo tipo de rede de dados que faz uso da tecnologia de ponta de diversas áreas de pesquisa, ou seja, aprendizagem de máquina, representação geral do conhecimento, rede de computadores, gerenciamento de rede e outros, para resolver alguns problemas que as redes atuais enfrentam e não conseguem solucionar.

CN é diferente de *Cognitive Radio* (CR), pois cobre eventos em todas as camadas de rede quando comparadas ao modelo de referência OSI, que é um modelo composto por 7 camadas, contemplando da superior para a inferior: aplicação, apresentação, sessão, transporte, rede, enlace e física (X.210, 1993).

No CR a aplicação principal relaciona-se às camadas mais baixas (enlace e física) com enfoque na otimização do uso do espectro de radiofrequências, não atuando inicialmente, portanto, com suporte à inteligência nas camadas superiores (MITOLA, 2000).

Existe uma evolução do CR, apresentada no próximo tópico que aborda conscientização e adaptação, com uso de tecnologias de *Web* semântica, visão computacional, navegação, e reconhecimento de fala e de síntese.

Thomas (2005) define a CN como uma rede com processo cognitivo, ou seja,

uma rede que pode perceber as condições atuais e planejar, decidir, ou agir de acordo com essas condições, além de aprender com as consequências de suas ações para poder decidir sobre futuros eventos visando a atingir o objetivo de entrega de dados fim-a-fim.

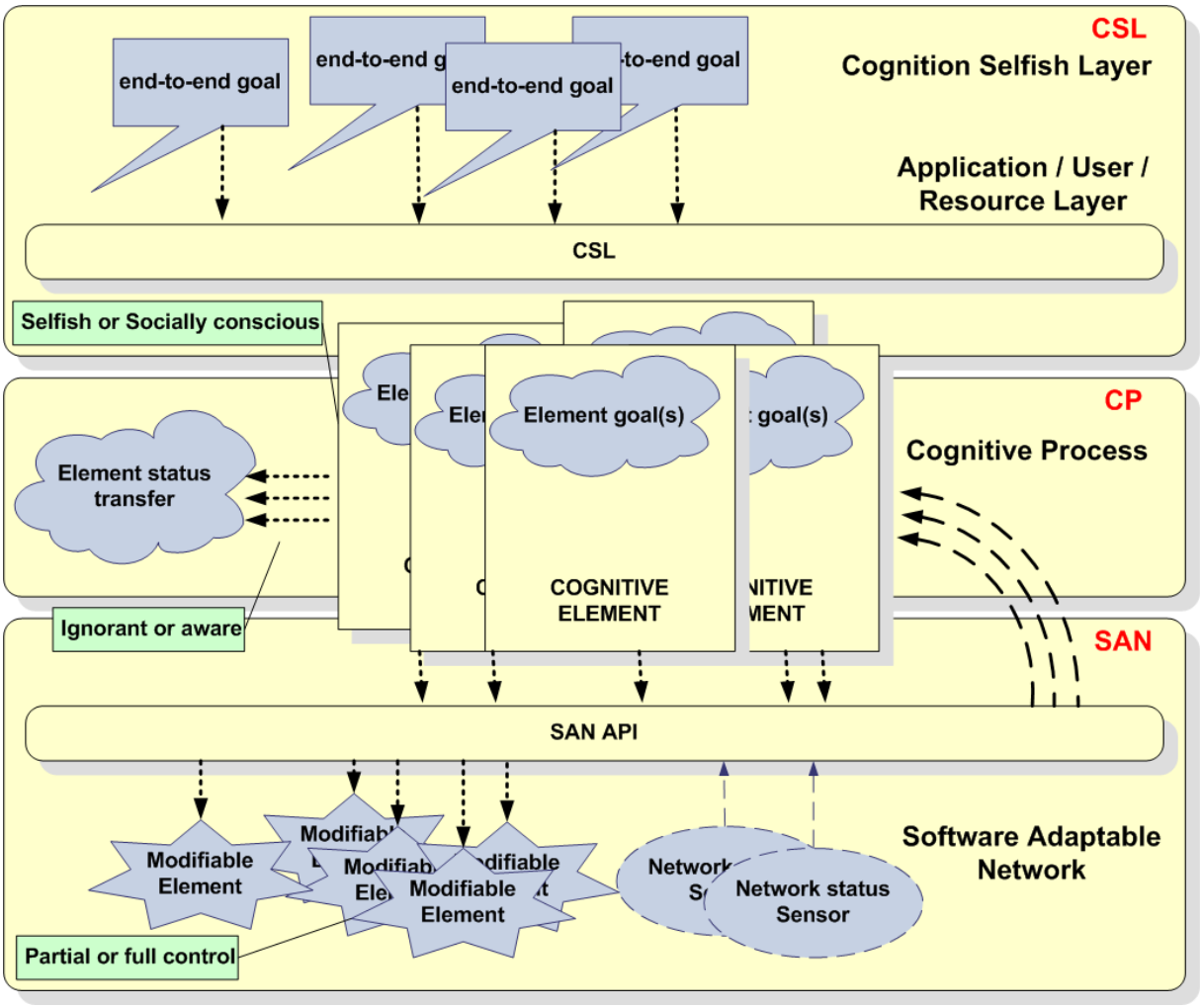
Para que a rede tenha este conhecimento, é executado um *loop* de cognição que permite sentir o ambiente, planejar as ações de acordo com os dados recolhidos dos sensores e conforme políticas de rede, decidir qual o cenário que se encaixa melhor para a finalidade usando um processo de raciocínio. Após o procedimento anterior, o sistema atua sobre o cenário escolhido, por exemplo, com o uso de adequações nas tabelas de roteamento ou mesmo, no caso de leitura de ambiente externo, executando ações via agentes no ambiente e realizando os respectivos ajustes.

A princípio, o sistema aprende com o passado (situações, planos, decisões e ações que ocorreram), pois deve manter um histórico da rede juntamente com os seus elementos e não somente em termos de tabela de roteamento, como nos protocolos de rede comuns, usufruindo deste conhecimento para melhorar as decisões futuras.

Apesar das semelhanças com as redes *cross-layers* comuns, as redes cognitivas alcançam um escopo bem maior, sendo que as CN devem suportar os *trade-offs* entre múltiplos objetivos, com a execução de *Multi-Objective Optimizations* (MOO), enquanto que os projetos *cross-layer* normalmente executam um único objetivo de otimização. Modelos *cross-layer* executam otimizações específicas e independentes, que não contam para os objetivos de desempenho geral da rede (THOMAS, 2006).

Na Figura 21, a arquitetura de rede apresenta 3 camadas, que contemplam da baixa para a alta, a *Software Adaptable Network* (SAN) para aplicações com ligação ao meio de transmissão e recepção de dados, *Cognitive Process* (CP) para atuar com elementos cognitivos a fim de alcançar os objetivos traçados e *Cognition Selfish Layer* (CSL) que apresenta os objetivos fim-a-fim, bem como a relação com a consciência própria e geral da rede.

Figura 21. *Cognitive framework* e assuntos críticos de projetos.



Fonte: Adaptado de Thomas (2006).

A Tese de Thomas (2007) abrange um enquadramento para a operação do CN e menciona que não especificou detalhes para um elemento cognitivo genérico, envolvendo capacidades funcionais, tais como repositório de dados, dispositivo cognitivo ou mesmo um *framework* de comunicação entre elementos, que são mecanismos ainda por definir e abertos à interpretação.

Com base nessas características e na definição de um novo elemento cognitivo de WSN, que remete ao uso do conceito de memória, execução de processos, lembranças, histórico e tomada de decisões, esta Tese trabalha com a criação de um sistema ou dispositivo cognitivo com atuação *multi-layer* e processamento cognitivo.

2.6.2. *Cognitive Radio*

Conforme Arslan (2007), no final de 1990 quase todos os rádios de telecomunicações eram construídos utilizando um *Digital Signal Processor* (DSP) para implementar modulação e funções de processamento de sinal e um *General Purpose Processor* (GPP) para desenvolver interface de operação, sinalização de rede e funções gerais do sistema. Esta arquitetura é atraente para os fabricantes, pois as mesmas eletrônicas básicas podem ser usadas repetidas vezes para cada novo projeto de rádio, reduzindo assim a engenharia de desenvolvimento e permitindo a compra de grandes volumes, otimizando a produção de uma plataforma comum, porém retendo a flexibilidade para formas de onda e protocolos sofisticados. Alguns fabricantes optaram por *Software-Defined Radio* (SDR), proporcionando a criação de sistemas que sejam capazes de acrescentar funcionalidades adicionais altamente solicitadas para os padrões atuais em aplicações específicas.

O *Cognitive Radio* (CR) representa a integração de substancial inteligência computacional, em especial *machine learning*, visão computacional e processamento de linguagem natural em um SDR (MITOLA III, 2006).

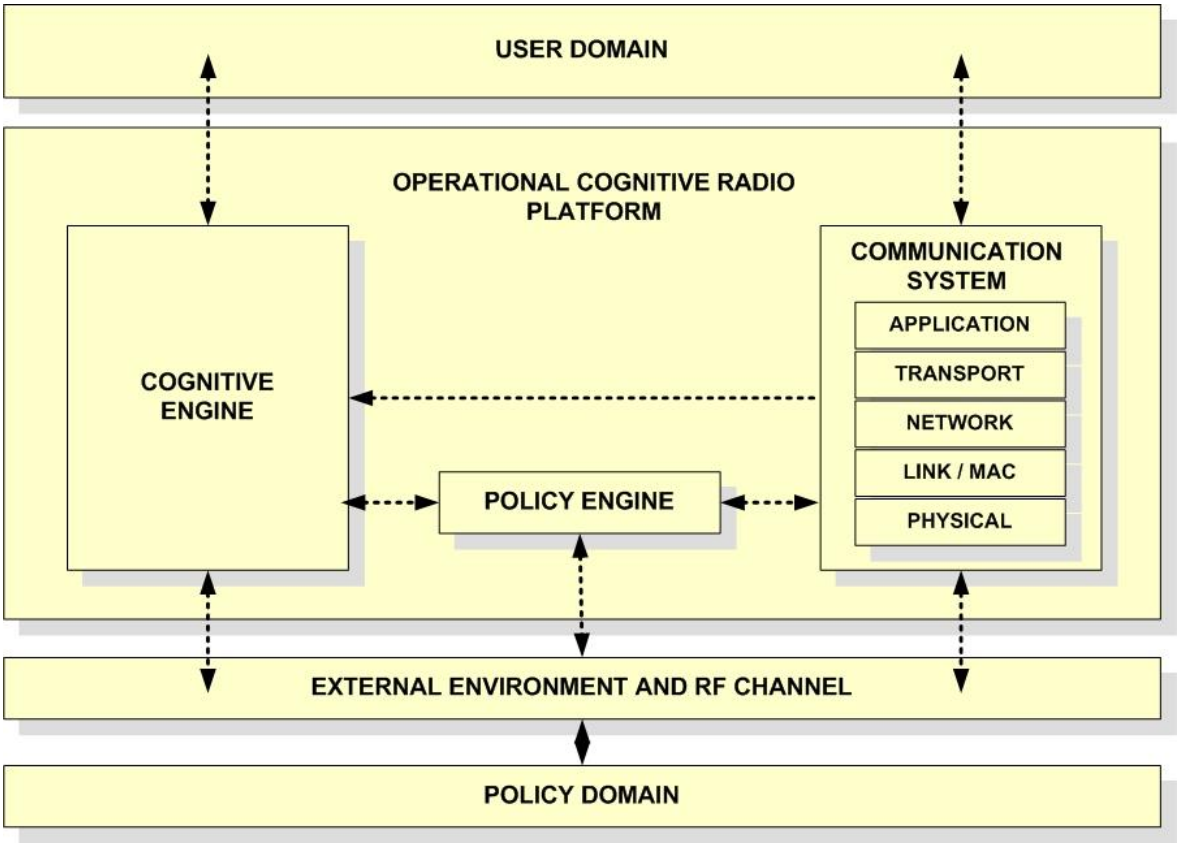
CR incorpora agente inteligente no domínio de RF como um *proxy radio* de acesso a informações do usuário, fazendo uma análise de rádio detalhada para tomar decisões sobre o comportamento do usuário através do uso de reconhecimento de padrões (não necessariamente da rede) para utilizar o espectro de radiofrequências de forma mais eficaz. A definição técnica é dada em ontologia computacional como *ideal Cognitive Radio* (iCR). Entre 1998 e 2000, Mitola III realizou a construção e um protótipo em linguagem Java denominado *Cognitive Wireless Personal Digital Assistant* (CWPDA), o qual mostrou potencial de valor econômico para mercados de rádios secundários de uso do espectro do iCR.

A FCC usa o termo cognitivo com o significado de “adaptativo” sem a necessidade de aprendizado de máquina. O iCR corresponde a um CR com a aprendizagem autônoma de máquina, visão (e não apenas o uso de uma câmera), e percepção de linguagem escrita e falada.

Foi realizada por Mitola III, uma adaptação e progressão de conscientização

para o CR, denominada *Aware, Adaptive and Cognitive Radio* (AACR), que aproveita as tecnologias de visão computacional, navegação, *Web* semântica e reconhecimento de fala e síntese. A percepção de máquina chega ao iCR como auto-percepção e percepção das necessidades de comunicação do usuário, incluindo prioridades e entendimento sobre características de espaço-tempo, além de situações em que o iCR atua de forma mais transparente e eficiente com informações úteis através de quaisquer meios sem fio que possam estar disponíveis.

Figura 22. Arquitetura interna de um CR.



Fonte: Adaptado de Arslan (2007).

A Figura 22 apresenta a arquitetura interna de um CR contendo a parte relativa à implementação cognitiva e relação com o domínio do usuário e o ambiente de RF, conforme políticas adotadas na rede. Conforme já notado, o CR está focado na otimização de uso do espectro de RF e não no mecanismo de atuação de um nó nas diversas camadas, conforme o sistema desenvolvido nesta Tese, com base na arquitetura CN de Thomas (2006).

Mitola III menciona ainda que o usual na abordagem de tecnologias sem fio *Always Best Connected* (ABC) é transformado pelo iCR, com foco na *Quality of*

Information (QoI), em *Always Better Informed* (ABI). Essa transformação é facilitada pelo uso de tecnologias *Web* semântica como o *eXtensible Markup Language* (XML) e *Ontology Web Language* (OWL) adaptadas para aplicações de rádio através de uma nova metalinguagem denominada *Radio* XML (RXML).

2.6.3. *Cognitive Sensor Network*

As *Cognitive Sensor Networks* (CSN) são equivalentes ao CN e diferem do conhecido *Cognitive Radio Sensor Network* (CRSN) ou CR, pois não ficam limitadas apenas às camadas mais baixas de leitura sobre o uso dos canais da rede com aplicação direta em otimizar o uso do espectro de radiofrequência, mas fazem referência a nós que possam trabalhar nas diferentes camadas da arquitetura de rede, desde as camadas mais baixas até a camada de aplicação, com atuação possível na formatação intercamadas dos nós e da rede. A seguir são apresentadas algumas abordagens de aplicação do CSN.

Shenai (2008) realiza a construção de um mecanismo baseado em CSN para atuação nas áreas de energia, agricultura, meio ambiente, saúde, transporte e defesa. O mecanismo baseia-se no uso de agentes inteligentes distribuídos na rede para o controle de segmentos locais que são sintetizados com base em uma especificação global de comportamento declarativo, bem como as políticas de gerenciamento do usuário em *Secure Operations Language* (SOL), que é uma linguagem de programação síncrona para especificação de sistemas distribuídos orientados a serviço.

Bhanu (2011) traz uma grande relação de trabalhos que envolvem a transmissão de vídeo distribuído por CSN. Destacam-se as redes de vídeo em larga escala, que estão se tornando cada vez mais importantes para uma ampla gama de aplicações críticas, ou seja, que são extremamente sensíveis a atraso e QoS, como vídeo vigilância, monitoramento de zonas de desastre e de tráfego, cuidado ao idoso, rastreamento de pessoas e veículos em ambientes lotados, bem como a aquisição de imagens mais realistas para dispositivos eletrônicos de consumo e entretenimento.

Na maioria das aplicações citadas por Bhanu, são utilizados múltiplos

sensores, como câmeras de vídeo, *InfraRed* (IR), ou sensores de alcance para reunir dados de vários pontos de vista, que são então enviados para uma unidade de processamento central. Na maioria dos sistemas existentes, não há processamento inteligente dos dados coletados localmente em cada câmera e de estações de videomonitoramento, que em geral são equipadas com pessoas para observar e, muitas vezes, armazenar múltiplos *streamings* de vídeo com um processo automatizado muito limitado.

Somov (2013) utiliza o conceito de CIoT em *smart-cities*, com o intuito de trazer considerável valor ao futuro de aplicações em *smart-cities* e *Intelligent Transport Systems* (ITS). Objetiva-se a redução do limiar de nível de entrada para desenvolvedores, uma vez que dá os meios para semanticamente enriquecer e agregar dados de uma forma que as partes interessadas em aplicativos possam explorar. Nos cenários de *smart-cities*, deve-se compreender como e quais informações extras são necessárias para enriquecer os dados detectados e publicá-los como dados compostos de *Resource Description Framework* (RDF) que podem ser de interesse dos fabricantes de sensores e, desta forma, ter vantagem para explorar o potencial de negócios de CIoT.

Cognitive Wireless Sensor Network (CogSeNet) é um sistema de rede inteligente de sensores sem fio que conta com processos cognitivos para fornecer uma capacidade dinâmica na configuração de RSSF (GOH, 2010). Considera-se também, como uma estrutura de rede formada por nós sensores equipados com módulos cognitivos que lhes permite observar o ambiente operacional e que realizam decisões.

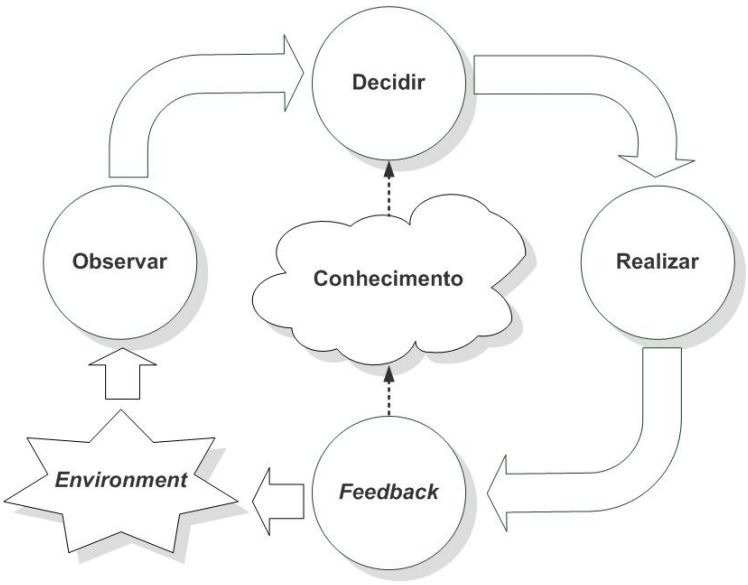
Uma série de ações é executada no CogSeNet, de modo que os nós possam se adaptar e atingir determinados objetivos dentro de uma política global. Esses objetivos podem ser tão simples como fornecer conectividade robusta ou tão complexos como negociar recursos adicionais de grupos de redes vizinhas para transmitir dados críticos. Goh define a CogSeNet como sendo:

> *"a networked group of sensor nodes capable of sustaining performance in a dynamic environment using its cognitive capability, all while meeting the users' requirement."* (GOH, 2010)

No CogSeNet a seleção dos modelos de inteligência e raciocínio não é

restringida, sendo possível desta forma utilizar qualquer módulo de inteligência artificial para formar a base do processo cognitivo. O objetivo fundamental é estabelecer um mecanismo de solução para o problema com uso de raciocínio baseado em regras. Dentro deste contexto, a Figura 23 apresenta o processo cognitivo do CogSeNet.

Figura 23. Processo cognitivo do CogSeNet.



Fonte: Adaptado de Goh (2010).

O conceito do CogSeNet, que segue a mesma linha do CN e CSN, também se aplica ao sistema desta Tese, uma vez que o sistema desenvolvido observa o ambiente e através do conhecimento toma decisões acerca do desempenho da rede, sempre levando em consideração as políticas da rede e a semântica adotada.

2.7. Protocolos de roteamento em CSN

Além dos protocolos de roteamento mais comuns para WSN, existem os protocolos de roteamento que foram desenvolvidos com base no pensamento de uso de conceitos cognitivos para rotear dados em redes de sensores sem fio e com aplicação em IoT.

2.7.1. *Efficient Traffic Aware Multipath Routing*

Efficient Traffic Aware Multipath Routing (ETAMR), segundo Li (2011) é um algoritmo de roteamento *multi-path*, que depende de um modelo de previsão

baseado no tráfego do *framework* da rede cognitiva. O protocolo possui forte adaptabilidade, robustez e menos *overhead*.

O ETAMR é implementado sob a arquitetura de rede cognitiva, sendo proativo, isto é, os sensores percebem a carga de tráfego em tempo real de cada nó do caminho principal e dos caminhos de *backup*, enquanto isso, o modelo de previsão de tráfego calcula a carga, para que na próxima vez que ocorrer, realize uma redução na carga de tráfego que excede o limite de tráfego, que é o resultado de aprendizagem e raciocínio do módulo. Se algum nó na via de transmissão atual é previsto para estar congestionado, o novo caminho de *backup* será desencadeado até que o congestionamento desapareça e o caminho primário ainda seja utilizado para o fluxo de dados.

2.7.2. *Trust Congestion Ant Colony Optimization*

Trust Congestion Ant Colony Optimization (TC-ACO) trabalha com o conceito de confiança e eliminação de nós problemáticos do percurso de encaminhamento de dados, o que proporciona minimização do congestionamento, melhoria da eficiência energética e do tempo de vida da rede.

Conforme Chakraborty (2013), WSN são geralmente operadas em modo de espera e, repentinamente, os nós tornam-se ativos em resposta a um evento detectado. Quando um grande número de nós sensores torna-se ativo, assim como para o transporte de dados para uma *Base Station* (BS), colisões de pacotes e nível de congestionamento da rede são bastante prováveis. Além disso, o *buffer overflow* ocorre devido ao tamanho limitado do *buffer* dos nós sensores, causando o descarte de pacotes que, por sua vez, diminui a taxa de transferência de dados na rede e consequentemente causa a redução no desempenho.

Nós sensores baratos geralmente são propensos a falhas e os nós falhos podem criar várias ameaças de segurança que agravam o problema de congestionamento por propagar difusão de pacotes inúteis, executar *flooding* com mensagens falsas, interferência intermitente e retransmissão da mesma mensagem diversas vezes. O efeito resultante disso tudo é uma computação e comunicação redundante, que causa desperdício de recursos de energia e uma redução no tempo

de vida da rede (CHAKRABORTY, 2013).

TC-ACO é um esquema de roteamento ciente do congestionamento baseado no algoritmo *Ant Colony Optimization* (ACO), no qual nós falhos, conhecidos como nós maliciosos, são detectados e isolados, utilizando uma estrutura baseada em *framework* de confiança e uso de feromônio, no qual os pacotes de dados são considerados como formigas artificiais e são lançados a partir do nó de origem para encontrar a melhor rota para o nó de destino em cada ciclo, para finalmente, como todas as formigas, seguir o caminho mais curto (DORIGO, 2006). A gestão da confiança é uma ideia que é usada para a detecção de nós falhos, a fim de estabelecer um caminho de roteamento de dados confiável a partir do nó de origem para a BS.

O conceito de confiança é basicamente emprestado da sociedade humana, em que os nós sensores monitoram o comportamento de seus vizinhos durante as operações de transferência de dados que passam por esses nós, com base em alguns parâmetros conhecidos como *Trust Metrics* (TM). O congestionamento do nó de confiança é calculado através da estimativa do espaço livre na fila do *buffer*. No algoritmo TC-ACO, ACO é utilizado para fatores de distância e TC para confiança e congestionamento visando à eficiência de energia, confiabilidade e otimização de roteamento em WSN (CHAKRABORTY, 2013).

2.7.3. *Directed Diffusion*

Directed Diffusion (DD) é um paradigma de difusão direcionada, baseado na localização de pedestres por uma rede de sensores, com característica de ser centrado em dados com a comunicação baseada em dados nomeados. Todos os nós em uma DD estão cientes da aplicação, permitindo que a difusão economize energia pela seleção empírica de bons caminhos, realize armazenamento em cache e faça o processamento de dados na rede.

O protocolo DD foi testado em uma rede simples de vigilância remota de pedestres. Motivado pelos requisitos de robustez, escalabilidade e energia, os dados gerados pelos nós sensores são nomeados por pares de *attribute-value*. Um nó solicita dados através do envio de interesses pelos dados nomeados. Encontrando

os dados de interesse, uma pilha de sequência até o nó é gerada. Nós intermediários podem armazenar em cache, transformar os dados, ou dirigir interesses com base em dados previamente armazenados.

Nesse processo, uma consulta do operador humano é transformada em um interesse que é difundido em nós de determinadas regiões (X ou Y). Quando um nó em uma região recebe um interesse, ele ativa seus sensores que começam a coletar informações sobre pedestres. Quando os sensores relatam a presença de pedestres, esta informação retorna ao longo do caminho no sentido reverso de propagação do interesse, no qual os nós intermediários podem agregar dados com a localização dos pedestres através da combinação de relatórios de diversos sensores.

Uma característica importante do DD é que o interesse, propagação e agregação de dados são determinados pelas interações de localização, com mensagens trocadas entre os nós vizinhos ou nós dentro de arredores da rede (INTANAGONWIWAT, 2000).

2.7.4. *Associativity-based Ad hoc Multicast*

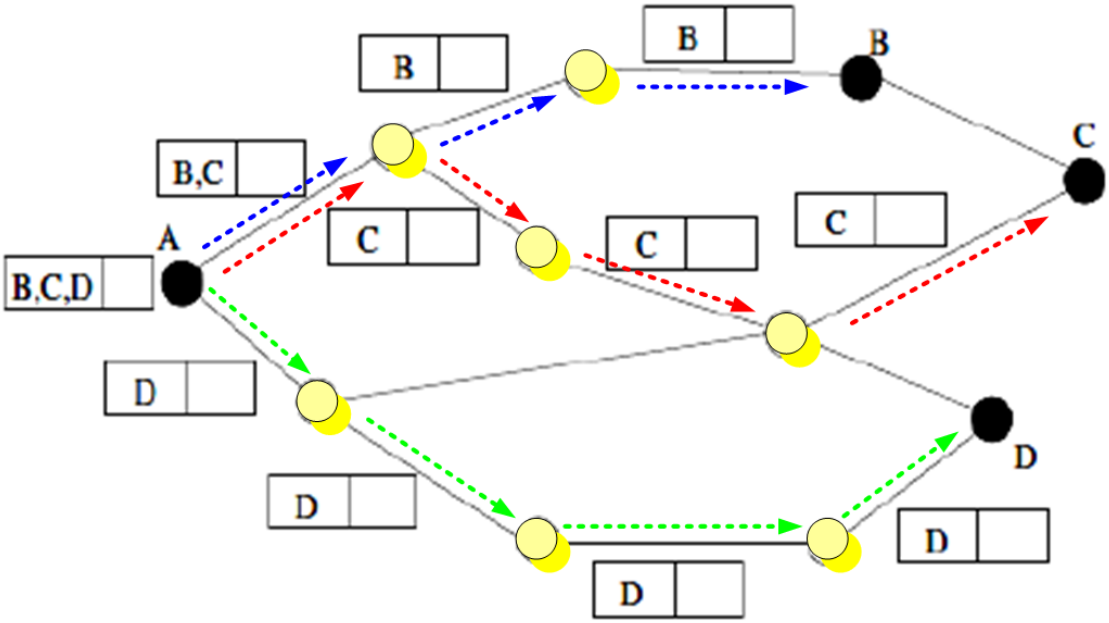
Associativity-based Ad hoc Multicast (ABAM) é um protocolo de roteamento *multicast* baseado no conceito de *spanning tree* e que trabalha sob demanda para redes móveis *Ad Hoc*. Uma árvore de rotas *multicast* é estabelecida no nó de transmissão para cada sessão *multicast*, baseado primeiramente na estabilidade de associação que se refere às características: espacial, temporal, de conexão e estabilidade de energia de um nó com seus vizinhos (TOH, 2000).

Após estarem estabelecidas e estáveis as árvores de rotas, um número baixo de reconfigurações são necessárias e o desempenho da comunicação na rede é melhorado. Este protocolo trabalha em 3 fases: estabelecimento das árvores de rotas *multicast*, reconfigurações das árvores *multicast* e exclusão de árvores *multicast*. Para iniciar uma sessão *multicast*, o nó transmissor envia mensagens *Multicast Broadcast Query* (MBQ) através da rede. Baseia-se também em comunicação de rotas na formatação de árvore como realizado no *Spanning Tree Protocol* (STP) (CISCO, 2015).

Os receptores *multicast* armazenam todas as informações dos MBQ e os nós com rotas mais estáveis enviam mensagens MBQ-REPLY em direção ao nó transmissor inicial, que por sua vez calcula a árvore *multicast* e gera mensagens MC-SETUP para estabelecer a árvore. O ABAM utiliza estratégia de reparação localizada com mensagens *broadcast Local Query* (LQ) e *Join Query* (JQ) para a inclusão do nó na árvore e manutenção. Na exclusão de árvores de rotas, usam-se mensagens LEAVE e DELETE para “podar” a árvore ou excluí-la totalmente.

Um exemplo de reconexão do protocolo ABAM é dado na Figura 24, na qual o protocolo de roteamento tenta reduzir o número de reconfigurações da árvore e a sobrecarga por processos de controle.O ABAM também tenta aumentar a distância em saltos entre o nó fonte e o de destino. Se existem muitos receptores do mesmo grupo *multicast*, isto leva ao congestionamento da maioria das rotas estáveis, o que leva ao aumento de atraso na rede e a redução na taxa de entrega de pacotes.

Figura 24. Exemplo de reconexão entre S e D pelo protocolo ABAM.



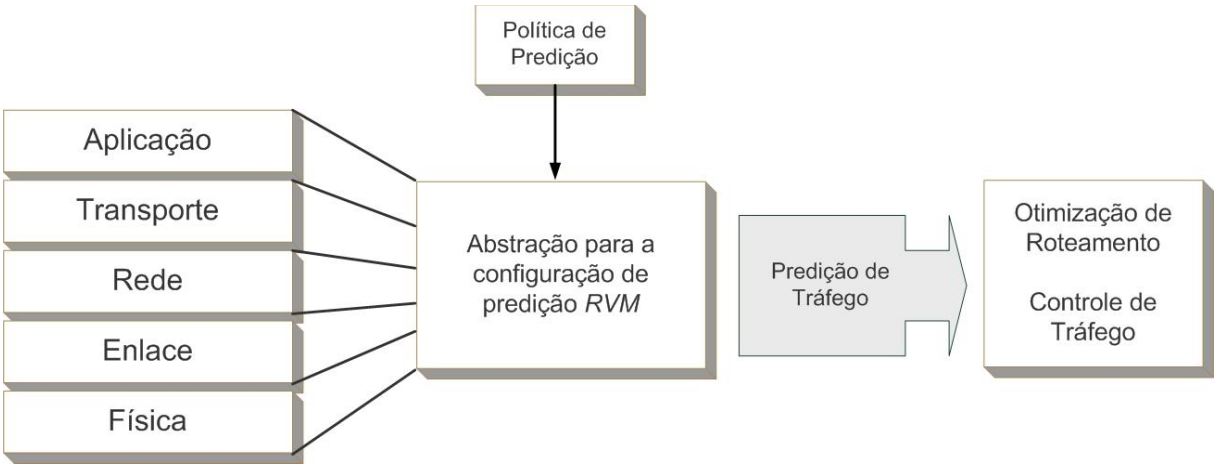
Fonte: Adaptado de Baker (2016).

Na Figura 24, existe a tentativa de reconexão entre o nó A e os nós B, C e D, em que os pacotes detêm o endereçamento do caminho entre A e os destinos traçados. Para alcançar B é executada a rota pelo caminho 1 (em azul), o mesmo ocorre para C pelo caminho 2 (em vermelho) e para D via caminho 3 (em verde).

2.7.5. *Relevance Vector Machine*

Relevance Vector Machine (RVM) é baseado no julgamento do espectro e de características de ambientes *wireless*, no qual o tráfego da rede pode ser previsto utilizando-se amostras periódicas de treinamento para formar um *loop* fechado de *feedback*. No trabalho de Zhang (2011), os resultados das simulações são comparados com um esquema de *Support Vector Machine* (SVM), demonstrando que o modelo proposto tem precisão promissora e viabilidade de desenvolvimento com relação à complexidade computacional.

Figura 25. Estrutura básica do módulo cognitivo do RVM.



Fonte: Adaptado de Zhang (2011).

A Figura 25 mostra a estrutura geral do protocolo, no qual o mecanismo de predição abstrai a informação de múltiplas camadas, treinando a informação como um conjunto de dados de predição, realizando a previsão de tráfego com a política de predição e, por fim, configurando a otimização do sistema ou executando o controle de dadas características do ambiente da rede (ZHANG, 2011).

2.8. Outros aspectos gerais em redes *wireless* e cognitivas

Dentro do campo de estudos de arquiteturas cognitivas, Langley (2009) fez uma seleção de arquiteturas conforme a frequência com que estas apareceram na literatura, incluindo diferentes graus de preocupação com referência ao comportamento humano. Em cada caso, é discutida a maneira de representação da arquitetura, organização, utilização e aquisição de conhecimento, juntamente com suas realizações.

Conforme Langley, uma arquitetura cognitiva especifica o uso de uma infraestrutura subjacente para um sistema inteligente. Resumidamente, uma arquitetura cognitiva inclui os aspectos de um agente cognitivo que são constantes ao longo do tempo e em diferentes domínios de aplicação.

Os agentes tipicamente incluem: memória de curto prazo e de longo prazo que armazenam conteúdo sobre as crenças, metas e conhecimento; representação dos elementos que estão contidos nestas memórias e a organização em estruturas mentais de larga escala; e processos funcionais que operam sobre estas estruturas, incluindo mecanismos de desempenho que os utilizam e os mecanismos de aprendizagem que os alteram.

Langley afirma ainda, que a pesquisa sobre arquiteturas cognitivas é importante, pois suporta um objetivo central de inteligência artificial e ciência cognitiva, que é a criação e compreensão de agentes sintéticos que suportam as mesmas capacidades dos seres humanos. Alguns trabalhos focam na modelagem de aspectos invariantes da cognição do ser humano, ao passo que outros esforços veem as arquiteturas como um caminho efetivo para a construção de agentes inteligentes.

2.8.1. Segurança

Segundo Fernández (2005), existem 4 limitações genéricas que todos os dispositivos *wireless* possuem: energia, largura de banda de comunicação, processamento e conexão de rede relativamente pouco confiáveis. A largura de banda disponível para sistemas sem fios é usualmente de pelo menos uma ordem de grandeza menor do que a disponível em dispositivos com fios.

O poder de processamento é limitado devido ao espaço, custo e restrições de energia de dispositivos *wireless*. A falta de confiabilidade na conexão é universal em todas as redes sem fio. Alguns protocolos foram projetados na tentativa de tentar melhorar e aproveitar esta falta de confiabilidade dos dispositivos, no entanto, na concepção destes protocolos, escolhas têm sido feitas sobre o tamanho dos pacotes e dos *frames* a serem usados, que acarretam um profundo impacto sobre a eficácia de protocolos de criptografia e de outras medidas de segurança.

Fernández afirma ainda que um fator que influencia bastante a área de segurança *wireless* é o fato de que há uma grande variedade de dispositivos utilizando diferentes arquiteturas, sistemas operacionais e funcionalidades. Acima de tudo, a segurança destes dispositivos, regularmente são maiores do que a dos dispositivos de rede cabeada e, isto se deve à própria natureza de utilização, por serem móveis, estarem na extremidade da rede, terem conexões não confiáveis e tenderem a ficar acidentalmente ou maliciosamente destruídos, além de poderem ser roubados, perdidos ou tornarem-se obsoletos.

Desta forma, é necessário ter processamento seguro, porém, em contrapartida, o ganho de processamento seguro gera o problema de facilmente sobrecarregar os processadores limitados de dispositivos sem fio, sendo um tema para muita pesquisa e conhecido como *security-processing gap*.

2.8.2. *Dynamic Cognitive Networks*

Dynamic Cognitive Networks (DCN) utiliza a ideia de *Fuzzy Cognitive Maps* (FCM) que corresponde a uma ferramenta de mapeamento cognitivo para modelagem do conhecimento humano, obtido utilizando-se termos linguísticos inerentes aos sistemas *Fuzzy*, facilitando o tratamento de dados, aprendizado e adaptação, em implementação de um algoritmo para DCN que permite que cada conceito selecione a sua relação de causalidade com os outros conceitos, de acordo com as exigências do sistema abordado.

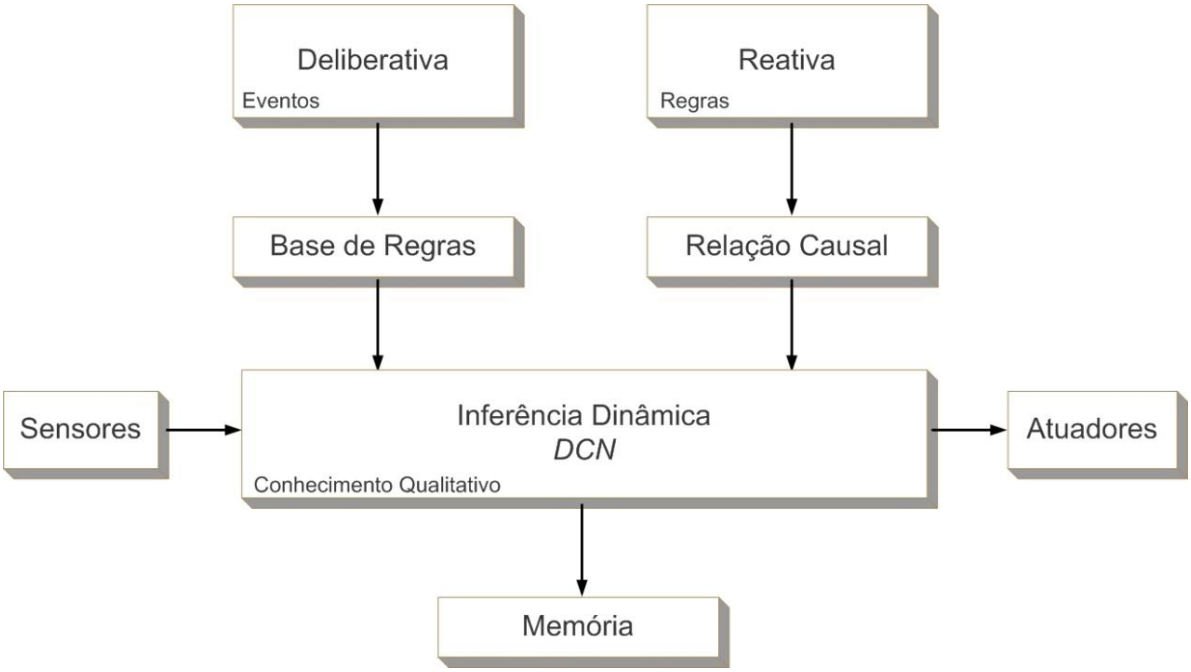
Além do que foi exposto, o protocolo realiza a introdução de procedimentos, visando a determinar como as causas produzem efeitos sazonais, utilizando para isto regras que identificam a ocorrência simultânea de diversos fatores, sendo, portanto, considerada como um modelo causal com habilidades dinâmicas em que as relações e os efeitos são variáveis dependentes do tempo (MENDONÇA, 2013).

A Figura 26 mostra o diagrama em blocos do DCN, representando a arquitetura com a inferência dinâmica e a relação de causas e efeitos, por meio de uso de regras. Mendonça exemplifica o uso do algoritmo vinculado à tomada de decisão em futebol de robôs, controle de sistemas supervisórios, misturador industrial, fermentador e navegação robótica, informando que utiliza a modelagem

matemática baseada em grafos para modelar a imprecisão e incertezas de um sistema dinâmico.

Mendonça afirma que a arquitetura desenvolvida permite a tomada de decisões dinâmicas através do mapeamento de 2 camadas com modelagem de diferentes tipos de conhecimento e que não está restrita apenas às aplicações mencionadas, podendo ser aplicada na construção de quaisquer tipos de sistemas que tenham a finalidade de modelar a tomada de decisão *on-line* a partir de eventos dinâmicos.

Figura 26. Arquitetura do DCN.



Fonte: Adaptado de Mendonça (2013).

A ideia básica de modelar sistemas dinâmicos com o uso de *Fuzzy* se aplica ao contexto desta Tese, porém, a arquitetura geral que oferece base de *framework* para a criação do sistema deste trabalho baseia-se na arquitetura cognitiva proposta por Thomas (2006), executando-se aqui a modelagem, desenvolvimento e simulação do módulo de processo cognitivo em CN ou CSN, aproveitando o legado de configurações básicas de nós de WSN, sendo que o mecanismo proposto e apresentado neste trabalho pode vir a ser inserido em dispositivos IoT ou aplicações em *smart-cities*.

2.8.3. *Trellis Coded Network*

Trellis Coded Network (TCNet) é um protocolo de roteamento que possibilita o encaminhamento de dados coletados por micro-sensores em cenários de redes *Ad Hoc*, com configuração de nós distribuídos aleatoriamente em uma área adversa, utilizando o conceito de máquinas de estados, modelagem de dígrafos e implementação de algoritmo de encaminhamento de pacotes com uso do conceito de códigos convolucionais (LIMA FILHO, 2015).

O protocolo apresenta a vantagem de possuir complexidade de implementação extremamente baixa, sendo compatível com os poucos recursos de *hardware* que os elementos de uma WSN usualmente possuem, porém, o protocolo não foi desenvolvido pensando-se em mobilidade dos nós.

É considerada a modelagem da rede em dígrafos, ou seja, utiliza-se a teoria dos dígrafos, na qual:

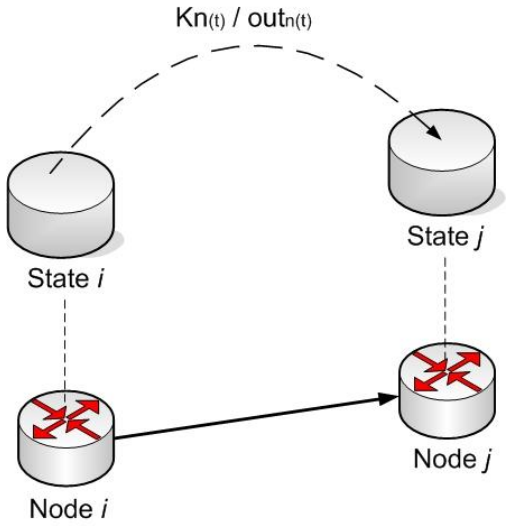
$$D = (V, E) \Longrightarrow \begin{cases} v_i \in V \\ E(v_i, v_j) \end{cases} \qquad 2.3$$

Onde: V é o conjunto dos vértices v_i que representam os nós ou equipamentos; e E é o conjunto de elementos v_i e v_j que são as arestas que representam as conexões físicas ou lógicas entre os vértices v_i e v_j.

O código convolucional, que é baseado em *Finite State Machine* (FSM), sugere uma analogia com os elementos de uma rede em que os nós representam os estados da FSM, utilizando para isto uma topologia de arestas orientadas.

Na Figura 27 é possível visualizar a relação de nós/*links* de uma rede WSN e estados/transições de uma FSM, na qual $k_n(t)$ representa um símbolo de entrada de uma sequência no instante t e que está relacionado com uma palavra código de saída $out_n(t)$ pela função de transferência ($k_n(t)/out_n(t)$), gerando a operação de uma FSM do $state_i$ para o $state_j$, representando portanto, uma analogia com um enlace de transmissão de dados do nó i para o nó j.

Figura 27. Representação de 2 nós no TCNet.



Fonte: Adaptado de Lima Filho (2015).

A FSM foi modelada como *Mealy Machine* (MM) por ser um gerador de baixa complexidade (WAGNER, 2009), construído com portas lógicas do tipo *eXclusive-OR* (XOR) e registradores de deslocamento, portanto, sendo atribuída a cada nó da rede. O funcionamento do mecanismo consiste em deslocar uma sequência $\{k_n\}$ modelada pela característica desejada na rede e utilizada como *label* do protocolo TCNet, através da MM e obter então, palavras-códigos que serão utilizadas como *header* dos *frames* com as informações necessárias sobre a rota na rede, que por sua vez, são definidas por meio de um procedimento *off-line*.

Utiliza-se o algoritmo de Viterbi (PROAKIS, 2008) para as tomadas de decisão sobre a realização da treliça, caracterizada como um *broadcast*, em que os nós da rede encontram-se na mesma área física e todos os nós podem receber o sinal. Neste cenário, considerando-se a semelhança entre topologia de grafos com o uso de treliças e transição de estados, verificou-se que no ambiente de WSN estes conceitos se comportam como uma transmissão de pacotes de dados.

2.8.4. Latência e Atraso fim-a-fim

A latência está relacionada com o atraso relativo da rede, ou seja, com o tempo necessário para que uma informação atravesse a rede, de um nó fonte (S) para um nó destino (D). Nesta Tese a latência foi medida como o atraso de ida e volta da informação enviada inicialmente pelo nó fonte. Desta forma, o *Round Trip*

Time (RTT) entre S e D de uma WSN foi definido como sendo o intervalo de tempo entre o momento em que um pacote está pronto para ser transmitido por S até o momento em que o pacote de resposta enviado por D para S é recebido pelo nó S. Neste ponto, a partir do momento em que a rota foi estabelecida entre os nós S e D, o nó D fica disponível para o recebimento do pacote de dados e envio de resposta deste recebimento.

O atraso fim-a-fim está diretamente relacionado com o atraso na entrega de um pacote de S para D, sendo medido como o intervalo de tempo entre o momento que o primeiro bit de um pacote é transmitido pelo nó S até o recebimento deste bit pelo nó D (KROPOTOFF, 2002).

A diferença entre a latência e o atraso fim-a-fim, é que para o atraso fim-a-fim considera-se a chegada de um pacote específico ao nó de destino, independentemente do caminho percorrido e, para a latência entende-se como o atraso em relação: à propagação do sinal, à transmissão devido ao meio *wireless*, ao roteamento de nós intermediários em diferentes caminhos entre S e D e ao processamento interno dos nós sobre os pacotes recebidos e retransmitidos entre S e D (BRASILWORK, 2016).

A medida de latência é diferenciada do atraso fim-a-fim pelas considerações apontadas e pelo RTT, conforme a equação 2.9. No atraso fim-a-fim considera-se apenas o tempo direto de transmissão e recepção de um pacote específico (geralmente tende-se a utilizar o tempo do primeiro pacote transmitido por S e recebido por D), conforme a equação 2.4.

$$A_{ff} = A_t + A_{pg} + A_{cp} \qquad 2.4$$

Onde: A_{ff} é o atraso fim-a-fim, A_t relaciona-se com o atraso de transmissão, A_{pg} com o atraso de propagação e A_{cp} ao atraso médio de espera na fila para a comutação de pacotes.

Para o cálculo do atraso de transmissão, utiliza-se a equação a seguir.

$$A_t = \frac{PL}{CT} \qquad 2.5$$

Onde: PL é o comprimento do pacote enviado e CT corresponde à taxa de transmissão do canal.

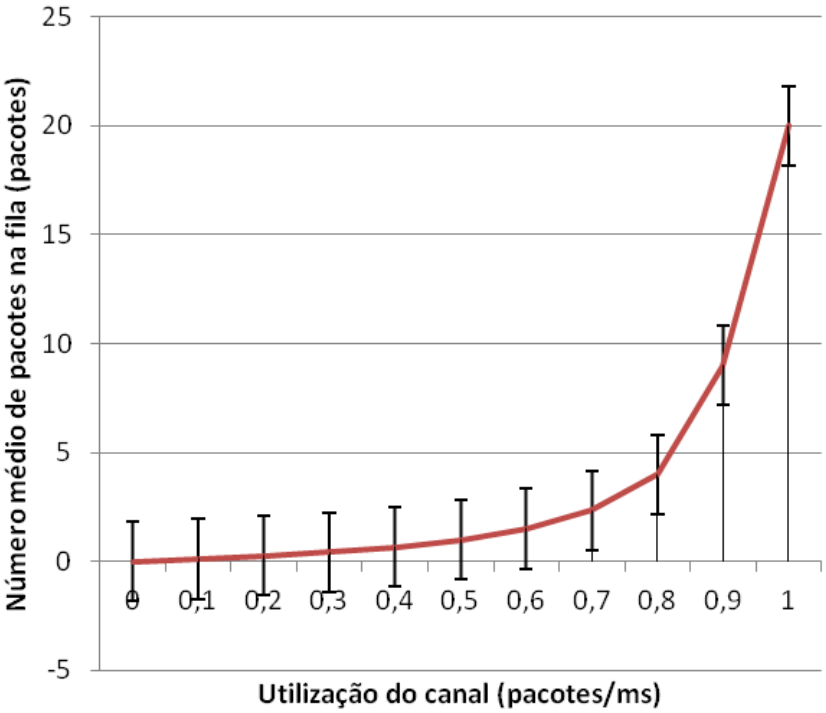
A equação 2.6 representa a obtenção do atraso de propagação.

$$A_{pg} = \frac{SDP}{CSPV} \qquad 2.6$$

Onde: SDP corresponde à distância percorrida pelo sinal e CSPV refere-se à velocidade de propagação do sinal no canal de transmissão.

A equação 2.7 representa o número médio de pacotes na fila de processamento dos nós, que por sua vez, é utilizada para a equação 2.8, que é utilizada para o cálculo do atraso médio de espera na fila de processamento para a comutação de pacotes.

Figura 28. Representação gráfica do número médio de pacotes na fila de processamento pela utilização do canal de transmissão.



Fonte: Autor.

$$N = \frac{\rho}{1-\rho} \qquad 2.7$$

Onde: ρ corresponde ao número de bits transmitidos pelo canal num intervalo

X de tempo e $1 - \rho$ refere-se ao número máximo de bits que podem ser transmitidos pelo canal no intervalo *X* de tempo.

$$A_{cp} = \frac{N}{\lambda} \qquad 2.8$$

Onde: λ corresponde à taxa média de chegada de pacotes na fila.

A equação 2.9 considera o período medido com relação ao RTT para determinar o valor da latência na rede.

$$L_{RTT} = A_{ffSD} + A_{ffDS} \qquad 2.9$$

Onde: A_{ffSD} é o atraso de ida do pacote partindo de S para D e A_{ffDS} é o atraso de retorno de D para S.

Ambas as métricas foram utilizadas nesta Tese para a execução das simulações, bem como a aquisição de dados para a determinação dos resultados.

2.8.5. Desvanecimento

O desvanecimento de um sinal corresponde basicamente à taxa na qual o sinal decai pela distância entre o nó transmissor e receptor (JACINTO, 2012). O desvanecimento pode ser do tipo plano (*path loss*) ou seletivo (*fading*).

Existem diferentes modelos aplicados ao desvanecimento plano, sendo modelos: de espaço livre, de 2 raios, *Log-distance path loss*, *shadowing*, Okumura, Hata, RIM (*Radio Irregularity Model*) e ERIM (*Enhanced Radio Irregularity Model*).

Dentre os modelos apresentados, o que foi adotado nesta Tese é o modelo de espaço livre, pois os cenários que foram implementados em *software* não possuiam obstáculos, impacto de relevo (apenas cenários de duas dimensões) ou mesmo áreas urbanas com possibilidade de sombras.

Quanto ao desvanecimento seletivo, existem diferentes distribuições que podem ser utilizadas na área de transmissão *wireless*, das quais as mais empregadas são: Rayleigh, Rice, Log-Normal e Normal.

A distribuição Rayleigh é aplicada quando não há uma componente de sinal dominante e a comunicação é via multi-percurso, sendo, portanto, um alcance indireto com múltiplos percursos entre S e D com sinais chegando em S com intensidades diferentes. A distribuição Rice se aplica a situações em que todas as componentes do sinal, resultantes de propagação multi-percurso, chegam à antena receptora com a mesma amplitude ou em comunicação direta entre S e D.

A distribuição Rayleigh foi utilizada na implementação dos testes desta Tese e a definição e determinação dos parâmetros estão descritos no Capítulo 4.

2.9. Lógica *Fuzzy*

Segundo o programa *Berkeley Initiative in Soft Computing* (BISC), que é um centro de pesquisa básica e aplicada em Lógica *Fuzzy*, que se encontra dentro de *Soft Computing* (SC), cujos principais constituintes são: *Fuzzy Logic* (FL), *Neural Networks theory* (NN) e *Probabilistic Reasoning* (PR), em conjunto com os posteriores campos de pesquisa correspondentes a rede de crenças, computação evolutiva (incluindo DNA *computing*), teoria do caos e partes da teoria de aprendizagem (BISC, 2016). As ideias básicas sobre SC e que ainda estão bem presentes na atualidade, tem relação direta com muitas influências anteriores e, entre elas, a descrição inicial sobre conjuntos *Fuzzy* (ZADEH, 1965).

Zadeh introduziu os conceitos de Lógica *Fuzzy* como uma teoria que busca a representar algo impreciso ou vago (lógica nebulosa) como são as expressões humanas de linguagem natural, oferecendo base para o raciocínio aproximado que dispõe sobre conceitos imprecisos e tornando-se uma ferramenta para modelar o conhecimento. Dentro destes conceitos, sustenta-se que os principais elementos do pensamento humano são as etiquetas linguísticas e não os números.

Nos conjuntos *crisp* ou clássicos, a representação de pertinência está relacionada aos valores 0 e 1, onde a função indicadora $f_A(x)$ representa um grau de pertinência restrito de x em A com duas hipóteses, conforme relaciona a equação a seguir:

$$f_A(x) = \begin{cases} 1 \leftrightarrow x \in A \\ 0 \leftrightarrow x \notin A \end{cases} \qquad 2.10$$

A função de pertinência de conjuntos *Fuzzy* relaciona-se com o intervalo [0,1], correspondendo a valores percentuais presentes entre 0 e 100%, em que tem-se $\mu_A(x)$: $X \rightarrow [0,1]$, onde $\mu_A(x)$ representa o grau de confiança em que uma determinada etiqueta linguística se aplica a um elemento particular do conjunto *Fuzzy*, evidenciando desta forma a compatibilidade. Apesar das funções de pertinência poderem adquirir diferentes formas, utiliza-se com mais frequência as funções gaussianas, triangulares e trapezoidais (MALVEZZI, 2010).

Segundo Zadeh (1973), variáveis linguísticas são aquelas em que os valores são um tipo de sentença de uma linguagem natural *L*, onde cada palavra *x* pode ser vista como uma descrição sumarizada de um subconjunto *Fuzzy M(x)* de um universo de discurso *U*, com *M(x)* representando o significado e, conforme Lee (1990), sendo caracterizada por meio de uma expressão que é apresentada a seguir:

$$f_A(x) = (x, T(x), G(x), M(x), U) \qquad 2.11$$

Onde: *x* representa o nome da variável, *T(x)* o conjunto de nomes atribuídos à variável *x*, *G(x)* a regra sintática para geração dos nomes dos valores de *x*, *M(x)* a regra semântica para associação de cada valor com o respectivo significado e *U* o universo de discurso.

O conjunto *Fuzzy* é totalmente caracterizado por sua função de pertinência, conforme a equação 2.12, na qual *A* é o conjunto *Fuzzy*, $\mu_A(x)$ a função de pertinência e *X* ou *U* o universo de discurso.

$$A = \{(x, \mu_A(x)) \mid x \in (X \vee U)\} \qquad 2.12$$

A teoria dos conjuntos *Fuzzy* acaba sendo uma extensão da teoria tradicional dos cojuntos, pois trabalha com o grau de verdade {verdadeiro, falso}, classificando os conjuntos com rótulos qualitativos em termos linguísticos (insuficiente, ruim, regular, bom, muito bom, excelente, etc.) e caracterizando as informações destes conjuntos pela variação do grau de pertinência.

A base de regras é fornecida por especialistas (comandos: se, então, e, ou, etc.), utilizando o formato de sentenças linguísticas, que podem estar associadas a ações de controle de operadores em função da observação de dados de entrada e saída de um processo (MALVEZZI, 2010).

Resumidamente, pode-se dizer que a Lógica *Fuzzy* permite o tratamento de expressões que envolvem quaisquer tipos de grandezas descritas de forma não exata, ou seja, que não possuam valores específicos como são os valores binários (NASCIMENTO JR., 2000).

2.9.1. Inferência e modelos Mamdani e Sugeno

Inferência é o ato ou processo de derivar conclusões lógicas a partir de determinadas premissas conhecidas ou assumidas como verdade. As regras da inferência são válidas e utilizadas no campo da lógica e podem ser definidas como meio não lógico, porém com significados racionais, através da observação de padrões, para indiretamente prover significados e contextos para o entendimento de algo. A inferência não determina conclusões, mas possibilita a verificação dos significados sobre observações efetuadas.

Existem dois tipos de inferência: indutiva e dedutiva. Na indutiva é priorizado o raciocínio, no qual as premissas são vistas como forte evidência para a verdade da conclusão e na dedutiva, o raciocínio dedutivo tem ligação direta com as conclusões, ou seja, se todas as premissas forem verdadeiras, contendo termos claros e seguindo as regras lógicas dedutivas, então a conclusão será necessariamente verdade. Portanto, a conclusão de um argumento dedutivo é dita como certa, enquanto que a verdade da conclusão de um argumento indutivo é provável com base em uma dada evidência.

A Lógica *Fuzzy* trabalha com três operadores lógicos definidos que podem estar em tabelas da verdade e que são utilizados em lógica digital, compondo o conceito de portas lógicas e Álgebra de Boole: ∧ (AND), ∨ (OR) e ¬ (NOT). Estes operadores fazem parte da relação para a criação de regras *Fuzzy*.

No algoritmo *Fuzzy* do modelo de controle Mamdani, cada regra é uma

proposição condicional, sendo que a implementação é feita mediante a definição dos operadores lógicos para o devido processamento dos dados que antecedem à regra e da função de pertinência que irá definir o valor resultante. A ação é definida pela agregação de *n* regras que compõem o algoritmo, mediante o uso do conectivo de vinculação, o qual pode ser aplicado por diferentes operadores. Esta agregação resulta no conjunto *Fuzzy* que define a saída do controlador, sendo que a saída efetiva é obtida por meio de um processo de defuzzificação aplicado ao conjunto de saída do controlador (ANDRADE, 2008).

No algoritmo *Fuzzy* do modelo de controle Sugeno, que é na verdade uma simplificação do modelo Mamdani, o resultado de cada regra é definido como uma função das variáveis linguísticas de entrada. A regra geral do Sugeno é mostrada na equação 2.13, na qual o resultado de cada regra é um valor numérico, que assume como peso o valor da pertinência resultante do processamento dos dados. Essa condição dispensa a definição de uma função de pertinência específica, pois a resposta final do controlador é obtida pela média ponderada das respostas das regras individuais, portanto, neste tipo de controlador não ocorre o processo de defuzzificação.

$$\begin{gathered}\text{SE } x = A_i \text{ AND } y = B_i \text{ ENTÃO } w = f_i(x, y) \rightarrow \\ \rightarrow \text{IF}\{x = A_i \ \wedge \ y = B_i\}, \text{THEN } w = f_i(x, y)\end{gathered} \qquad 2.13$$

O valor de *w* refere-se à função de A_i e B_i e pode ser definido como um valor constante, que pode ser interpretado como um conjunto *Fuzzy* de um único valor com pertinência igual a um e com todos os demais componentes com pertinência zero (*singleton*) (ANDRADE, 2008).

2.9.2. Fuzzificação

Fuzzificação refere-se a um mapeamento do domínio de números reais para o domínio *Fuzzy*, atribuindo valores linguísticos e descrições qualitativas com relação a funções de pertinência vinculadas às variáveis de entrada. Pode ser considerada como um pré-processamento de categorias ou classes dos sinais de entrada.

As funções de pertinência podem ser um conjunto tabulado de valores numéricos que podem acelerar a fuzzificação através do uso de procedimento para consulta às tabelas. É possível calcular os valores dos vetores de possibilidades (produção de p_n vetores *Fuzzy* baseados na fuzzificação de x_n vetores discretos) pelas equações que representam as funções de pertinência.

2.9.3. Defuzzificação

Na defuzzificação, executa-se a interpretação do conjunto *Fuzzy* de saída, no qual o valor da variável linguística de saída inferida pelas regras *Fuzzy* é traduzido em valor discreto, objetivando ter um único valor numérico discreto que melhor represente os valores *Fuzzy* inferidos sobre a variável linguística de saída, abordando desta forma a distribuição de possibilidades. A seleção do método apropriado de defuzzificação pode ser baseada no centroide ou nos valores máximos que ocorrem na função de pertinência resultante, dos quais destacam-se: *Center of Area* (CoA), *Center of Maximum* (CoM) ou *Mean of Maximum* (MoM) (SHAW, 1999).

Para o CoA o cálculo do centroide da área se dá através da equação 2.14, na qual $\mu_{out}(u_i)$ é a área de uma função de pertinência e u_i é a posição do centroide da função de pertinência individual. Este método pode apresentar pequenos problemas nos casos em que as funções de pertinência não possuam sobreposição ou quando houver duas funções de mesma saída *Fuzzy* com sobreposição de áreas, o que acarreta na não contabilização de uma das áreas.

$$u_{\text{defuzzy}} = \frac{\sum_{i=1}^{N} u_i \mu_{\text{out}}(u_i)}{\sum_{i=1}^{N} \mu_{\text{out}}(u_i)} \qquad 2.14$$

No CoM os picos das funções de pertinência representados no universo de discurso da variável de saída são usados, ignorando-se as áreas das funções e trabalhando-se com as contribuições múltiplas de regras. O valor de saída defuzzificado e discreto é determinado achando-se o ponto de apoio onde os pesos

ficam equilibrados entre os vetores de peso posicionados nos picos da função. A saída é calculada através de uma média ponderada dos máximos, cujos pesos resultam das inferências. Na equação 2.15, $\mu_{0,k}(u_i)$ indica os pontos em que ocorrem os máximos das funções de pertinência de saída.

$$u_{\text{defuzzy}} = \frac{\sum_{i=1}^{N} u_i \cdot \sum_{k=1}^{n} \mu_{0,k}(u_i)}{\sum_{i=1}^{N} \sum_{k=1}^{n} \mu_{0,k}(u_i)} \qquad 2.15$$

No MoM a média de todos os máximos pode ser utilizada nos casos em que existam funções de pertinência que tenham mais do que um máximo, na qual u_m representa o m-ésimo elemento no universo de discurso e a função $\mu_{out}(u_i)$ tenha um máximo com M sendo a quantidade destes elementos, desconsiderando portanto, o formato das funções de pertinência de saída, conforme pode-se ver na equação 2.16.

$$u_{\text{defuzzy}} = \sum_{m=1}^{M} \frac{u_m}{M} \qquad 2.16$$

O método de defuzzificação pode ser considerado contínuo, se uma mudança infinitamente pequena numa variável de entrada não causa uma mudança abrupta em quaisquer variáveis de saída. Para sistemas com malha fechada, o CoM apresenta resultados mais coerentes pois saltos na saída do controlador podem causar instabilidade e oscilações enquanto que o MoM apresenta melhor desempenho em reconhecimento de padrões pois o vetor de possibilidades de saída é o resultado da classificação entre a similaridade do sinal e objetos padrão. Ambos fornecem suporte à decisão, com o CoM sendo adotado para decisões quantitativas (como alocação de recursos) e o MoM em decisões qualitativas (por exemplo em detecção de fraude).

3. METODOLOGIA

A metodologia aborda os aspectos gerais relevantes sobre WSN, cognição, escolha de *software* para simulação, criação de algoritmo e desenvolvimento de sistema cognitivo.

O sistema desenvolvido analisa aspectos da rede abordando as situações para que atendam às políticas da rede, buscando sempre a atingir a QoS definida como base nas relações de semântica.

Aspectos da área cognitiva foram incorporados ao ACS, abrangendo histórico, memória e tomada de decisão, que possibilitaram ao mecanismo optar pelas melhores situações na rede, conforme estas forem aparecendo e de acordo com o aprendizado dos nós controladores.

O estudo trata sobre conceitos de redes de sensores, envolvendo teoria, desenvolvimento, composição dos nós e da rede, tipos de protocolos de roteamento, arquitetura de rede e interna dos nós, tipos de conexões e tipos de topologia, conceito de cognição, envolvendo casos de aplicação e reconhecimento, teoria de máquinas, interface homem-máquina, aspectos cognitivos em diversos casos, inteligência artificial, estudo de aspectos cognitivos (como memória, lembrança, histórico, atividade cerebral e tomada de decisão) e padrões de implementação de inteligência em máquinas/*softwares*.

A pesquisa engloba também o estudo de Lógica *Fuzzy*, padrões de implementação de fuzzificação (*Mamdani* e *Sugeno*), tipos de entradas e saídas, modelos de análise, estudo sobre regras, inferência e defuzzificação, além de interpretação de gráficos de regras e de relação de inferência para a geração de defuzzificação.

Para a escolha sobre a ferramenta a ser utilizada como base de implementação e desenvolvimento em *software* desta pesquisa, foram considerados 3 aspectos: WSN, *Fuzzy* e cognição. Portanto, foram selecionados os seguintes *softwares*, dentre os mais conhecidos e em uso para a área de redes e análise acadêmica:

- OPNET (OPNET, 2012);
- OMNET++ (OMNET, 2013);
- NS-2 (NS-2, 2013);
- TOSSIM (TOSSIM, 2013);
- MATLAB – *Prowler* (PROWLER, 2015); e
- *Software* R (R SOFTWARE, 2013).

Neste passo, foi escolhido o software *Prowler* para simular WSN ao invés do NS-2, OMNET++ ou OPNET, pois o MATLAB fornece diferentes ferramentas (HANSELMAN, 2003) complementares às redes de sensores, como a facilidade para implementação de Lógica *Fuzzy* e de modelos de simulação genéricos através do *Simulink*. O mesmo com relação ao *Software* R, pois o MATLAB também fornece base para implementação e trabalho de vetores de informação com geração de gráficos analíticos.

O *software* MATLAB foi escolhido para a elaboração deste trabalho, pois fornece base forte em implementação de simulação, execução de testes e geração de gráficos de resultados simultaneamente com a simulação da rede de sensores, além de oferecer um *toolbox* já pronto para implementação da Lógica *Fuzzy* e apresentar ferramentas de interface gráfica que possibilitam a integração via *Simulink* com um modelo que unifica a Lógica *Fuzzy* com a simulação em WSN proporcionada pelo *Prowler*. Portanto, oferecendo a integração entre Lógica *Fuzzy*, WSN e *Simulink tools* que o *Software* R não contempla, bem como o OPNET e o NS-2 (EZREIK, 2012).

Bajaj (1999) afirmou que o uso de simulação em pesquisas da área de redes desempenha um papel valioso por proporcionar um ambiente no qual o desenvolvimento e testes de novas tecnologias de rede possuam baixo custo, bem como auxilia na complexidade de construção de bancos de ensaio. Embora não seja um substituto completo para *testbeds*, uma estrutura padrão para simulação usada por um conjunto diverso de pesquisadores aumenta a confiabilidade e aceitação de resultados de simulações.

No contexto de simulação de redes de sensores, Korkalainen (2009) fez um comparativo do *Prowler* com os simuladores NS-2, OMNET++, TOSSIM e OPNET, chegando à conclusão de que:

- O NS-2 (SINGH, 2012) e OMNET++ necessitam de extensão ou modificação para chegar a resultados mais precisos em simulações de WSN;
- Todos os simuladores possuem extensões customizadas para serem escritas ou baixadas da Internet para modelagem de WSN;
- Simulações de redes móveis e mudanças de topologias dinâmicas são possíveis em todos os simuladores, exceto no TOSSIM;
- Quando considerada a escalabilidade para redes com muitos nós, o NS-2 tem desempenho mais lento em relação aos outros simuladores;
- Todas as ferramentas têm carência por modelagem de um ambiente real em 3D, exceto o OPNET para a modelagem de rádio;
- O *Prowler* é bem adequado para o desenvolvimento de protocolos e algoritmos, e também pode proporcionar as métricas de QoS sobre o desempenho da aplicação testada.

Outro ponto que Korkalainen colocou foi que a exportação de códigos de simulação, por exemplo, o C/C++ ou TinyOS, de algoritmos de roteamento (ou aplicações) para os nós sensores é mais fácil quando utilizadas plataformas comerciais dependentes de emuladores TinyOS, uma vez que rodam mais códigos usados em nós sensores.

Dentro de plataformas de simuladores independentes de propósitos gerais, como o NS-2, a troca de códigos-fonte é possível quando as interfaces e bibliotecas necessárias estão presentes na plataforma de destino.

A Tabela 3 apresenta um comparativo entre os simuladores de WSN mencionados por Korkalainen, detalhando os aspectos gerais relativos aos *softwares*. Outros pontos não destacados no comparativo referem-se à inserção de cognição e ao trabalho com Lógica *Fuzzy*, que influenciaram bastante na escolha do *software* para simular redes de sensores desta Tese.

Tabela 3 – Comparação entre diferentes simuladores de WSN.

Descrição	NS-2	OMNET++	*Prowler*	OPNET	TOSSIM
License	GPL	*Academic and commercial*	*Academic*	*Academic and commercial*	*Open source*
Sensor platform support	*Universal*	*Universal*, TinyOS	*Universal*, TinyOS	*Universal*	TinyOS (*Berkeley* MICA *Mote*)
Simulation code support	C++, C	C++, C, nesC	m-file	C++, C	C++, C, nesC
Simulation code exportable	*Limited*	*Limited*, (NesC *yes*)	*No*	*Limited*	*Yes*
Scalability to large networks (n>100)	*Fair* (*some cases*)	*Good*	*Excellent*	*Excellent*	*Excellent*
Protocol design/optimization	*Possible*	*Possible*	*Possible*	*Possible*	*Possible*
Mobile network simulation	*Yes*	*Yes*	*Yes*	*Yes*	*No*
Dynamic network topology change possible	*Yes*	*Yes*	*Yes*	*Yes*	*No*
Real-time OS/SW *execution time modeling*	*No*	*No*	*No*	*No*	*Limited*
Needs extensions for proper WSN simulation	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*
Custom extensions can be written	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*
3D radio modeling	*No*	*No*	*No*	*Yes*	*No*
Mobile network simulation	*Yes*	*Yes*	*Yes*	*Yes*	*No*
Standard MAC and routing support	802.11, 802.15.4, DSDV, DSR, TORA, AODV	802.11, *fair* (*Ad Hoc routing*)	802.15.4, *fair* (*Ad Hoc routing*)	802.11, 802.16, UMTS, *Smart* MAC, 802.15.4, AODV, DSR, GRP, OLSR, OSPFv3, TORA	CSMA, *primitive*
Standard energy modeling (*battery/radio*)	*Primitive* / T_X, R_X, *idle*	*Primitive* / T_X, R_X, *sleep*	*Primitive*	*Primitive* / T_X, *idle*, *sleep*	*None*
Publicly available extensions for WSN	*Energy models*, *protocols*, *SensorSim*	*Energy models*, *protocols*, *SensorSim*, EWSNSIM, *Castalia*	*Plugins for energy models*, *protocols*	*Several under license*	TinyViz, *Power*TOSSIM
Implemented radio models	*Shadowing*, 2-*ray ground, free space*	*Free space*, 2-*ray ground* (*experimental*, Castalia)	*Probabilistic and deterministic*	*Free space*, CCIR, Hata, *Longley-Rice*, TIREM, *Walfish-Ikegami*	*Probabilistic bit error*

Fonte: Adaptado de Korkalainen (2009).

Com base no *software* escolhido realizou-se um estudo do *Prowler*, focando no funcionamento, dados gerados de saída, dados selecionados como entrada, arquivos tipo **.m* e demais características do *software*.

Para a implementação da Lógica *Fuzzy* no MATLAB, utilizou-se a ferramenta *toolbox Fuzzy*, na qual foram criadas as funções de pertinência de entrada e saída, regras *Fuzzy* e inferência para a defuzzificação.

No desenvolvimento da arquitetura do sistema CPMod, foi criado um modelo com interface gráfica, abrangendo a Lógica *Fuzzy*, que foi implementado com interligação com a simulação de rede executada no *Prowler*.

Para a realização de testes, conforme descrito no método científico, foram abordados os cenários, configurações da rede, protocolos de roteamento, métricas para análise e geração de gráficos.

3.1. Método científico

Conforme os objetivos da Tese, procurou-se estabelecer uma sequência de procedimentos de testes e análises dos resultados, com base na implementação de simulação no *software Prowler* e modelagem matemática acerca do comportamento dos nós na WSN.

3.1.1. Método para o desenvolvimento do algoritmo de variabilidade

Para a modelagem da característica de variação dos nós sensores, desenvolveu-se um algoritmo de detecção de variabilidade, baseado em:

- Mobilidade do nó na rede;
- Alcance da antena do nó sensor;
- Quantidade de conexão com os nós vizinhos; e
- Quantidade de nós que compõem a rede.

Para tanto, foram realizadas observações sobre o ambiente de WSN, com relação às características apontadas. Após as observações, foi realizada a

adequação das equações relacionadas com cada característica apontada, abrangendo um comportamento genérico dos nós.

Com as equações definidas, pode-se representar o funcionamento das características de variabilidade de WSN através de um pseudo-código desenvolvido, com a finalidade de detectar a variabilidade na rede e facilitar o entendimento do funcionamento do mecanismo de acordo com as características observadas.

3.1.2. Método para o desenvolvimento do ACS e CPMod em *software*

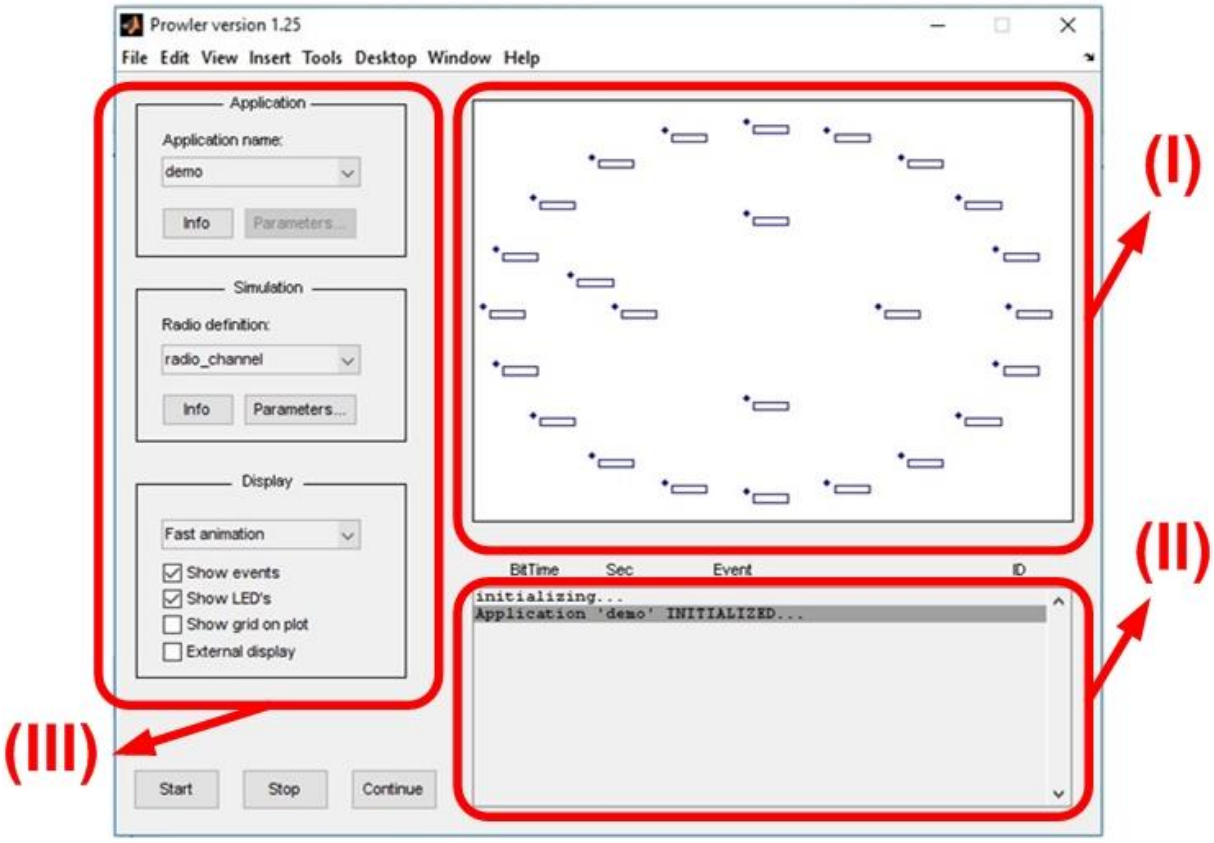
Para o desenvolvimento do ACS utilizou-se o MATLAB, com implementação e teste na forma de *software*, no qual foram criadas todas as interfaces de característica cognitiva (representando os três aspectos apresentados anteriormente), abrangendo também o uso do *Simulink* do MATLAB, envolvendo o desenvolvimento de Lógica *Fuzzy* e de modelos de simulação. Como o objetivo do trabalho é a inserção de características cognitivas em uma rede de sensores *wireless* já existente, foi realizada adaptação do programa *Prowler*.

Optou-se por utilizar o *Prowler*, pois o trabalho não trata sobre criação de redes e topologias, bem como sobre desenvolvimento de padrões de roteamento ou encaminhamento de dados. O mecanismo desenvolvido neste trabalho funciona em múltiplas camadas, realizando decisões *multi-layer* sobre características a serem alteradas na rede. Para que os nós dos cenários de testes fossem monitorados e controlados, foi desenvolvida uma ferramenta de atuação para as devidas alterações na busca em atender à semântica adotada para a rede.

Com o pseudo-código desenvolvido na parte matemática, executou-se a implementação do mecanismo para a detecção de variabilidade na rede, através da implementação das equações de variabilidade no MATLAB.

Para a realização de simulação de WSN, utilizou-se o *Prowler*, cuja tela principal é mostrada na Figura 29, onde (I) representa a tela do cenário demonstrativo de rede com os nós distribuídos em sua maioria de forma circular, (II) um visualizador de eventos e (III) seleção de parâmetros para serem adotados durante a simulação.

Figura 29. Interface principal do *Prowler*.



Fonte: Adaptado de *Prowler* (2015).

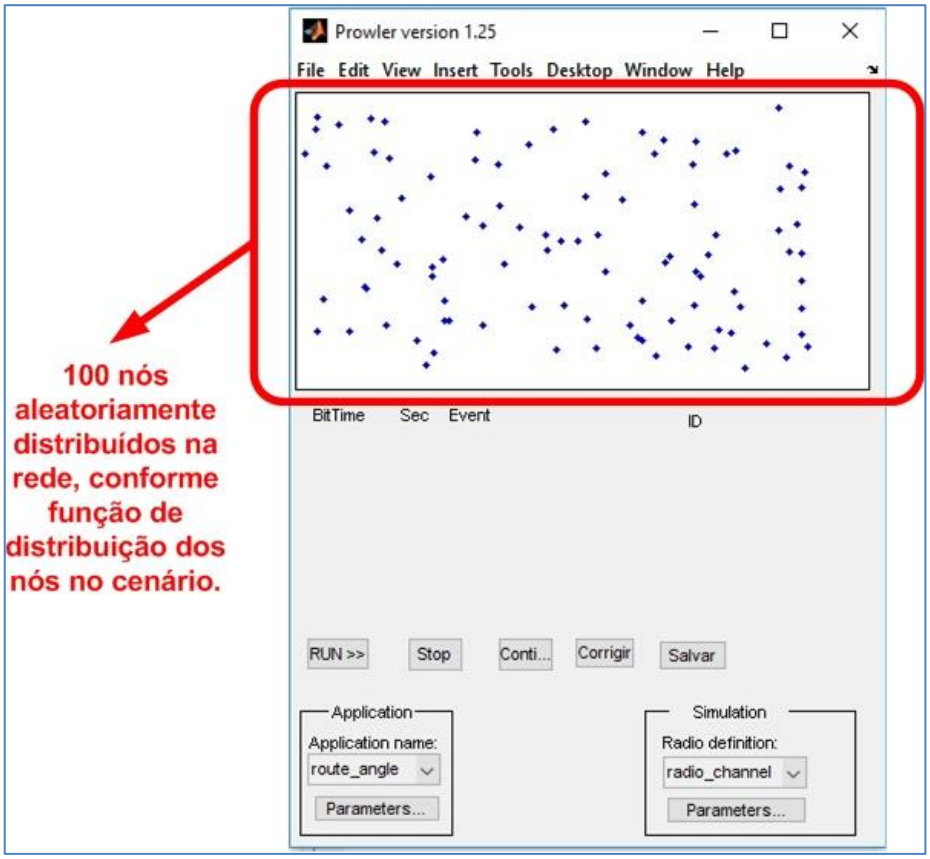
A criação da Lógica *Fuzzy*, foi baseada na implementação de regras *Fuzzy*, com geração de funções de pertinência para fuzzificação e defuzzificação, e realização de inferência, com base em *Quality of Experience* (QoE) de redes de sensores e na semântica aplicada à WSN.

O desenvolvimento do CPMod, abrange a criação de um modelo com inserção no *Simulink* do MATLAB, juntamente com conexão ao *Prowler* adaptado da Figura 30 e ao bloco *Fuzzy* implementado.

Na Figura 30, a tela principal do *Prowler* foi alterada para suprir as necessidades deste trabalho, na qual ficou mantida a parte que demonstra o cenário e a de visualização de eventos.

Foram mantidos os botões de iniciar a simulação, parar e continuar a simulação, e acrescentados botões para a correção e salvamento dos dados, caso seja necessária a intervenção manualmente.

Figura 30. Interface do *Prowler* adaptado.



Fonte: Adaptado de *Prowler* (2015).

São coletadas do *Prowler* as seguintes informações, que servem como base de dados para análises e como *structs* e *cells* (CHAPMAN, 2010) para uso pelo CPMod:

- Quantidade total de energia dos nós e da rede;
- Intensidade de sinal na recepção pelos nós;
- Tempo de simulação;
- Quantidade de pacotes enviados;
- Quantidade de pacotes recebidos;
- Quantidade de pacotes com colisão;
- Mensagem transmitida nos pacotes;
- Posição dos nós;
- Quantidade de nós da rede.

Após a coleta dos dados, são definidos 4 vetores de 1 dimensão cada, que abrangem dados de energia dos nós da rede, tempo de propagação dos pacotes até o destino, quantidade de pacotes descartados e quantidade de atraso fim-a-fim, para servirem como entrada do bloco CPMod.

Com o CPMod criado, deve-se então estabelecer a semântica da rede, ou seja, o conjunto de conceitos linguísticos sobre as tendências de QoS, visando dar significado aos dados resultantes da defuzzificação com relação ao que é aceitável ou não para a rede, desenvolvendo-se o bloco de memória semântica. Com base nisto, uma consulta ao arquivo de dados históricos da rede é realizada abrangendo o bloco de memória episódica (lembrança), para então verificar se já ocorreu tal situação na rede.

Em caso positivo de ocorrência da situação na rede, o CPMod adotará as medidas da semântica, caso contrário o ACS executará *rounds* de análises verificando a cada *round* a condição da rede e o limiar de decisão baseado na semântica, para que então seja possível estabelecer a continuidade de execução dos *rounds* ou a finalização e armazenamento dos dados na memória episódica, ou seja, no histórico de melhores situações da rede para futura lembrança.

3.1.3. Nós sensores

Geralmente os nós sensores captam informação (podendo ser de diversos tipos, como: temperatura, luminosidade, presença, entre outros), armazenam (conforme capacidade de armazenamento do nó) e transmitem periodicamente estas informações ou quando um nó interessado entra no raio de cobertura do sinal do sensor envolvido (ou seja, quando os sensores têm proximidade suficiente para transmitir e receber informação dentro do alcance da antena transmissora/receptora (T_X/R_X)).

O nó sensor deve possuir memória interna para armazenamento de informações, processador e capacidade de roteamento de dados provenientes de outros nós. Para atender aos requisitos estipulados, foi selecionado o uso de nós sensores do tipo MICA *motes*, que pertencem à última geração de sensores e foram desenvolvidos inicialmente pela Universidade da Califórnia (BERKELEY, 2015).

Tabela 4 – Características dos sensores MICA *motes*.

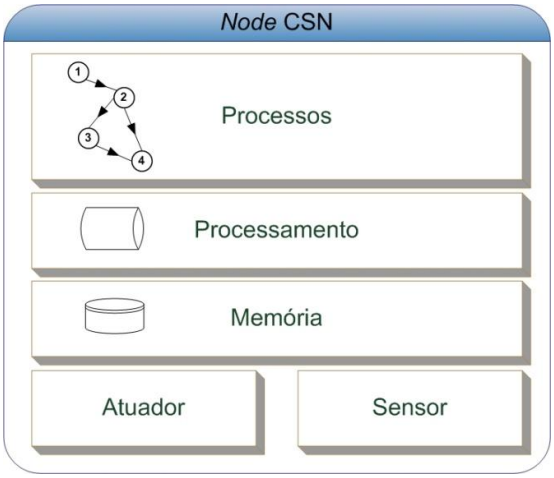
Características	Sensores MICA *motes*		
	MICA *mote*	**MICA 2 *mote***	**MICAz *mote***
Modelo	MOT300 - MPR300CA	MICA2 - MPR400CB	MICAz - MPR2400CA
Rádio Transceptor	TR1000	CC1000	CC2420 - IEEE 802.15.4 RF *transceiver*
Frequência	916 MHz	868/916 MHz, 433 MHz (MPR410CB) ou 315 MHz (MPR420CB)	2,4 GHz até 2,48 GHz
Taxa de Transmissão	Disponibilidade para operar nos modos OOK (10 kbps) or ASK (115 kbps), com máxima capacidade de transmissão de 50 kbps	38,4 kbaud na configuração Manchester	250 kbps em *High Data Rate Radio*
Processador	Microcontrolador Atmel Atmega 103. AT90LS8535 de 4 MHz, 35 pinos (I/O), voltagem de operação de 2,7 a 5,5 V. Consome 19,2 mJ/s no modo *Active*, 5,7 mJ/s no modo *Idle* e menos de 3 µJ/s no modo *Sleep* (medidas de consumo para 4 MHz, 3V e 20 ºC). Utiliza 0,75 mW, com 12 mA para transmitir e 1,8 mA na recepção	Microcontrolador Atmel Atmega128L, que é um microcontrolador de baixa potência. Na transmissão 27 mA, recepção 10 mA e no modo *Sleep* <1µA.	Microcontrolador Atmel Atmega128L. O rádio MICAz (MPR2400), oferece alta velocidade (250 kbps) juntamente com segurança de hardware (AES-128). Utiliza 8mA no modo *Active* e menos que 15 µA no modo *Sleep*
Comunicação	Rádio Transceptor RF em 916 MHz, com capacidade de transmitir em média 10 kbps. Possui gastos de energia com transmissão na ordem de 36 mJ/s e recepção entre 5,4 mJ/s a 14,4 mJ/s. Alcance de 20 m aproximadamente	Rádio Transceptor RF em 916 MHz. A potência de transmissão pode ser regulada entre 0,01 mW e 3,16 mW, variando o alcance entre 5 e 91 m	Rádio Transceptor RF em 2,4 GHz. O alcance externo é de 75 m a 100 m e interno de 20 m a 30 m
Sistema Operacional	TinyOS. Ocupa apenas 178B de memória	TinyOS. Ocupa apenas 178B de memória	TinyOS. Ocupa apenas 178B de memória
Memória	128 kB de memória programável, 512 kB de memória SRAM para dados e 32 kB de EEPROM	128 kB de memória programável, 512 kB de *flash* e 4 kB de EEPROM de configuração	128 kB de memória programável, 512 kB de *flash* e 4 kB de EEPROM de configuração
Dimensões	7.62 cm x 2.54 cm x 1.27 cm	5,71 cm x 3,17 cm x 0,63 cm, tendo aproximadamente 11cm^3. O peso do *mote* é de 18gramas	58 mm x 32 mm x 7 mm. O peso do *mote* é de 18gramas
Tipos de Sensores Disponíveis	Luz, temperatura, acelerômetro, ruído e sensores sísmicos	Luz, temperatura, acelerômetro, ruído, magnetômetro, umidade e pressão barométrica	Na expansão: luz, temperatura, RH, pressão barométrica, acelerômetro/Sísmico, acústica, magnético e outros tipos MEMSIC *Sensors*
Bateria	Utiliza 2 baterias AA com tensão de operação 3V. Pode funcionar por aproximadamente 1 ano em modo de baixo consumo	Utiliza 2 baterias AA com tensão de operação de 2,7V a 3,3V. Pode funcionar por aproximadamente 1 ano em modo de baixo consumo	2 baterias AA. Tensão entre 2,7V e 3,3V. Pode funcionar por aproximadamente 1 ano em modo de baixo consumo
Características Especiais	Expansão (51 para conectar outros elementos ao sensor)	Expansão (51 para conectar outros elementos ao sensor)	Expansão (51 para conectar outros elementos ao sensor)
Aplicação	WSN com capacidade de roteamento em redes de larga escala (+1000 nós), *Ad Hoc* e Monitoramento	WSN com capacidade de roteamento em redes de larga escala (+1000 nós), *Ad Hoc* e Monitoramento	WSN com capacidade de roteamento em redes de larga escala (+1000 nós), *Ad Hoc mesh networking* e Monitoramento

Fonte: Adaptado de Silva (2014) e Leopold (2007).

Os sensores MICA *motes* (MICAz) têm característica de pouco consumo de energia, suporte à comunicação sem fio e maior capacidade de armazenamento. Os *motes* são padronizados pelo IEEE 802.15.4, em uma frequência de 2,4 GHz e largura de banda de aproximadamente 250 kbps, garantindo baixo custo, baixa potência e confiabilidade (SILVA, 2014).

A Tabela 4 apresenta a descrição das características da família de sensores MICA *motes*, relacionando as informações necessárias para a escolha do sensor a ser utilizado na WSN do *Prowler* para simulação. TinyOS é um sistema operacional dirigido a eventos que foi desenvolvido para aplicação em nós sensores. Foi projetado para suportar as operações intensivas de concorrência ao canal que são comuns nas redes de sensores e utilizar poucos recursos de HW, pois ocupa apenas 178B de memória. Neste panorama, é possível que um nó não somente atue na sensibilidade de detecção de fenômenos no ambiente, mas também, como roteador, atuador e identificador de processos referentes à aplicação. Para visualizar o funcionamento do mecanismo interno do nó, a Figura 31 mostra a pilha de interação básica de um nó genérico do ambiente de rede proposto em CSN.

Figura 31. Pilha com blocos genéricos internos dos nós sensores.



Fonte: Autor.

Devido ao fato de ser o mais avançado dos modelos estudados e por oferecer: grande capacidade de transmissão; maior espaço para armazenamento de dados; operar no padrão IEEE 802.15.4; e suportar o TinyOS como sistema operacional de “baixa ocupação de espaço em disco”, o MICAz da família MICA *motes* foi adotado para as simulações em *software*.

4. DESENVOLVIMENTO

Os próximos tópicos mostram a formatação do nó sensor juntamente com o desenvolvimento em blocos do CPMod, e após será descrita a realização matemática para a parte inicial da ideia do sistema com o algoritmo para a detecção de variabilidade, incluindo aspectos inerentes à tomada de decisão, que posteriormente foram alterados para implementação em *software* MATLAB, com a modelagem do CPMod, uso de Lógica *Fuzzy* e *rounds* de análises.

O algoritmo de variabilidade apenas detecta se houve ou não variação de dados de características no nó, conforme explicado a seguir, contudo o mecanismo que auxilia na decisão sobre os índices da Tabela 6 é incorporado em Lógica *Fuzzy*, fazendo o papel do *Decision Making* (DM) juntamente com o bloco de semântica da rede.

4.1. Desenvolvimento do algoritmo de detecção de variabilidade

Visando cumprir a proposta presente nos objetivos, que é a criação de um algoritmo para detecção de variações de características básicas dos nós sensores, realiza-se nesta etapa o desenvolvimento de um algoritmo de detecção de variabilidade.

Para a modelagem dos nós sensores, é prevista a aplicação em ambientes dinâmicos ou diferenciados, ou seja, o estudo envolve redes com mobilidade aleatória dos nós e não está limitado a um único tipo de cenário. O ambiente de rede é adaptativo, ou seja, qualquer conexão entre os nós pode ser refeita a cada alteração no meio, de forma que o sistema se adapte ao ambiente e às configurações físicas do mesmo, seja para monitoramento ou atuação.

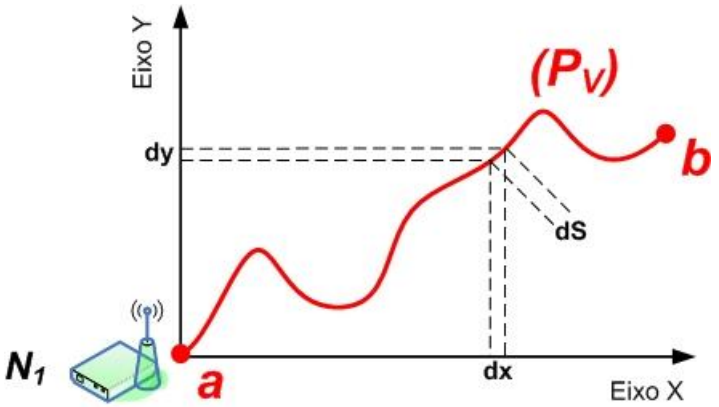
Com a finalidade de modelar 4 características básicas de um nó sensor, foi elaborada a equação 4.1, que contempla a movimentação do nó, alcance da antena, quantidade de conexões possíveis com nós vizinhos e número de nós na rede, sendo calculada em determinados instantes na rede, caracterizando uma equação para eventos discretos, abrangendo pequenos segmentos de cada característica.

$$F_V = (\alpha.P_V + \beta.R_V + \gamma.L_V + \sigma.N_V) \quad 4.1$$

Onde: F_V representa a hipótese da característica de variabilidade do nó e, é composta por: P_V variação de posição do nó, R_V variação de alcance entre os nós, L_V a variação na quantidade de conexões entre os nós e N_V a variação no número de nós da rede.

Os valores de P_V, R_V, L_V e N_V são obtidos das relações a seguir, onde *t* representa o tempo atual e *t-1* o tempo num passo anterior. Os valores das variáveis de variabilidade do sistema P_V, R_V, L_V e N_V são obtidos através dos cálculos das equações a seguir. Para a definição da variável de posição P_V, utilizou-se a característica de movimentação aleatória do nó N_1 com deslocamento de *a* para *b*, conforme é representado na Figura 32.

Figura 32. Característica de movimentação aleatória do nó na WSN.



Fonte: Autor.

Supondo-se que *dS* seja um comprimento inclinado do traçado de movimentação do nó N_1 de *a* até *b* no segmento delimitado por *dx* e *dy*, então tem-se pelo Teorema de Pitágoras que:

$$dS = \sqrt{dx^2 + dy^2} \quad 4.2$$

O traçado de *a* até *b* representa a curva *C* em função dos eixos X e Y, portanto, considerando-se um traçado contínuo de deslocamento *P* do nó, tem-se:

$$P = \int_C f_{(X,Y)} dS \quad 4.3$$

Substituindo-se a equação 4.2 em 4.3, resulta em:

$$P = \int_C f_{(X,Y)} dS = \int_a^b f_{(x,y)}.\sqrt{dx^2 + dy^2} = \int_a^b f_{(x,y)}.\sqrt{dx^2.\left(1 + \frac{dy^2}{dx^2}\right)} \Longrightarrow$$

$$\Longrightarrow P = \int_a^b f_{(x,y)}.dx\sqrt{1 + \left(\frac{dy}{dx}\right)^2} = \int_a^b f_{(x,y)}.\sqrt{1 + (y')^2}dx \mid y' = \frac{dy}{dx} \qquad 4.4$$

Portanto, a variação de posição do nó obedece à equação 4.5 resultante, adotando-se a relação de tempo para a verificação de deslocamento do nó no percurso da Figura 33, e $f_{(x,y)}$ representa a função entre *a* e *b*, que pode ser uma junção de diversas funções básicas concatenadas para cobrir a trajetória de *a* até *b*.

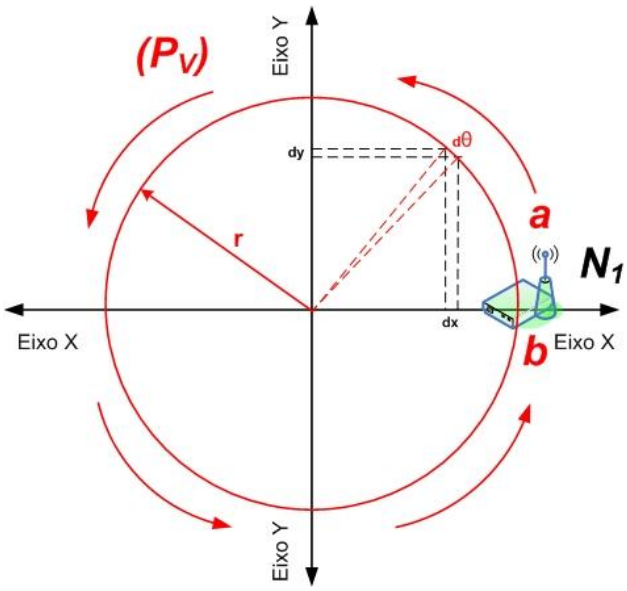
$$P_V = \left\{P_t - P_{t-1} \mid t \geq 1;\ P = \int_C f_{(X,Y)} dS = \int_a^b f_{(x,y)}.\sqrt{1 + (y')^2}dx \ \mid y' = \frac{dy}{dx}\right\} \qquad 4.5$$

Caso o percurso do nó seja circular, conforme a Figura 33, a seguinte relação deve ser abordada considerando-se o ângulo de deslocamento *θ* e os eixos X e Y:

$$\begin{cases} X \rightarrow x(\theta) = r.\cos\theta \\ Y \rightarrow y(\theta) = r.\sin\theta \end{cases} \qquad 4.6$$

Na Figura 33, *r* refere-se ao raio do traçado executado pelo nó e *dθ* o quanto houve de deslocamento no traçado circular.

Figura 33. Característica de movimentação circular do nó na WSN.



Fonte: Autor.

Dadas as informações anteriores (nas equações sobre a característica de movimentação com a posição do nó) e considerando-se:

$$\frac{d\theta}{d\theta} = 1\ , \quad d\theta = \sqrt{d\theta^2}, \quad dS = \sqrt{dx^2 + dy^2} \quad \text{e} \quad f_{(x(\theta),y(\theta))} = 2\pi r \qquad 4.7$$

Tem-se que:

$$dS = \frac{d\theta}{d\theta}.\sqrt{dx^2 + dy^2} \rightarrow dS = d\theta.\sqrt{\frac{dx^2 + dy^2}{d\theta^2}} \rightarrow dS = d\theta.\sqrt{\frac{dx^2}{d\theta^2} + \frac{dy^2}{d\theta^2}} \rightarrow$$

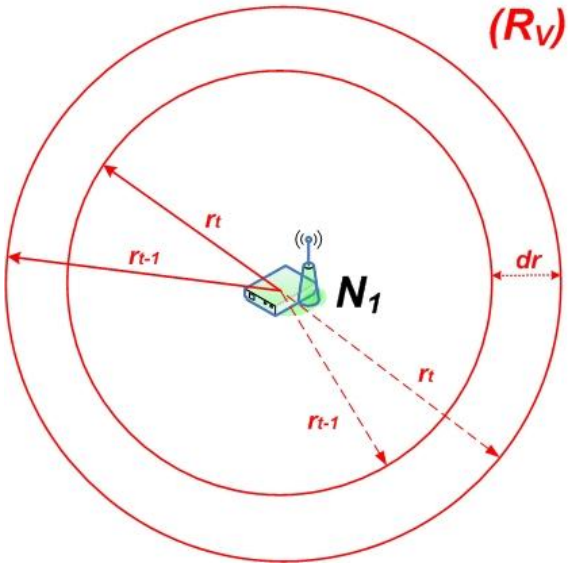
$$\rightarrow dS = \sqrt{\left(\frac{dx}{d\theta}\right)^2 + \left(\frac{dy}{d\theta}\right)^2}\, d\theta \rightarrow P_C = \int_0^{2\pi} f_{(x(\theta),y(\theta))}.\sqrt{\left(\frac{dx}{d\theta}\right)^2 + \left(\frac{dy}{d\theta}\right)^2}\, d\theta \qquad 4.8$$

Portanto, para movimentação circular do nó P_{VC} tem-se:

$$P_{VC} = \left\{ P_t - P_{t-1} \mid t \geq 1;\ P_C = \int_0^{2\pi} f_{(x(\theta),y(\theta))}.\sqrt{\left(\frac{dx}{d\theta}\right)^2 + \left(\frac{dy}{d\theta}\right)^2}\, d\theta \right\} \qquad 4.9$$

Pensando-se no raio de cobertura ou alcance do sinal da antena do nó, adotaram-se as relações para a variação de intensidade de sinal, ou seja, a diferença na distância limite de alcance em relação ao nó, conforme a Figura 34.

Figura 34. Característica de cobertura do sinal do nó na WSN.



Fonte: Autor.

Na Figura 34 observam-se duas formatações de condição do sinal de cobertura de raio *r* do nó, sendo que a linha contínua representa uma diminuição no alcance do nó e a linha tracejada um ganho no alcance do sinal da antena.

Com a variação entre o alcance atual r_t e o de um momento anterior r_{t-1} e, considerando-se uma diferença denominada *dr* entre os mesmos, obtém-se então o valor do alcance para o momento atual como sendo:

$$R = r_{t-1} + \int_{r_{t-1}}^{r_t} dr \mid r_t \neq r_{t-1} \qquad 4.10$$

Portanto, a variação do alcance R_V estará condicionada à *R*, que pode ser igual a r_{t-1} caso não exista alteração no raio de cobertura do nó. Considera-se na equação 4.11 a variação de *r* até o infinito apenas para demonstrar que o alcance da antena não tem um limite pré-definido, o que é correto do ponto de vista da matemática, porém, incoerente do ponto de vista de redes devido ao alcance real das antenas dos sensores, que em sua maioria são em dezenas de metros.

$$R_V = \left\{ R_t - R_{t-1} \mid t \geq 1;\ 0 \leq r \ll \infty;\ r_{t-1} \leq R \leq \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr \right) \right\} \qquad 4.11$$

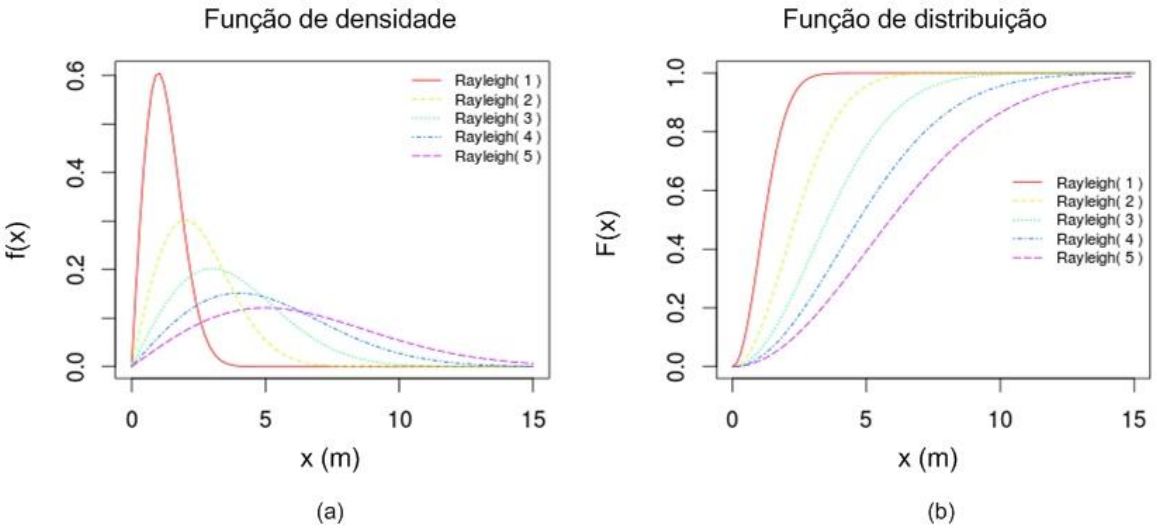
O alcance do sinal da antena também está diretamente ligado ao efeito de desvanecimento do sinal no meio de transmissão, que no caso de comunicações sem fio pode ser adotado como do tipo Rayleigh, ou seja, quanto maior a distância entre os nós, maior será a queda de sinal conforme a curva da FDP Rayleigh, que está diretamente relacionada com a distância em relação ao nó fonte, sendo apresentada na Figura 35 e tendo a notação $R \sim \mathrm{Rayleigh}(\sigma)$, na qual σ representa o fator de escala.

A Figura 35a mostra a FDP para uma sequência de 5 simulações de σ e na Figura 35b é apresentada a função de distribuição para as mesmas 5 variações de σ. Nota-se também, através da Figura 35a, que quanto maior o valor de σ mais suave se torna a curva da função e na Figura 35b que conforme o valor de σ aumenta, mais distribuído fica a função em relação à *x*.

Para o entendimento sobre Rayleigh tem-se que, conforme duas variáveis

aleatórias independentes *V* e *U*, sendo que $V \sim N(0, \sigma^2)$ e $U \sim N(0, \sigma^2)$, uma equação $X = \sqrt{V^2 + U^2}$ segue a distribuição de Rayleigh. Dadas estas informações, a função densidade de probabilidade conjunta de *V* e *U* é dada pela equação 4.12.

Figura 35. Função Densidade de Probabilidade do tipo Rayleigh para canal de comunicação.



Fonte: Adaptado de PortalAction (2016).

Na equação 4.12 tem-se a função densidade de probabilidade conjunta, enquanto que a equação 4.13 apresenta a função de distribuição.

$$f_{(v,u)} = f_{(v)} f_{(u)} \rightarrow f_{(v,u)} = \left[\frac{1}{\sigma\sqrt{2\pi}} e^{\left\{\frac{-v^2}{2\sigma^2}\right\}}\right] . \left[\frac{1}{\sigma\sqrt{2\pi}} e^{\left\{\frac{-u^2}{2\sigma^2}\right\}}\right] \Rightarrow$$

$$\Rightarrow f_{(v,u)} = \frac{1}{2\pi\sigma^2} e^{\left\{\frac{-(v^2+u^2)}{2\sigma^2}\right\}} \rightarrow f_{(v,u)} = \frac{1}{2\pi\sigma^2} e^{\left\{\frac{-1}{2}\left(\frac{v^2}{\sigma^2}+\frac{u^2}{\sigma^2}\right)\right\}} \Rightarrow \qquad 4.12$$

$$\Rightarrow f_{(v,u)} = \frac{1}{2\pi\sigma^2} e^{\left\{\frac{-1}{2}\left(\frac{v^2+u^2}{\sigma^2}\right)\right\}}, (v,u) \in \Re^2$$

Na equação 4.12, $B_x = \{(v,u) \in \Re^2 : v^2 + u^2 \leq x^2\}$.

$$F_{(x)} = P_{(X \leq x)} = \int_{B_x} f_{(V,U)} d_{(V,U)} = \int_{B_x} \frac{1}{2\pi\sigma^2} e^{\left\{\frac{-1}{2}\left(\frac{v^2+u^2}{\sigma^2}\right)\right\}} d_{(V,U)} \qquad 4.13$$

Considerando as coordenadas polares com $v = r.\cos\theta$ e $u = r.\operatorname{sen}\theta$, pode-se dizer que:

$$v^2 + u^2 = r^2 \cos^2\theta + r^2 \operatorname{sen}^2\theta = r^2(\cos^2\theta + \operatorname{sen}^2\theta) = r^2 \qquad 4.14$$

Utilizando-se a equação 4.14 em 4.13, tem-se:

$$F_{(x)} = \int_0^{2\pi}\int_0^x \frac{1}{2\pi\sigma^2} e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} r dr d\theta = \frac{1}{2\pi\sigma^2}\int_0^{2\pi} d\theta \int_0^x e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} r dr \Rightarrow$$

$$\Rightarrow F_{(x)} = \frac{1}{\sigma^2}\int_0^x e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} r dr = \int_0^x \frac{r}{\sigma^2} e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} dr \rightarrow F_{(x)} = \int_0^x f_{(r)}\, dr \qquad 4.15$$

Portanto a equação 4.16 é a função densidade probabilidade de *X*.

$$f_{(x)} = \frac{x}{\sigma^2} e^{\left\{\frac{-x^2}{2\sigma^2}\right\}}, x \geq 0 \qquad 4.16$$

A função de distribuição de *X* fica, portanto:

$$F_{(x)} = \int_0^x f_{(r)}\, dr = \int_0^x \frac{r}{\sigma^2} e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} dr \rightarrow \{q = r^2, dq = 2rdr\} \Rightarrow$$

$$\Rightarrow F_{(x)} = \int_0^{x^2} \frac{1}{2\sigma^2} e^{\left\{\frac{-q}{2\sigma^2}\right\}} dq \rightarrow F_{(x)} = \int_0^{x^2} \frac{1}{2\sigma^2} e^{\left\{\frac{-q}{2\sigma^2}\right\}} dq \Rightarrow \qquad 4.17$$

$$\Rightarrow F_{(x)} = -\, e^{\left\{\frac{-q}{2\sigma^2}\right\}}\Big|_0^{x^2} \rightarrow F_{(x)} = 1 - e^{\left\{\frac{-x^2}{2\sigma^2}\right\}}$$

A Esperança de *X*, que corresponde ao valor médio é:

$$E_{(r)} = \int_0^{\infty} f_{(r)}\, dr \rightarrow E_{(X)} = \sigma\sqrt{\frac{\pi}{2}} \rightarrow E_{(X)} \approx 1{,}253.\sigma \qquad 4.18$$

O valor da variância corresponde a:

$$\sigma_r^2 = E_{(r^2)} - E_{(r)}^2 = \int_0^{\infty} \frac{r^2}{\sigma^2} e^{\left\{\frac{-r^2}{2\sigma^2}\right\}} dr - \frac{\sigma^2\pi}{2} \rightarrow \sigma_r^2 = \sigma^2\left(2 - \frac{\pi}{2}\right) \Rightarrow$$

$$\Rightarrow Var_{(X)} = \sigma^2\left(\frac{4-\pi}{2}\right) \rightarrow Var_{(X)} \approx 0{,}429.\sigma^2 \qquad 4.19$$

Portanto, considerando-se o desvanecimento da equação 4.16 e aplicando-se

$F_{(x)}$ da equação 4.17 na equação de posição 4.11, tem-se:

$$R_V = \left\{R_t - R_{t-1}\,;\; r_{t-1}.\left(1 - P_{(R\leq r)}\right) \leq R \leq \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr\right).\left(1 - P_{(R\leq r)}\right)\right\} \Rightarrow$$

$$\Rightarrow R_V = \left\{R_t - R_{t-1}\,;\; r_{t-1}.\left(1 - F_{(x)}\right) \leq R \leq \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr\right).\left(1 - F_{(x)}\right)\right\}, \qquad 4.20$$

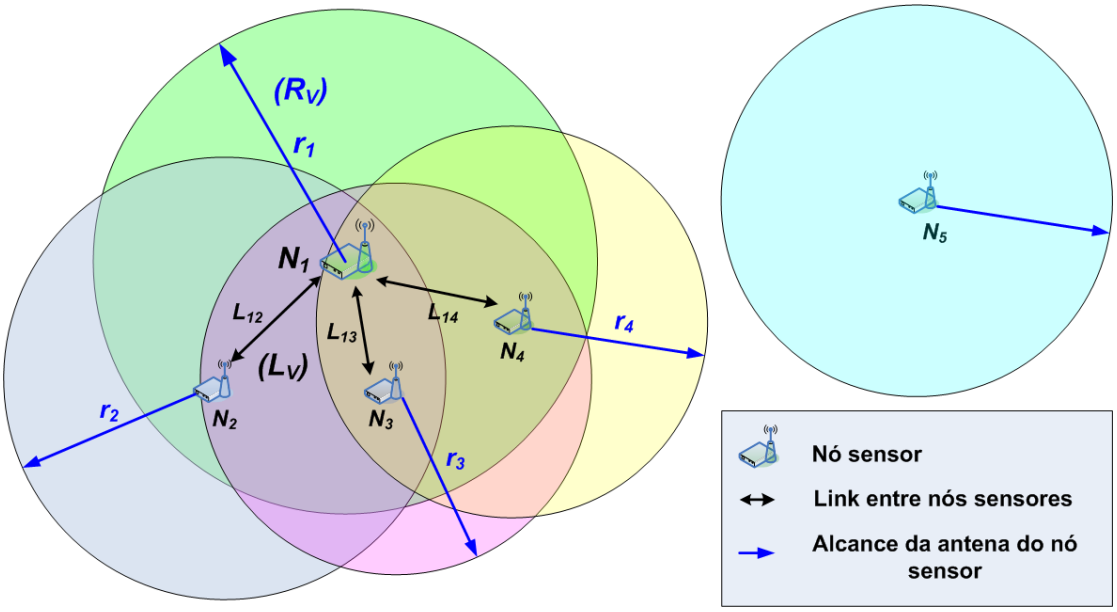
$$1 \leq \sigma \leq 2;\; 0 \leq r \ll \infty$$

Para a modelagem do número de conexões possíveis, visualizou-se o cenário da Figura 36, na qual foi considerada a quantidade de *links* de um nó, no universo de sensores dentro da mesma rede, ou seja, a quantidade total de nós *N* na rede que estejam dentro do alcance do nó de origem N1, conforme a equação 4.21.

$$L_V = \{L_t - L_{t-1} \mid t \geq 1;\; 0 \leq L \leq (N-1)\} \qquad 4.21$$

A Figura 36 considera uma situação de conexão do nó fonte N1 com outros 3 nós vizinhos que estão na região de alcance (em verde), enquanto que o nó N5 apesar de fazer parte da mesma rede, não tem conexão direta com o N1, devido ao alcance da antena de ambos.

Figura 36. Característica de possibilidade de conexões do nó na WSN.



Fonte: Autor.

Como as possibilidades de conexão de um nó com os nós vizinhos dentro de

uma rede estão diretamente ligados ao raio de cobertura, considera-se a equação 4.22 para a adequação do nó, na qual o número de nós vizinhos N_{VIZ} deve estar dentro do alcance do nó fonte $R_{V(S)}$.

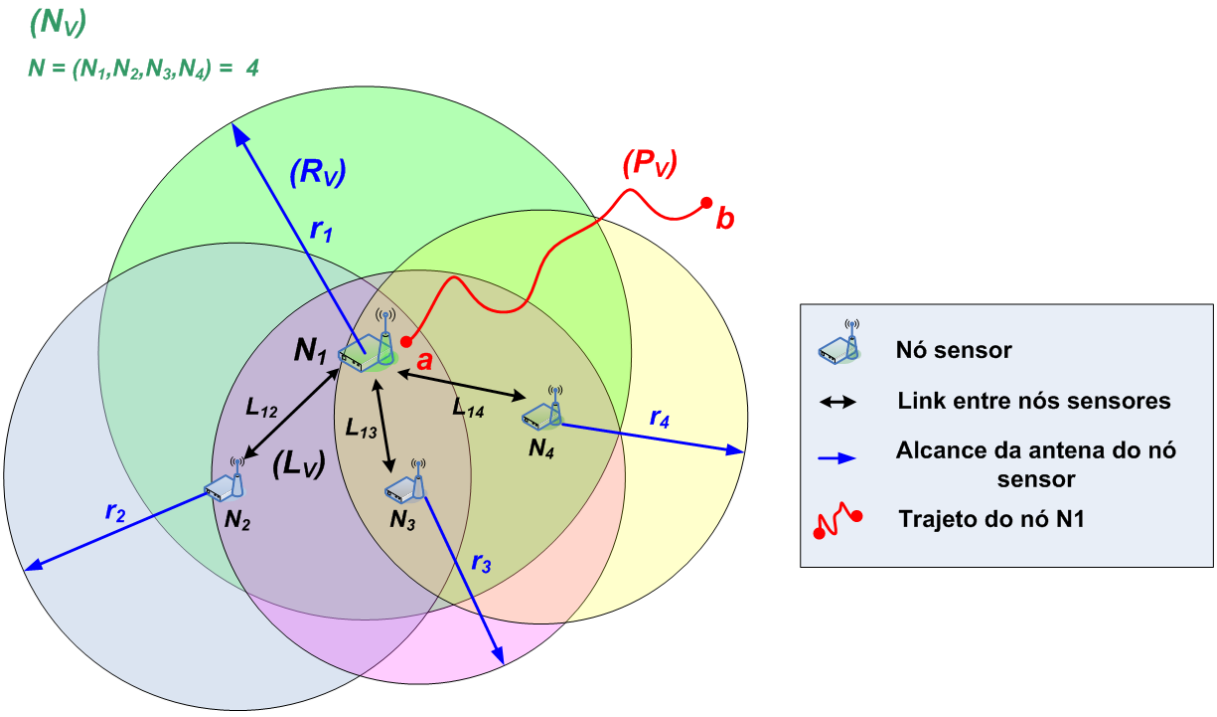
$$L_V = \{L_t - L_{t-1} \mid t \geq 1;\ 0 \leq L \leq (N-1); N_{VIZ} \subset R_{V(S)}\} \qquad 4.22$$

Finalizando as variáveis de características dos nós e que estão presentes na equação 4.1, tem-se a equação 4.23, que considera a situação na rede em que pode-se existir uma quantidade variável de nós presentes no ambiente ou *cluster* específico, por entrada de novos nós ou perdas de nós que já fazem parte da rede.

$$N_V = \{N_t - N_{t-1} \mid t \geq 1\} \qquad 4.23$$

Portanto, diante do exposto pelas fórmulas anteriores, tem-se a Figura 37 com o exemplo de cenário geral para 4 nós, onde P_V representa a trajetória de um nó partindo de um ponto *a* até *b*, R_V ao alcance da antena do nó N1 e *r* ao raio de alcance, L_V ao número de conexões de N1 e N_V ao número de nós da rede.

Figura 37. Características gerais da WSN.



Fonte: Autor.

A relação de proporção (peso) entre os fatores multiplicativos adotados na equação 4.1 para o algoritmo está presente na equação 4.24.

$$\sum_{i=1}^{N}(\alpha+\beta+\gamma+\sigma)_i = 1 \Longrightarrow$$

$$\Longrightarrow \begin{cases} \alpha = \frac{1}{4}.\left|\frac{P_t - P_{t-1}}{\text{argmax}(P_t - P_{t-1})}\right|, \forall\alpha \in \mathbb{R} \mid 0 \leq \alpha \leq 25\% \\ \beta = \frac{1}{4}.\left|\frac{R_t - R_{t-1}}{\text{argmax}(R_t - R_{t-1})}\right|, \forall\beta \in \mathbb{R} \mid 0 \leq \beta \leq 25\% \\ \gamma = \frac{1}{4}.\left|\frac{L_t - L_{t-1}}{\text{argmax}(L_t - L_{t-1})}\right|, \forall\gamma \in \mathbb{R} \mid 0 \leq \gamma \leq 25\% \\ \sigma = \frac{1}{4}.\left|\frac{N_t - N_{t-1}}{\text{argmax}(N_t - N_{t-1})}\right|, \forall\sigma \in \mathbb{R} \mid 0 \leq \sigma \leq 25\% \end{cases} \qquad 4.24$$

A equação 4.24 refere-se à proporção relacionada aos fatores de variabilidade do sistema, que individualmente representam: α a proporção de posição, β a proporção de alcance do nó, γ a proporção de conexões possíveis do nó e σ a proporção da quantidade de nós na rede.

A relação com as características de variabilidade representa 1/4 da característica analisada pela máxima variação da característica, ou seja, supondo-se que para uma dada variação de posição máxima, o cálculo da proporção de posição resultará em $\alpha = 0{,}25$, que neste caso representa 1/4 de F_V.

Em uma rede, a quantidade de nós deve ser definida (ou monitorada, caso já exista), baseando-se na necessidade do meio a ser monitorado ou aplicado. Se o meio a ser monitorado for pequeno e necessitar de poucos nós, a quantidade de nós *N* passa a ser um parâmetro conhecido e não mais variável, limitando a característica de variabilidade a 3 variáveis com a quantidade de nós aleatória e com no mínimo 3 nós para a possibilidade de manter a característica de múltiplos saltos na rede, e redistribuindo as proporções das características em 1/3.

No caso de existir a necessidade de adesão aleatória de nós na rede, então se deve trabalhar com a fórmula completa de 4 variáveis da mesma forma que a equação 4.1. Se o resultado das subtrações das equações 4.5, 4.9, 4.20, 4.22 e 4.23 for zero, as variáveis do sistema terão valores binários 0_2, que corresponde a não ter variação nos parâmetros e caso seja qualquer outro valor diferente de zero então é assumido o valor binário 1_2, que indica existência de variação nos parâmetros da rede.

$$F_N = \bar{X}_{\left(F_{V(n)}\right)} = \frac{\sum_{n=1}^{N} F_{V(n)}}{N} \qquad 4.25$$

Com o cálculo da característica de variabilidade F_N da rede, realiza-se o monitoramento dos nós aplicando-se as relações da equação 4.1 para cada nó da rede, conforme a média aritmética simples de F_V representada pela equação 4.25, para uso posterior pelo algoritmo.

Figura 38. Pseudo-código da variabilidade do sistema.

Algorithm for sensor variability detection

N = Number of nodes into network;
$n = 0$;
$t = 0$;
$WHILE\ (n \leq N)$
$WHILE\ (t > 0)$
$LET\ P_{V(n)} = \phi_0,\quad \alpha_{(n)} = \theta_0$
$R_{V(n)} = \phi_1,\quad \beta_{(n)} = \theta_1$
$L_{V(n)} = \phi_2,\quad \gamma_{(n)} = \theta_2$
$N_{V(n)} = \phi_3,\quad \sigma_{(n)} = \theta_3$
{ $P_{V(n)}$, $R_{V(n)}$, $L_{V(n)}$ and $N_{V(n)}$ are readings of network by node measurements. }
$FOR\quad i = 0\ to\ 3$
$DO\quad \theta_i = \frac{1}{4}.\left|\frac{\phi_{i(t)} - \phi_{i(t-1)}}{\arg\max(\phi_{i(t)}, \phi_{i(t-1)})}\right|$
$F_{Vi(n)} = \prod_{i=0}^{3} \theta_i \phi_{i(t)}$
$F_{V(n)} = \sum_{i=0}^{3} F_{Vi(n)}$
{ $F_{V(n)}$ represents the percentage relative to the variability in nth node. }
$END\ FOR$
$t = t + 1$
$n = n + 1$
$END\ WHILE$
$END\ WHILE$

Fonte: Autor.

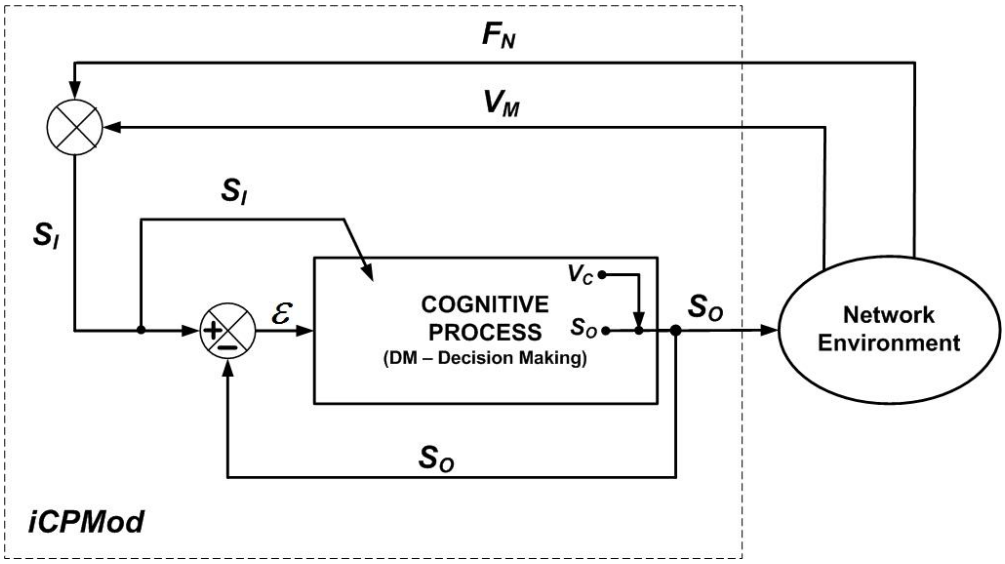
F_N é um dos parâmetros para o *Decision Making process* (DMp), por parte do processo cognitivo, que será descrito adiante nas características do sistema, e $F_{V(n)}$ é a característica de variabilidade do n-ésimo nó. Na Figura 38 é apresentado o pseudo-código referente ao uso das equações expostas na construção dos *rounds* de análises do sistema. O algoritmo apresentado no pseudo-código representa a sequência de implementação das equações apresentadas 4.1 a 4.25, para uma formatação que auxilia a implementação em *software*.

O algoritmo faz a relação inicial com a quantidade N de nós na rede, para então fazer os *rounds* com as 4 variáveis ϕ_i e os pesos dos coeficientes θ_i vinculados às mesmas (equações 4.1 e 4.24) em cada nó, determinando os valores de variabilidade do nó com relação à posição, alcance, quantidade de conexões e de nós que compõem a rede.

4.1.1. Configuração inicial do processo cognitivo

O processo cognitivo inicialmente trabalha com vetor de memória V_M, característica de variabilidade F_N e sinal de entrada S_I, para gerar o sinal de saída composto por S_O e V_C de controle do processo cognitivo. No processo cognitivo o resultado do comparador, que é o erro quadrático médio $\mathcal{E}$ de estimativa entre o sinal recebido S_I e o sinal gerado S_O, é utilizado para tentar adequar o sistema (ou meio) à melhor condição possível dentro da política da rede.

Figura 39. Diagrama em blocos de interconexão do processo cognitivo.



Fonte: Autor.

A Figura 39 mostra as interfaces e conexões que ocorrem na parte interna do processo cognitivo do *initial Cognitive Processor Module* (iCPMod). O sinal é derivado da combinação de F_N e V_M, enquanto que o V_C atua somente no S_O. F_N e V_M são indicadores da situação do ambiente do sistema. Com a análise destes dois parâmetros, o sinal S_I pode ser composto para o cálculo de erro $\mathcal{E}$.

4.1.2. *Memory vector*

$$V_M = (V_S, V_F, V_R) \qquad 4.26$$

Na equação 4.26, V_M corresponde ao vetor de memória, sendo composto por: V_S é o vetor de sucesso de conexões entre os nós, V_F vetor de falha ocorrida nos nós e V_R para representar o vetor de rotas válidas. A composição de V_S, V_F e V_R está descrita na Tabela 5.

Tabela 5 – Componentes do *Memory vector*.

Vector Type	***Components and Description***		
	Description	***Components***	***Decision***
V_S	*Success Vector*	Conexão fim-a-fim bem sucedida, Atraso fim-a-fim, Taxa de dados efetivamente entregues.	Pelas rotas com maior probabilidade de sucesso fim-a-fim.
V_F	*Fail Vector*	Nó desconectado por 3 vezes seguidas, Nó que não responde com *reply message* (RREP).	Por evitar passar por estes nós em certo intervalo de tempo.
V_R	*Route Vector*	Rotas válidas, Nó de origem, Nó de destino, Quantidade de saltos até o destino.	Pela melhor rota, conforme histórico de rotas válidas.

Fonte: Autor.

O V_M permite ao processo cognitivo iCPMod armazenar informações da rede e dos nós, estabelecendo histórico das situações de rede e memória. Este vetor é o resultado da coleta de dados instantâneos sobre o meio monitorado em intervalos que podem ser adotados.

Com o V_M é possível atuar no meio utilizando o conceito de DMp para gerar os vetores *Decision Making* (DM).

4.1.3. *Error vector*

Basicamente o vetor de erro $\mathcal{E}$ representa o erro médio quadrático relativo às medidas dos sinais de entrada S_I e saída S_O do processo cognitivo, sendo, portanto, a média ponderada de probabilidade sobre todas as amostras coletadas para N nós sensores. Utiliza-se este parâmetro para verificação da entrada do iCPMod a fim de realizar os devidos ajustes no sinal com uso do V_C.

$$\begin{cases} S_I = (F_N, V_M) \rightarrow \zeta S_I = \text{Valor estimado de } S_I \\ S_O = (V_C, V_M) \rightarrow \xi S_O = \text{Valor real de } S_O \end{cases} \Rightarrow$$

$$\Rightarrow \mathcal{E}_{(\%)} = \sum_{i=1}^{N} \frac{(\zeta S_I - \xi S_O)_i^2}{S_O} . 100 \mid 0 \leq \mathcal{E} \leq 100\% \qquad 4.27$$

4.1.4. *Decision Making vector*

$$DM = (F_N, V_M, \mathcal{E}) \qquad 4.28$$

O mecanismo *Decision Making process* (DMp) é realizado pelo processo cognitivo e leva em consideração a característica de variabilidade do meio (F_N), o vetor de memória (V_M) e o erro ($\mathcal{E}$).

Tabela 6 – Tomada de decisão conforme o DL.

DM *Level*	**DL *Configuration***	
	***Decision Limit* (%)**	***Action about* DM (*with* DL)**
Urgente	80,0 → 100	Reinicia a tabela de roteamento nos nós. Processo examina o índice (F_N, V_M, ε) que gerou a situação de urgência. V_C é enviado aos nós.
Crítico	70,0 → 79,9	O processo examina a memória (V_M) e o erro (ε). Ajusta o sinal S_O e refaz leitura do DL. Caso se mantenha a situação, um vetor V_C de controle é enviado ao nó.
Atenção	50,0 → 69,9	Ajustar S_O com a correção do sinal, conforme erro $\mathcal{E}$ detectado.
Regular	30,0 → 49,9	Sem necessidade de mudança, porém, decisões podem ser tomadas para beneficiar o QoS. O sistema está operando normalmente.
Controlado	0 → 29,9	Rede sem necessidade de mudança. O sistema está operando no melhor caso.

Fonte: Autor.

No caso Urgente, V_C é enviado aos nós para, conforme os índices F_N, V_M e $\mathcal{E}$ em prol da otimização da rede e ajuste do protocolo de roteamento utilizado: alterar a potência de transmissão do nó (dentro do limite do nó), adoção de outro protocolo de roteamento, alterar o tamanho do buffer disponível (impactando na fila e processamento) e alterar o tamanho do pacote de dados (para dar mais vazão de dados na rede e diminuir a latência).

O DMp é configurado conforme a QoS esperada. A ação do DMp está relacionada diretamente com o *Decision Limit* (DL). O DL corresponde a um valor probabilístico de ajuste do sistema da combinação de F_N, V_M e $\mathcal{E}$, conforme consta na Tabela 6, que refere-se ao nível de medição sobre a situação atual da rede.

Foram adotados 5 níveis de abstração, baseando-se em *Quality of Experience* (QoE), pois com esta quantidade de níveis é possível representar uma quantidade expressiva de situações sobre as condições da rede e atuar com o V_C para ajustar os dados nas camadas envolvidas.

O cálculo de DL trabalha com o inverso da média da composição de DM, conforme equação 4.20 e a Figura 40 mostra um exemplo da relação gráfica de DL.

$$\bar{X}_{(F_N,V_M,\mathcal{E})} = \frac{\sum_{i=1}^{N}\left(F_{N_i}, V_{M_i}, \mathcal{E}_i\right)}{N} \rightarrow \begin{cases} \bar{X}_{(F_N,V_M,\mathcal{E})} \geq 1, DL_{(\%)} = \left(\frac{1}{\bar{X}_{(F_N,V_M,\mathcal{E})}}\right).100 \\ \bar{X}_{(F_N,V_M,\mathcal{E})} < 1, DL_{(\%)} = 100\ \% \end{cases} \qquad 4.29$$

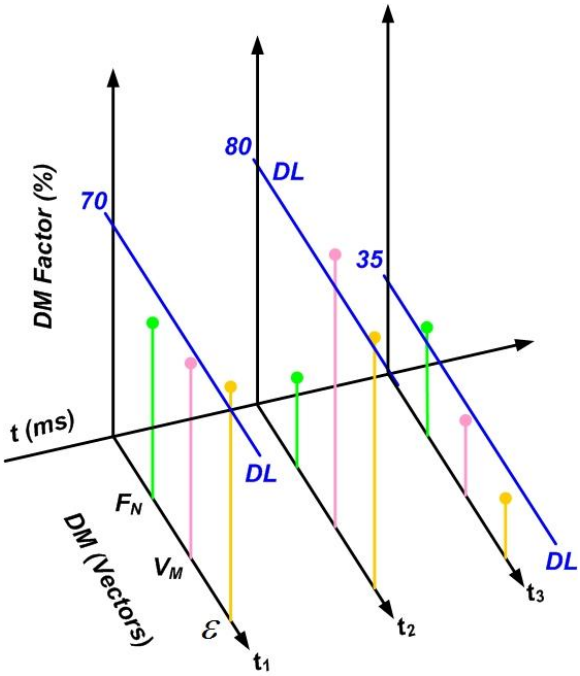
Na equação 4.29, quanto menor for o valor de DL em porcentagem, melhor será a condição da rede e, por outro lado, quanto maior o DL então pior a situação da rede e ações mais drásticas devem ser tomadas pelo iCPMod conforme os níveis presentes na Tabela 6.

Portanto, para que a rede tenha nível DM controlado ou regular, é necessário que $\bar{X}$ seja um valor alto. Quanto mais baixo o valor de $\bar{X}$ dos vetores de variabilidade, memória e erro, mais crítica será a condição da rede. Na Figura 40 existe um exemplo de leitura de dados de DM, composto pelos vetores F_N, V_M e $\mathcal{E}$, em determinados intervalos t_1, t_2 e t_3.

Neste passo é que se pode utilizar o processo cognitivo com aplicação da

Lógica *Fuzzy*, podendo ser implementado com as funções de pertinência de fuzzificação e defuzzificação, baseando-se nos níveis do DMp da Tabela 6, que variam de acordo com a situação de leitura de informações atuais da rede.

Figura 40. Relação do limiar de decisão (DL) com o DMp.



Fonte: Autor.

A Tabela 6 servirá de base para a adoção de regras *Fuzzy* e criação de funções de pertinência no desenvolvimento em software do CPMod, que é uma evolução do iCPMod.

4.1.5. *Control vector*

O módulo de processamento cognitivo com o DM envia em S_O um vetor de dados de controle para definir alterações aos nós da rede conforme necessidade de mudança através do uso de um vetor de controle V_C. Este vetor é definido na equação 4.21 e pode ser alterado para acréscimo de outras atuações conforme a necessidade da rede e configuração.

$$V_C = (\text{Energy, Route,Queue,Memory}) \qquad 4.30$$

Com o V_C sobre o sinal de saída S_O é possível ajustar parâmetros nos nós da rede. Pelo vetor de controle final os nós da rede recebem e fazem alterações sobre dimensionamento de quantidade de memória para a execução de tarefas, de fila de

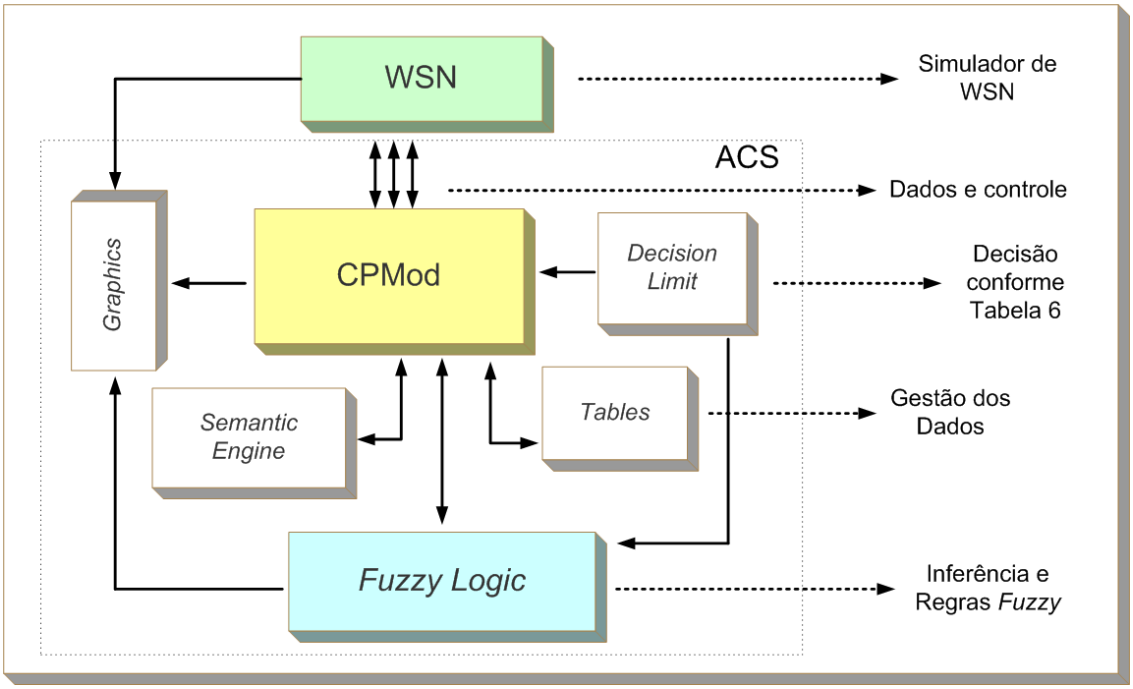
processamento interno do nó, alteração no protocolo de roteamento e economia de energia.

O processamento cognitivo permite ao sistema se reajustar e ao mesmo tempo executar a tarefa de monitoramento da rede. Este vetor de controle é interpretado no ACS como a saída do processo de defuzzificação, o qual direcionará as medidas a serem adotadas na rede.

4.2. Modelagem do ACS

Conforme proposto no objetivo do trabalho foi desenvolvida uma arquitetura de sistema cognitivo adaptativo, com a finalidade de executar o controle da WSN em múltiplas camadas. Para a modelagem do *Adaptive Cognitive System* (ACS), foi definida a relação em blocos da Figura 41, na qual existe interação do CPMod com os blocos WSN e *Fuzzy*.

Figura 41. Diagrama em blocos da arquitetura do ACS.



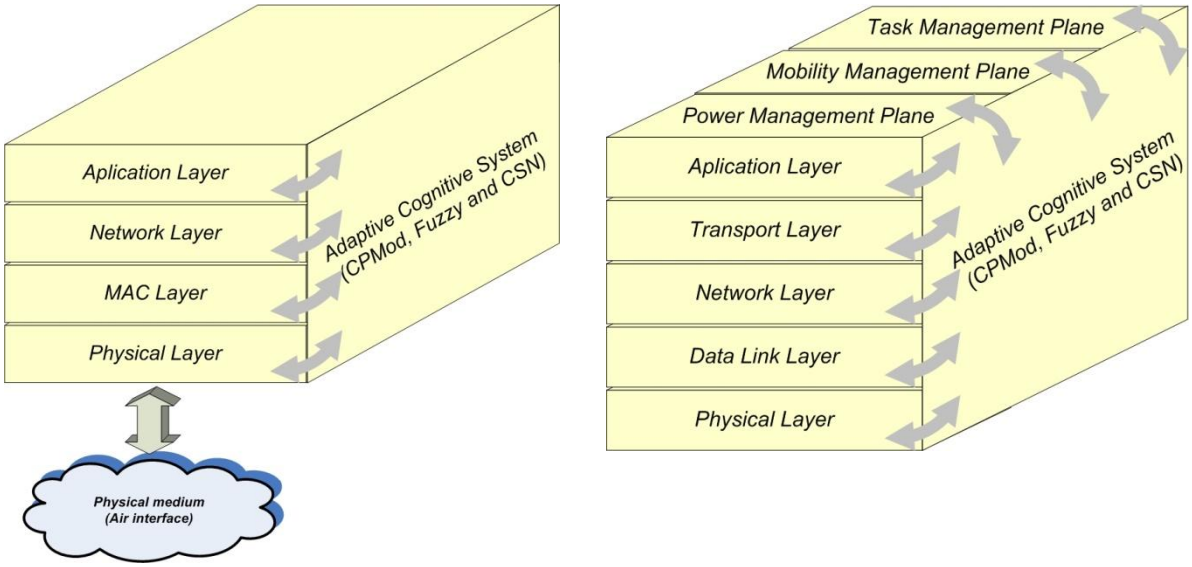
Fonte: Autor.

A arquitetura de CN (THOMAS, 2006) é considerada no ACS, na interação lógica entre os blocos. No CN, a camada SAN faz a conexão com o meio de rede, ou seja, no ACS a WSN realiza interface com o CPMod. A camada CP do CN, que contempla elementos cognitivos a fim de alcançar os objetivos traçados, é

representada pelo bloco CPMod. A camada CSL, que apresenta os objetivos fim-a-fim e conhecimento da rede, é representada pelo ACS juntamente com o bloco de Lógica *Fuzzy*.

Além da representação do ACS como sistema autônomo (juntamente com as conexões apresentadas), pensando-se na arquitetura específica de WSN, a Figura 42 mostra como o ACS está relacionado com as diferentes camadas de rede, abrangendo Lógica *Fuzzy* e CPMod com execução de ajustes nas diferentes camadas.

Figura 42. Arquitetura do ACS. (a) modelo com 4 camadas; (b) modelo genérico tridimensional com 5 camadas.



(a) (b)

Fonte: Autor.

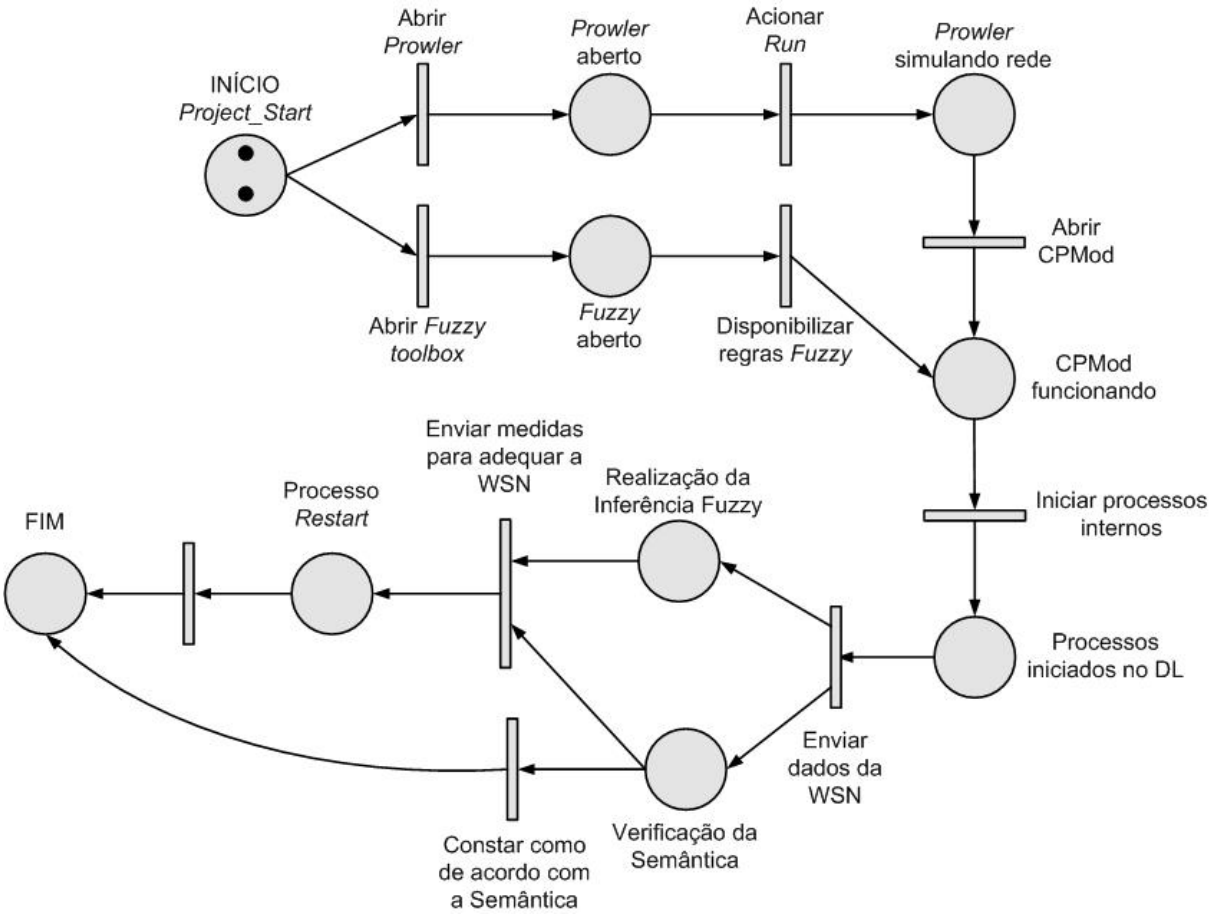
Em ambos os modelos, pode-se observar a característica *multi-layer* do ACS, bem como a abstração da arquitetura de rede em 4 e 5 camadas. Neste trabalho, não foi implementada a análise com alteração de aspectos da camada de transporte, como por exemplo, alteração entre *Transmission Control Protocol* (TCP) e *User Datagram Protocol* (UDP), sendo, portanto, mais coerente a adoção do modelo com 4 camadas presente na Figura 40a.

Como o nó sensor escolhido para a simulação é da família MICA *motes* (MICAz), então o modelo da Figura 42a é mais coerente, pois atende especificamente ao padrão IEEE 802.15.4. O modelo de 5 camadas da Figura 42b é válido também para o ACS, porém, com alterações necessárias no CPMod para a

realização de análise de *rounds* envolvendo a camada de transporte.

Para facilitar o entendimento sobre o funcionamento do ACS, foi elaborada a Rede de Petri da Figura 43, que contempla a visão geral das interações que ocorrem no sistema. Na representação, os círculos são estados, as barras são transições e as setas de conexão são os arcos do sistema. A cada interação na rede, existe a passagem de um estado para outro através dos arcos e transições.

Figura 43. Rede de Petri do funcionamento geral do sistema.



Fonte: Autor.

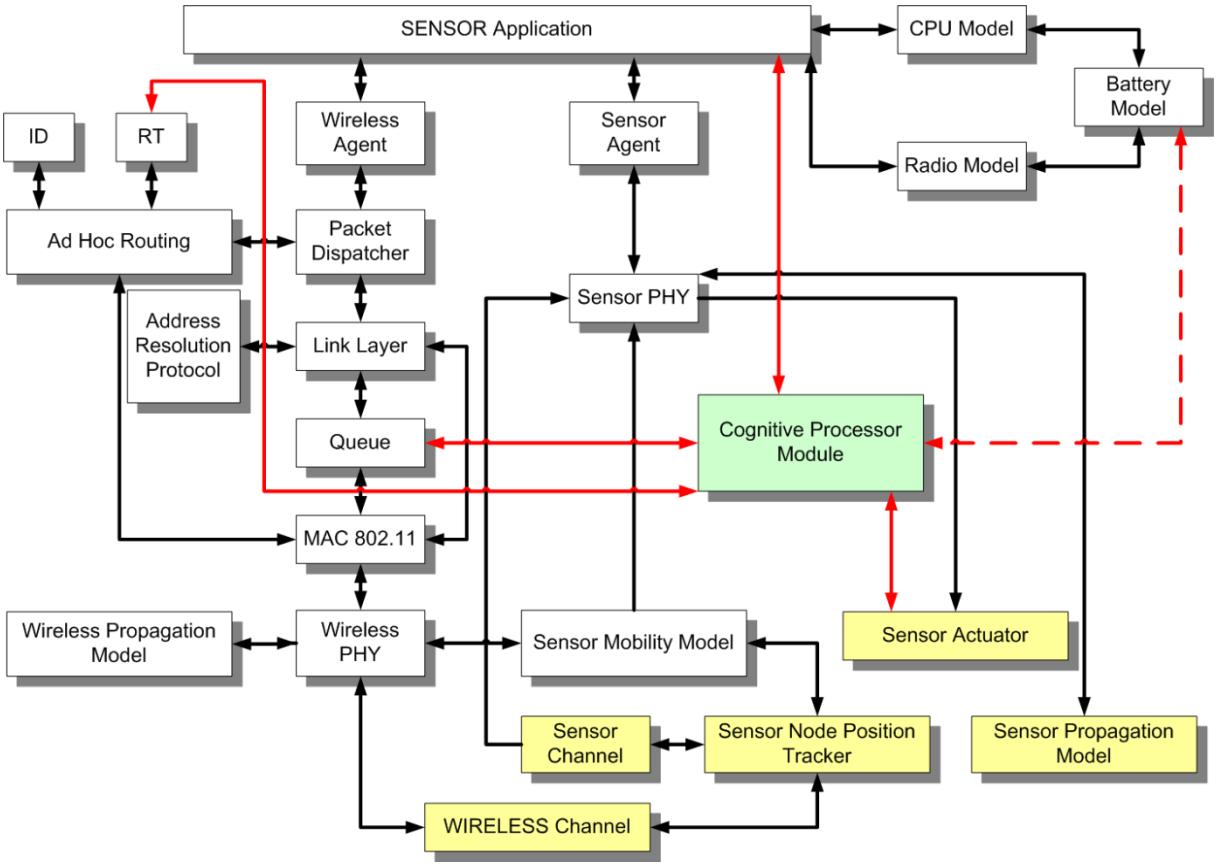
Na Figura 43, tem-se duas marcações na posição inicial que vão circulando pelos estados (círculos) conforme ocorrem as transições (barras). Esta Rede de Petri contempla o início dos processos de análise da rede abordando o funcionamento do ACS. Ao iniciar o processo do ACS, através dos arcos que fazem conexão entre estados e transições, é realizada a abertura dos processos referentes aos dois blocos *Prowler* e *Fuzzy*, para então habilitar as regras *Fuzzy* e acionar a simulação da WSN.

Com a simulação inicial da WSN realizada, é então acionado o bloco do CPMod, que inicia os processos de análise primeiramente pelo DL, para então passar ao bloco *Fuzzy* para a inferência e enquanto isto, a análise sobre a semântica também é realizada, para adequar a rede caso existam alterações determinadas pelos processos. No caso da leitura dos dados da rede estar de acordo com a semântica, que por sua vez, agrega melhoria de desempenho e, portanto tende a suprir a política de rede, o procedimento é finalizado.

4.3. CPMod

Como continuidade ao cumprimento da proposta deste trabalho, desenvolveu-se o *Cognitive Processor Module* (CPMod), que é a parte inteligente para aplicação nos sensores, envolvendo aspectos cognitivos para executar o processo de tomada de decisão, atuando conforme a necessidade, sobre aspectos específicos em sensores, que estão relacionados à modelagem de memória, tipo de protocolo de roteamento, fila de processamento e economia de energia.

Figura 44. Estrutura geral do nó com o CPMod.



Fonte: Autor.

A Figura 44 apresenta a estrutura correspondente a um nó sensor comum com capacidade de roteamento, juntamente com a inserção de um módulo cognitivo responsável pela execução de processos cognitivos.

No esquema, os blocos brancos representam a interface interna tradicional do nó, os blocos amarelos são a representação do meio físico sem fio e o bloco verde representa o módulo cognitivo desenvolvido. No CPMod, as interfaces vermelhas contínuas foram implementadas inicialmente para processamento, memória e roteamento, e a tracejada corresponde à parte conectada à energia, o que proporciona a geração dos 4 vetores de análises para os *rounds*.

O sistema atua de acordo com o padrão 802.15.4 para redes móveis do tipo *Ad Hoc* WSN em WLAN, onde ID corresponde à identificação do nó, RT à *Routing Table* (roteamento) e *Address Resolution Protocol* (ARP).

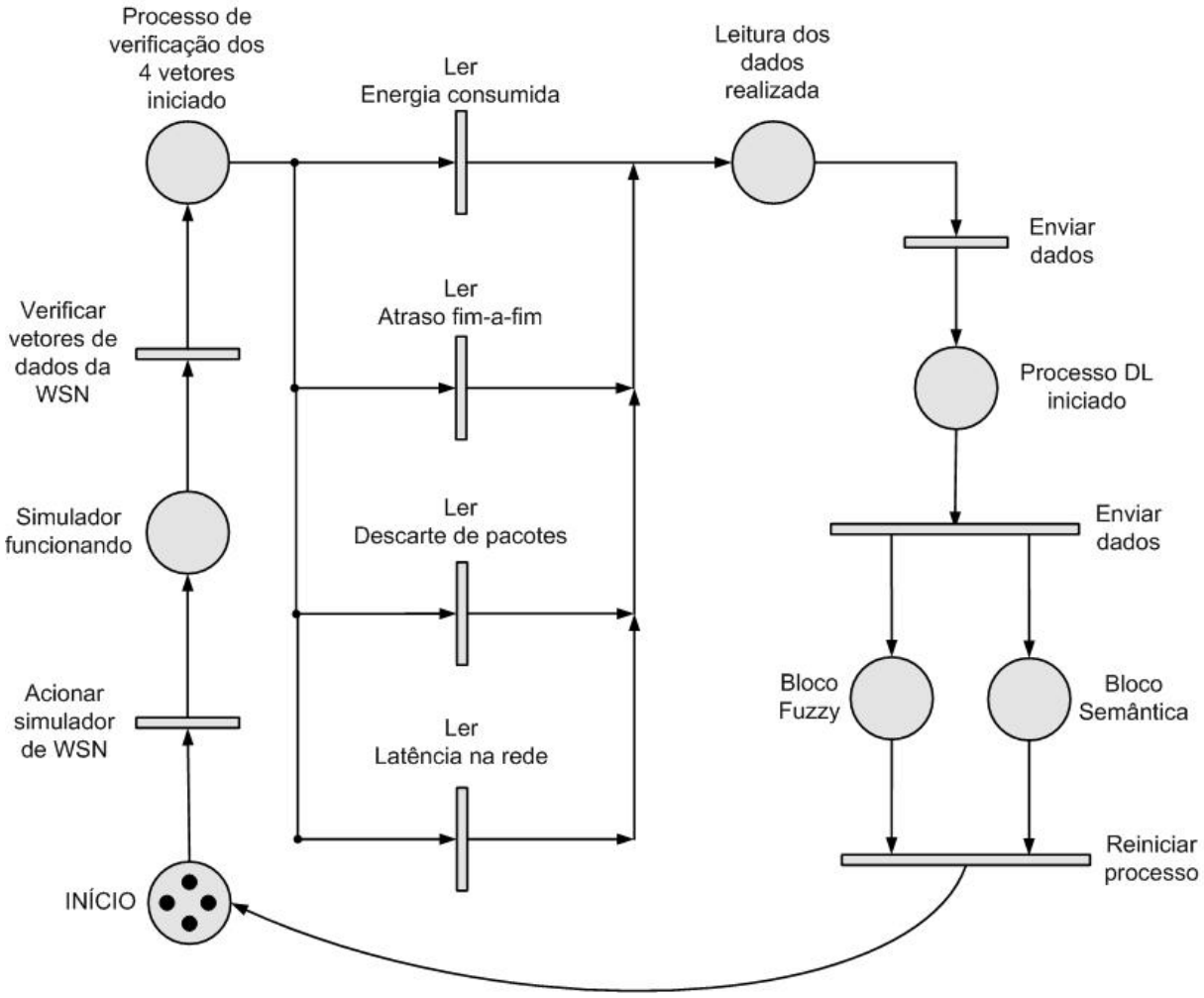
O CPMod tem relação com a CogSeNet, pois faz observação dos dados do meio, decide com base na inferência Fuzzy e realiza ajustes na rede, aguardando *feedback* sobre os novos acontecimentos. O CPMod pode ser executado na camada de aplicação diretamente dentro do nó (como um *software* embarcado) para realizar as operações e análises de *rounds*. Para o estudo desta Tese, visando à visualização dos testes e implementação em *software*, o processo foi implementado através de modelos criados no *Simulink* do MATLAB.

A execução do CPMod ocorre na camada de aplicação, porém não altera o significado e finalidade do processo, ou seja, a realização da tomada de decisão em execução no módulo cognitivo irá afetar parâmetros internos de outras camadas, tais como adaptação nas filas de processamento dos pacotes, alocação de memória e modificações de tipos de protocolos de roteamento adotados para otimizar as rotas de encaminhamento de pacotes, influenciando diretamente no desempenho da rede.

A Figura 45 apresenta a Rede de Petri do funcionamento do CPMod. Neste módulo existem quatro marcações que contemplam a relação da quantidade de dados a serem analisados pelo CPMod. É feita a leitura dos dados provenientes da rede de sensores, que referem-se à energia consumida pelos nós na rede, atraso fim-a-fim, quantidade de pacotes descartados e latência na rede. Com a leitura

destes dados, é inicializado o processo de análise interna pelo DL, que por sua vez, redireciona as marcações conforme o bloco selecionado com base na Tabela 6. Ao término das análises inicia-se o procedimento novamente.

Figura 45. Rede de Petri do CPMod.



Fonte: Autor.

4.4. Desenvolvimento em *Software*

Foi realizada uma modificação na configuração do *Prowler* para incorporar o sistema cognitivo com atuação em diferentes camadas de rede, denominado CPMod. Em relação aos protocolos de roteamento utilizados para a camada de rede *Ad Hoc* desta Tese, o CPMod utiliza uma interface que verifica a alteração (ou opção por manter) de protocolos de roteamento disponíveis nos nós sensores da rede.

Com a redução do tamanho de componentes eletrônicos em geral e de Circuitos Integrados (CI), mantendo ou aumentando o desempenho e capacidade de

armazenamento de informações dos dispositivos móveis, mais sistemas embarcados que envolvem algoritmos, rotinas e protocolos podem estar presentes. Aliado a tudo isto, componentes digitais de processamento de informações mais eficazes também vêm sendo otimizados e melhorados de forma geral.

O sistema proposto e desenvolvido neste trabalho foi implementado de forma a sempre tomar uma decisão dentro dos parâmetros estipulados pela semântica da rede, ou seja, nunca vai ficar travado ou parado sem executar uma solução. A busca do sistema é feita pela melhor solução, ou seja, a que mais se aproximar da semântica estipulada.

A opção de alterar dados na rede sempre será em busca da solução ótima, porém, se for verificado pelo sistema que esta meta não é possível, então o mesmo trará a solução mais adequada conforme ajustes realizados em passos anteriores de análises (*rounds*). A solução pode trazer alterações grandes na rede, caso necessário, por exemplo, com a troca de protocolos de roteamento.

4.4.1. Configuração geral do sistema

Inicialmente, foram definidos e dimensionados 4 vetores de dados (ENERGY, LATENCY, DELAY e DROP), para que fosse elaborado o modelo de análises do CPMod no MATLAB (**.mdl*), abrangendo consulta de dados no *workspace* ou num arquivo do tipo **.mat* (MATSUMOTO, 2002).

O modelo implementado no MATLAB para a simulação do ACS e CPMod comporta-se como um nó central das informações da rede, ou seja, faz o papel de um nó *Sink*, com capacidade de processamento e armazenamento maior de dados, sendo utilizado para tanto, um microcomputador do tipo *desktop* com as seguintes características: processador core i7 2600 3,4 GHz, memória de 6 GB 1333 MHz e HD de 2 TB Sata II.

Como a intenção deste trabalho não é a de apenas utilizar o *Prowler* para simulações de WSN, foi realizada alteração da interface principal (GUI) do *software*, vinculando os 4 vetores de saída mencionados ao modelo do CPMod. Para banco de dados de salvamento e armazenamento de informações acerca da WSN, foi

criada conexão com arquivos do tipo *.xls*, servindo como banco de dados de armazenamento do histórico da rede. Para facilitar a realização de análises em sistemas embarcados, sugere-se trabalhar com arquivos *.*dat* ou *.*txt*.

O CPMod irá realizar análise sobre a melhor condição de acordo com os passos e executar o salvamento dos dados em outro arquivo, conforme os valores definidos no bloco de semântica.

A semântica está relacionada com a parte do CPMod de análise de significado, ou seja, conforme ocorre o andamento dos *rounds* de análises no sistema, o CPMod utiliza o bloco *Fuzzy* para verificação da condição atual da rede, confrontando os dados recolhidos da WSN com os valores presentes na semântica.

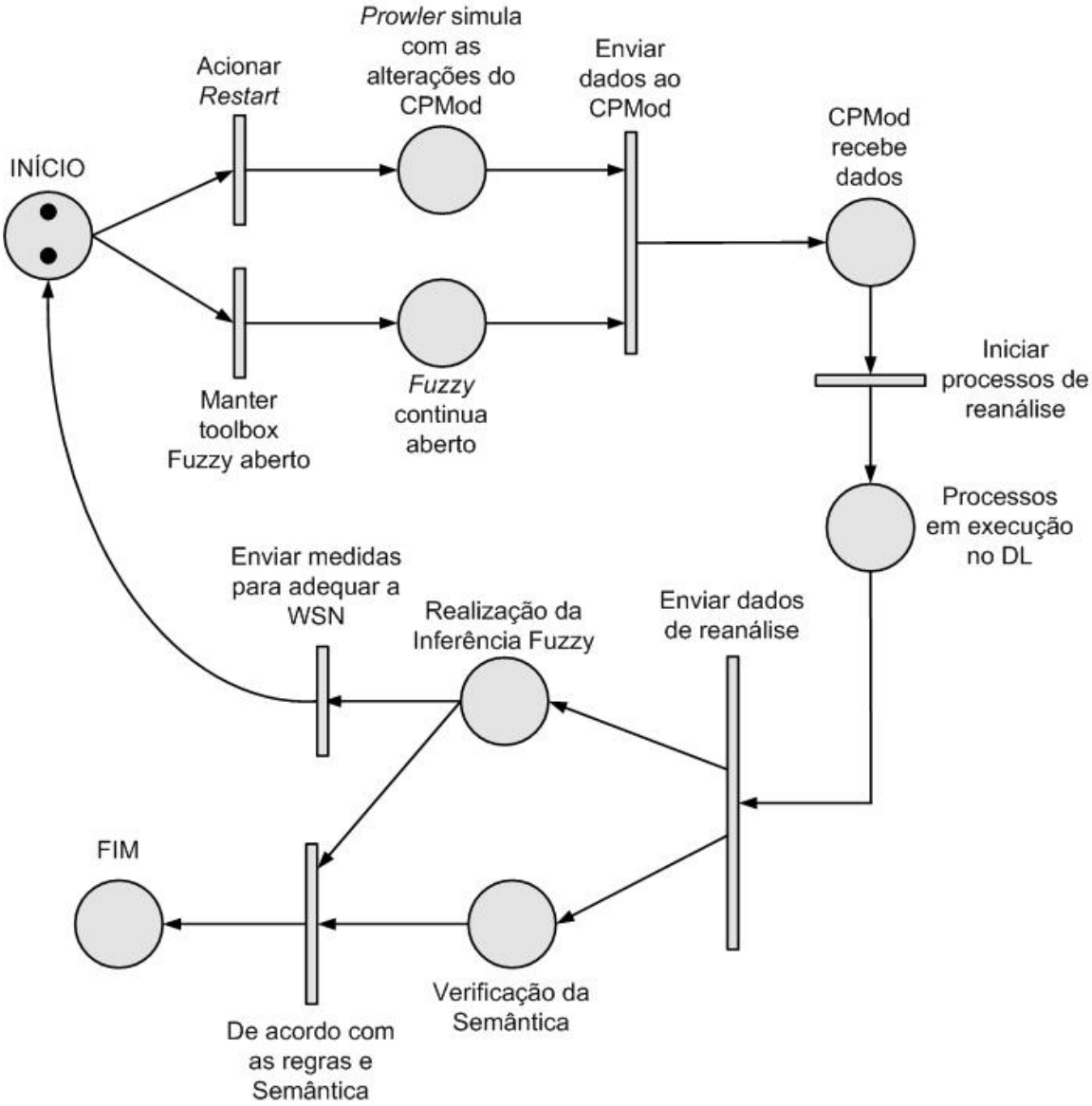
Com o salvamento dos dados sobre as melhores condições para os eventos ocorridos de acordo com a semântica adotada, a memória episódica vai se formando e construindo o histórico da rede.

Logicamente a capacidade de armazenamento de um nó sensor não é infinita, tendo certa restrição. Devido a isto foi escolhido o nó sensor MICAz (ANEXO III), pois contempla capacidade de 128 kB de memória programável e 512 kB de memória *flash*, com possibilidade de conexão com muitos nós e taxa de transmissão de pacotes de até 250 kbps em 2,4 GHz.

O CPMod também executa consulta aos dados de QoS desejada na rede, através da verificação do bloco de política da rede. Caso seja identificado que os dados coletados da WSN não contemplem a semântica e a política da rede na execução de um *round*, um novo *round* deve ser executado com os devidos ajustes apontados pelos vetores de saída da defuzzificação, conforme as funções de pertinência aplicadas, que estão presentes no arquivo *.*fis* e com interface no arquivo *.*mdl* do CPMod.

Com a finalidade de facilitar o entendimento sobre a parte relacionada aos processos de análises, foram elaborados a Rede de Petri da Figura 46 e o fluxograma dos processos que ocorrem na consulta ao bloco de Lógica *Fuzzy* da Figura 47.

Figura 46. Rede de Petri sobre procedimento de adequação da rede.

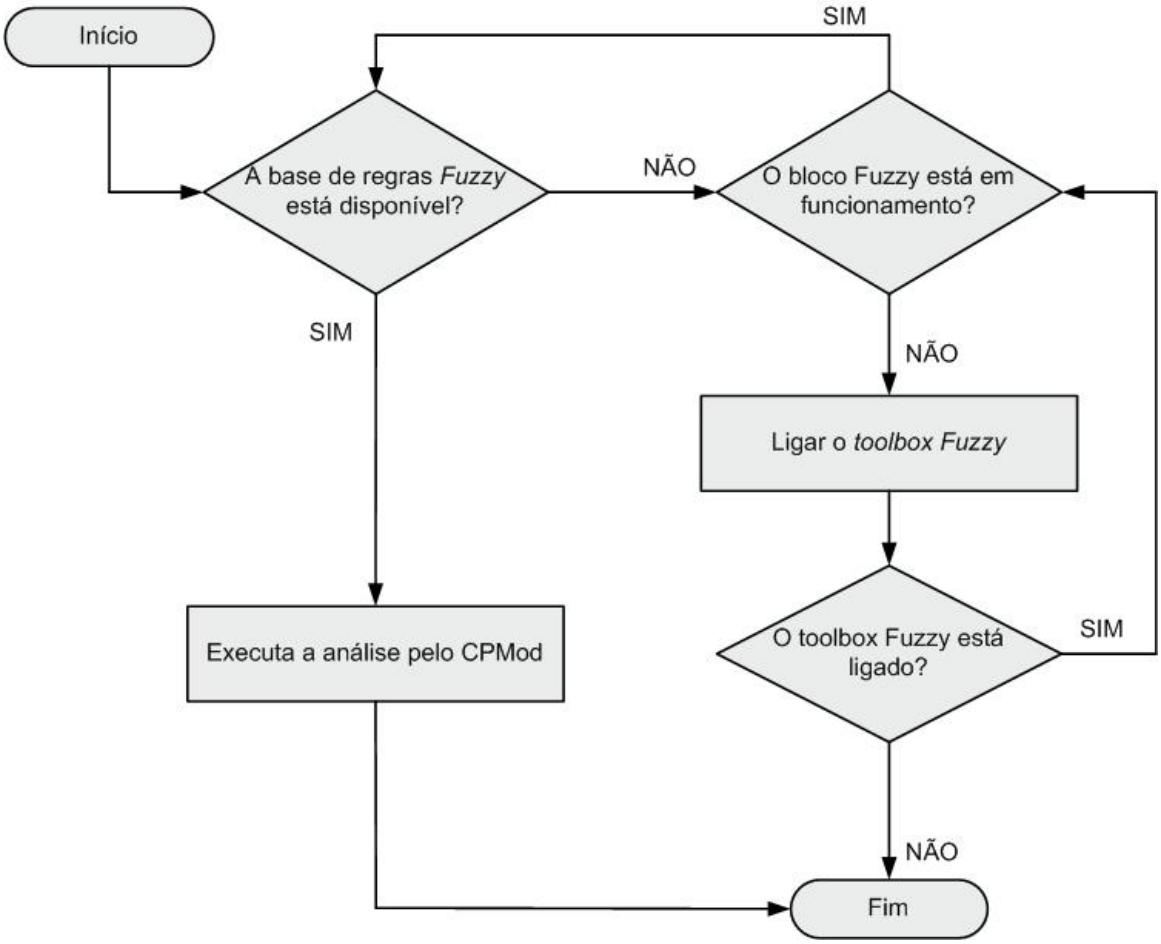


Fonte: Autor.

Na Figura 46 ocorre o processo de adequação da rede pelo CPMod, no qual deve-se manter os blocos *Fuzzy* e *Prowler* em funcionamento. O CPMod recebe os dados atualizados da rede e realiza o processo de reanálise, incluindo o DL, inferência *Fuzzy* e consulta à semântica da rede, que contempla o significado sobre o que se espera como adequado no funcionamento e na política de rede. Como o processo de análise deve ser executado enquanto a rede estiver ativa, significa que caso exista adequação a ser aplicada, reinicia-se o procedimento.

Na Figura 47 é executada a verificação sobre o funcionamento do bloco *Fuzzy*, para então ser realizada a análise pelo CPMod.

Figura 47. Fluxograma sobre a Lógica *Fuzzy*.



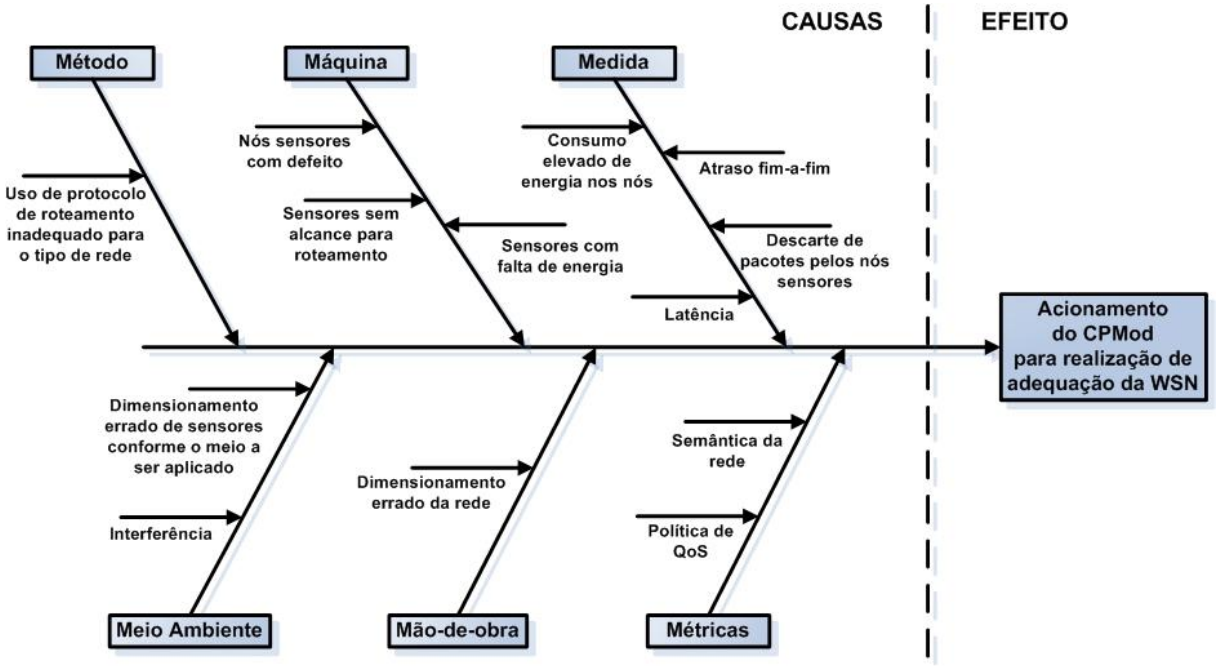
Fonte: Autor.

Para iniciar as análises em *software*, foi desenvolvido o diagrama de causa e efeito de Ishikawa, conhecido como gráfico de espinha de peixe, que representa uma visão geral sobre as possíveis causas que acarretam determinados efeitos, que no caso de redes sem fio podem estar relacionados com problemas de conexão, anomalias na rede, baixo desempenho dos nós, entre outros.

Como esta Tese não se refere ao estudo de todas as possíveis causas e efeitos ocasionados em uma WSN, então o Gráfico 1 apresenta as relações de causa para o efeito de tomada de decisão do CPMod.

No Gráfico 1 são mostradas as possíveis causas que podem ocasionar o acionamento do CPMod em um sistema genérico. Os índices que foram testados diretamente neste trabalho estão nas categorias presentes no diagrama como: medida, métricas e método.

Gráfico 1. Relação de causa e efeito sobre o CPMod.

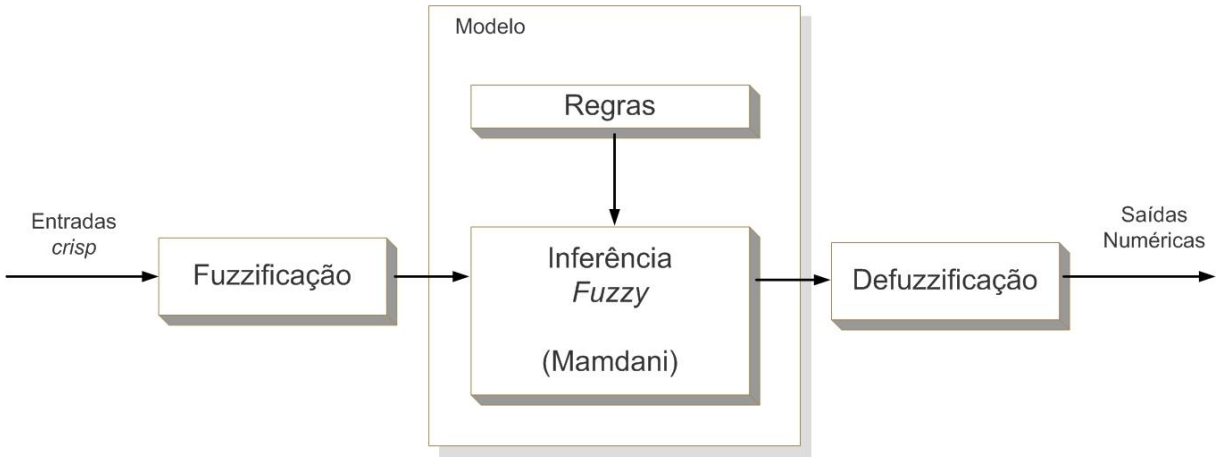


Fonte: Autor.

4.4.2. Implementação da Lógica *Fuzzy*

Para a implementação da Lógica *Fuzzy* como bloco de interligação com o CPMod, utilizou-se o *toolbox Fuzzy* presente no MATLAB, que facilitou apenas a configuração via interface gráfica, ou seja, todas as funções de pertinência e regras *Fuzzy* foram desenvolvidas neste trabalho.

Figura 48. Diagrama em blocos da Lógica *Fuzzy*.



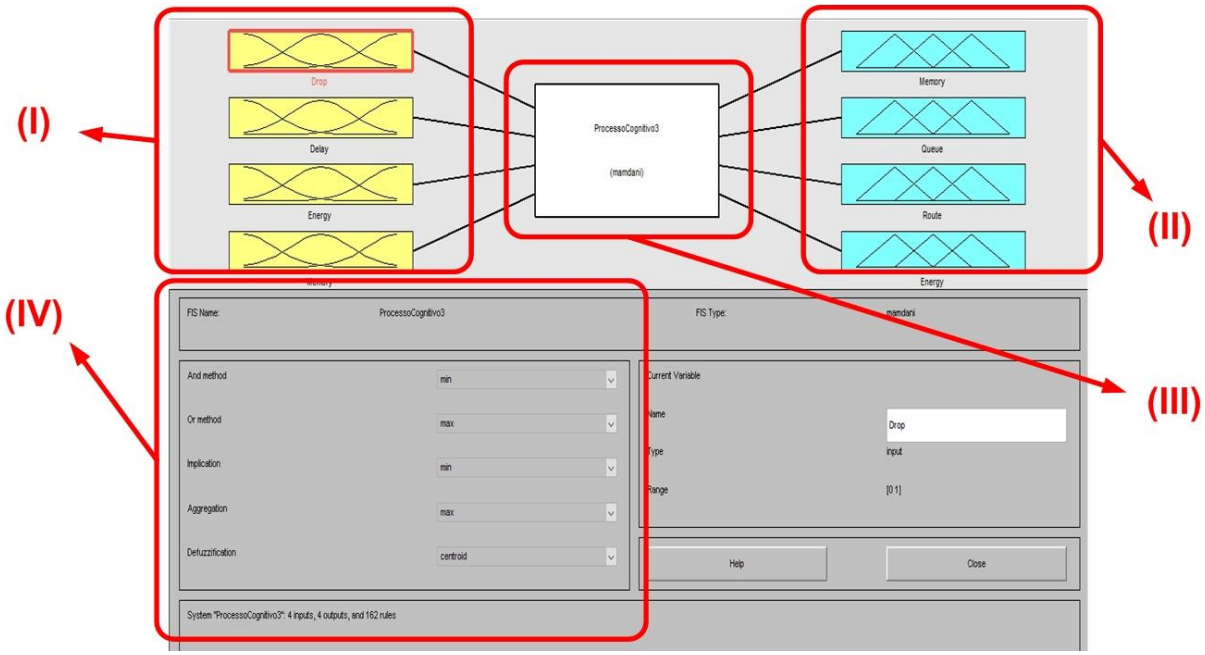
Fonte: Autor.

A arquitetura central do CPMod foi desenvolvida no *Simulink* do MATLAB, incorporando a interligação com o bloco de fuzzificação e defuzzificação, bem como

o uso do modelo Mamdani para a inferência e uso de regras *Fuzzy*. A Figura 48 apresenta o diagrama em blocos da Lógica *Fuzzy* elaborada com o modelo Mamdani.

A tela inicial da Lógica *Fuzzy* é apresentada na Figura 49 e representa o modelo de análise de dados da estrutura inicial, na qual foram criadas 4 funções de fuzzificação para dados coletados da WSN (I) e 4 de defuzzificação correspondentes aos 4 elementos presentes na equação 4.30 do V_C (II). O item (III) representa o uso do modelo Mamdani para a inferência das funções de pertinência de fuzzificação e o (IV) está ligado aos operadores e ao método de defuzzificação adotado, que no caso é baseado no centroide das funções de pertinência de defuzzificação.

Figura 49. Tela inicial de *Fuzzy* no MATLAB.



Fonte: Autor.

Considerando-se o disposto na equação 4.31, são demonstradas as funções de pertinência aplicadas na fuzzificação e defuzzificação, para uso na inferência.

$$A = \{(x, \mu_A(x)) | x \in X\} \; e \; B = \{(x, \mu_B(x)) | x \in X\}, \mu_A(x) = mf \Rightarrow$$

$$\Rightarrow \begin{cases} \mu_{A \cap B} = \text{argmin}(\mu_A(x), \mu_B(x)), \forall x \in X \\ \mu_{A \cup B} = \text{argmax}(\mu_A(x), \mu_B(x)), \forall x \in X \end{cases} \quad 4.31$$

Para as funções de pertinência, consideraram-se as funções triangulares e

trapezoidais, conforme as equações 4.32 e 4.33, respectivamente.

$$\mathrm{Trimf}_{(x;a,b,c)} = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \qquad 4.32$$

As funções de pertinência dos tipos triangulares e trapezoidais podem ser utilizadas em conjunto para formar um *Fuzzy Inference System* (FIS), conforme constam nas Figuras 50 e 52.

$$\mathrm{Trapmf}_{(x;a,b,c,d)} = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \qquad 4.33$$

As equações 4.35 até 4.42 assumem valores com base na equação 4.34, que relaciona a quíntupla *Fuzzy* do grau de pertinência de *x* em *A*. Sendo que *x* representa o nome do vetor de dados, $T_{(x)}$ o conjunto de significados do vetor, $G_{(x)}$ o conjunto de regras sintáticas aplicadas a *x*, $M_{(x)}$ a regra semântica que associa a cada valor gerado por $G_{(x)}$ um conjunto *Fuzzy* em *x* e *X* o universo de discurso que representa o eixo X.

$$f_A(x) = (x, T(x), G(x), M(x), X) \qquad 4.34$$

Os gráficos de formatação de entrada do sistema são apresentados na Figura 50. Nestes gráficos são realizadas as inserções das relações de entrada de fuzzificação.

A equação 4.35 representa a função de pertinência da Figura 50a.

$$f_A(x) = \begin{cases} x\text{: Drop} \\ T(x)\text{: \{Considerable, Verification, Problem\}} \\ X\text{: Quantidade de pacotes} \\ G(x)\text{: \{regras\}} \\ M(x)\text{: Significado\{regras\}} \end{cases} \qquad 4.35$$

A equação 4.36 representa a função de pertinência da Figura 50b, na qual tem-se em $T_{(x)}$ as variáveis linguísticas *very low* (VL), *low* (L), *mean* (M), *high* (H), *very high* (VH) e *extreme* (E).

$$f_A(x) = \begin{cases} x\text{: Delay} \\ T(x)\text{: }\{VL, L, M, H, VH, E\} \\ X\text{: Tempo em milissegundos} \\ G(x)\text{: }\{\text{regras}\} \\ M(x)\text{: Significado}\{\text{regras}\} \end{cases} \quad 4.36$$

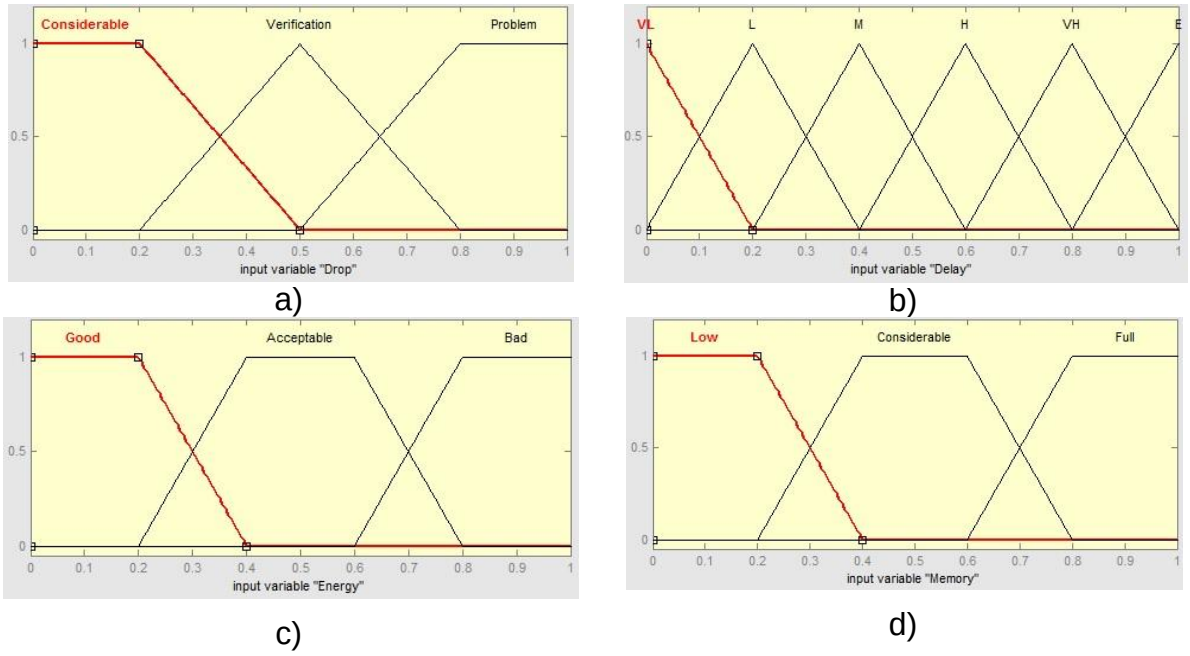
A equação 4.37 representa a função de pertinência da Figura 50c.

$$f_A(x) = \begin{cases} x\text{: Energy} \\ T(x)\text{: }\{\text{Good, Acceptable, Bad}\} \\ X\text{: Quantidade de energia consumida} \\ G(x)\text{: }\{\text{regras}\} \\ M(x)\text{: Significado}\{\text{regras}\} \end{cases} \quad 4.37$$

A equação 4.38 representa a função de pertinência da Figura 50d.

$$f_A(x) = \begin{cases} x\text{: Memory} \\ T(x)\text{: }\{\text{Low, Considerable, Full}\} \\ X\text{: Quantidade de memória utilizada} \\ G(x)\text{: }\{\text{regras}\} \\ M(x)\text{: Significado}\{\text{regras}\} \end{cases} \quad 4.38$$

Figura 50. Tela de gráficos de entrada de fuzzificação no MATLAB.



Fonte: Autor.

Uma parte do núcleo $G(x)$: $\{\text{regras}\}$ com o modelo Mamdani é mostrado a seguir e contempla na totalidade a cobertura de 162 linhas de regras diferentes, conforme QoE (Apêndice IV). É importante deixar claro que estas regras podem ser

modificadas facilmente a qualquer momento.

Figura 51. Tela de regras *Fuzzy* aplicadas ao módulo Mamdani no MATLAB.

Fonte: Autor.

Na Figura 51, (I) representa a relação de funções de pertinência de entrada e (II) as de saída, dentro da inferência baseada no modelo Mamdani e das regras *Fuzzy* estipuladas.

A seguir são mostradas as funções de pertinência dos gráficos de defuzzificação que representam a saída do sistema e que foram utilizadas para a inferência.

A equação 4.39 representa a função de pertinência da Figura 52a.

$$f_A(x) = \begin{cases} x\text{: Memory} \\ T(x)\text{: \{Low, Good, High\}} \\ X\text{: Quantidade de memória disponível} \\ G(x)\text{: \{regras\}} \\ M(x)\text{: Significado\{regras\}} \end{cases} \qquad 4.39$$

A equação 4.40 representa a função de pertinência da Figura 52b.

$$f_A(x) = \begin{cases} x\text{: Queue} \\ T(x)\text{: \{Low, Reasonable, High\}} \\ X\text{: Vazão de pacotes} \\ G(x)\text{: \{regras\}} \\ M(x)\text{: Significado\{regras\}} \end{cases} \qquad 4.40$$

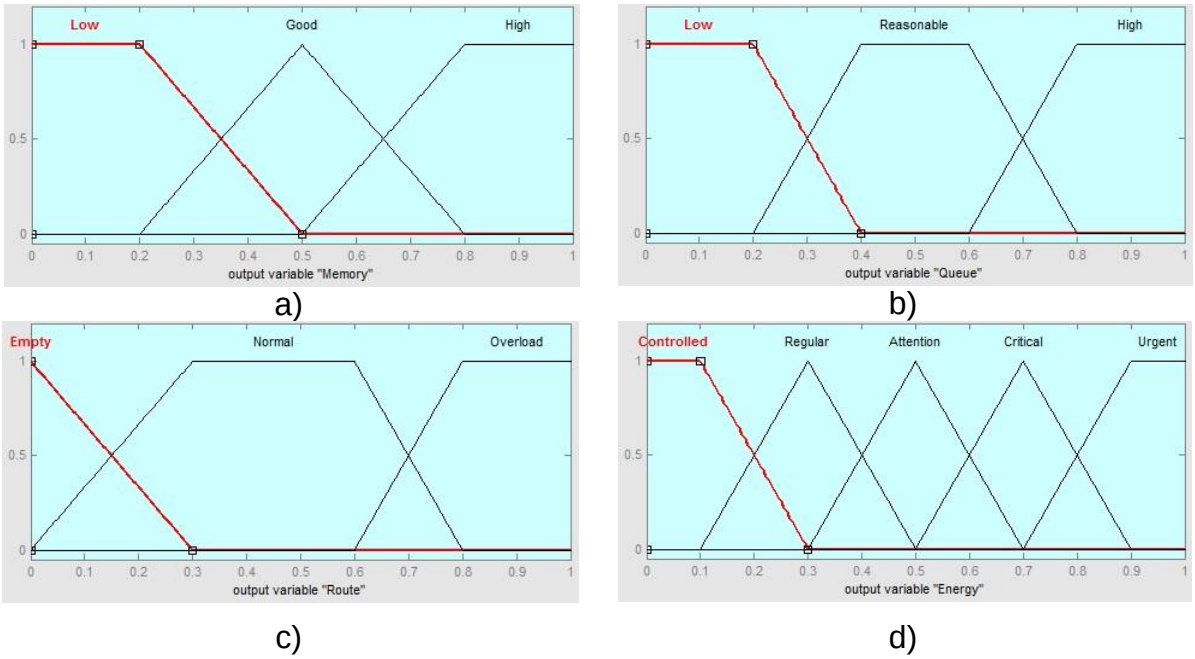
A equação 4.41 representa a função de pertinência da Figura 52c.

$$f_A(x) = \begin{cases} x\text{: Route} \\ T(x)\text{: \{Empty, Normal, Overload\}} \\ X\text{: Atraso na entrega de pacotes} \\ G(x)\text{: \{regras\}} \\ M(x)\text{: Significado\{regras\}} \end{cases} \quad 4.41$$

A equação 4.42 representa a função de pertinência da Figura 52d.

$$f_A(x) = \begin{cases} x\text{: Energy} \\ T(x)\text{: \{Controlled, Regular, Attention, Critical, Urgent\}} \\ X\text{: Quantidade de energia consumida} \\ G(x)\text{: \{regras\}} \\ M(x)\text{: Significado\{regras\}} \end{cases} \quad 4.42$$

Figura 52. Tela de gráficos de saída da defuzzificação no MATLAB.



Fonte: Autor.

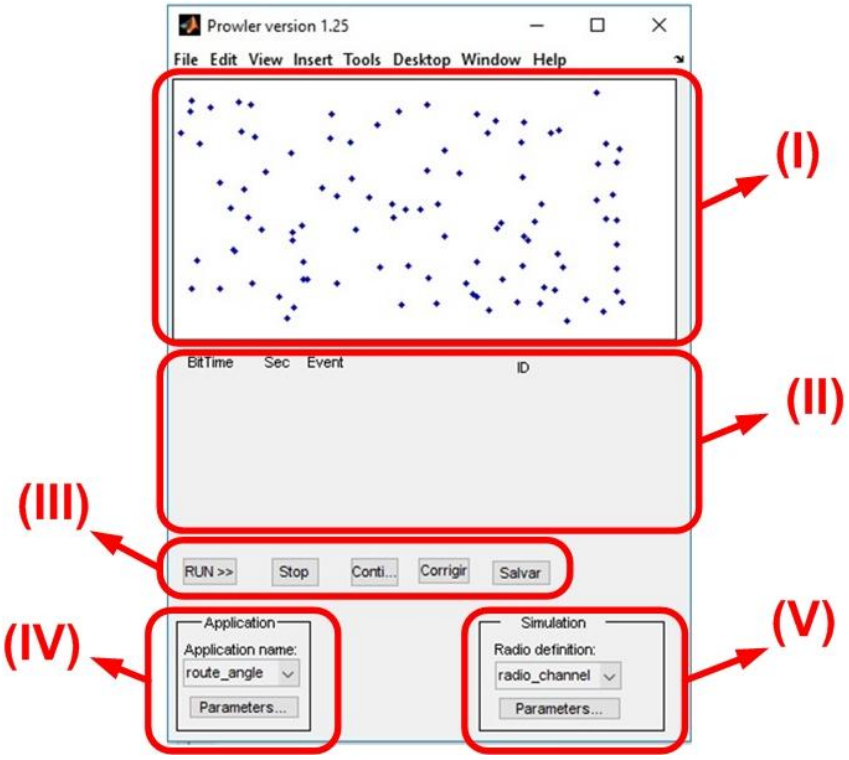
A inferência é executada sobre as funções de pertinência da defuzzificação, que no caso desta Tese está relacionada com o centroide de acordo com as regras aplicadas nas funções.

4.4.3. Configuração do *Prowler*

A Figura 53 mostra a tela inicial alterada do *software Prowler* que foi desenvolvida neste trabalho, na qual (I) representa a topologia do cenário da WSN

com distribuição aleatória dos nós, resultante da função de distribuição que consta na Tabela 7 (Camada Física), (II) refere-se à parte destinada à visualização dos eventos em tempo real, (III) contempla os botões de início da simulação (RUN>>), interrupção (Stop), continuação (Conti...), realização de *rounds* (Corrigir) e armazenamento do ocorrido na rede (Salvar). A parte (IV) refere-se à escolha do tipo de protocolo a ser utilizado inicialmente na rede. A parte (V) é utilizada para a escolha das características do canal de transmissão e recepção de pacotes de dados.

Figura 53. Tela do *Prowler* adaptado.



Fonte: Autor.

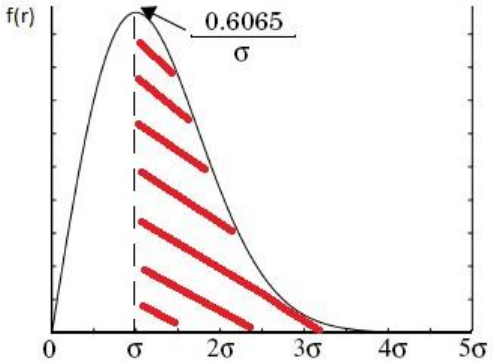
Na Figura 53, para iniciar a simulação deve-se clicar no botão RUN, que iniciará o processo de envio de dados entre os nós S e D. Conforme o tipo de protocolo de roteamento inicialmente escolhido no item (IV), o trajeto e sequência de envio dos pacotes de dados são mostrados na topologia durante a simulação no item (I). Durante a simulação os dados relativos ao sistema são coletados e mostrados no item (II), abrangendo as 4 características usadas na Lógica *Fuzzy*.

Foi escolhido o canal com FDP do tipo Rayleigh, ou seja, o meio de transmissão sem fio tem desvanecimento conforme a curva da Figura 54. A área

hachurada representa $P(x > 1)$, ou seja, demonstra a área da curva de distribuição de Rayleigh para $\sigma = 1$. A equação 4.42 representa a função de densidade do sinal da Figura 54.

$$f_{(x;\sigma)} = \frac{x}{\sigma^2} e^{\left\{\frac{-x^2}{2\sigma^2}\right\}}, x \geq 0 \; e \; \sigma \geq 1 \qquad 4.42$$

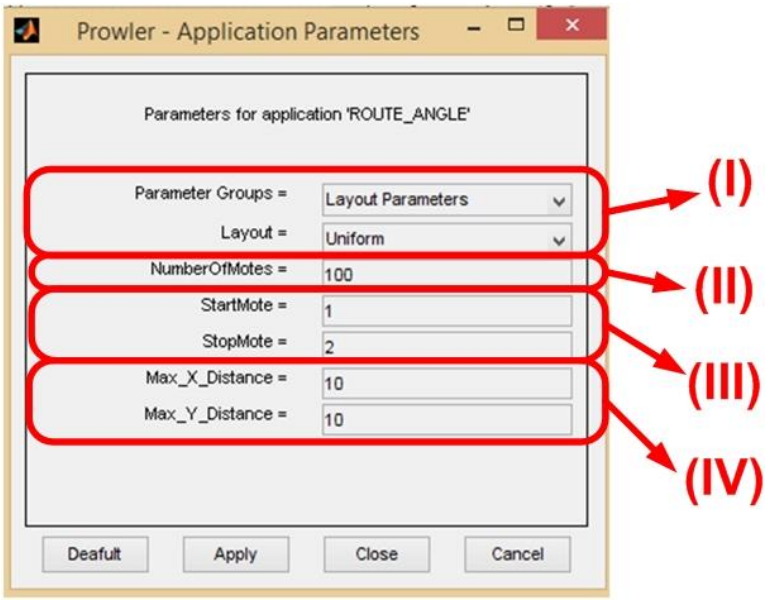
Figura 54. Função Densidade de Probabilidade do tipo Rayleigh para o canal de comunicação da rede simulada no *Prowler*.



Fonte: Autor.

A janela apresentada abaixo corresponde aos dados designados ao sistema e que são utilizados na simulação do ambiente de rede.

Figura 55. Tela de dados básicos para simulação do *Prowler*.

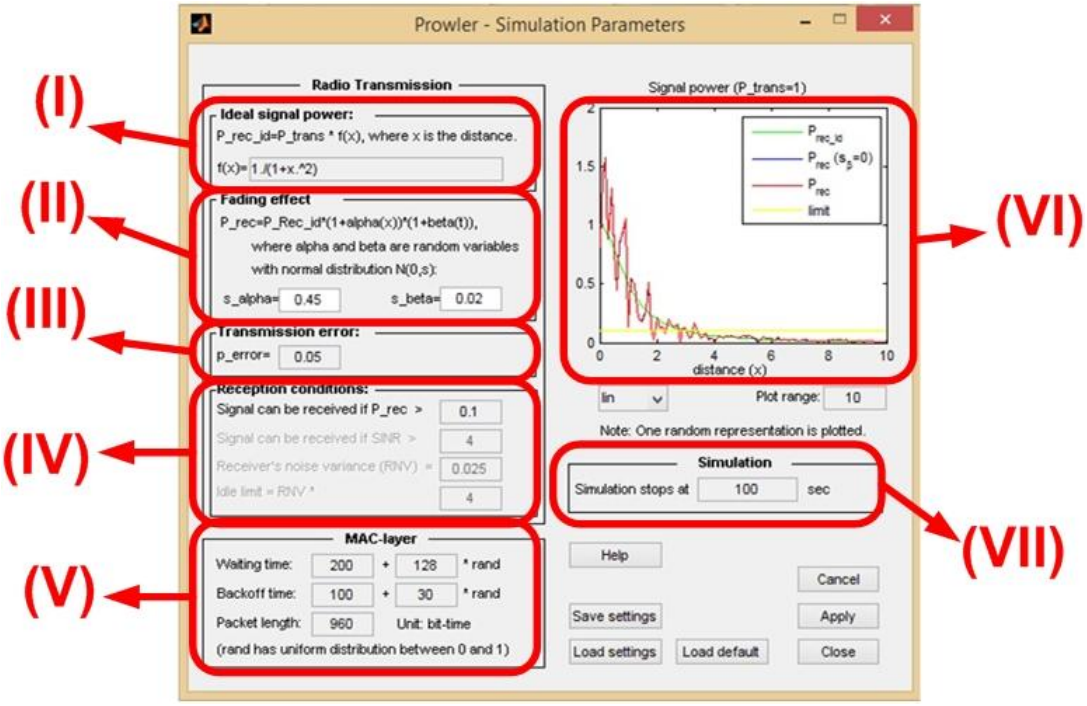


Fonte: Adaptado de *Prowler* (2015).

Na Figura 55, (I) corresponde às configurações gerais de *layout* da rede, (II) à quantidade total de nós que estarão presentes na rede, (III) à definição sobre os nós

fonte e destino e (IV) à máxima distância bidimensional entre os nós na rede. Ao clicar em parâmetros relativos ao modelo de propagação dos dados entre os nós da rede, a tela da Figura 56 é apresentada.

Figura 56. Tela de configuração de parâmetros para simulação do *Prowler*.



Fonte: Adaptado de *Prowler* (2015).

Conforme a Figura 56, (I) apresenta a expressão para a potência do sinal na relação de transmissão e recepção, (II) representa a característica de *fading* geral em um canal comum, (III) a probabilidade de erro de transmissão entre os nós, (IV) a condição para a recepção do sinal, (V) aos dados da camada MAC, (VI) à energia do sinal de acordo com a FDP em relação à distância do nó fonte e (VII) ao tempo total de simulação. O *software* já contempla a geração automática do gráfico referente à quantidade de energia pela distância entre os nós da rede conforme a simulação é executada.

Como o intuito deste trabalho não é apenas a análise de quantidade de energia despendida entre os nós para a execução de tarefas, como o caso do envio de uma mensagem entre S e D, foi realizada a alteração do *software* para abordar outras métricas e análises de dados, bem como o acréscimo de inteligência de rede e tomada de decisão com o uso de Lógica *Fuzzy* e rodadas de reanálise da rede como um todo.

A Tabela 7 corresponde às configurações realizadas no *Prowler* para as simulações deste trabalho. Contempla a relação geral de dados para os cenários de testes e descrição do utilizado ou feito nas camadas de WSN.

Tabela 7 – Configurações básicas do *Prowler*.

Parte da simulação	Características específicas	Descrição / Dados
Cenários	Quantidade de nós sensores	50 a 200 (incremento de 50 nós em cada cenário).
	Tipo de nós sensores	MICAz (família MICA *motes*).
	Tempo de simulação	Foi definido 10 s de simulação para cada cenário e cada *round*, que contempla tempo suficiente para o encaminhamento de pacotes fim-a-fim.
	Relação de coleta de amostras	Foi definido 10 ms para a amostragem do sinal, bem como realização de verificação de condições IF dos blocos internos do ACS.
	Parâmetro de simulação da rede	Envio de pacotes em tempos randômicos de 3 a 5 segundos após o envio do pacote anterior.
	Estáticos	Foram simulados cenários com nós estáticos com distribuição aleatória dos nós conforme a função $n_x \times n_y \times (d_x + d_y - 1)$.
Camadas	Aplicação	Envio de mensagem de texto com quantidade de caracteres que contemple o comprimento de 960 b.
	Rede	Tratamento inicial de pacotes de 960 bits de dados.
		Possibilidade de trabalhar com diferentes protocolos de roteamento.
	MAC	Utilização do CSMA/CA.
		Uso de uma variável randômica com distribuição uniforme entre 0 e 1 (*rand*).
		Tempo de espera é $t_e = 200 + 128 \times \{\text{rand}\}$.
		Tempo de *backoff* com $t_b = 100 + 30 \times \{\text{rand}\}$.
		Comprimento do pacote foi assumido inicialmente como 960 b.
	Física	Efeito de *fading* do tipo Rayleigh com $\sigma = 1$.
		Probabilidade de erro de transmissão correspondente à $p_{erroT} = 5\ \%$.
		Variância de ruído no receptor (RNV) de 2,5 %.
		Limite de desocupação do canal pelo nó é $\text{Idle}_{\text{limit}} = RNV \times 4$.
		Modelo de rádio conforme: $P_{\text{rec,ideal}}(d) = P_{\text{trans}}.\left(\frac{1}{1+d^{\gamma}}\right)$ $P_{\text{rec}}(i,j) = P_{\text{rec,ideal}}(d_{ij}).(1 + a(ij)).(1 + \beta(t))$ $\alpha: N(0, \sigma_\alpha), \sigma_\alpha \leftarrow 0{,}45$ $\beta: N(0, \sigma_\beta), \sigma_\beta \leftarrow 0{,}02$ $i \leftarrow j \Leftrightarrow P_{\text{rec}}(i,j) > \Delta$

Fonte: Autor.

Após finalizado o envio dos dados entre os nós, é iniciado o processo de análise via uso de regras *Fuzzy*, conforme modelo desenvolvido no *Simulink* e apresentado a seguir.

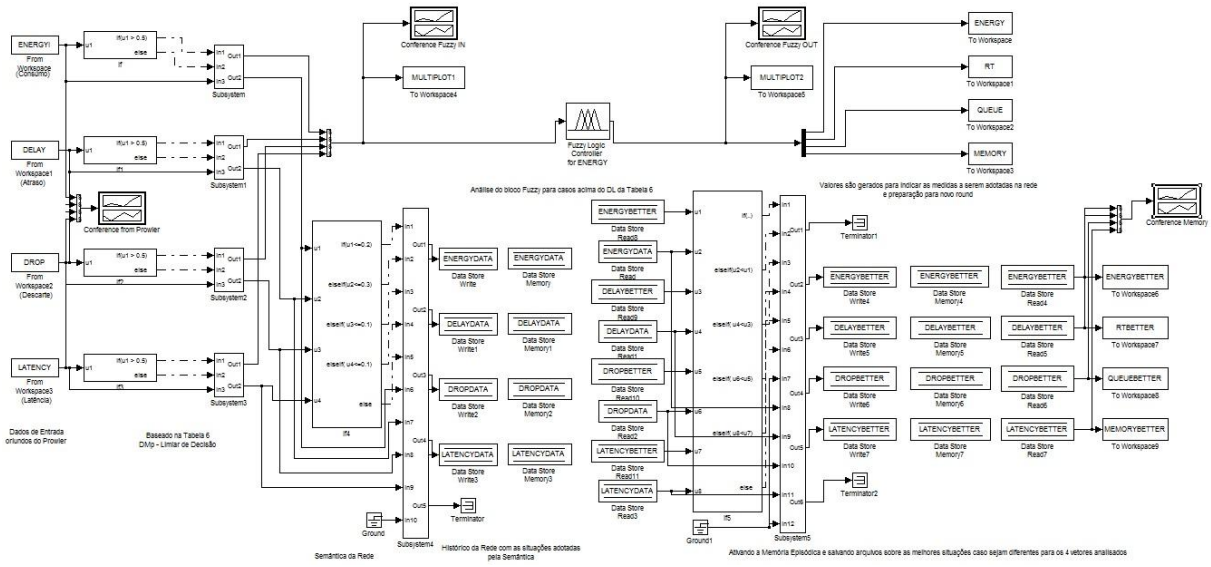
4.4.4. Implementação do ACS e CPMod no *software* de simulação

Para que fosse possível a implementação das arquiteturas desenvolvidas do ACS e seu módulo de inteligência baseada na semântica, denominado CPMod, foi necessário realizar alterações nos arquivos base do *Prowler*.

Como a parte de rotinas de *software* é muito grande, serão expostos aqui os aspectos mais importantes para o entendimento do sistema e demais informações sobre subsistemas dos blocos.

Na implementação da arquitetura do CPMod, utilizou-se o *Simulink* do MATLAB para a criação do modelo de análises de *rounds*. A Figura 57 apresenta o modelo desenvolvido para a simulação (também presente no Apêndice V).

Figura 57. Modelo do CPMod desenvolvido no *Simulink* do MATLAB.



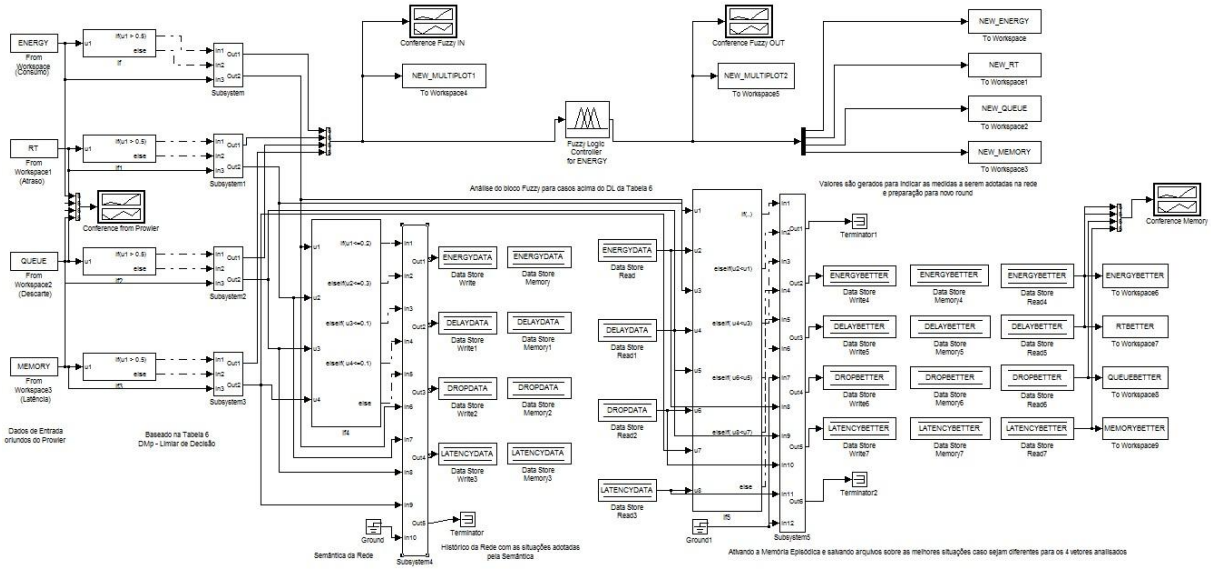
Fonte: Autor.

Com o bloco *Fuzzy* definido e inserido no modelo da Figura 55, são geradas as saídas do modelo inicial, que correspondem aos dados iniciais de entrada da atuação de correção do sistema, com alteração de dados conforme mostra a Figura 58 (também presente no Apêndice VI).

As Figuras 57 e 58 são compostas pelos blocos: entrada de dados provenientes do *Prowler*, DMp com base no DL, semântica, histórico de eventos ocorridos na rede, memória episódica com os eventos mais importantes para a

tomada de decisão e Lógica *Fuzzy*. Estes blocos são os responsáveis pelas diferentes funções presentes no ACS, conforme detalhado a seguir.

Figura 58. Modelo de correção aplicado ao CPMod que foi desenvolvido no *Simulink* do MATLAB.



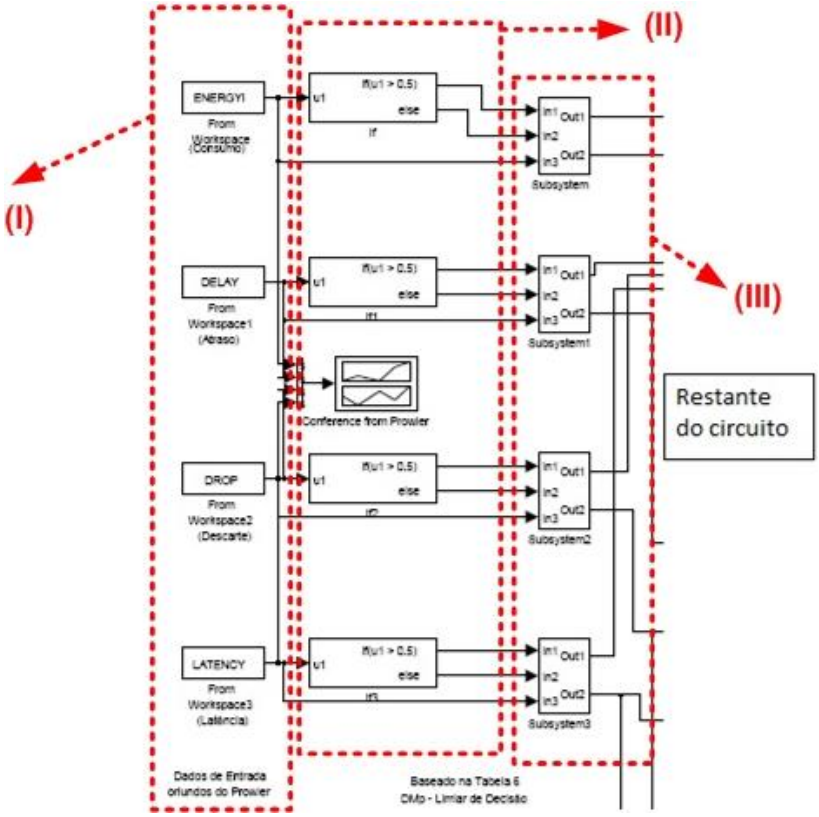
Fonte: Autor.

Do modelo de correção ilustrado na Figura 58, tem-se a execução da reanálise dos dados oriundos da primeira iteração ou de realização anterior. Os blocos NEW_X, com $X = \{\mathrm{ENERGY}, \mathrm{ROUTE}, \mathrm{QUEUE}, \mathrm{MEMORY}\}$, referem-se aos novos dados gerados pelas regras *Fuzzy* para a aplicação em nova rodada de simulação no *Prowler* adaptado neste trabalho.

Na Figura 59 são apresentados os blocos (I) de entrada de dados (que no caso está vinculada ao *Prowler*, mas pode ser conectada a qualquer outro dispositivo que forneça informações acerca de quantidade de energia consumida, atraso fim-a-fim, descarte de pacotes e latência na rede) e (II) de análise de dados de DL com base na Tabela 6, que contém a lógica condicional para determinar o caminho inicial dos dados, se para a semântica ou Lógica *Fuzzy*.

A separação dos dados é realizada com base na Tabela 6, ou seja, se os valores de entrada forem maiores ou iguais a 0,5, com a leitura do DL verifica-se que a característica está no mínimo no nível de atenção, portanto sendo enviado ao bloco *Fuzzy* para tratamento. Caso contrário, o dado já pode ser enviado pelos subsistemas (III) diretamente à semântica para a verificação.

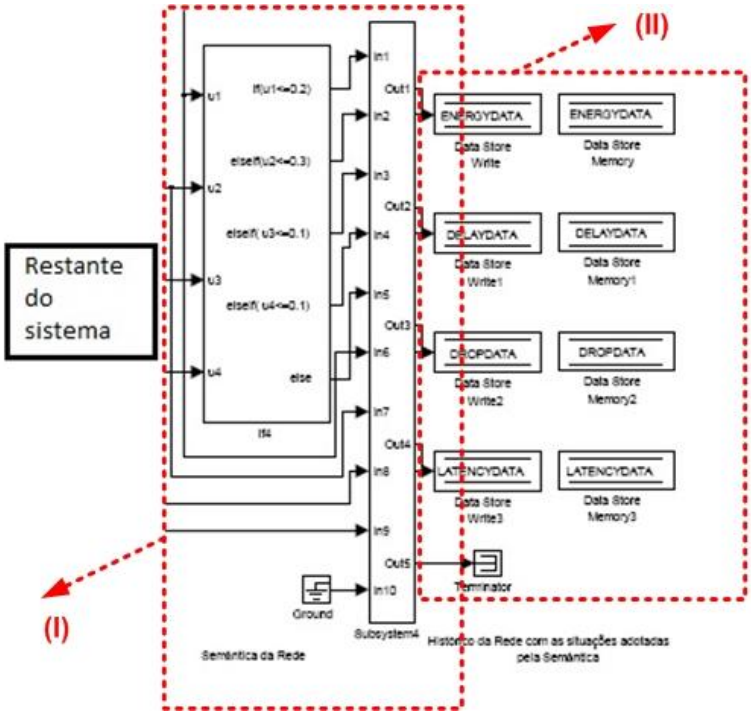
Figura 59. Blocos de entrada de dados provenientes do *Prowler* e de DMp com base no DL.



Fonte: Autor.

A Figura 60 mostra os blocos de semântica (I) e de histórico da rede (II).

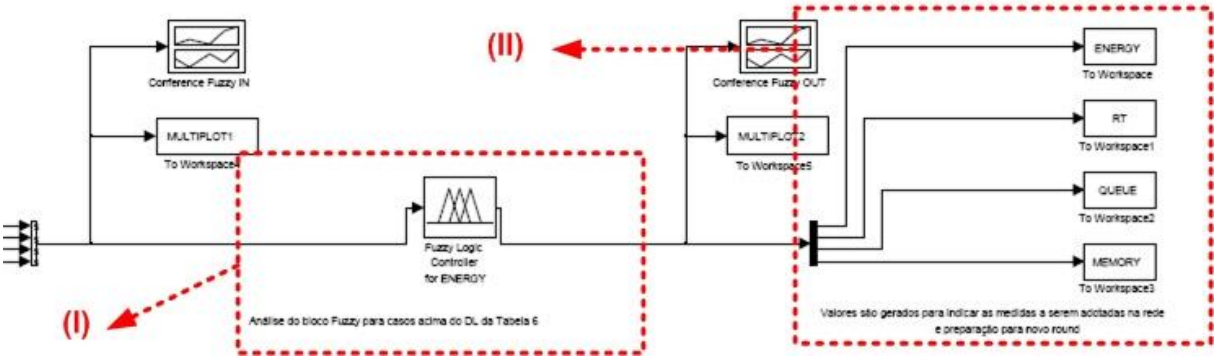
Figura 60. Blocos de semântica e de histórico da rede.



Fonte: Autor.

Na Figura 60, a semântica corresponde aos limites aceitáveis com relação aos significados e o histórico da rede refere-se ao bloco de armazenamento de eventos significativos da rede. A Figura 61 mostra a parte do sistema desenvolvido que contempla a Lógica *Fuzzy*.

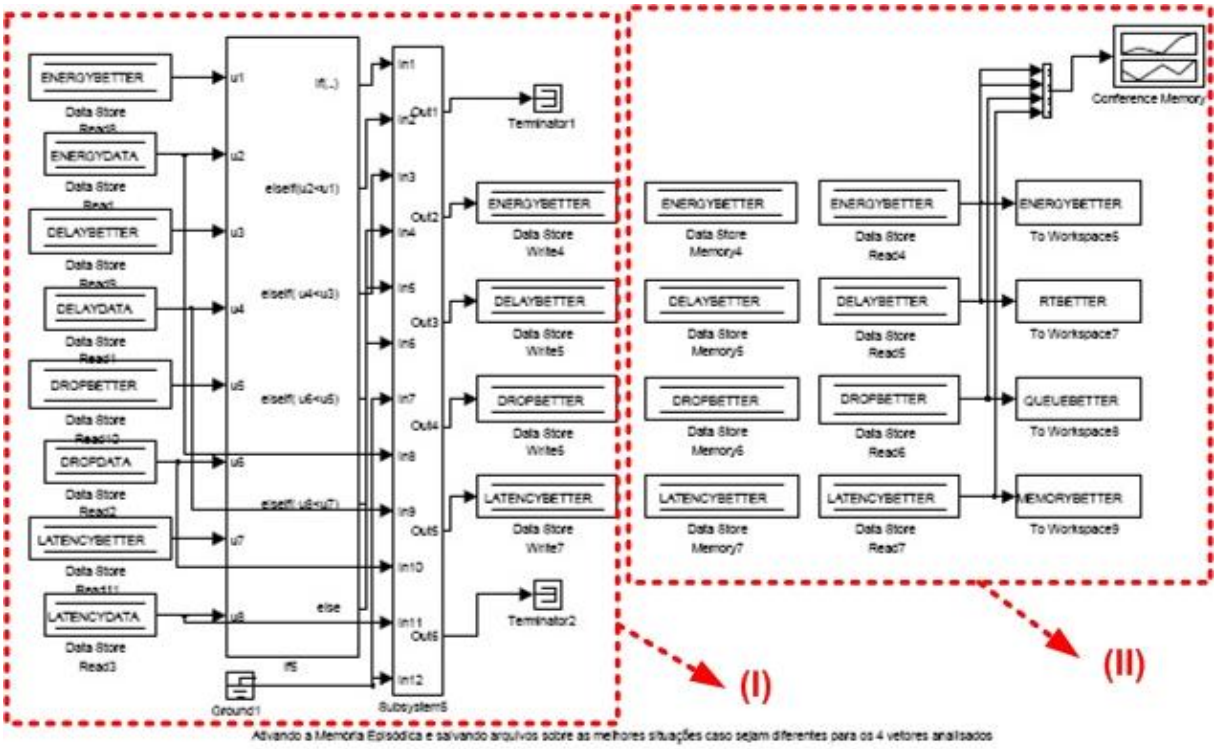
Figura 61. Bloco de Lógica *Fuzzy*.



Fonte: Autor.

Na Figura 61, (I) representa o bloco *Fuzzy* com a realização da inferência, de acordo com as regras e funções de pertinência. (II) está relacionado com a saída da Lógica *Fuzzy*, com a relação de V_C para ser adotado pelo CPMod.

Figura 62. Bloco de memória episódica do início de verificação da rede.



Fonte: Autor.

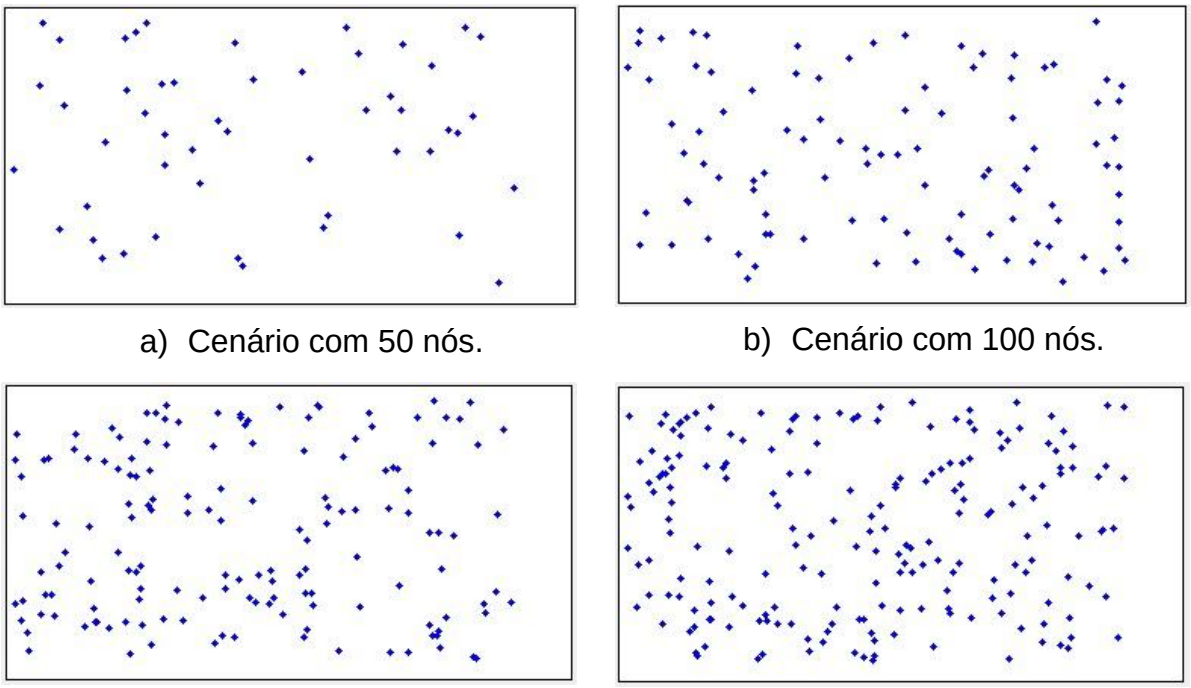
A Figura 62 mostra o bloco de memória episódica correspondente à Figura 57. (I) representa a parte condicional da memória para a verificação sobre os dados a serem armazenados em busca da solução adotada conforme a semântica e (II) corresponde ao bloco de lembrança sobre os eventos ocorridos na rede.

O bloco de memória episódica da Figura 58 realiza a mesma ação, porém, com comparação entre os dados que chegam dos novos *rounds* executados.

4.5. Simulação em *Software*

A Figura 63 mostra os cenários de testes para 50, 100, 150 e 200 nós, com distribuição aleatória dos nós conforme a função $f_{(\text{posição})} = n_x \times n_y \times (d_x + d_y - 1)$, onde (n_x, n_y) corresponde ao tamanho do cenário que é de (800,500) e (d_x, d_y) o deslocamentos nos eixos X e Y.

Figura 63. Cenários de 50, 100, 150 e 200 nós.



a) Cenário com 50 nós.

b) Cenário com 100 nós.

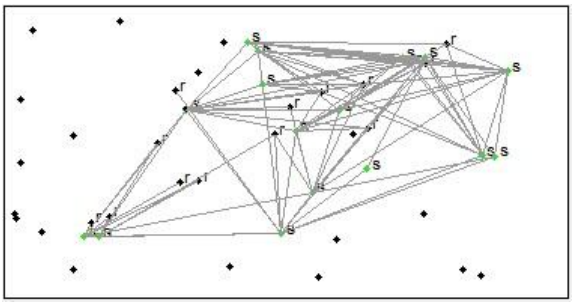
c) Cenário com 150 nós.

d) Cenário com 200 nós.

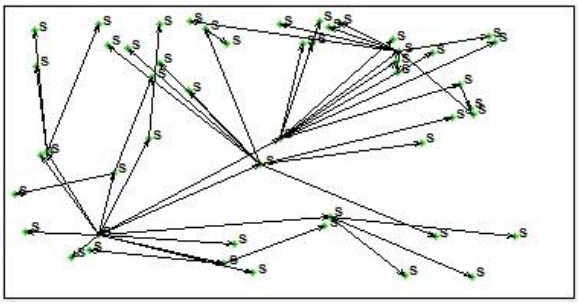
Fonte: Autor.

A Figura 64 apresenta a aplicação de diferentes protocolos de roteamento nos cenários de testes, neste caso, abrangendo os de característica *spanning tree* (ABAM) e *route angle* (ARP).

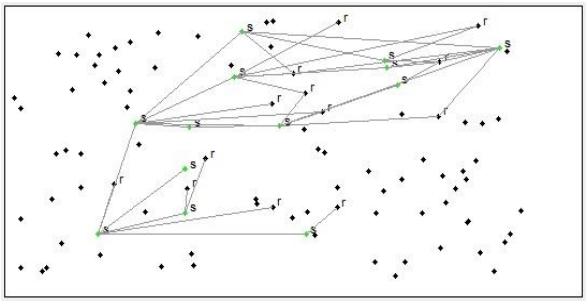
Figura 64. Primeira rodada de testes finalizada conforme a quantidade de nós.



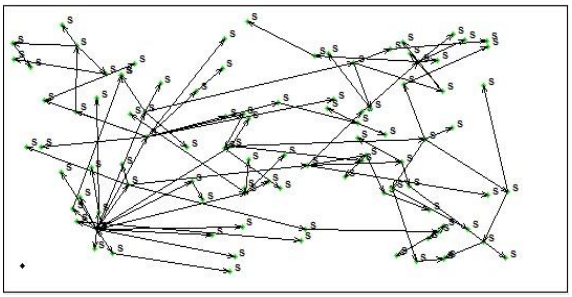
a) 50 nós ARP



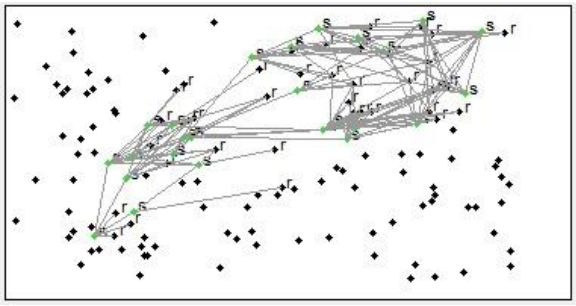
b) 50 nós ABAM



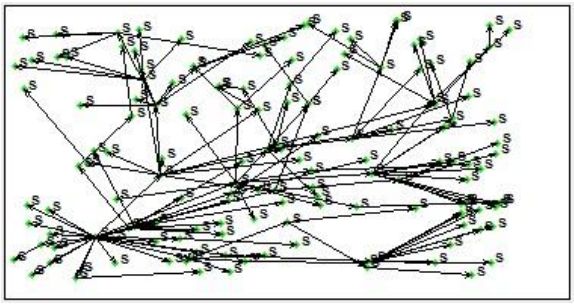
c) 100 nós ARP



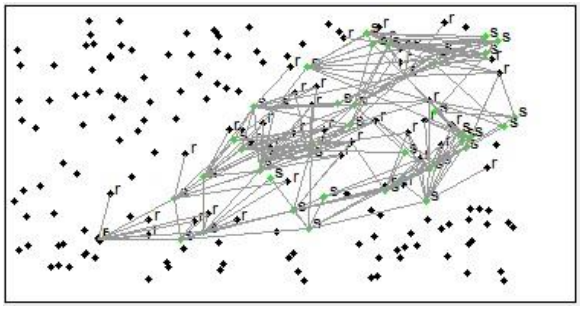
d) 100 nós ABAM



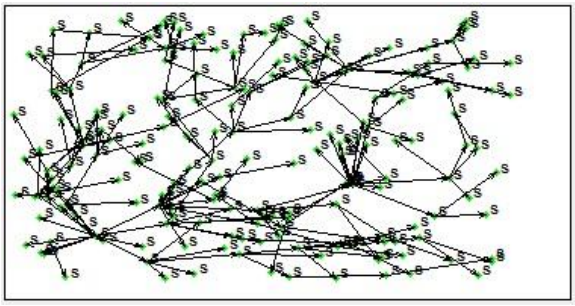
e) 150 nós ARP



f) 150 nós ABAM



g) 200 nós ARP

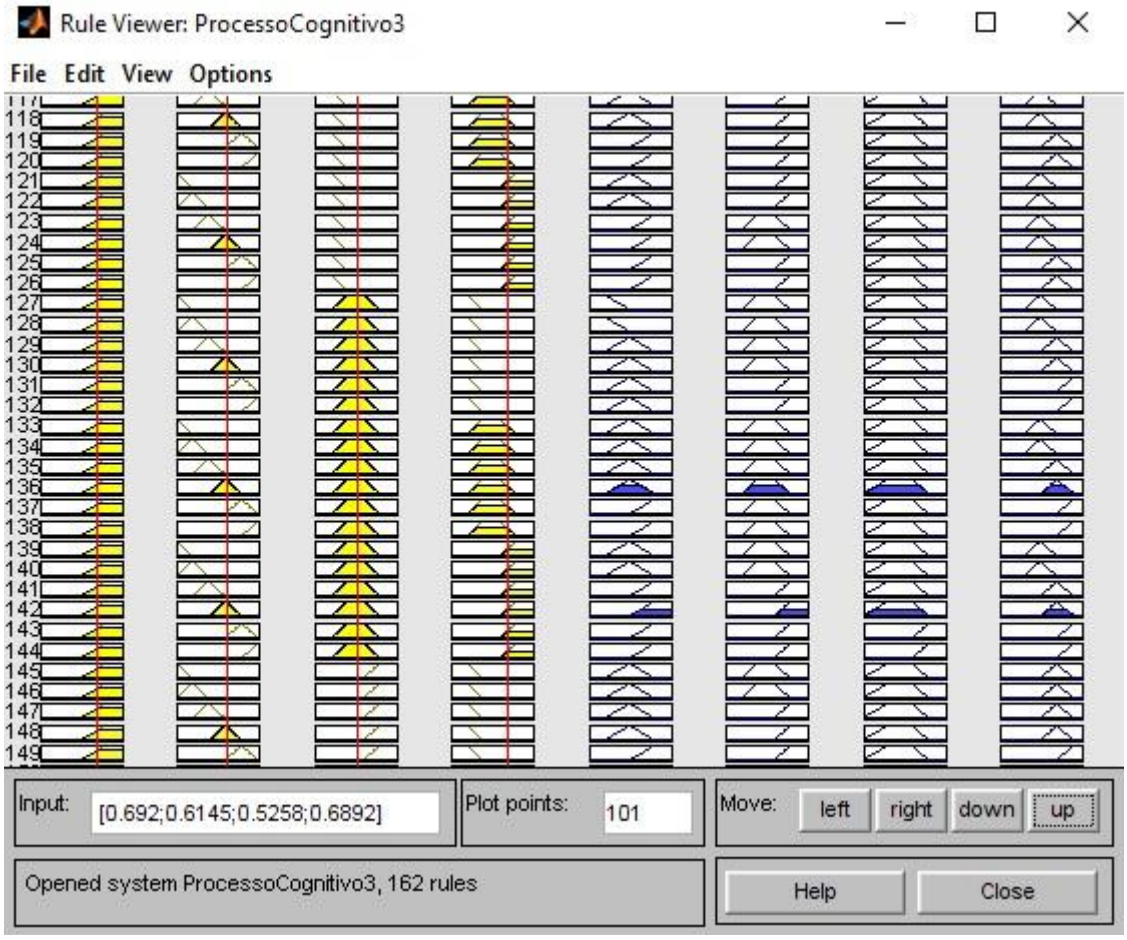


h) 200 nós ABAM

Fonte: Autor.

A Figura 65 mostra o processo de inferência *Fuzzy* aplicado sobre as regras (APÊNDICE IV), com base no centroide, que fornece suporte ao CPMod para a tomada de decisão sobre as características a serem alteradas para as diferentes camadas de rede.

Figura 65. Uso da inferência para auxiliar o CPMod na adoção de tomada de decisão.



Fonte: Autor.

Na Figura 65 visualizam-se linhas vermelhas verticais, que correspondem à varredura de acordo com as funções de pertinência, determinando em azul do lado direito da figura a relação de inferência para o evento ocorrido no instante de leitura do sinal de entrada do processo.

A cada iteração da rede, ou seja, a cada incremento de *round* executado na rede, gera-se uma nova coluna de dados em uma tabela simples de formato *.xls*, que serve como manutenção temporária e está representada na Figura 41 como parte do módulo *Tables*. Esta tabela é modificada conforme a quantidade de *rounds* executados. A última coluna gerada na tabela será a que mantém a mais recente modificação na rede e, portanto, deve ser mantida no histórico da rede como a solução que registrou a melhor ação para o evento detectado. Desta forma, é copiada esta última coluna da tabela para uma nova tabela com vínculo ao evento registrado e constatado pelo V_C.

4.5.1. Configuração da camada MAC do *Prowler*

O protocolo MAC que foi utilizado no *Prowler* é o *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA). O nós podem ser baseados em um transceptor rádio de 2,5 GHz com controle de intensidade de sinal e sensibilidade, operando em três modos: transmissão, recepção e desligar. A energia consumida por 1 bit para a transmissão é de 1 µJ e para a recepção é 0,5 µJ.

A opção padrão no *Prowler* é o modelo probabilístico, que depende de dois fatores: α que implica no *fading* devido à mudança de topologia e β que implica no *fading* ao longo do tempo.

O modelo de propagação determinístico estima a intensidade de sinal do transmissor para o receptor com base na equação 4.43, onde P_r é a potência de recepção, P_t é a potência de transmissão, d é a distância e $\gamma = 2$ para forçar o sinal a variar com o quadrado da distância.

$$P_{r_{\text{ideal}}}(d) = P_t.\left(\frac{1}{1 + d^{\gamma}}\right) \qquad 4.43$$

Na recepção do sinal, existem dois modelos, sendo que o primeiro baseia-se no limite de recepção para a intensidade do sinal recebido, que pode ser estipulado alterando-se o parâmetro no *software*, e a outra é baseada no parâmetro de variância do ruído do receptor σ_n^2 que é mais preciso, porém, mais lento conforme a equação 4.44.

$$\text{SNIR} = \frac{P_r(i,j)}{\sigma_n^2.\sum_{k \neq j} P_r(\imath,k)} \qquad 4.44$$

Portanto, a intensidade total de sinal no nó *i* é dada por:

$$P_{r_{\text{total}}}(i) = \sum_{k} P_r(i,k) \qquad 4.45$$

A camada MAC/Enlace do *Prowler* utiliza o protocolo CSMA e a camada de aplicação é provida com eventos relacionados às aplicações selecionadas na rede

juntamente com GUI de interações. O CSMA com prevenção de colisão é usado no *Prowler*, porém, como não é utilizada nenhuma política de reconhecimento, as colisões resultam em perdas de pacotes encaminhados pelos nós, assim como ocorre nas filas de processamento dos nós, o que influencia nos testes de simulação, bem como nos dados de entrada do CPMod.

5. RESULTADOS OBTIDOS

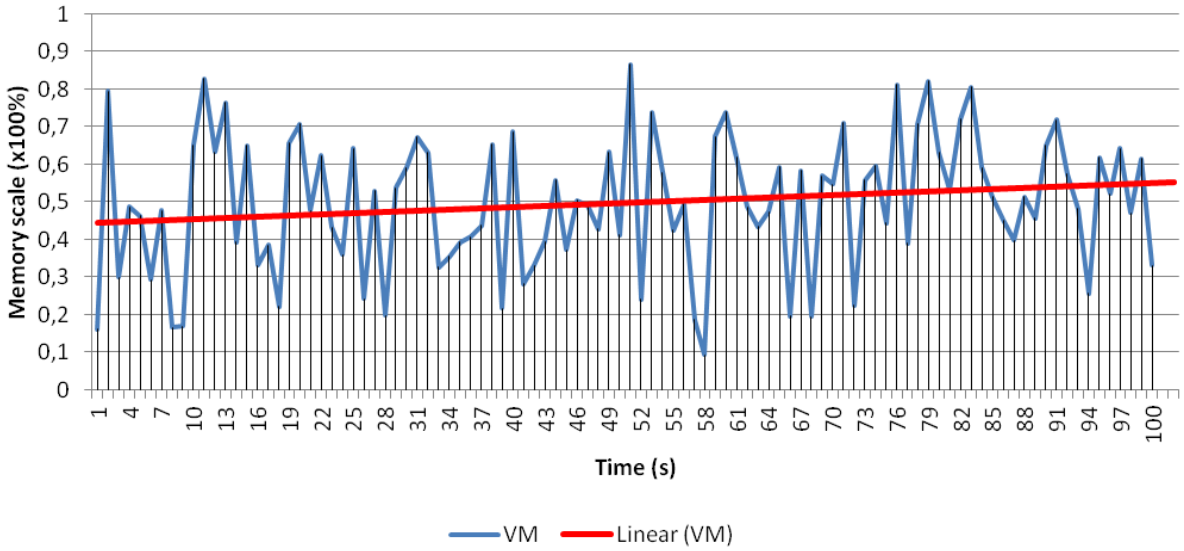
Com base nos dados gerados das simulações, realizaram-se análises sobre os gráficos obtidos, com relação às métricas definidas e semântica, abordando o desempenho dos nós do ponto de vista da rede e do usuário.

5.1. Resultados específicos do algoritmo de variabilidade

Aplicando-se o algoritmo de variabilidade dos nós sensores, puderam-se obter os seguintes resultados:

A memória semântica refere-se a uma memória de busca das situações atuais e verificação de acordo com um significado para a rede. A memória episódica é a que armazena os dados relacionados ao histórico da rede, referentes aos eventos ocorridos. O Gráfico 2 demonstra que quanto mais passos de eventos acontecem na rede, que devem ser notados e armazenados, maior é o consumo de espaço de memória para o processo, o que resulta em um limite para aplicação do ACS em aproximadamente 600s de simulação considerando-se a linha de tendência do gráfico apresentado.

Gráfico 2. Relação de capacidade de armazenamento na memória episódica.

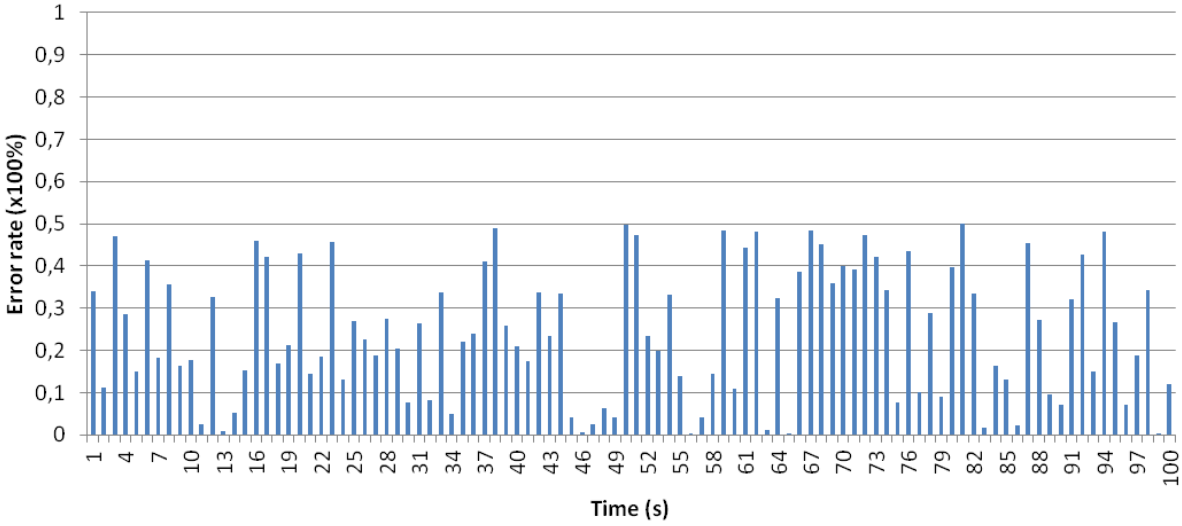


Fonte: Autor.

Contudo, deve-se considerar a taxa de erro calculada sobre os intervalos dispostos no Gráfico 2. Para a verificação sobre a relação de erro entre o sinal S_l

(coletado da rede) e S_O (em direção à rede), tem-se o Gráfico 3.

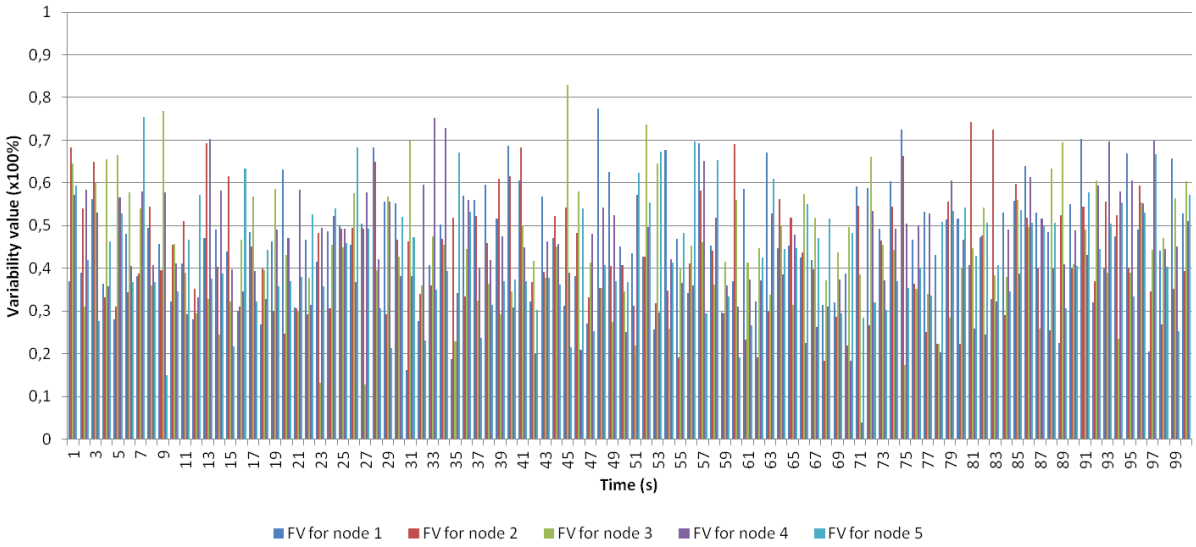
Gráfico 3. Taxa de erro relativa aos eventos da rede.



Fonte: Autor.

O Gráfico 4 apresenta a variabilidade do sistema F_N, calculada pela equação 4.1, com simulação sobre 5 nós na rede.

Gráfico 4. Grau de variabilidade dos nós sensores.



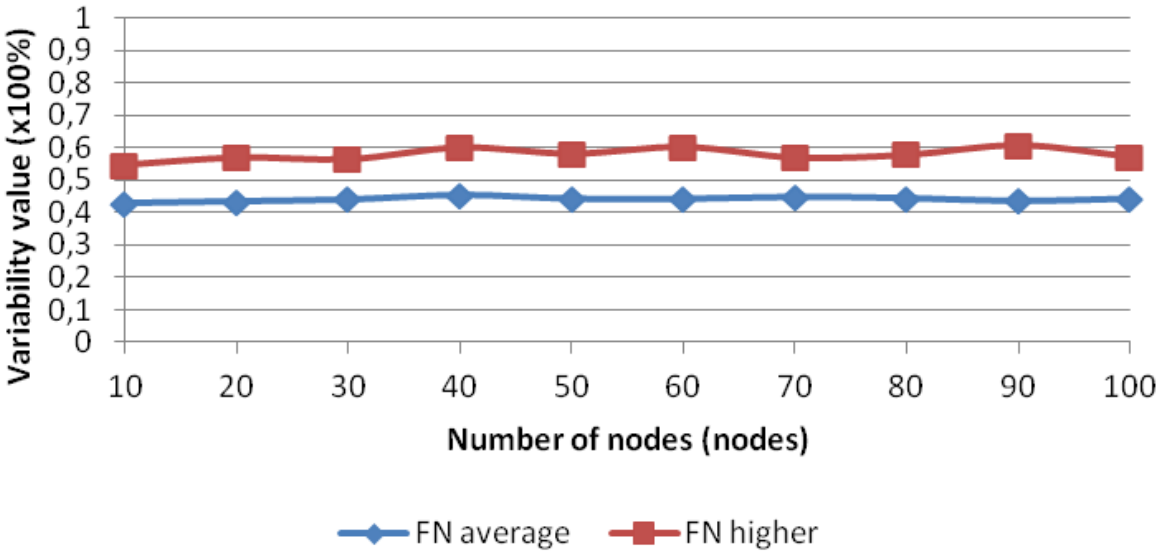
Fonte: Autor.

Nota-se através do Gráfico 4 que o grau de variabilidade dos nós sensores na rede assume valores com picos elevados, ultrapassando 70% em 10 casos, o que remete à instabilidade na rede ao uso de ajustes pelo V_C.

O Gráfico 5 mostra a variabilidade aplicada a uma quantidade maior de nós

sensores, que resulta em uma média em torno de 45%. No gráfico tem-se também a relação de picos de variabilidade detectados conforme a quantidade de nós.

Gráfico 5. Variabilidade para um número maior de nós.



Fonte: Autor.

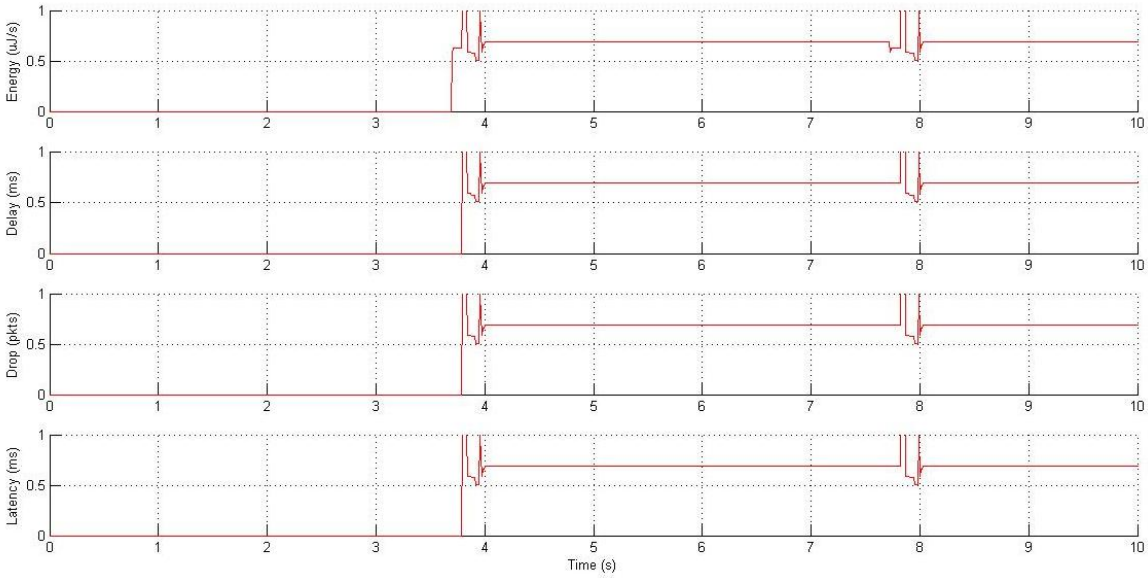
Verificou-se que os picos geralmente atingem máximos de 60% e que a variabilidade corresponde ao nó, pois mede características de sensores individualmente, mas que afeta a rede como um todo, portanto, sendo necessário aplicar um sistema de controle e adequação da rede para diferentes cenários, sendo o ACS a solução para atuação em múltiplas camadas de rede.

5.2. Resultados das simulações no MATLAB

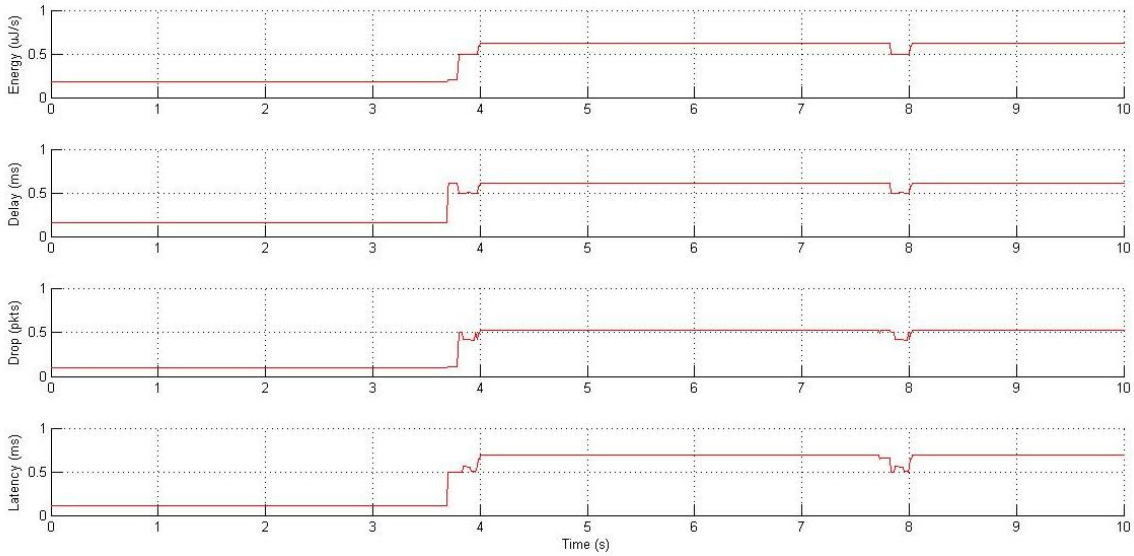
Dadas as descrições dos cenários de teste em *software*, os Gráficos 6 até 19 representam simulações para cenários com 100 nós. A mesma sequência de testes foi realizada para 50, 150 e 200 nós, constando aqui os gráficos com a compilação dos resultados de diversos *rounds*.

Para o caso de 100 nós, o Gráfico 6 mostra o resultado para a primeira rodada de testes. Na qual já se considera o filtro de seleção condicional para dados de entrada da Lógica *Fuzzy* sobre os casos de necessidade de atuação, que corresponde a F_{IN}>50%, conforme a Tabela 6.

Gráfico 6. Entrada e saída do módulo *Fuzzy* da primeira rodada de testes.



a) Entrada do bloco *Fuzzy* para início dos testes.



b) Saída do bloco *Fuzzy* para início dos testes.

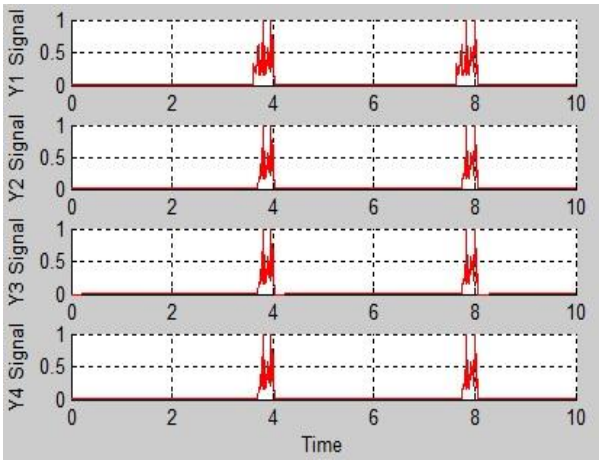
Fonte: Autor.

O Gráfico 6b mostra que houve necessidade de atuação do ACS sobre 3 itens da rede, onde $V_C = \{1, 1, 0, 1\}$, sendo que 1_2 corresponde à atuação do CPMod sobre a característica e 0_2 que não há necessidade de mudanças. Devido a isto, o Gráfico 8 mostra o resultado para a primeira interação e *round* do sistema com o uso de V_C.

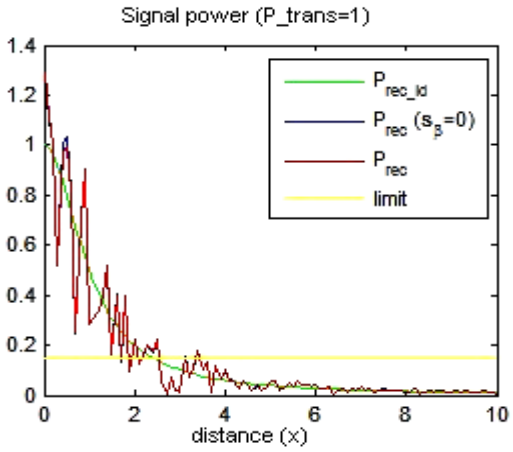
O Gráfico 7a apresenta a saída do bloco *Prowler* (que é a entrada do sistema com o DL), o Gráfico 7b a curva que representa a energia do sinal em relação à distância para a recepção por um nó vizinho e o Gráfico 7c a representação de uso

da memória para armazenamento das melhores condições.

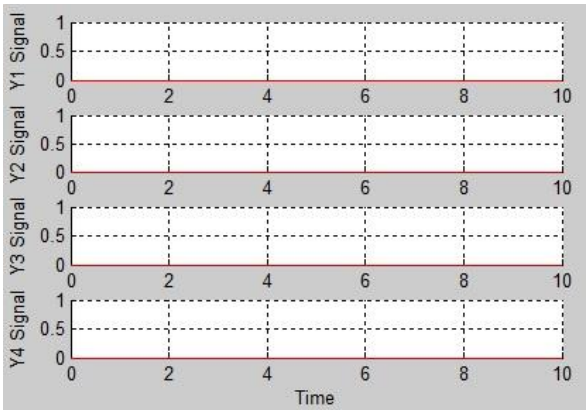
Gráfico 7. Dados sobre o sinal gerado entre os nós na primeira rodada.



a) Entrada do ACS.



b) Energia x Distância em um nó.



c) Armazenamento das melhores situações.

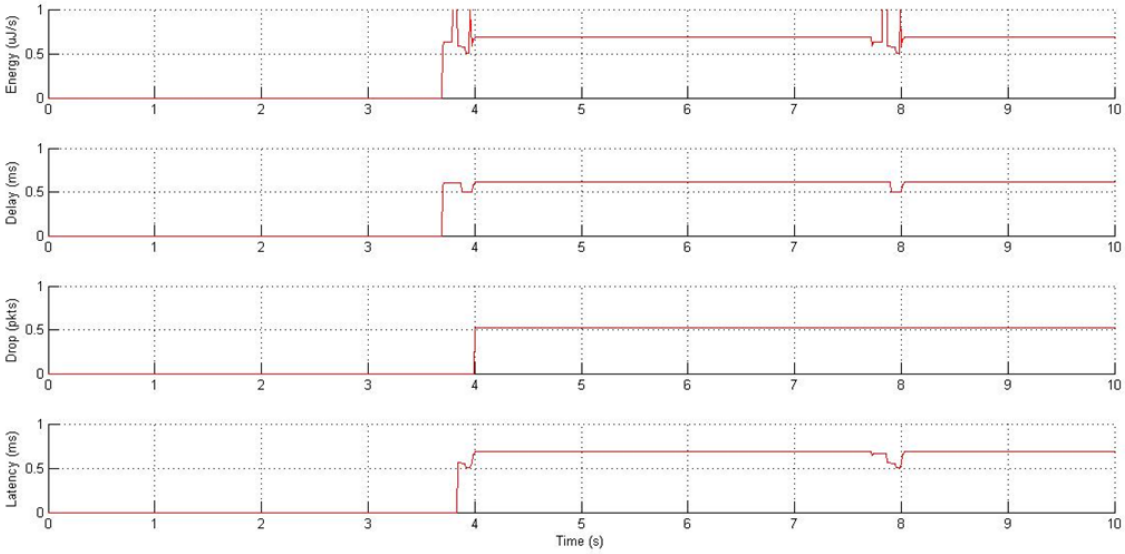
Fonte: Autor.

O sinal do Gráfico 7a mostra que houve duas fases críticas na rede, alcançando o grau de máxima urgência para atuação do CPMod em direcionar o V_C (conforme Tabela 6) para adequar a rede ao determinado pela semântica. O Gráfico 7b mostra o efeito do desvanecimento sobre a transmissão dos nós na rede, onde β implica a relação de *fading* ao longo do tempo, P_{rec_id} à potência de recepção ideal conforme a curva Rayleigh, P_{rec} à potência de sinal efetivamente aplicado ao cenário de testes para a recepção pelos nós sensores e *limit* corresponde ao limite estabelecido para que o nó consiga realizar o *link* com os nós vizinhos. O Gráfico 7c mostra que esta situação não foi relevante para armazenamento do histórico.

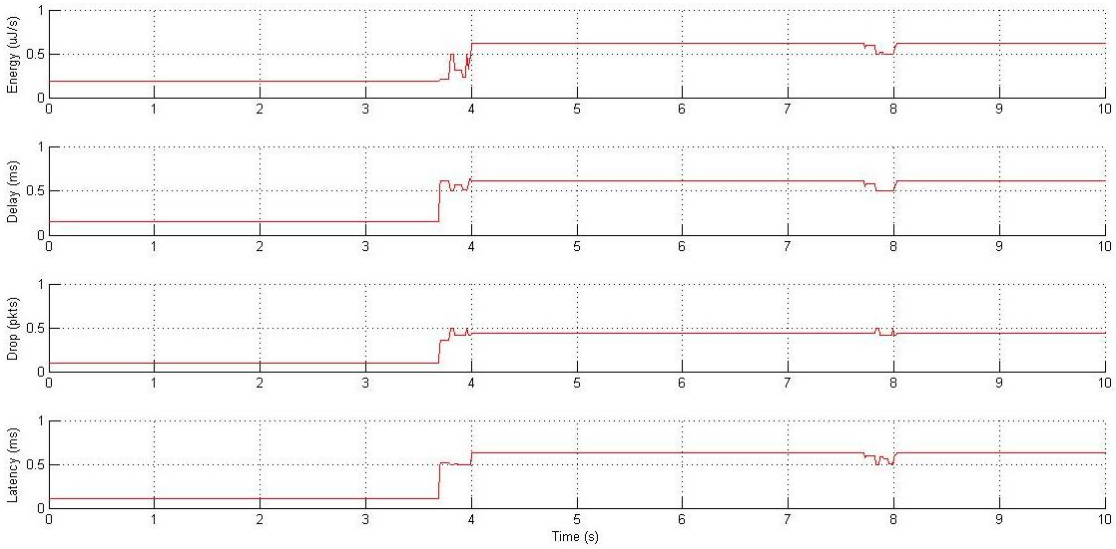
O processo de lembrança foi modelado de forma a procurar as melhores situações que foram armazenadas, visando atribuir as mesmas soluções caso um

evento semelhante aconteça, isto dentro da análise realizada pelo bloco de semântica da rede.

Gráfico 8. Entrada e saída do módulo *Fuzzy* do primeiro *round* de testes.



a) Entrada do bloco *Fuzzy* do primeiro *round*.



b) Saída do bloco *Fuzzy* do primeiro *round*.

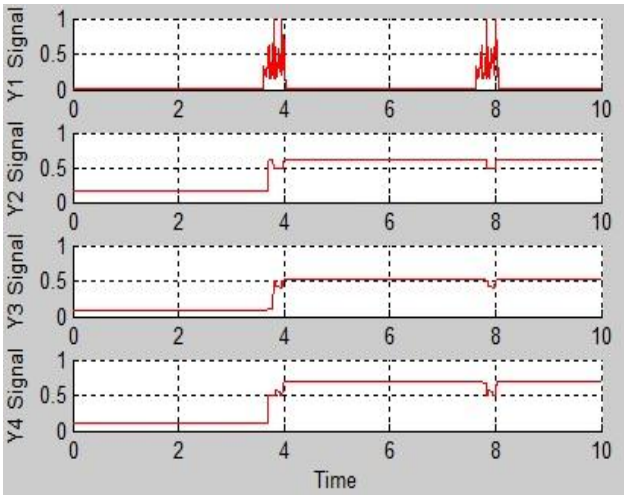
Fonte: Autor.

O Gráfico 8a mostra que na entrada do bloco *Fuzzy*, ou seja, com os valores obtidos da rede para atraso fim-a-fim, descarte de pacotes e latência na rede, já houve melhora com relação ao cenário inicial. O Gráfico 8b aponta para a saída do bloco *Fuzzy* com geração novamente de um $V_C = \{1, 1, 0, 1\}$ para mais adequações.

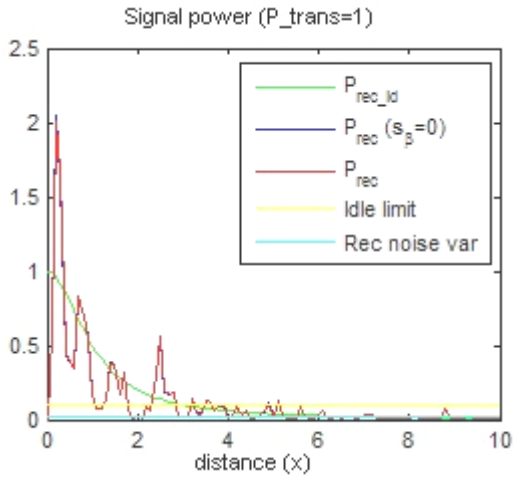
O Gráfico 9a apresenta a saída do bloco *Prowler*, o Gráfico 9b a curva que representa a energia do sinal em relação à distância para a recepção por um nó

vizinho e o Gráfico 9c a representação de uso da memória para armazenamento das melhores condições, para o primeiro *round* executado. No Gráfico 9a, já nota-se melhora quanto ao atraso fim-a-fim (Y2 *Signal*), descarte de pacotes na rede (Y3 *Signal*) e latência da rede (Y4 *Signal*). No Gráfico 9c, percebe-se que os dados relacionados com energia (Y1 *Signal*) e atraso fim-a-fim foram considerados no histórico da rede. No Gráfico 9a, os picos de sinal representam períodos de envio de dados (pacotes) na rede.

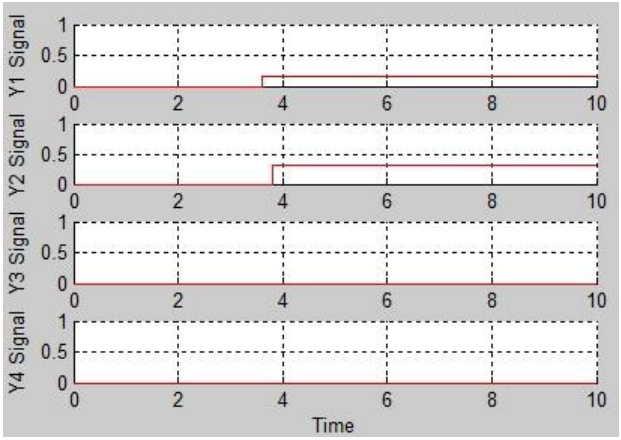
Gráfico 9. Dados sobre o sinal gerado entre os nós no primeiro *round*.



a) Entrada do ACS.



b) Energia x Distância em um nó.



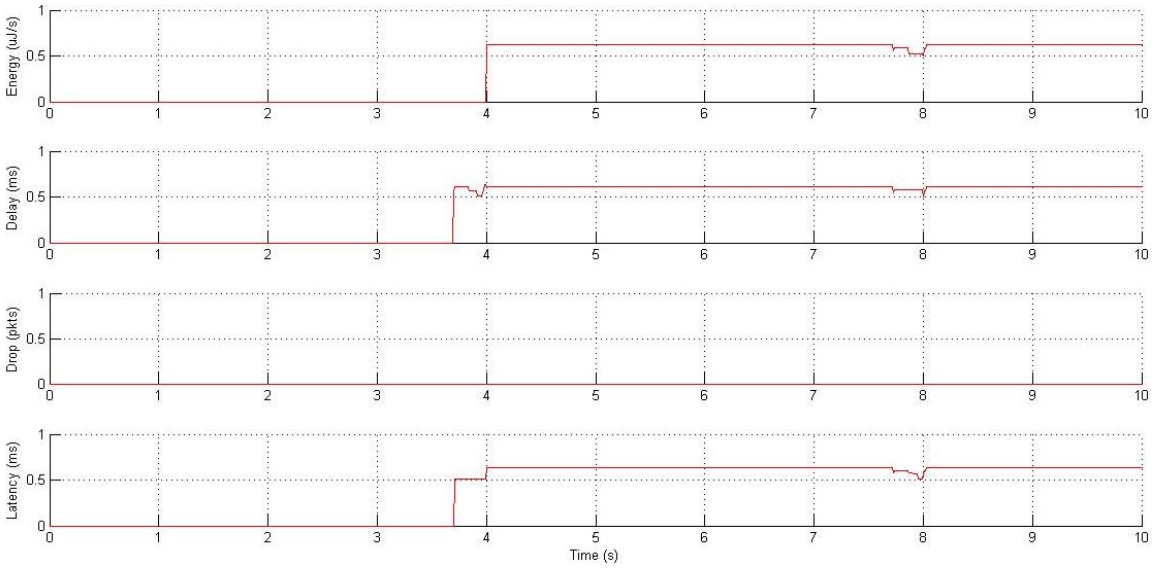
c) Armazenamento das melhores situações.

Fonte: Autor.

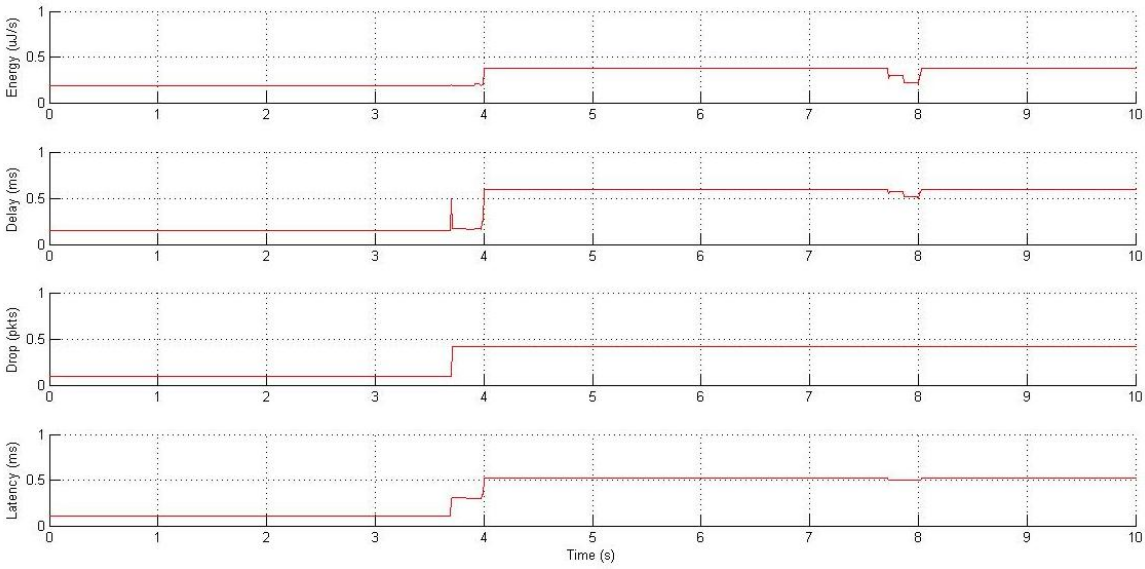
O Gráfico 10a expõe que na entrada do bloco *Fuzzy* já houve melhora com relação ao cenário do primeiro *round*, pois não ocorreram picos de intensidade igual ou acima de 0,7 no vetor de energia consumida pelos nós da rede.

O Gráfico 10a mostra ainda, que na entrada do bloco *Fuzzy*, ou seja, com os valores obtidos da rede para atraso fim-a-fim, descarte de pacotes e latência na rede, já houve melhora com relação ao cenário inicial. No Gráfico 10b, visualiza-se a necessidade do bloco *Fuzzy* em gerar novamente um $V_C = \{0, 1, 0, 0\}$, para mais adequações.

Gráfico 10. Entrada e saída do módulo *Fuzzy* do segundo *round* de testes.



a) Entrada do bloco *Fuzzy* do segundo *round*.



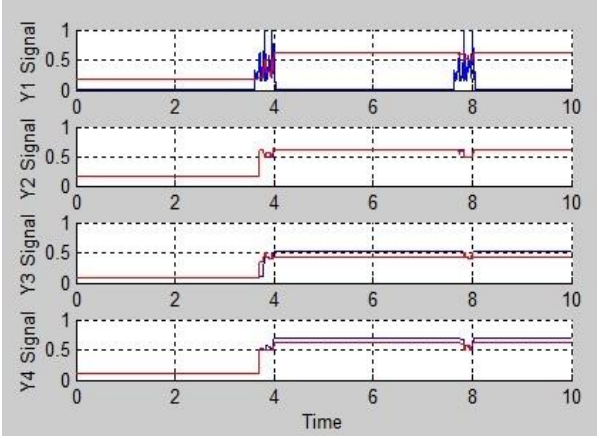
b) Saída do bloco *Fuzzy* do segundo *round*.

Fonte: Autor.

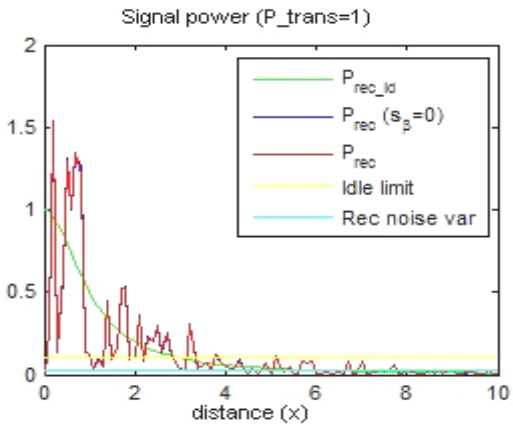
No Gráfico 11a, a linha vermelha que representa o estado atual da rede está comparada com a azul, que representa o *round* anterior. Percebe-se neste ponto a melhoria da rede com a adequação realizada pelo CPMod. Com o Gráfico 11b,

verifica-se que a intensidade de sinal se manteve na mesma proporção e que no Gráfico 11c a característica do nó foi considerada para o histórico.

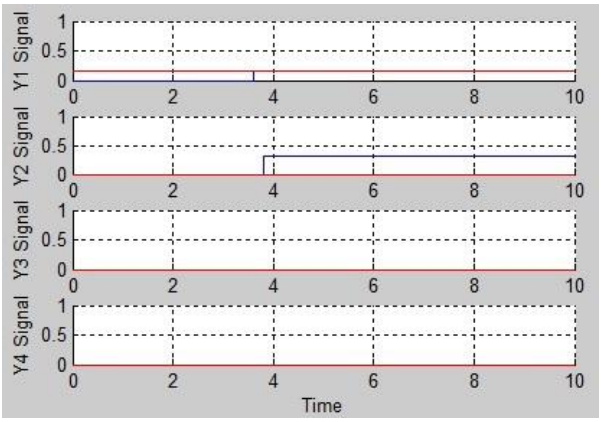
Gráfico 11. Dados sobre o sinal gerado entre os nós no segundo *round*.



a) Entrada do ACS.



b) Energia x Distância em um nó.



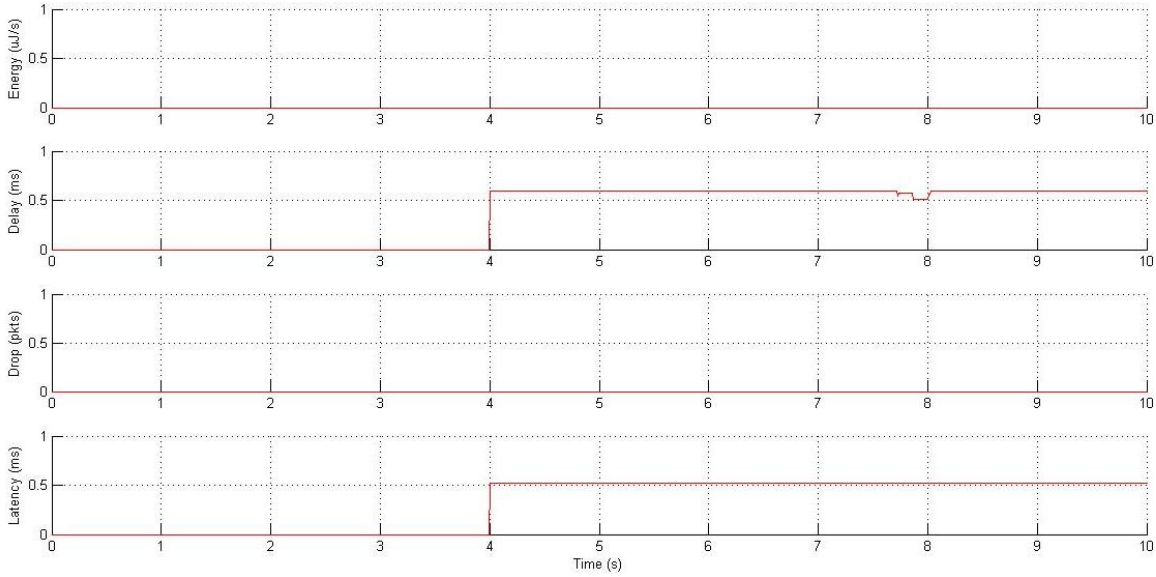
c) Armazenamento das melhores situações.

Fonte: Autor.

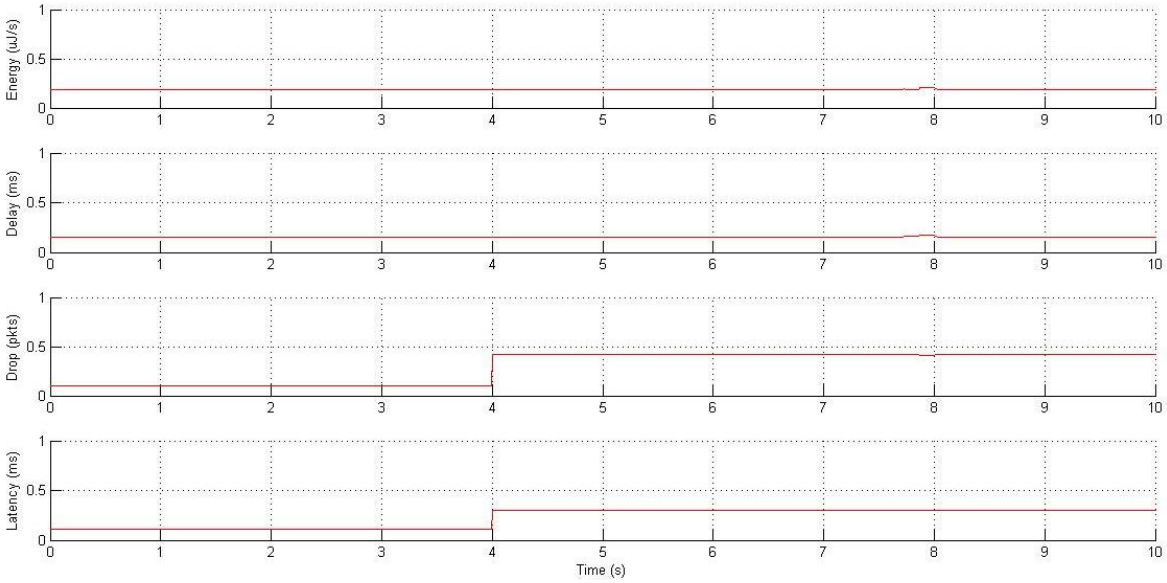
O Gráfico 12a mostra que na entrada do bloco *Fuzzy* já houve melhora com relação ao cenário do segundo *round*, pois não foram identificados problemas quanto ao consumo de energia pelos nós da rede. A saída do bloco *Fuzzy* indica que não existem mais adequações necessárias, conforme Gráfico 12b.

Nos cenários existe um *delay* de aproximadamente 4 segundos para o envio de pacotes, devido ao estabelecimento das conexões de ângulos pelo ARP e de criação de árvores no ABAM, sendo que no *spanning tree* este período praticamente dobra. Percebe-se que isto ocorre, pois no *route angle* o procedimento de descoberta é mais diretivo e no *spanning tree* ocorre a descoberta entre os nós vizinhos para a formação de cada ramo da árvore.

Gráfico 12. Entrada e saída do módulo *Fuzzy* do terceiro *round* de testes.



a) Entrada do bloco *Fuzzy* do terceiro *round*.

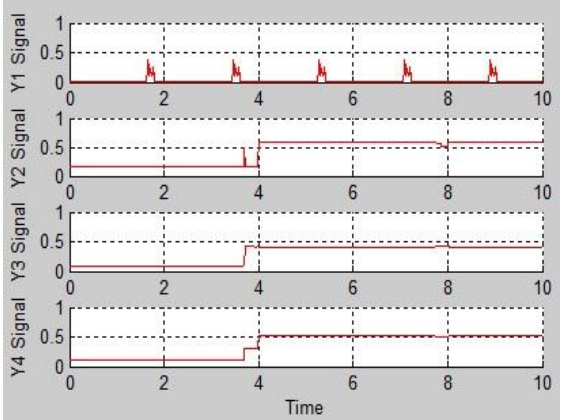


b) Saída do bloco *Fuzzy* do terceiro *round*.

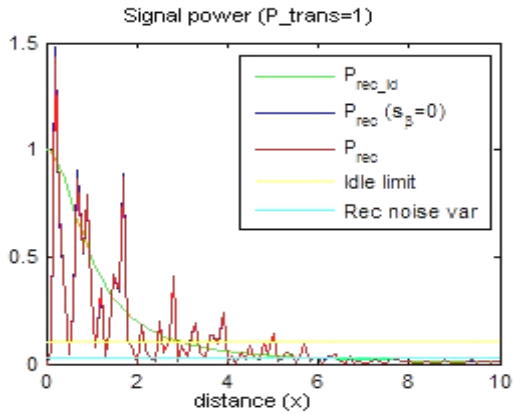
Fonte: Autor.

No Gráfico 13a, demonstra que o atraso está em nível de atenção, mas não existe necessidade de atuação pelo CPMod nesta situação, pois contempla algo aceitável pela semântica. O Gráfico 13b mostra um certo ganho de sinal, pois o alcance neste *round* atingiu aproximadamente 5m. O Gráfico 13c apresenta o armazenamento de informação sobre o ocorrido quanto à energia e ao atraso fim-a-fim.

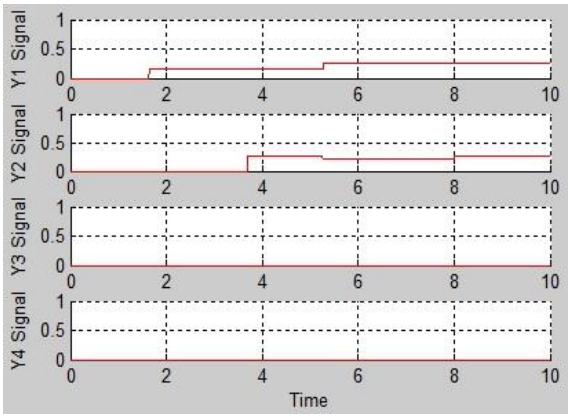
Gráfico 13. Dados sobre o sinal gerado entre os nós no terceiro *round*.



a) Entrada do ACS.



b) Energia x Distância em um nó.



c) Armazenamento das melhores situações.

Fonte: Autor.

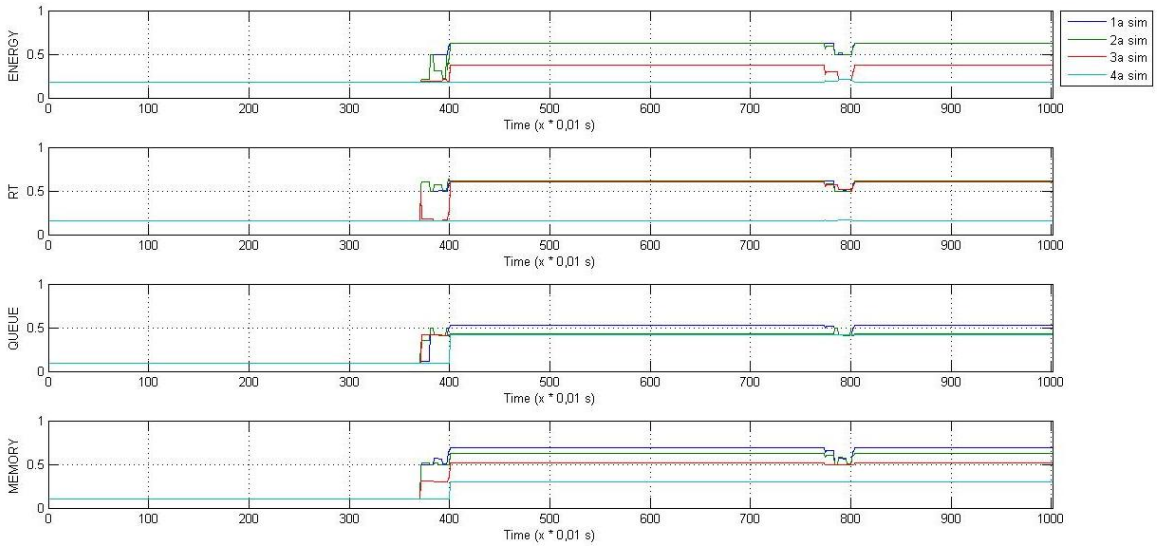
Mesmo com o resultado do terceiro *round*, buscando-se estudar o ACS, foi dada continuidade nas sequências de *rounds*, executando-se outro passo na rede e a seguinte situação foi verificada.

Este procedimento foi adotado para verificar a real influência do CPMod sobre as tomadas de decisão na rede e de que forma estas afetam o comportamento e adequações dos nós sensores.

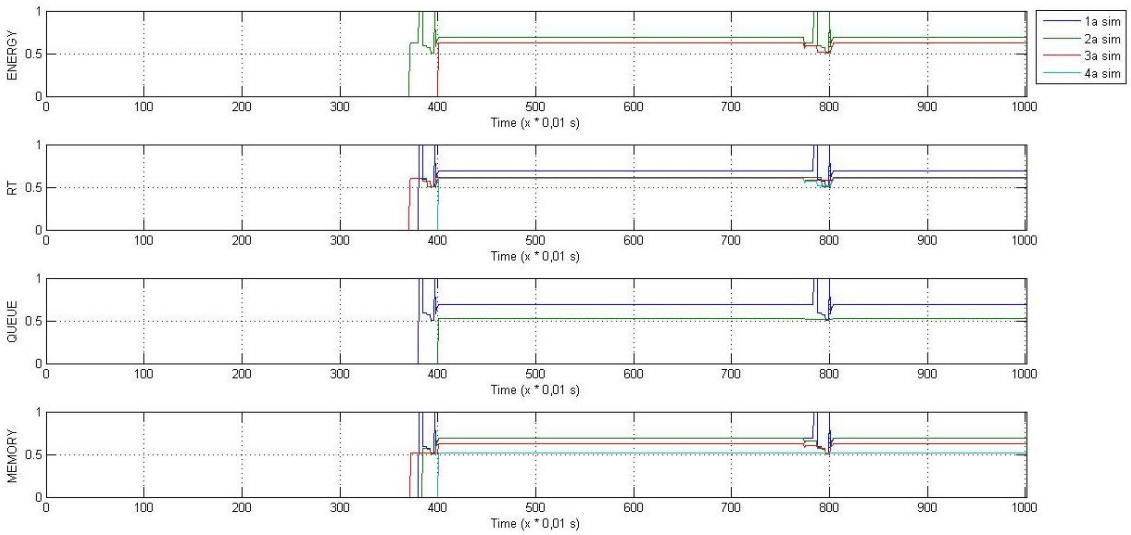
Quando o vetor ENERGY é acionado, então o CPMod realiza ajuste na potência de transmissão, para mais ou menos, conforme a situação apresentada na rede, afetando desta forma a camada física. Da mesma forma, quando o vetor QUEUE é apresentado pelo bloco *Fuzzy* ao CPMod, o módulo executa particionamento de pacotes, melhorando a vazão pelos nós e o tráfego na rede, com atuação na camada de enlace. Com o vetor ROUTE acionado, simboliza ao CPMod

a necessidade por troca de protocolo de roteamento, atuando na camada de rede. Quando ocorre o vetor MEMORY, o CPMod verifica a necessidade e possibilidade do nó sensor disponibilizar mais memória visando à melhoria no processamento de informações pelo nó.

Gráfico 14. Entrada e saída do módulo *Fuzzy* do quarto *round* de testes.



a) Entrada do bloco *Fuzzy* do quarto *round*.



b) Saída do bloco *Fuzzy* do quarto *round*.

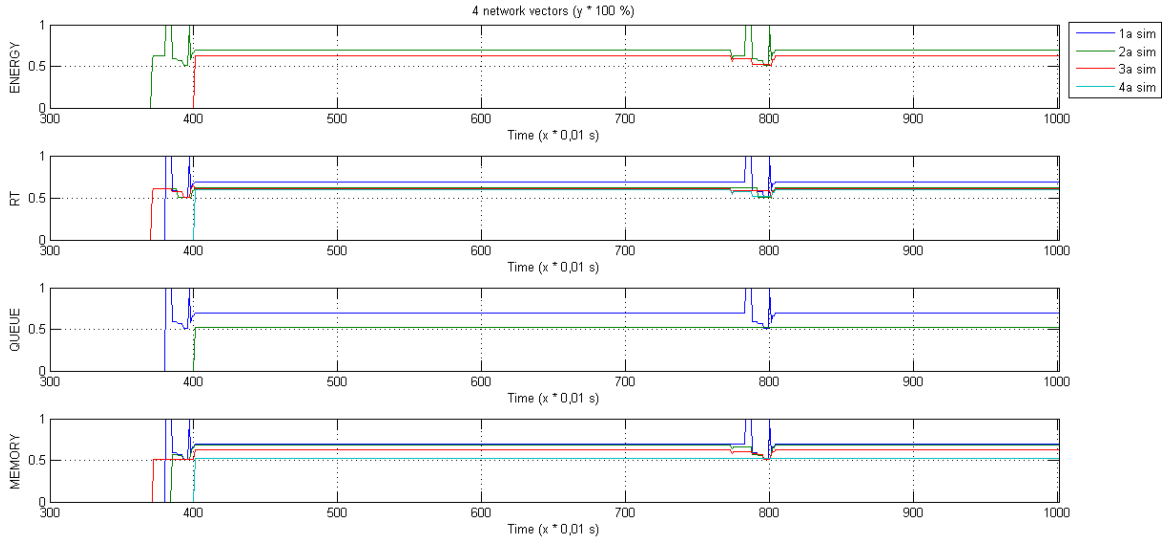
Fonte: Autor.

No Gráfico 14a, pôde-se verificar que com 3 *rounds* de análises executados (3ª simulação), a rede já teve um desempenho melhor quando comparado ao sistema sem aplicação do ACS. Portanto, com a execução de mais um *round*, observa-se que houve um ganho maior, porém com o custo de sobrecarregar a memória do nó com armazenamento desnecessário de histórico e o tempo

desperdiçado na execução de um *round* completo. O Gráfico 14b mostra a influência do bloco *Fuzzy* nas tomadas de decisões sobre os nós na rede.

O Gráfico 15 mostra os detalhes para a entrada do bloco *Fuzzy* presente no modelo desenvolvido, com a utilização do ARP.

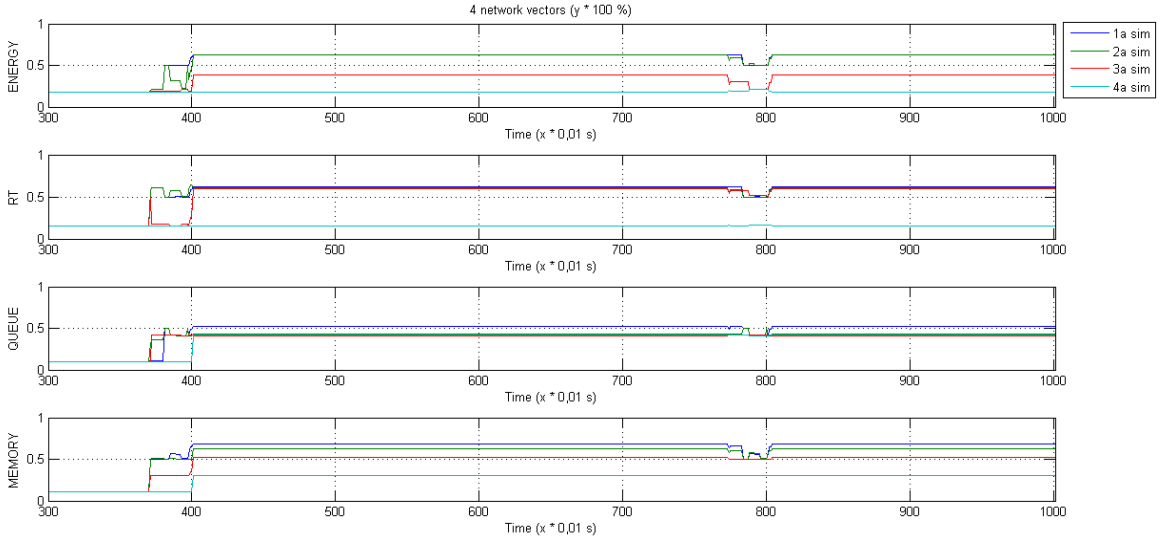
Gráfico 15. Entrada do bloco *Fuzzy* utilizando o ARP.



Fonte: Autor.

O Gráfico 16 mostra os detalhes para a entrada do bloco *Fuzzy* presente no modelo desenvolvido com a utilização do ARP. Nota-se que a atuação é requerida pelo CPMod a partir de 3s de simulação, visto que no bloco DL identificou-se a situação *Urgente*, com base na solução proposta pelo bloco *Fuzzy*.

Gráfico 16. Saída do bloco *Fuzzy* utilizando o ARP.

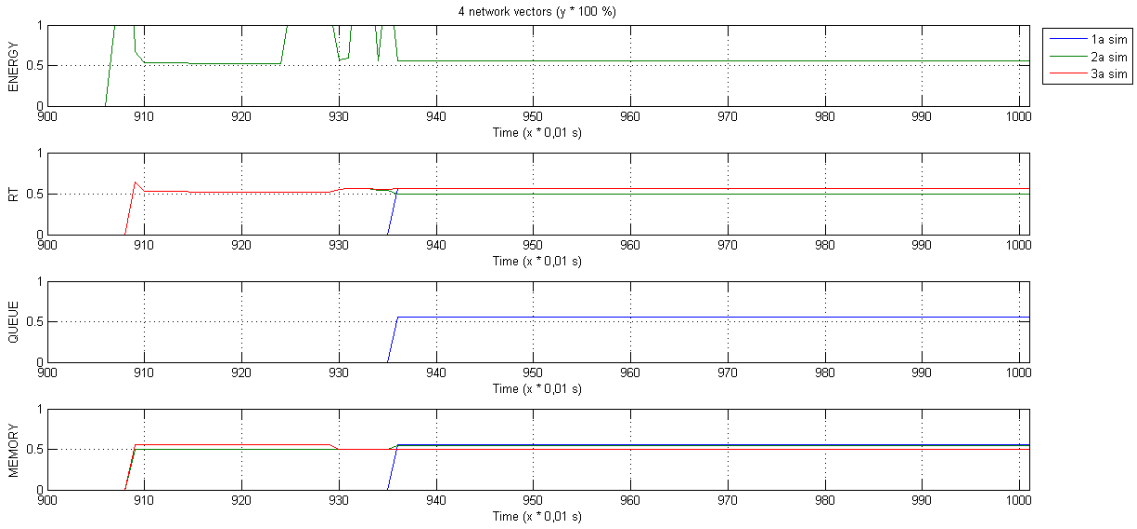


Fonte: Autor.

Como se observa nos Gráficos 15 e 16, conforme as situações foram ocorrendo na rede, o bloco *Fuzzy* definiu melhorias que foram realizadas pelo CPMod de acordo com os *rounds*, melhorando o desempenho na rede.

Os Gráficos 17 a 19 mostram o resultados de análises do ACS com uso do protocolo de roteamento do tipo *spanning tree* (ABAM).

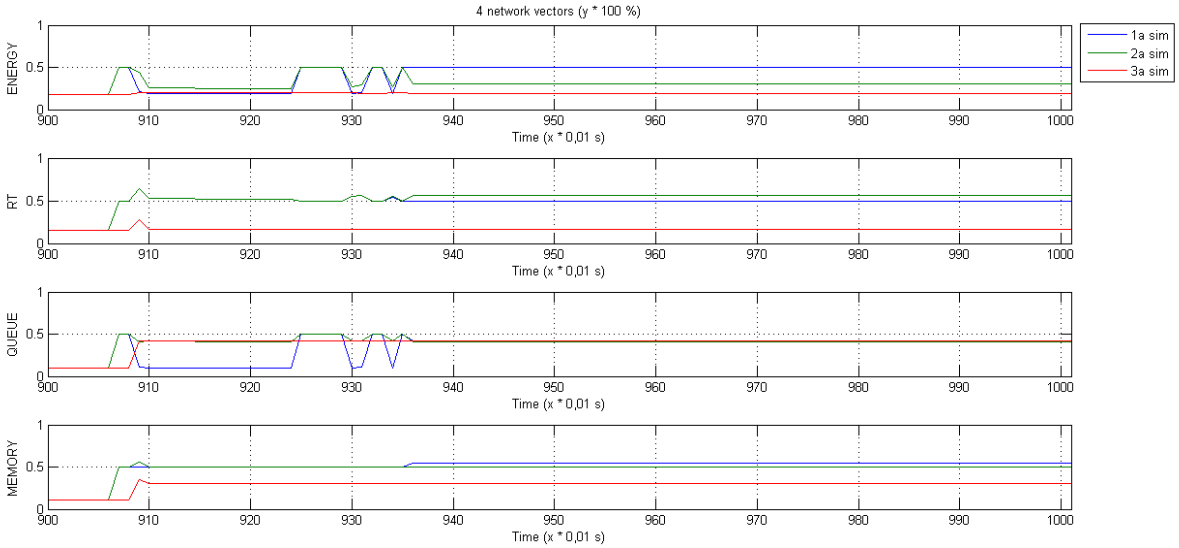
Gráfico 17. Entrada do bloco *Fuzzy* utilizando o ABAM.



Fonte: Autor.

O Gráfico 17 mostra situações de extrema atenção relacionadas à energia consumida pelos nós a partir de 9s de simulação.

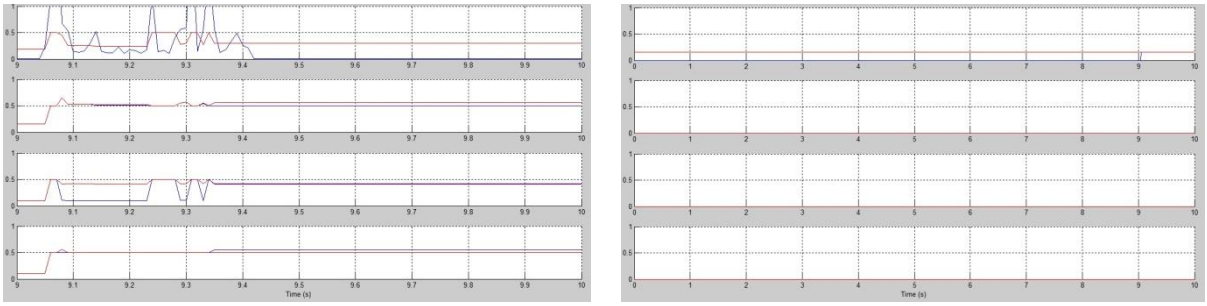
Gráfico 18. Saída do bloco *Fuzzy* utilizando o ABAM.



Fonte: Autor.

O Gráfico 18 apresenta a melhora gradativa no desempenho da rede, conforme foram ocorrendo os *rounds*.

Gráfico 19. Relação de eventos na rede e salvamento de casos sobre o ABAM.



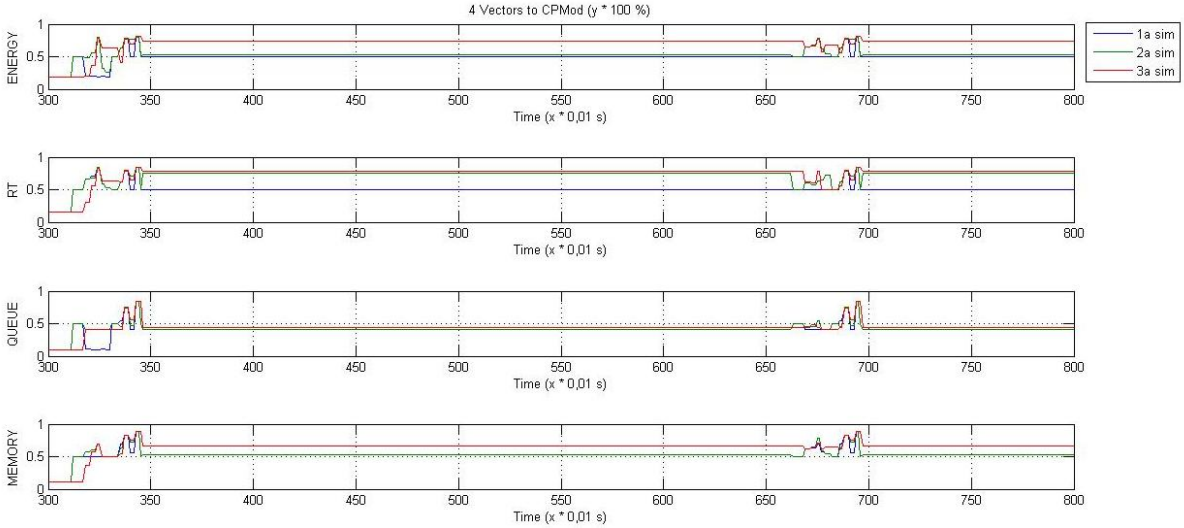
a) Entrada do ACS. b) Energia x Distância em um nó.

Fonte: Autor.

No Gráfico 19 verifica-se que os casos de uso excessivo de energia pelos nós no processo de identificação para a criação da árvore foram documentados na memória episódica.

O Gráfico 20 apresenta o caso de troca de protocolo de roteamento pelo ACS para um cenário de 50 nós, ou seja, a saída do segundo *round* resultou em um $V_C = \{0, 1, 0, 0\}$, indicou a mudança para o CPMod, que por sua vez, fez a troca do protocolo corrente *route angle* pelo *spanning tree*.

Gráfico 20. Saída do bloco *Fuzzy* com troca de protocolos de roteamento.

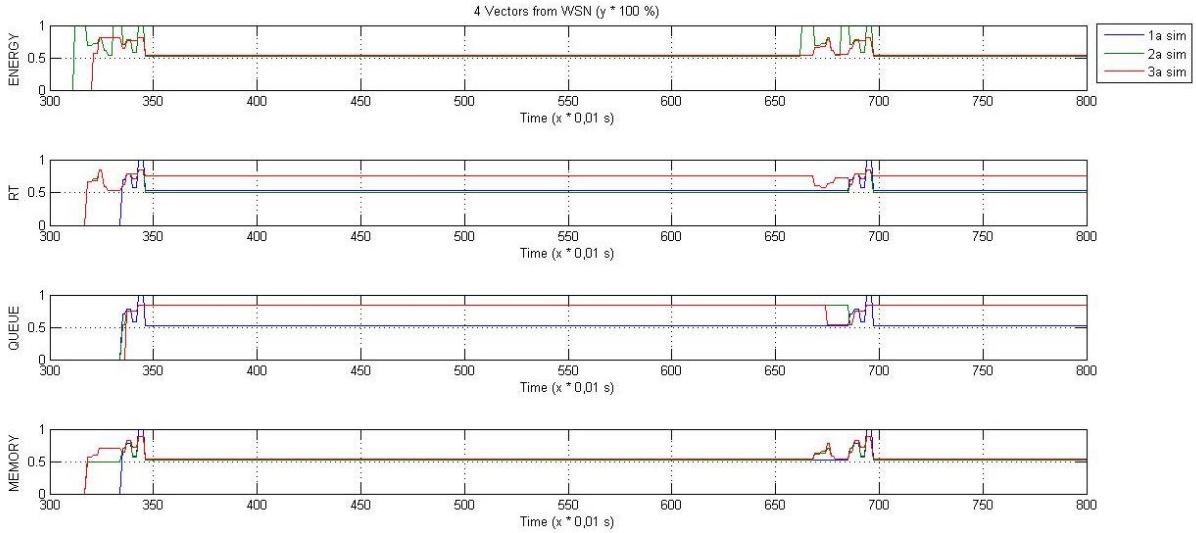


Fonte: Autor.

Percebe-se com clareza no Gráfico 21 que o uso do ABAM neste caso ocasionou numa elevação excessiva de carga de roteamento e das filas de

encaminhamento de pacotes pelos nós da rede.

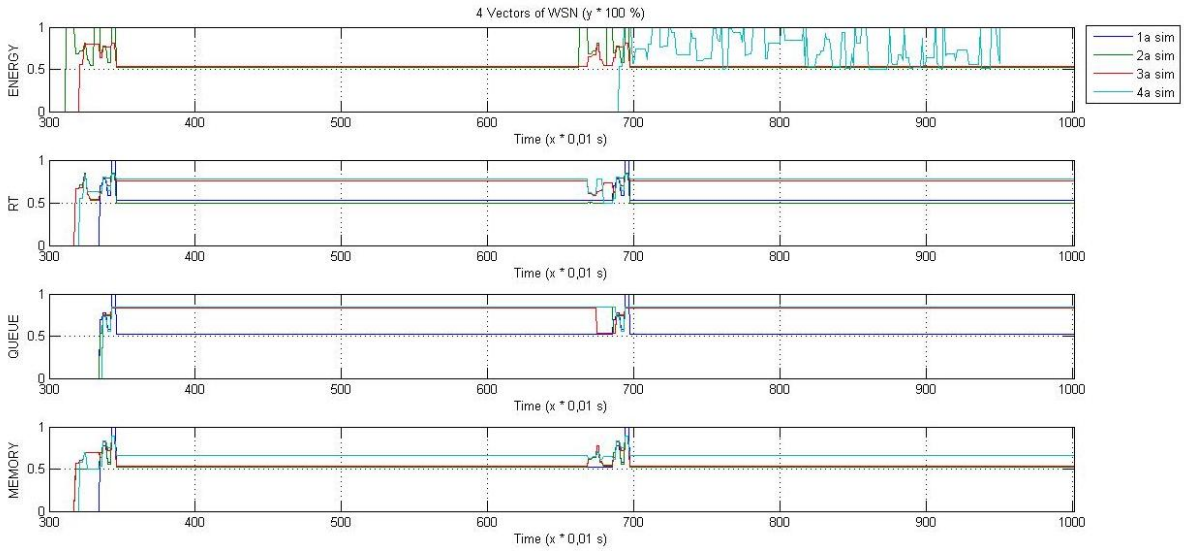
Gráfico 21. Entrada do bloco *Fuzzy* com troca de protocolos de roteamento.



Fonte: Autor.

Com o terceiro *round* do Gráfico 20 identifica-se um $V_C = \{1, 1, 1, 1\}$, pois com a troca feita de protocolos de roteamento, outros problemas foram ocasionados. Portanto com este V_C, retornou-se ao ARP. A impressão é de que retorne a situação anterior ao evento de troca inicial de protocolos, porém o Gráfico 22 mostra que houve gasto de energia desnecessário pelos nós para adequarem-se ao uso do novo protocolo de rede.

Gráfico 22. Implicação no caso de retorno do ABAM para o ARP.

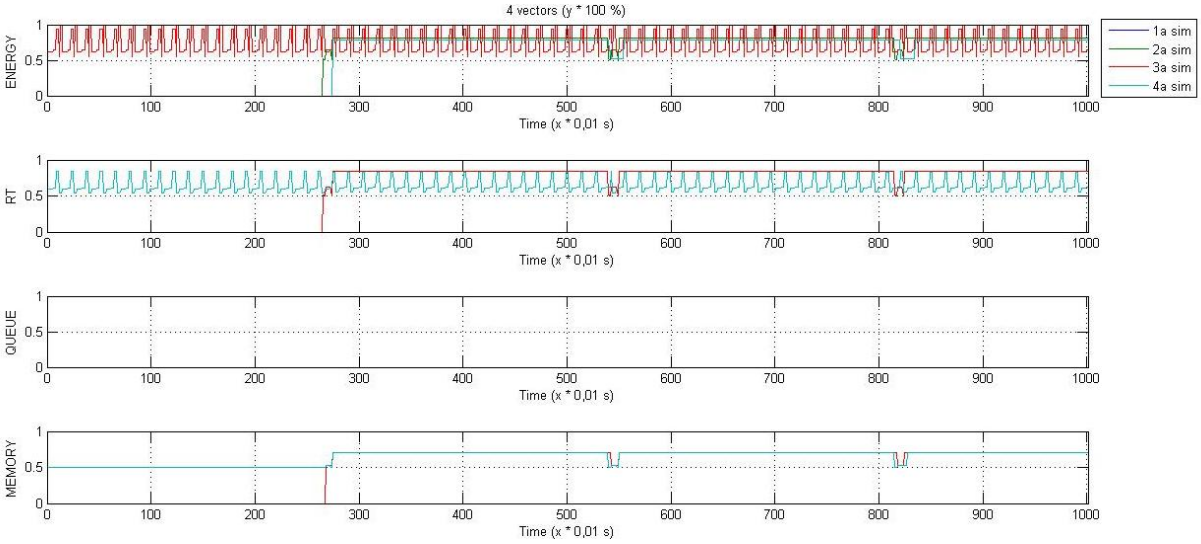


Fonte: Autor.

Estes eventos sobre a troca de protocolos de roteamento foram

documentados, portanto, salvos nos arquivos gerados pelo bloco de memória episódica.

Gráfico 23. Entrada do bloco *Fuzzy* para simulação com uso inicial do ARP para 210 nós.



Fonte: Autor.

Conforme pode-se observar no Gráfico 23, o desempenho para uma quantidade de nós sensores acima de 200 resultou em não eficácia na correção conforme adequação realizada pelo CPMod referente à energia consumida, roteamento e quantidade de energia, tanto para o terceiro como para o quarto *round* do sistema, impactando em falta de acesso às informações relativas ao processamento sobre as filas nos nós.

A arquitetura central do CPMod foi desenvolvida no *Simulink* do MATLAB, incorporando a interligação com o bloco de fuzzificação e defuzzificação, bem como o uso do modelo Mamdani para a inferência e uso de regras *Fuzzy*. A Figura 48 apresenta o diagrama em blocos da Lógica *Fuzzy* elaborada com o modelo Mamdani.

A relação do V_C com o que ocorre na rede foi executada nas iterações dos *rounds* e a Tabela 8 mostra um resumo das métricas analisadas e as correções executadas.

O Gráfico 24 mostra a relação de densidade de alcance dos *motes* MICAz utilizados para as simulações. Identificou-se através da figura que, conforme a quantidade de *motes* aumenta na rede, a área de cobertura tende a alcançar o limite

da rede, sendo referenciada no gráfico como 100 %.

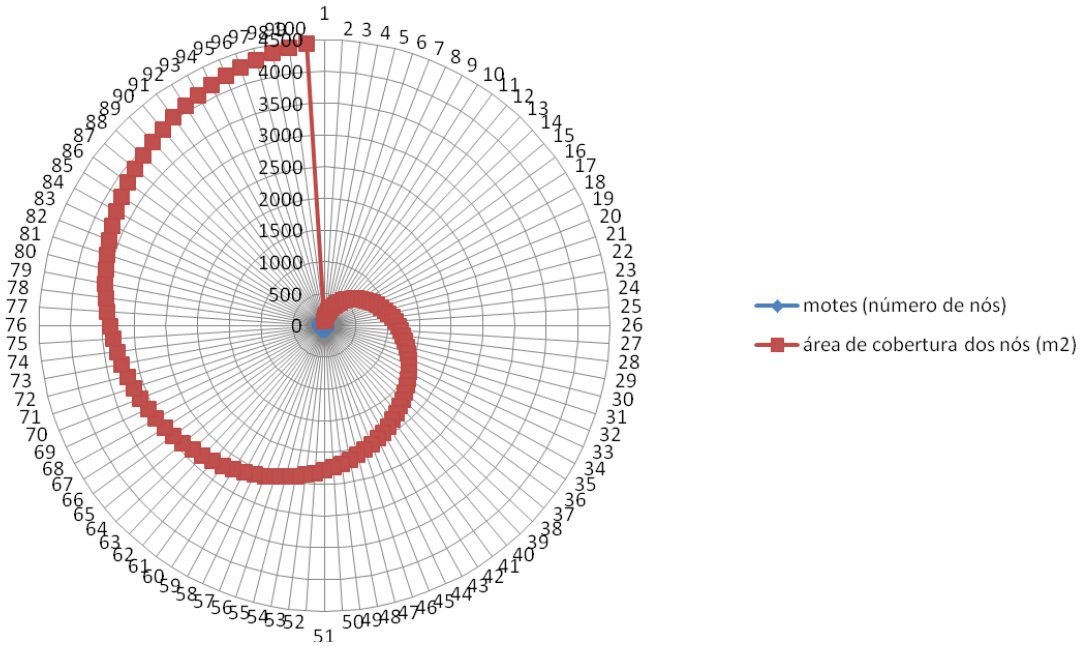
Tabela 8 – Resumo das atribuições do vetor de controle.

Índice	Resumo das atuações	
	Métricas	**Atuação**
V_C	ENERGY	Alteração da potência de transmissão dentro do limite do nó, conforme alcance determinado pelo F_N (conforme limite de recepção do sinal).
	ROUTE	Troca do protocolo de roteamento, devido ao atraso detectado na rede (*delay*). Neste aspecto, utilizou-se a leitura de atraso fim-a-fim.
	QUEUE	Alteração do tamanho do *buffer* disponível para o processamento dos pacotes, devido ao descarte de pacotes na rede, conforme estipulado pelo CPMod.
	MEMORY	Alteração do tamanho do pacote (em divisões de 50% da parte de dados), devido à latência. Com a divisão dos pacotes de dados, melhora a vazão de pacotes na rede.

Fonte: Autor.

Para melhor visualizar a relação de densidade dos nós na rede com relação ao ACS desenvolvido e implementado nesta Tese, o Gráfico 25 traz a relação com a estimação de *motes* pela área de cobertura.

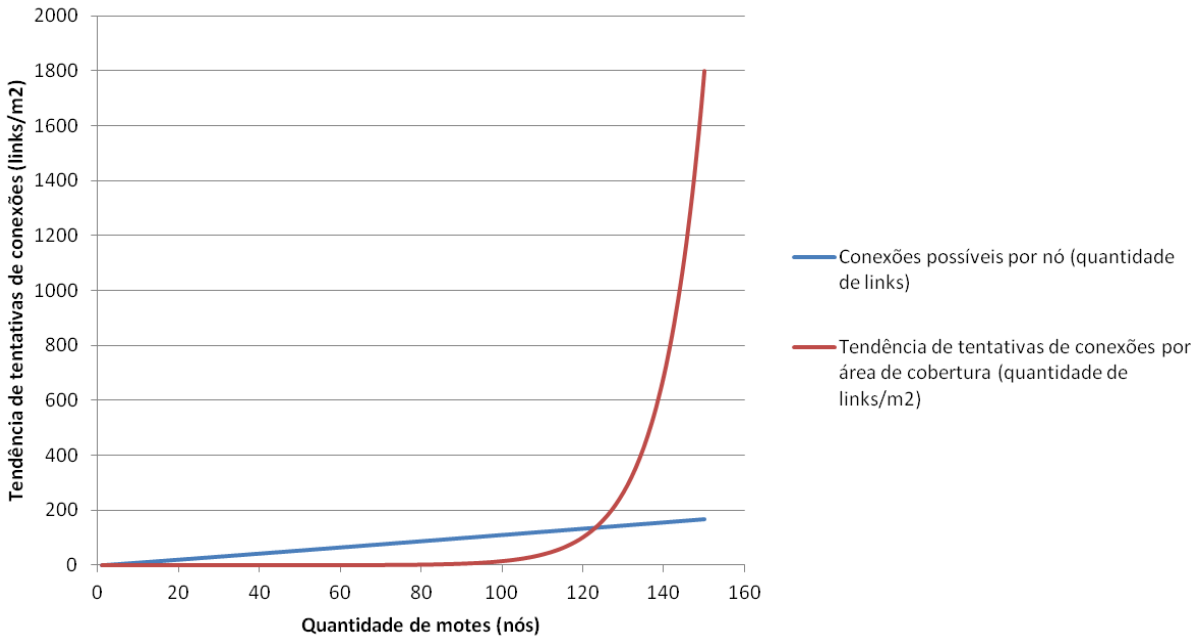
Gráfico 24. Densidade de alcance dos *motes* MICAz.



Fonte: Autor.

A quantidade de possíveis conexões de um nó na rede segue a linha de tendência do Gráfico 25 conforme o acréscimo no número de nós.

Gráfico 25. Tendência da quantidade de conexões estimadas por *mote*.



Fonte: Autor.

Desta forma, a limitação apontada para 200 nós não é uma limitação inerente do sistema desenvolvido nesta Tese, mas sim à limitação inerente do próprio sistema, pois o tamanho dos cenários não foi alterado durante a realização dos testes, como forma de identificar as situações na rede desde uma rede com muitos recursos até uma rede sobrecarregada e com poucos recursos disponíveis.

6. CONCLUSÃO

Com base nos objetivos propostos, foi desenvolvido um algoritmo de detecção de variabilidade dos nós sensores na rede, em que se observou que a variabilidade dos nós sensores afeta diretamente o desempenho da rede, devido à necessidade de reestabelecimento de *links* e rotas entre os nós.

Verificou-se através dos gráficos do algoritmo de detecção que os picos de variabilidade dos nós da rede geralmente atingem máximos de 60% e que mesmo com a variabilidade correspondendo a características individuais de sensores, afeta a rede como um todo, sendo necessário aplicar um sistema de controle e adequação da rede para diferentes cenários, sendo o ACS a solução proposta para atuar em múltiplas camadas de rede.

Atendendo também à proposta do trabalho, foi criado um sistema cognitivo baseado na arquitetura CN proposta por Thomas (2006), que engloba o desenvolvimento de um sistema cognitivo denominado ACS, que por sua vez, inclui blocos de WSN, Lógica *Fuzzy*, semântica, DL e memória episódica, sob o controle de um módulo cognitivo chamado CPMod.

Verificou-se através de simulação em *software* que conforme a execução de *rounds* de análises para detectar a resposta mais próxima à semântica, o sistema ACS se comportou de forma eficaz para redes com até 200 nós, ou seja, com a atuação das medidas adotadas pelo CPMod, conforme as indicações fornecidas pelo resultado da inferência *Fuzzy*. O ACS demonstrou potencial para a solução de problemas relacionados com as métricas de atraso fim-a-fim, latência, quantidade de pacotes descartados e de energia consumida pelos nós na rede, realizando ajustes em múltiplas camadas de rede do padrão IEEE 802.15.4.

A limitação apontada para 200 nós não é uma limitação inerente do sistema desenvolvido nesta Tese, mas sim à limitação inerente do próprio sistema, pois o tamanho dos cenários não foi alterado durante os testes realizados a fim de identificar as situações na rede desde uma rede com muitos recursos até uma rede sobrecarregada e com poucos recursos disponíveis. Portanto, com o aumento de nós e mantendo-se o tamanho dos cenários em 800×500 (m^2), houve um aumento

de densidade dos nós por unidade de área conforme cada incremento de nós na rede, ocasionando, desta forma, maior envio de pacotes de dados, que por conseguinte gerou maior descarte de pacotes, mesmo com medidas de adequação de memória (pois existe limitação deste recurso nos nós), aumentando assim, as filas de processamento, descarte de pacotes e congestionamento na rede. Esta situação se torna cíclica, o que significa que independentemente da quantidade de *rounds* executados ou mesmo de lembranças da memória episódica, a rede encontra-se em uma situação que mesmo com a atuação do ACS/CPMod/VC, os ajustes não proporcionarão melhoria, pois a rede como um todo já está em uma situação “irreversível” (rede travada).

A mobilidade também influenciou para a limitação de 200 nós na rede. Para a correção do cenário “irreversível” exposto, deve-se alocar melhor os recursos da rede, dimensionar a rede para um índice de densidade que contemple atender à velocidade de processamento nos nós, buffer dos nós para a recepção e colocação de pacotes em filas de processamento e quantidade de memória dos nós.

Com os resultados dos testes, e baseando-se nos gráficos obtidos, verificou-se que o sistema realiza melhora do desempenho da rede com a execução dos *rounds*, como por exemplo, o resultado do Gráfico 14 apresenta melhorias quanto ao consumo de energia, roteamento de dados e uso de memória. Identificou-se também, que a troca de protocolos de roteamento nem sempre é benéfica para o desempenho da rede, como no caso do Gráfico 21, que na troca do ARP pelo ABAM, gerou excessiva carga de roteamento e de filas de encaminhamento de pacotes pelos nós da rede.

O sistema é viável do ponto de vista da rede, pois o ACS consegue realizar a melhoria do desempenho da rede relacionado com o consumo de energia dos nós, a identificação do melhor protocolo de rede a ser utilizado conforme a situação da mesma, à métrica de atraso fim-a-fim e à latência com aumento da vazão de pacotes pelos nós intermediários.

O mecanismo proposto e apresentado neste trabalho pode vir a ser inserido em dispositivos IoT ou em aplicações de dispositivos para *smart-cities*, com a adaptação do ACS/CPMod para operação em nós distintos presentes na WSN e não

somente como base de análise de tráfego na Camada de Aplicação.

Parte deste trabalho relacionado com o desenvolvimento matemático foi apresentada no IEEE *Fifth International Conference on Communication Systems and Network Technologies* (CSNT) 2015, 4 a 6 de abril de 2015, em Gwalior, na Índia, com o *paper* intitulado *Adaptive Cognitive System Applied on Wireless Sensor Networks Nodes Decisions* (WAGNER, 2015) e está registrado no IEEEXplore através do DOI 10.1109/CSNT.2015.65.

Parte deste trabalho relacionado à implementação em Lógica *Fuzzy* do mecanismo desenvolvido foi publicada no *International Journal of Communication System and Network Technology* (IJCSNT), vol. 5, *issue* 01, em fevereiro de 2016, com publicação *on-line*, com o título *Adaptive Cognitive System Applied to WSN Decisions at Nodes with a Fuzzy Logic Approach*, e está registrado através do DOI 10.18486/ijcsnt.2016.5.1.01.

Trabalhos futuros:

- Inclusão de outros tipos de protocolos de roteamento para a simulação e verificação de reação da rede com relação ao QoS e em compatibilidade com a WSN;

Dentre todos os protocolos apresentados, foi escolhido o ARP (ângulo) e o ABAM (árvore) para a simulação, pois para analisar o fluxo de dados na rede, bem como a criação e armazenamento de tabelas de roteamento, estes protocolos são interessantes do ponto de vista de conhecimento de nós vizinhos e *flooding* na rede.

Os protocolos de WSN convencionais (AODV, DSDV, OLSR, DSR e TORA) poderão ser testados neste tipo de análise, para a verificação do desempenho da rede em comparação aos que fazem estimativas, previsões ou mesmo uso de outras técnicas diferenciadas como, por exemplo, posicionamento, colônia de formigas ou uso de treliças (GLS, SFN-DCRP, ETAMR, TC-ACO, DD, RVM, DCN, TCNet e RPL).

Neste ponto, o uso do RPL auxiliaria na verificação de eficiência entre a troca de protocolos de roteamento dentro de um mesmo perfil, ou seja, com análise sobre

protocolos do tipo *spanning tree*.

- Inserção de um módulo para controle específico de mobilidade na rede;

Dada a estrutura do ACS, um bloco específico para tratar da leitura e controle da mobilidade dos nós seria bom, no sentido de descentralizar este tipo de análise do CPMod.

- Realizar a implementação em HW do sistema desenvolvido nesta Tese, podendo-se utilizar os MICA *motes* descritos anteriormente;

Foi utilizado o padrão IEEE 802.15.4, para o qual foi desenvolvido um sistema de controle e atuação nos nós, portanto, a implementação em *hardware* do ACS/CPMod utilizando os *motes* citados torna-se interessante do ponto de vista acadêmico e comercial.

Neste ponto a análise de viabilidade de uso do ACS/CPMod com descentralização é interessante, pois possibilitaria o processamento e controle distribuídos.

- Inclusão de análise de *rounds* envolvendo a camada de transporte, para sistemas baseados no padrão 802.11;

Uma vez que as simulações desta Tese ocorreram para nós dentro da padronização IEEE 802.15.4, o teste do mesmo mecanismo para diferentes padrões torna-se interessante do ponto de vista acadêmico.

- Realizar alterações no ACS para atender redes com maior densidade de nós;

Como já é de conhecimento, as redes simuladas nesta Tese atingiram até a quantidade de 200 nós, portanto, deixando aberta a oportunidade para a continuidade do trabalho no sentido de acréscimo de mais nós na rede e alterações nos tamanhos dos cenários.

- Verificar os testes para modificação na topologia;

Utilizar diferentes topologias para a análise do ACS/CPMod torna-se

interessante, pois desta forma pode-se obter resultados específicos conforme cada cenário. Outro ponto a ser abordado, seria a implementação em 3 dimensões, uma vez que o sistema criado nesta Tese tem característica bidimensional.

- Realização de testes em ambiente real com configuração IoT;

Neste contexto, seria interessante agregar não somente a conexão entre os nós da rede, de forma reativa como ocorre hoje, e sim com a inserção de “inteligência” na rede, com múltiplas possibilidades de encaminhamento e tratamento de dados, para auxiliar nas DM pelos nós da rede.

- Aplicação de Redes Neurais para a parte inteligente do ACS/CPMod.

A aplicação de conceitos ou mesmo implementação de Redes Neurais para representar o ACS/CPMod é um tema muito importante e que deve ser explorado do ponto de vista de desempenho da rede, podendo até ser confrontado com os resultados obtidos nesta Tese.

REFERÊNCIAS

ABI RESEARCH. More than 30 billion devices will wirelessly connect to the internet of everything in 2020. Disponível em: <https://www.abiresearch.com/press/more-than-30-billion-devices-will-wirelessly-conne>. Acesso em: 24 nov. 2013.

ALCARAZ, C. et al. Wireless sensor networks and the internet of things: do we need a complete integration? Computer Science Department of University of Malaga. In: INTERNATIONAL WORKSHOP ON THE SECURITY OF THE INTERNET OF THINGS (SecIoT'10), 1.Tokyo. **Anais…** Tokyo, Japan, 2010.

ANATEL. Agência Nacional de Telecomunicações. **Resolução nº 614**, 28/05/2013, que aprova o Regulamento do Serviço de Comunicação Multimídia e altera os Anexos I e III do Regulamento de Cobrança de Preço Público pelo Direito de Exploração de Serviços de Telecomunicações e pelo Direito de Exploração de Satélite. Brasília/DF, 2013. Disponível em: <http://www.anatel.gov.br/legislacao/resolucoes/2013/465-resolucao-614>. Acesso em: 09 nov. 2013.

ANDRADE, M. e JAQUES, M. A. P. Estudo comparativo de controladores de Mamdani e Sugeno para controle de tráfego em interseções isoladas. **Revista Transportes**. ANPET – Associação Nacional de Pesquisa e Ensino em Transportes. v. 16, n. 2, Dezembro, 2008. Disponível em: <http://www.revistatransportes.org.br/anpet/article/view/24>. Acesso em: 19 jan. 2016.

AODV. Ad hoc On-demand Distance Vector Routing. Freie Universität Berlin. Disponível em: http://www.des-testbed.net/content/ad-hoc-demand-distance-vector-routing-aodv**.** Acesso em: 02 set. 2013.

ARAÚJO, A. et al. Security in cognitive wireless sensor networks. Challenges and open problems. **Journal on Wireless Communications and Networking**. EURASIP. Madrid, Spain: 2012. Disponível em: <http://www.jwcn.eurasipjournals.com/content/2012/1/48>. Acesso em: 09 jan. 2016.

ARSLAN, H. **Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems**: Signals and communication technology. Dordrecht, The Netherlands: Springer, 2007.

ASHTON, K. That 'internet of things' thing, in the real world things matter more than ideas. **RFID Journal**. June, 2009. Disponível em: <http://www.rfidjournal.com/articles/pdf?4986>. Acesso em: 24 nov. 2013.

BAJAJ, S. et al. Improving simulation for network research. USC Computer Science Department. **Technical Report 99-702b**. University of Southern California. March, 1999.

BAKER, M. R. A Survey of Multicast Routing Protocols in Ad-Hoc Networks. Gazi **University Journal of Science**. p. 451-462. December, 2010. Disponível em: <https://www.researchgate.net/publication/281176358_A_Survey_of_Multicast_ Rou ting_Protocols_in_Ad-Hoc_Networks>. Acesso em: 17 jun. 2016.

BANKA, R. K. and XUE, G. Angle Routing Protocol: Location Aided Routing for Mobile Ad-Hoc Networks Using Dynamic Angle Selection. In: PROCEEDINGS OF IEEE MILITARY COMMUNICATIONS CONFERENCE MILCOM. USA, Anaheim, **Proceedings…** USA, Anaheim, October, 2002.

BASAGNI, S. et al. A Distance Routing Effect Algorithm for Mobility (DREAM). In: PROCEEDINGS OF INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING MobiCom'98. Dallas, Texas, **Proceedings…** Dallas, Texas, USA, 1998.

BERKELEY. Network Embedded Systems Technology (NEST) research Project at UC Berkeley Website. Disponível em: <http://nest.cs.berkeley.edu/related.html>. Acesso em: 08 de julho de 2015.

BHANU, B. et al. **Distributed Video Sensor Networks**. London, England: Springer, 2011.

BISC. Berkeley Initiative in Soft Computing (BISC). Disponível em: <http://www.cs.berkeley.edu/~zadeh/>. Acesso em: 08 de janeiro de 2016.

BLUETOOTH. Bluetooth Website. Disponível em: <https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth>. Acesso em: 08 de julho de 2015.

BOOTH, A. Dynamic Systems Theory. (Presentation), 30 July 2012. Disponível em: <https://www.youtube.com/watch?v=qyOQyw7ws-c>. Acesso em: 14 jan. 2016.

BRASIL. Presidência da República, Casa Civil, Subchefia para assuntos jurídicos. **Decreto n. 6.654,** 20/11/2008, que Aprova o Plano Geral de Outorgas de Serviço de Telecomunicações prestado no regime público. Brasília/DF, 2008. Disponível em: <http://www.planalto.gov.br/ccivil_03/_Ato2007-2010/2008/Decreto/D6654.htm>. Acesso em: 17 dez. 2008.

BRASILWORK. O que é latência em redes? Disponível em:<https://brasilwork.com. br/duvidas/e-latencia-em-redes/>. Acesso em: 15 jun. 2016.

CAMPOS, L. B. Camada de rede – roteamento. **Notas de aula** (Redes-I). Universidade Federal do Vale do São Francisco – UNIVASF, agosto, 2008.

CANDYCAT. Candycat blog site. Disponível em: <http://blog.csdn.net/candycat1992/ article/details/8087255>. Acesso em: 04 jan. 2016.

CHAKRABORTY, A. et al. A Trust Based Congestion Aware Hybrid Ant Colony Optimization Algorithm for Energy Efficient Routing in Wireless Sensor Networks (TC-ACO). In: INTERNATIONAL CONFERENCE ON ADVANCED COMPUTING, 5. Chennai, India. **Conference...** ICoAC'2013. p. 137-142, Chennai, India, December, 2013.

CHAPMAN, S. J. **Programação em MATLAB para engenheiros**. 2ª ed. São Paulo: Cengage Learning, 2010.

CISCO. Entendendo e configurando o Spanning Tree Protocol (STP) em switches catalyst. Disponível em: <http://www.cisco.com/cisco/web/support/BR/8/82/82594_5.html>. Acesso em: 29 de dezembro de 2015.

DARGIE, W. and POELLABAUER, C. **Fundamentals of wireless sensor networks**: theory and practice. Chichester, United Kingdom: John Wiley and Sons, 2010.

DORIGO, M. and SOCHA, K. An introduction to ant colony optimization. **IRIDIA – Technical Report Series**. Université Libre de Bruxelles, Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, TR/IRIDIA/2006-010, n. 010, April, 2006. Disponível em: <http://iridia.ulb.ac.be/~mdorigo/ACO/publications.html>. Acesso em: 15 jan. 2016.

DSR. Dynamic Source Routing. Freie Universität Berlin. Disponível em: <http://www.des-testbed.net/content/dynamic-source-routing-dsr>. Acesso em: 21 ago. 2015.

ERIKSSON, M.; ARIF, M. Dynamic single frequency networks in wireless multihop networks - energy aware routing algorithms with performance analysis. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION TECHNOLOGY, 10. **Conference…** (CIT'2010). p. 400-406, Bradford, July, 2010.

EZREIK, A. and GHERYANI, A. Design and simulation of wireless network using NS-2. In: INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE AND INFORMATION TECHNOLOGY (ICCSIT'2012), 2. Singapore. **Conference…** Singapore, April, 2012.

FERNÁNDEZ, E. B. et. al. Some security issues of wireless systems. In: INTERNATIONAL SCHOOL AND SYMPOSIUM ON ADVANCED DISTRIBUTED SYSTEMS – ISSADS'2005, 5. LNCS 3563. Guadalajara, **Anais…** p. 388-396, jan. 2005. Revised Selected Papers. Disponível em: <http://link.springer.com/chapter/10.1007%2F11533962_35>. Acesso em: 07 jan. 2016.

FRACKOWIAK, R. and MARKRAM, H. The future of human cerebral cartography: a novel approach. Philosophical **Transactions** B Royal Society Publishing. January, 2015.

GEORGEFF, M. et al. The Belief-Desire-Intention Model of Agency. In: INTERNATIONAL WORKSHOP ON INTELLIGENT AGENTS V, AGENT THEORIES, ARCHITECTURES, AND LANGUAGES, 5. **Proceedings…** ATAL'98. p. 1-10. Springer-Verlag: London, UK, 1999.

GOH, H. G. et al. CogSeNet: A Concept of Cognitive Wireless Sensor Network. In: IEEE CONSUMER COMMUNICATIONS AND NETWORKING CONFERENCE, 7. **Conference…** (CCNC'2010). p. 1-2, Las Vegas, NV, January, 2010.

HANSELMAN, D. e LITTLEFIELD, B. **MATLAB 6**: curso completo. São Paulo: Prentice Hall, 2003.

IDTECHEX. IDTechEx – Internet of Things and WSN USA 2013. Disponível em: <http://www.idtechex.com/wireless-sensor-networks-usa-13/home.asp>. Acesso em: 24 nov. 2013.

IEEE. Standard 802.15.4 – Standard for Local and metropolitan area networks – Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Computer Society. 5 September 2011. Disponível em: <https://standards.ieee.org/about/get/802/802.15.html>. Acesso em: 31 jan. 2016.

IEEE. Institute of Electrical and Electronics Engineers. Disponível em: <http://www.ieee.org.br/>. Acesso em: 02 fev. 2013.

INFOWESTER. InfoWester Website. Disponível em: <http://www.infowester.com/wifi.php>. Acesso em: 07 de julho de 2014.

INTANAGONWIWAT, S.; GOVINDAN, R. and ESTRIN, D. Directed Diffusion**:** a scalable and robust communication paradigm for sensor networks. In: INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING. **Proceedings...** MobiCom 2000. Boston, Aug, 2000.

IOT BRASIL. Site IoT Brasil. Disponível em: <http://www.iotbrasil.com.br/new/informacoes-sobre-o-conteudo-do-seminario-do-forum-de-5-set-2014-sao-paulo-iot-aplicada-na-melhoria-de-vida-nas-cidades/>. Acesso em: 07 out. de 2014.

JACINTO, R. M. P. Modelação da propagação numa rede de sensores sem fios. Dissertação de Mestrado. Faculdade de ciências e tecnologia. Universidade Nova de Lisboa. 108p. Março, 2012. Disponível em:<https://run.unl.pt/bitstream/10362/8162/1/Jacinto_2012.pdf>. Acesso em: 20 mai. de 2016.

JAMAJ. JamaJ Electronics – Jamaj Automation, Robotics: Robótica, microcontroladores, automação. Disponível em: <https://sites.google.com/site/jamajautomationrobotics/>. Acesso em: 15 out. 2013.

JOHNSON, T. Wireless BR. Disponível em: <http://www.wirelessbrasil.org/ wirelessbr/blog_sensores/sensores_01.html>. Acesso em: 15 out. 2013.

JUCHEM, M. e BASTOS, R. M. Arquiteturas de agentes. **Technical Report Series**. Faculdade de Informática da PUC-RS, n. 013, Abril, 2001. Disponível em: <http://www3.pucrs.br/pucrs/files/uni/poa/facin/pos/relatoriostec/tr013.pdf>. Acesso em: 03 jan. 2016.

KO, Y-B.; VAIDYA, N. H. Location-Aided Routing (LAR) in mobile ad hoc networks. **Journal Wireless Networks**. Springer-Verlag, New York, USA. v. 6, n. 4, p. 307–321, July, 2000.

KORKALAINEN, M. et al. Survey of wireless sensor networks simulation tools for demanding applications. In: INTERNATIONAL CONFERENCE ON NETWORKING AND SERVICES, 5. **Conference…** (ICNS'2009). p. 102-106, Valencia, Spain, April, 2009.

KRISTIANSEN, G. et al. **Cognitive linguistics**: current applications and future perspectives. Berlin, Germany: Mouton de Gruyter, 2006.

KROPOTOFF, A. B. Um emulador paramétrico de conexões fim-a-fim em redes IP. Dissertação de Mestrado. Universidade Federal do Rio de Janeiro. 116p. Abril, 2002. Disponível em:<http://www.ravel.ufrj.br/emulador-parametrico-conexoes-fim-fim-em-redes-ip>. Acesso em: 25 mai. de 2016.

LANGLEY, P.; LAIRD, J. E. and ROGERS, S. Cognitive architectures: research issues and challenges. **Journal of Cognitive Systems Research**. v. 10, n. 2, p. 141–160, Elsevier, Amsterdam, The Netherlands, June, 2009. Disponível em: <http://dl.acm.org/citation.cfm?id=2298716>. Acesso em: 08 jan. 2016.

LEE, C. C. Fuzzy logic in control systems: fuzzy logic controller – Part I. **IEEE Transactions on Systems, Man and Cybernetics**. v. 20, n. 2, p. 404-418, March/April, 1990.

LEE, C. C. Fuzzy logic in control systems: fuzzy logic controller – Part II. **IEEE Transactions on Systems, Man and Cybernetics**. v. 20, n. 2, p. 419-435, March/April, 1990.

LEOPOLD, M. **Network sensor motes**: portability & performance. 2007. 123p. Ph.D. dissertation. Department of Computer Science, Faculty of Science, University of Copenhagen, Copenhagen, December, 2007.

LI, J. et al. A Scalable Location Service for Geographic Ad Hoc Routing. M.I.T. Laboratory for Computer Science. In: ANNUAL INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING MobiCom, 6. **Conference…** 2000. ACM, New York, USA. p. 120-130, 2000.

LI, J. et al. Efficient Traffic Aware Multipath Routing Algorithm in Cognitive Networks. INTERNATIONAL CONFERENCE ON GENETIC AND EVOLUTIONARY COMPUTING, 5. (ICGEC). **Conference…** Xiamen, September, 2011.

LIM, K. H. **Performance Enhancement of the Temporally-Ordered Routing Algorithm**. 2007. 67p. Honour Thesis. School of Computer Science and Software Engineering. The University of Western Australia. Crawley, Australia, 2007.

LIMA FILHO, D. F. **Projeto e implementação de um novo algoritmo e protocolo de encaminhamento de pacotes baseado em códigos convolucionais usando TCNet: Trellis Coded Network**. 2015. 116p. Tese (Doutorado). Sistemas Eletrônicos – Escola Politécnica, Universidade de São Paulo, São Paulo, 2015.

LOBO NETTO, M. e RINALDI, L. C. A. Vida Artificial: conceitos e aplicações. Universidade de São Paulo, USP. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 10. Anais... São João Del-Rei, MG, 2012. Disponível em: <https://scholar.google.com.br/scholar?oi=bibs&cluster=15058287230455199654&btnI=1&hl=pt-BR>. Acesso em: 14 jun. 2016.

LOUREIRO, A. A. F. et al. Redes de sensores sem fio. Universidade Federal de Minas Gerais, UFMG. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, 21. **Anais...** Natal, CE, 2003. Disponível em: <www.dcc.ufmg.br/~loureiro/cm/docs/sbrc03.pdf>. Acesso em: 09 jan. 2016.

MALVEZZI, W. R. **Uma ferramenta baseada em teoria Fuzzy para o acompanhamento de alunos aplicado ao modelo de educação presencial mediado por tecnologia**. 2010. 123p. Dissertação (Mestrado). Sistemas Digitais – Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

MATSUMOTO, E. Y. **MATLAB 6.5**: fundamentos de programação. São Paulo: Érica, 2002.

MENDONÇA, M. et al. Redes Cognitivas Dinâmicas, Fundamentos e Aplicações. **Revista Semina: Ciências Exatas e Tecnológicas**. Londrina, v. 34, n. 2, p. 245-260, Julho, 2013.

MITOLA III, J. **Cognitive radio:** an integrated agent architecture for software defined radio. Ph.D. Dissertation (Doctor of Technology). Royal Institute of Technology. Kista, Sweden, 2000.

MITOLA III, J. **Cognitive radio architecture:** the engineering foundations of radio XML. New Jersey, USA: John Wiley and Sons, September, 2006.

MÜLLER, J. P.; SINGH, M. P. and RAO, A. S. Intelligent Agents V: Agent Theories, Architectures and Languages. **Lecture Notes in Computer Science**, v. 1555. Berlin,

Germany: Springer-Verlag, 1999.

MUNEER, S. **Design and Evaluation of SFN based Distributed Cooperative Routing Protocol for Multi-hop Ad-hoc Networks**. 2013. 53p. M.Sc. Thesis. Computer Engineering. The Department of Information Technology and Media (ITM). Mid Sweden University. Sweden, 2013.

NARRA, H. et al. Destination-Sequenced Distance Vector (DSDV) routing protocol implementation in ns-3. In: PROCEEDINGS OF THE 4TH INTERNATIONAL ICST CONFERENCE ON SIMULATION TOOLS AND TECHNIQUES, 4. SIMUTools'11. **Proceedings...** Brussels, Belgium, p. 439-446, ICST, Brussels, Belgium, 2011.

NASCIMENTO JR., C. L. e YONEYAMA, T. **Inteligência Artificial em Controle e Automação**. São Paulo: Edgard Blücher, 2000.

NS-2. **Network Simulator** – NS-2. Disponível em: <http://www.isi.edu/nsnam/ns/>. Acesso em: 01 set. 2013.

OMNET. **Software de simulação de redes** – OMNET++ – Discrete Event Simulator. Disponível em: <https://omnetpp.org/>. Acesso em: 24 dez. 2013.

OPNET. **Software de Simulação de Redes** – Riverbed application and network performance management solutions. Disponível em: <http://www.opnet.com/ university_program/itguru_academic_edition/>. Acesso em: 01 dez. 2012.

OPENCADD. MATLAB Software – OPENCADD Advanced Technology Consulting Group. Disponível em: <http://opencadd.com.br/produto/mathworks/>. Acesso em: 01 out. 2015.

PALMER, G. B. Talking about thinking in Tagalog. **Cognitive Linguistics Journal**. v. 14, n. 2-3, p. 251–280, Walter de Gruyter, Berlin, 2003. Disponível em: <http://www.degruyter.com/view/j/cogl. 2003.14.issue-2-3/cogl.2003.010/cogl.2003. 010.xml>. Acesso em: 11 out. 2015.

PERKINS, C. E. and BHAGWAT, P. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In: PROCEEDINGS OF FLAGSHIP ANNUAL CONFERENCE OF THE SPECIAL INTEREST GROUP ON DATA COMMUNICATION SIGCOMM'94. **Conference…** London, England, UK, 1994.

PERKINS, C. E. and ROYER, E. M. Ad-hoc on-demand distance vector routing. In: IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS, 2. (WMCSA '99). New Orleans, LA, **Proceedings…** p. 90-100, New Orleans, LA, 1999.

PORTALACTION. Portal Action – Distribuição de Rayleigh. Disponível em: <http://www.portalaction.com.br/probabilidades/619-distribuicao-de-rayleigh>. Acesso em: 25 jan. 2016.

PROAKIS, J. and SALEHI, M. **Digital Communications**. 5th ed. New York: McGraw Hill, 2008.

PROWLER. **Software em MATLAB**. Disponível em: <http://www.isis.vanderbilt.edu/ projects/nest/prowler/>. Acesso em: 10 fev. 2015.

QUADRI, I. R. Performance Evaluation of Routing Algorithms in Wireless Mobile Ad-Hoc Networks. Wireless and Mobile Networks. Disponível em: <http://faculty.kfupm. edu.sa/COE/ashraf/RichFilesTeaching/COE022_543/Projects/>. Acesso em: 07 jan. 2016.

QUALITÄT. Qualität Systems – embedded Systems, VLSI & software. Disponível em: <http://qualitatsystems.com/SuccessStories/WN%20AODV.aspx>. Acesso em: 05 out. 2015.

R SOFTWARE. **Software R**: The R Project for Statistical Compunting. Disponível em: <http://www.r-project.org/>. Acesso em: 10 out. 2013.

RADIO-ELECTRONICS. Resources and Analysis for Electronics Engineers. Disponível em: <http://www.radio-electronics.com/info/wireless/uwb/ultra-wideband-technology .php>. Acesso em: 18 de setembro de 2015.

RFC 3561. Request for comments 3561. Ad hoc On-Demand Distance Vector (AODV) Routing. The Internet Society (2003). Internet Engineering Task Force (IETF). July 2003. Disponível em: <https://www.ietf.org/rfc/rfc3561.txt>. Acesso em: 30 jan. 2016.

RFC 3626. Request for comments 3626. Optimized Link State Routing Protocol (OLSR). The Internet Society (2003). Internet Engineering Task Force (IETF). October 2003. Disponível em: <https://www.ietf.org/rfc/rfc3626.txt>. Acesso em: 30 jan. 2016.

RFC 4728. Request for comments 4728. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. The Internet Society (2007). Internet Engineering Task Force (IETF). February 2007. Disponível em: <https://www.ietf.org/rfc/rfc4728.txt>. Acesso em: 30 jan. 2016.

RFC 6550. Request for comments 6550. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). The Internet Society (2012). Internet Engineering Task Force (IETF). March 2012. Disponível em: < https://tools.ietf.org/html/rfc6550>. Acesso em: 30 mai. 2016.

SARKAR, S. K.; BASAVARAJU, T. G. and PUTTAMADAPPA, C. **Ad Hoc Mobile Wireless Networks**: Principles, Protocols, and Applications. New York, USA: Auerbach Publications, 2007.

SHAW, I. S. e SIMÕES, M. G. **Controle e Modelagem Fuzzy**. São Paulo: Edgard Blücher, 1999.

SHENAI, K. and MUKHOPADHYAY, S. Cognitive sensor networks. In: PROCEEDINGS INTERNATIONAL CONFERENCE ON MICROELECTRONICS, 26. (MIEL'2008). Nis, Serbia, **Proceedings...** p. 315-320, Nis, Serbia, May, 2008.

SILVA, F. A. et al. Tecnologia de Nós sensores sem fio. **Revista Controle e Instrumentação**. UFMG, 2014. Disponível em: <http://homepages.dcc.ufmg.br/~linnyer/bianossensores.htm>. Acesso em: 15 fev. 2015.

SINGH, N.; LAL DUA, R. and MATHUR, V. Network Simulator NS2-2.35. **International Journal of Advanced Research in Computer Science and Software Engineering** (IJARCSSE). v. 2, n. 5, May, 2012.

SOHRABY, K.; MINOLI, D. and ZNATI, T. **Wireless sensor networks**: technology, protocols, and applications. New Jersey, USA: John Wiley and Sons, 2007.

SOMOV, A.; DUPONT, C. and GIAFFREDA, R. Supporting Smart-city Mobility with Cognitive Internet of Things. In: CONFERENCE PROCEEDINGS OF FUTURE NETWORK & MOBILESUMMIT 2013. Paul Cunningham and Miriam Cunningham (Eds). International Information Management Corporation (IIMC). **Conference...** Portugal, Lisboa, July, 2013.

SRIVASTAVA, L. The Internet of Things. "Back to the Future". European Commission Internet of Things Conference in Budapest. (Presentation), 16 May 2011. Disponível em: <http://www.youtube.com/watch?v=CJdNq7uSddM>. Acesso em: 23 nov. 2013.

SURESH, P. et al. Angle based Routing Protocol for MANET. **International Journal of Scientific Engineering and Research** (IJSER). v. 2, n. 2, February, 2014.

THAGARD, P. **The cognitive science of science**: explanation, discovery, and conceptual change. 2nd ed. p. 376. Cambridge, MA: MIT Press, April, 2012.

THOMAS, R. W.; DASILVA, L. A. and MACKENZIE, A. B. Cognitive Network. In: IEEE INTERNATIONAL SYMPOSIUM ON NEW FRONTIERS IN DYNAMIC SPECTRUM ACCESS NETWORKS, 1. DySPAN 2005. p. 352-360, Baltimore, USA, November, 2005.

THOMAS, R. W. et al. Cognitive Networks: Adaptation and Learning to Achieve End-to-End Performance Objectives. In: TOPICS IN RADIO COMMUNICATIONS – IEEE

Communications Magazine. **IEEE Communications Society.** v. 44, n. 12, p. 51-57, December, 2006.

THOMAS, R. W. **Cognitive Networks**. 2007. 199p. Dissertation of PhD in Computer Engineering. Faculty of the Virginia Polytechnic Institute and State University. Blacksburg, Virginia, 2007.

TOH, C. K. et al. ABAM: on-demand associativity-based multicast routing for ad hoc mobile networks. In: IEEE VTS FALL VEHICULAR TECHNOLOGY CONFERENCE, 52. VTC'2000. Boston, MA. **Conference...** v. 3, p. 987-993, Boston, MA, September, 2000.

TOSSIM. **Software TinyOS Simulator**. Disponível em: <http://tinyos.stanford.edu/ tinyos-wiki/index.php/TOSSIM>. Acesso em: 10 out. 2013.

UFPA. Universidade Federal do Pará – Laboratório de Redes de Computadores e Comunicação Multimídia (GERCOM). Disponível em: <http://gercom.ufpa.br/>. Acesso em: 01 nov. 13.

UFPE. Universidade Federal de Pernambuco – Seminário sobre OLSR. Disponível em: <http://www.cin.ufpe.br/~if738/arquivos/seminarios/>. Acesso em: 04 jan. 2016.

VIEIRA, A. S. S. e CELESTINO Jr., J. Lógica Fuzzy para Melhorar o Desempenho do Protocolo AODV em Redes Ad Hoc Veiculares. Universidade Estadual do Ceará, UECE. In: CONGRESSO TECNOLÓGICO INFOBRASIL TI & TELECOM – INFOBRASIL, 4. **Anais...** Fortaleza, CE, 2011.

VIJAY, G.; BDIRA, E. B. A. and IBNKAHLA, M. Cognition in Wireless Sensor Networks: A Perspective. **IEEE Sensors Journal**. v. 11, n. 3, March, 2011.

WAGNER, M. S. **Influência de protocolos de segurança sobre o desempenho de redes UMTS**. 2009. 141p. Dissertação (Mestrado). Sistemas Eletrônicos – Escola Politécnica, Universidade de São Paulo, São Paulo, 2009.

WAGNER, M. S.; RAMÍREZ, M. A. and ZUCCHI, W. L. Adaptive Cognitive System Applied on Wireless Sensor Networks Nodes Decisions. In: IEEE INTERNATIONAL CONFERENCE ON COMMUNICATION SYSTEMS AND NETWORK TECHNOLOGIES (CSNT'2015), 5. **Conference...** Gwalior, Índia, april, 2015.

WAGNER, M. S.; RAMÍREZ, M. A. and ZUCCHI, W. L. Adaptive Cognitive System Applied to WSN Decisions at Nodes with a Fuzzy Logic Approach. International Journal of Communication System and Network Technology. v. 5, n. 1, England, UK, 2016.

WANG, Q. Traffic Analysis & Modeling in Wireless Sensor Networks and Their Applications on Network Optimization and Anomaly Detection. Macrothink Institute. **Journal of Network Protocols and Algorithms**. v. 2, n. 1, Sundsvall, Sweden, 2010.

WANG, Y. Yuan Wang's Home. Disponível em: <http://www-users.york.ac.uk/~yw679/ styled-6/styled-3/styled/index.html>. Acesso em: 07 ago. 2013.

WI-FI ALLIANCE. Wi-Fi Alliance Website – Discover Wi-Fi. Disponível em: <http://www.wi-fi.org/discover-wi-fi>. Acesso em: 09 de agosto de 2015.

WILSON, R. A. and KEIL, F. C. **The MIT encyclopedia of the cognitive sciences**. M.I.T. Cambridge: MIT Press, 1999.

X.210. Data Networks and Open System Communications; Open System Interconnection – General; Information Technology – Open Systems Interconnection – Basic Reference Model: Conventions for the Definition of OSI Services. International Telecommunication Union – Telecommunication Sector (ITU-T). 1993. Disponível em: <https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.210-199311-I!!PDF-E&type=items>. Acesso em: 13 jun. 2015.

ZADEH, L. A. Fuzzy sets. Electronics Research Laboratory, University of California, Berkeley, **Information and control Journal**, v. 8, n. 3, p. 338-353, June, 1965. Disponível em: <http://www.sciencedirect.com/science/journal/00199958/8/3>. Acesso em: 18 jan. 2016.

ZADEH, L. A. Outline of a new approach to the analysis of complex systems and decision processes. **IEEE Transactions on Systems, Man and Cybernetics**. v. SMC-3, p. 28-44, January, 1973. Disponível em: <http://trid.trb.org/ view.aspx?id=134506>. Acesso em: 18 jan. 2016.

ZHANG, N.; GUAN, J. and XU, C. Traffic Prediction Model for Cognitive Networks. In: PROCEEDINGS OF INTERNATIONAL CONFERENCE ON ADVANCED INTELLIGENCE AND AWARENESS INTERNET (AIAI 2011). **Conference…** China, Shenzhen, October, 2011.

ZHANG, M. et al. Cognitive Internet of Things: concepts and application example. **International Journal of Computer Science Issues** (IJCSI), v. 9, issue 6, n. 3, November, 2012.

APÊNDICES

APÊNDICE I	–	Texto auxiliar sobre o *Prowler*
APÊNDICE II	–	Gráficos das simulações
APÊNDICE III	–	*Software* do *Prowler* adaptado
APÊNDICE IV	–	Regras *Fuzzy*
APÊNDICE V	–	Módulo do CPMod
APÊNDICE VI	–	Módulo de correção do CPMod

APÊNDICE I

Texto auxiliar sobre o *Prowler*

Prowler é um programa desenvolvido na linguagem MATLAB para simulação de redes de sensores com quantidade variável de nós e distribuição aleatória dos mesmos, criado pelo *Institute for Software Integrated Systems* da *Vanderbilt University*, que fica no *Tennessee*, nos Estados Unidos (PROWLER, 2015).

O *Prowler* é um simulador de redes orientado a eventos, que é executado no ambiente do MATLAB, possibilitando a facilidade de prototipagem de aplicações e uso de interface gráfica (GUI). Este *software*, na configuração básica, é capaz de simular transmissão e recepção de RF entre os nós, propagação de dados na rede e operação na camada MAC/Enlace de redes do tipo *Ad Hoc*. Os modelos de rádio são baseados em cálculos específicos de intensidade de sinal combinados com erros aleatórios dentro de uma margem estipulada, sendo, portanto, uma plataforma adequada para o desenvolvimento de protocolos e algoritmos vinculados a *Ad Hoc* WSN (ARAÚJO, 2012).

Este trabalho utiliza o *Prowler* para a simulação de WSN, que em conjunto com o *toolbox Fuzzy* e o *Simulink*, visando a implementar um sistema cognitivo com tomada de decisão. Segundo Muneer (2013), uma grande variedade de plataformas de redes de sensores utilizam em sua maioria *softwares* de simulação direcionados para uma plataforma particular.

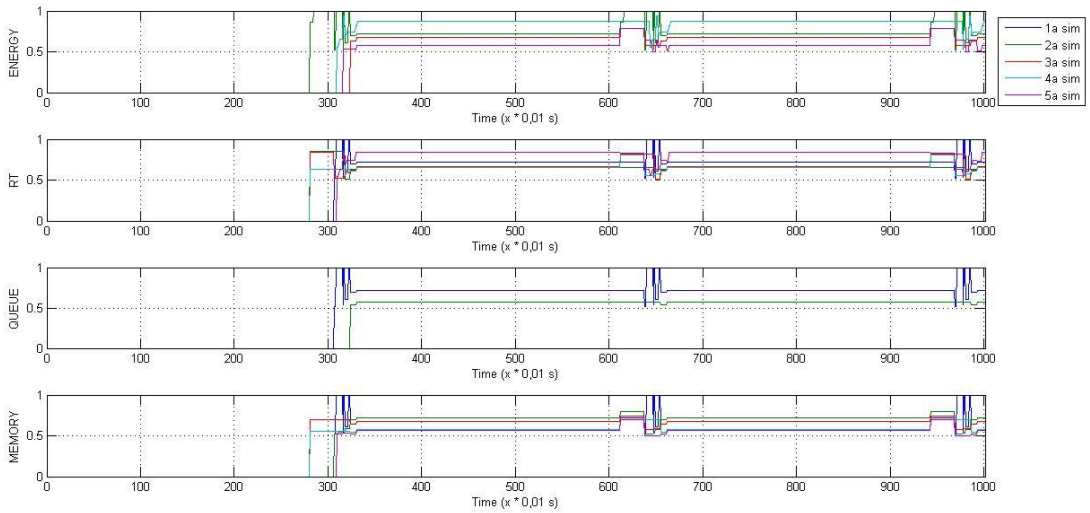
Berkeley (2015) apresenta uma grande relação de trabalhos suportados pelo projeto de pesquisa *Network Embedded Systems Technology* (NEST), inclusive com participação da *Vanderbilt University*, para o desenvolvimento de protocolos e realização de simulações que podem ter modificações de diferentes tipos de sistemas. O modelo de propagação pode ser determinístico ou probabilístico.

APÊNDICE II

Gráficos das simulações

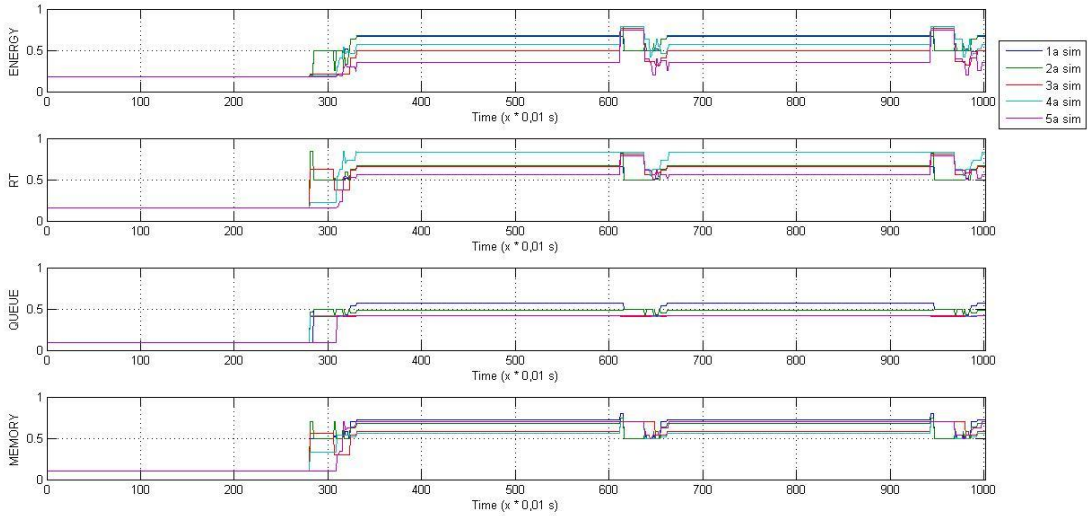
Os Gráficos 26 e 27 representam testes realizados com 50 nós no cenário com o *route angle*.

Gráfico 26. Entrada do bloco *Fuzzy* para 50 nós sensores.



Fonte: Autor.

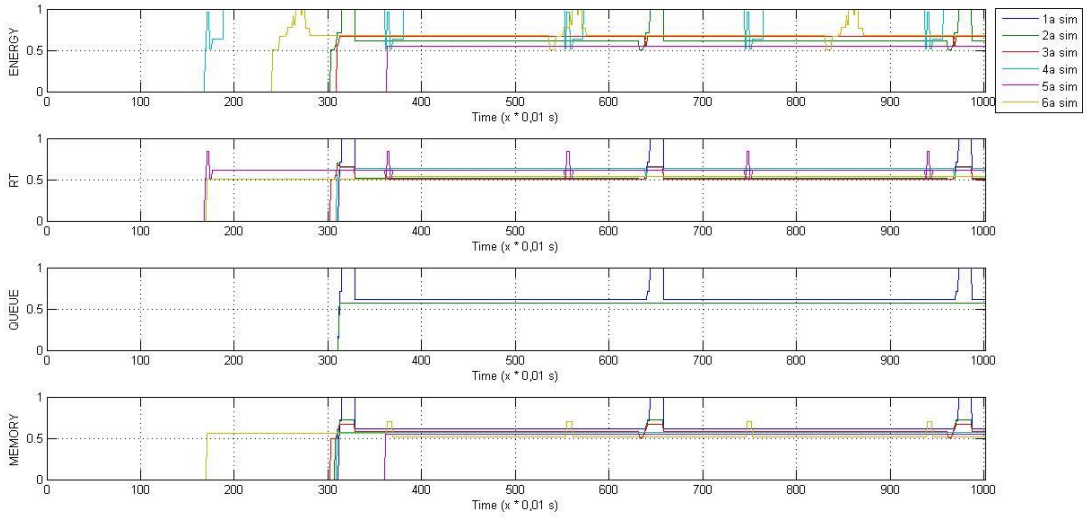
Gráfico 27. Saída do bloco *Fuzzy* para 50 nós sensores.



Fonte: Autor.

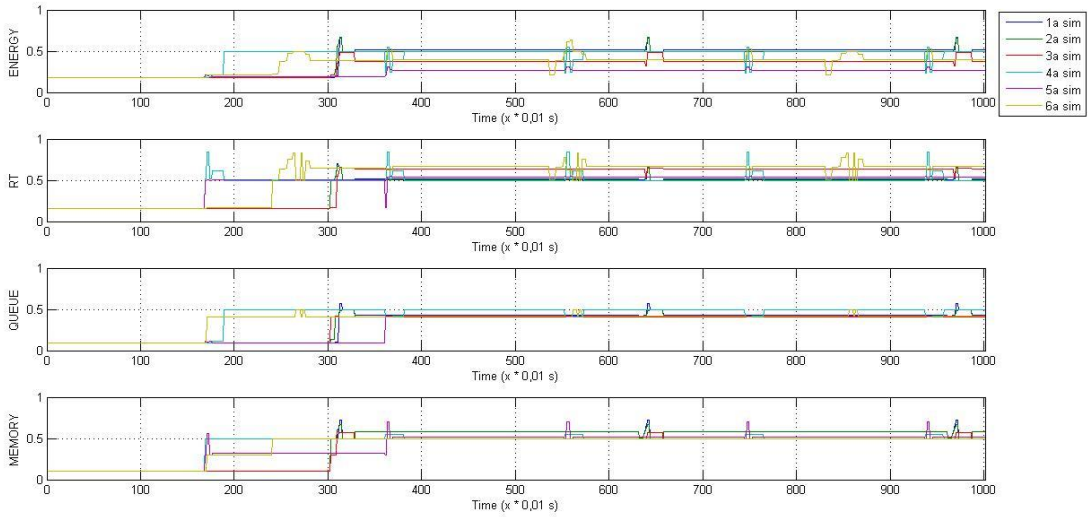
Os Gráficos 28 e 29 representam testes realizados com 100 nós no cenário com o *route angle* e *spanning tree*.

Gráfico 28. Entrada do bloco *Fuzzy* para 100 nós sensores.



Fonte: Autor.

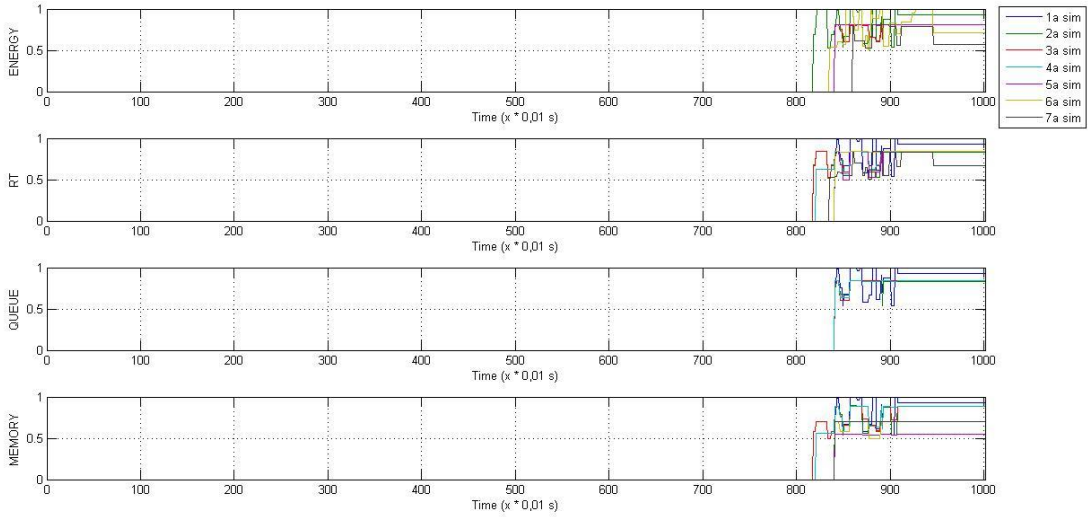
Gráfico 29. Saída do bloco *Fuzzy* para 100 nós sensores.



Fonte: Autor.

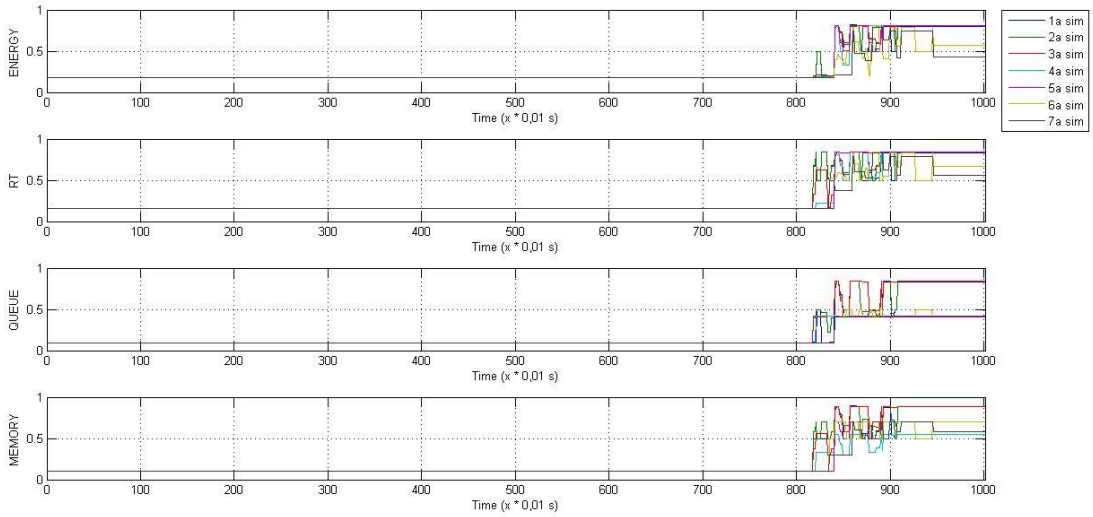
Os Gráficos 30 e 31 representam testes realizados com 150 nós no cenário com o *route angle*.

Gráfico 30. Entrada do bloco *Fuzzy* para 150 nós sensores.



Fonte: Autor.

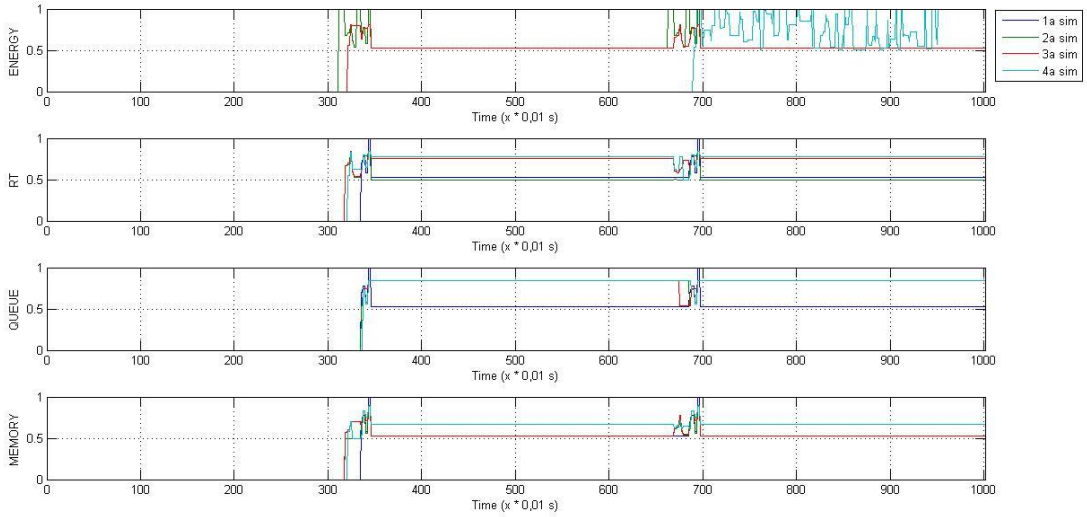
Gráfico 31. Saída do bloco *Fuzzy* para 150 nós sensores.



Fonte: Autor.

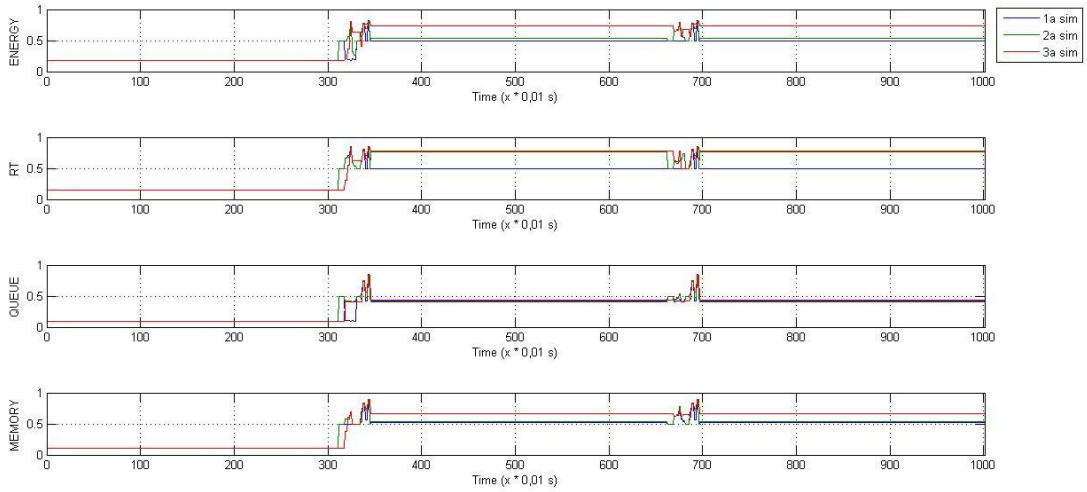
Os Gráficos 32 e 33 representam testes realizados com 200 nós no cenário com o *route angle* e *spanning tree*.

Gráfico 32. Entrada do bloco *Fuzzy* para 200 nós sensores.



Fonte: Autor.

Gráfico 33. Saída do bloco *Fuzzy* para 200 nós sensores.



Fonte: Autor.

APÊNDICE III

Software do *Prowler* adaptado

```
function varargout=prowler(command, varargin)
%  prowler - PROBABILISTIC WIRELESS NETWORK  SIMULATOR - Main simulation program
%
%  Command line options:
%    initialize: prowler('Init')
%    simulate:   prowler('StartSimulation')
%
% A graphical user interface can be invoked by typing prowler.
%
% See also: radio_channel, sim_params, demo_application, simstats, demo_opt

% ***
% ***    Copyright 2002, Vanderbilt University. All rights reserved.
% ***
% ***    This program is distributed in the hope that it will be useful,
% ***    but WITHOUT ANY WARRANTY; without even the implied warranty of
% ***    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
% ***
% ***

% Written by Gyula Simon, gyula.simon@vanderbilt.edu
% Last modified: Jan 28, 2004  by GYS


persistent event_Q  event_Q_ix topology mote_IDs topology_update_stamp
persistent radio app_name  sim_t  real_sim_T
global global_event_Q


if nargin<1
    command='OpenGui';
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL INÍCIO %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%v5
global     mediaENERGY   mediaDELAY  mediaDROP   mediaLATENCY



 %v2

if strcmpi(command, 'ReStartSimulation')

    sim_t=0;

    prowler('RefreshApplicationAndRadioInfo');
    SetApplicationParams(app_name);  % if exists _params file, set the default parameters
    %prowler('RefreshTopologyInfo', 'init');
    prowler('show_animation');
    prowler('show_events');


    prowparams('select_active_params')  % if parameters window is open, enable/disable radio specific UICs

    event_Q=[]; global_event_Q=[];
    prowler('InsertEvents2Q', make_event(0, 'Init_Radio', -999));
    for mote_ID=mote_IDs
        prowler('InsertEvents2Q', make_event(0, 'Init_Application', mote_ID));
    end
    event_Q_ix=1;
    print_event(['Application ''' app_name ''' initialized...'])
    plot_event('init')
    AdjustTipButton(app_name);

    prowler('show_LEDs')
    real_sim_T=0; % measure simulation time

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL FIM %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



 elseif strcmpi(command, 'Init')
    sim_t=0;

    prowler('RefreshApplicationAndRadioInfo');
    SetApplicationParams(app_name);  % if exists _params file, set the default parameters
    prowler('RefreshTopologyInfo', 'init');
    prowler('show_animation');
    prowler('show_events');


    prowparams('select_active_params')  % if parameters window is open, enable/disable radio specific UICs

    event_Q=[]; global_event_Q=[];
    prowler('InsertEvents2Q', make_event(0, 'Init_Radio', -999));
```

```
    for mote_ID=mote_IDs
        prowler('InsertEvents2Q', make_event(0, 'Init_Application', mote_ID));
    end
    event_Q_ix=1;
    print_event(['Application ''' app_name ''' initialized...'])
    plot_event('init')
    AdjustTipButton(app_name);

    prowler('show_LEDs')
    real_sim_T=0; % measure simulation time
elseif strcmpi(command, 'StartSimulation')
    sim_t=0;
    % housekeeping A1: remove those events from  global_event_Q which were added when simulation was suspended (by stop
button)
    % see housekeeping A2
    NUM_EVENTS_STOP=1000; % max number of events shown when stopped
    sim_params('set', 'SIMULATION_RUNNING',1);
    if length(global_event_Q)>0
        ix_end=length(global_event_Q)-length(event_Q)+event_Q_ix-1;
        print_event(global_event_Q(max(1,end-NUM_EVENTS_STOP):ix_end))
        global_event_Q=global_event_Q(1:end-length(event_Q));
    end
    % simulation
    % disable buttons and pulldown menus
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    if ~isempty(h_fig)
        a=guihandles(h_fig);
        set([a.Application_def, a.Radio_def, a.Simulation_start, a.Simulation_continue], 'enable', 'off')
    end
    tic; last_draw=clock; last_print=clock;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    a=cell(1001,1);
    marcel=1;


    %v07
    indiceAjuste = 0;

    % v1
    signals=struct('values',[],'dimensions',1,'labels','');

    ENERGYI=struct('time',[],'signals',[],'blockName','ENERGYI');
    DELAY=struct('time',[],'signals',[],'blockName','DELAY');
    DROP=struct('time',[],'signals',[],'blockName','DROP');
    LATENCY=struct('time',[],'signals',[],'blockName','LATENCY');

    %v5 media

    mediaENERGY=0;   %ENERGY
    mediaDELAY=0;    %RT
    mediaDROP=0;     %QUEUE
    mediaLATENCY=0;  %MEMORY


    Init_Radio=struct([]);
    Init_Application=struct([]);
    Channel_Request=struct([]);
    Channel_Idle_Check=struct([]);
    Packet_Receive_Start=struct([]);
    Packet_Receive_End=struct([]);
    Packet_Transmit_Start=struct([]);
    Packet_Transmit_End=struct([]);
    Packet_Sent=struct([]);
    Packet_Received=struct([]);
    Collided_Packet_Received=struct([]);
    Clock_Tick=struct([]);

    Radio=1;
    Application=1;
    Request=1;
    Idle_Check=1;
    Receive_Start=1;
    Receive_End=1;
    Transmit_Start=1;
    Transmit_End=1;
    Sent=1;
    Received=1;
    Collided=1;
    Tick=1;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    while sim_t<sim_params('get', 'STOP_SIM_TIME') & event_Q_ix <= length(event_Q) & sim_params('get','SIMULATION_RUNNING')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        a{marcel}=event_Q;
        marcel=marcel+1;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        event=event_Q(event_Q_ix);
        print_event(event)
        plot_event(event)
        last_sim_t=sim_t;
        [sim_t, event_name, ID, data]=get_event(event);
        if last_sim_t~=sim_t % new time instant, perhaps screen update necessary
            upd=sim_params('get', 'ANIMATE');
            if upd==1
                drawnow, last_draw=clock;
            else % slow update or no update
                if etime(clock, last_draw)>1
                    drawnow, last_draw=clock;
                end
            end
        end

        if etime(clock, last_print)>1, % this is to prevent gui from freezing when no animation is done
            last_print=clock;
            if ~sim_params('get', 'PRINT_EVENTS'), print_event(make_event(sim_t, 'Simulation running...',0), 0, 0), end
        end

        % decide to whom the event should be sent
        switch event_name
        case {'Init_Radio', 'Channel_Request', 'Channel_Idle_Check', ...
                    'Packet_Receive_Start', 'Packet_Receive_End', ...
                    'Packet_Transmit_Start', 'Packet_Transmit_End'}
            % event to radio layer
            feval(radio, event);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL INÍCIO %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %v1

        if ( strcmpi(event_name , 'Init_Radio'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Init_Radio{Radio,1} = event;
            Radio = Radio + 1;

        elseif ( strcmpi(event_name ,'Channel_Request'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Channel_Request{Request,1} = event;
            Request = Request + 1;

        elseif ( strcmp(event_name ,'Channel_Idle_Check'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Channel_Idle_Check{Idle_Check,1} = event;
            Idle_Check = Idle_Check + 1;

        elseif ( strcmpi(event_name ,'Packet_Receive_Start'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            signals.values(Receive_Start,1)=event.data.signal_strength;
            ENERGYI.signals=signals;

            %v15
            mediaENERGY = mediaENERGY + signals.values(Receive_Start,1);    %ENERGY
            cntENERGY = Receive_Start;

            Packet_Receive_Start{Receive_Start,1} = event;
            Receive_Start = Receive_Start + 1;

        elseif ( strcmpi(event_name ,'Packet_Receive_End'))
            event.time=event.time*sim_params('get', 'BIT_TIME');

            signals.values(Receive_End,1)=event.time - Packet_Receive_Start{Receive_End,1}.time;
            DELAY.signals=signals;

            %v5
            mediaDELAY=mediaDELAY + signals.values(Receive_End,1);     %RT
            cntDELAY = Receive_End;

            Packet_Receive_End{Receive_End,1} = event;
            Receive_End = Receive_End + 1;

        elseif ( strcmpi(event_name ,'Packet_Transmit_Start'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Packet_Transmit_Start{Transmit_Start,1} = event;
            Transmit_Start = Transmit_Start + 1;

        elseif ( strcmpi(event_name ,'Packet_Transmit_End'))
            event.time=event.time*sim_params('get', 'BIT_TIME');

            Packet_Transmit_End{Transmit_End,1} = event;
            Transmit_End = Transmit_End + 1;
        end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL FIM %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        case {'Init_Application', 'Packet_Sent', 'Packet_Received', ...
                    'Collided_Packet_Received', 'Clock_Tick'}
            % event to application layer
            feval([app_name, '_application'], event);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL INÍCIO %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
        %v1


        if ( strcmpi(event_name ,'Init_Application'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Init_Application{Application,1} = event;
            Application = Application + 1;

        elseif ( strcmpi(event_name ,'Packet_Sent'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Packet_Sent{Sent,1} = event;
            Sent = Sent + 1;

        elseif ( strcmpi(event_name ,'Packet_Received'))
            event.time=event.time*sim_params('get', 'BIT_TIME');

            signals.values(Received,1)=event.time;
            LATENCY.signals=signals;

            %v15
            mediaLATENCY = mediaLATENCY + signals.values(Received,1);  %MEMORY
            cntLATENCY = Received;

            Packet_Received{Received,1} = event;
            Received = Received + 1;

        elseif ( strcmpi(event_name ,'Collided_Packet_Received'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Collided_Packet_Received{Collided,1} = event;
            Collided = Collided + 1;

        elseif ( strcmpi(event_name ,'Clock_Tick'))
            event.time=event.time*sim_params('get', 'BIT_TIME');
            Clock_Tick{Tick,1} = event;
            Tick = Tick + 1;
        end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        otherwise
            error(['Unknown event: ' event_name])
        end
        event_Q_ix=event_Q_ix+1;

    end % while

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    assignin('base', 'teste11', a);

    % v15

for i=1:Sent-1
            signals.values(i,1)=(Packet_Receive_Start{i,1}.time - Packet_Receive_End{i,1}.time);

            DROP.signals=signals;

            %v5
            mediaDROP = mediaDROP + DROP.signals.values(i);     %QUEUE
end

    cntDROP = Sent;


    assignin('base', 'Init_Application', Init_Application);
    assignin('base', 'Init_Radio', Init_Radio);
    assignin('base', 'Channel_Request', Channel_Request);

    assignin('base', 'Packet_Receive_Start', Packet_Receive_Start);
    assignin('base', 'Packet_Receive_End', Packet_Receive_End);

    assignin('base', 'Packet_Transmit_Start', Packet_Transmit_Start);
    assignin('base', 'Packet_Transmit_End', Packet_Transmit_End);

    assignin('base', 'Packet_Sent', Packet_Sent);
    assignin('base', 'Packet_Received', Packet_Received);

    assignin('base', 'Collided_Packet_Received', Collided_Packet_Received);
    assignin('base', 'Channel_Idle_Check', Channel_Idle_Check);
    assignin('base', 'Clock_Tick', Clock_Tick);

    % teste de validação para verificar se a seleção de eventos está correta

%     assignin('base', 'Radio', Radio-1);
%     assignin('base', 'Application', Application-1);
%     assignin('base', 'Request', Request-1);
%     assignin('base', 'Idle_Check', Idle_Check-1);
%     assignin('base', 'Receive_Start', Receive_Start-1);
%     assignin('base', 'Receive_End', Transmit_End-1);
%     assignin('base', 'Transmit_Start', Transmit_Start-1);
%     assignin('base', 'Transmit_End', Transmit_End-1);
%     assignin('base', 'Sent', Sent-1);
%     assignin('base', 'Received', Received-1);
%     assignin('base', 'Collided', Collided-1);


    assignin('base', 'ENERGYI', ENERGYI);
    assignin('base', 'DELAY', DELAY);
```

```
    assignin('base', 'DROP', DROP);
    assignin('base', 'LATENCY', LATENCY);


    %v5 media

    mediaENERGY = mediaENERGY / cntENERGY;   %ENERGY
    mediaDELAY = mediaDELAY / cntDELAY;    %RT
    mediaDROP = mediaDROP / cntDROP;     %QUEUE
    mediaLATENCY = mediaLATENCY / cntLATENCY;  %MEMORY

    assignin('base', 'mediaENERGY', mediaENERGY);
    assignin('base', 'mediaDELAY', mediaDELAY);
    assignin('base', 'mediaDROP', mediaDROP);
    assignin('base', 'mediaLATENCY', mediaLATENCY);

%v07
indiceAjuste = evalin( 'base', 'indiceAjuste' );
if indiceAjuste == 0
energyRel(:,1)=ENERGYI.signals.values(:,1);
rtRel(:, 1)=DELAY.signals.values(:,1);
queueRel(:, 1)=DROP.signals.values(:,1);
memoryRel(:, 1)=LATENCY.signals.values(:,1);

multiplot1rel1(:, 1)=ENERGYI.signals.values(:,1);
multiplot1rel2(:, 1)=ENERGYI.signals.values(:,1);
multiplot1rel3(:, 1)=ENERGYI.signals.values(:,1);
multiplot1rel4(:, 1)=ENERGYI.signals.values(:,1);

multiplot2rel1(:, 1)=ENERGYI.signals.values(:,1);
multiplot2rel2(:, 1)=ENERGYI.signals.values(:,1);
multiplot2rel3(:, 1)=ENERGYI.signals.values(:,1);
multiplot2rel4(:, 1)=ENERGYI.signals.values(:,1);

assignin('base', 'energyRel', energyRel);
assignin('base', 'rtRel', rtRel);
assignin('base', 'queueRel', queueRel);
assignin('base', 'memoryRel', memoryRel);

assignin('base', 'multiplot1rel1', multiplot1rel1);
assignin('base', 'multiplot1rel2', multiplot1rel2);
assignin('base', 'multiplot1rel3', multiplot1rel3);
assignin('base', 'multiplot1rel4', multiplot1rel4);

assignin('base', 'multiplot2rel1', multiplot2rel1);
assignin('base', 'multiplot2rel2', multiplot2rel2);
assignin('base', 'multiplot2rel3', multiplot2rel3);
assignin('base', 'multiplot2rel4', multiplot2rel4);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% MARCEL FIM %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    try % try provided for compatibility reasons, older applications cannot handle the following events
        if sim_params('get','SIMULATION_RUNNING') % event queue empty
            for mote_ID=mote_IDs
                feval([app_name, '_application'], make_event(0, 'Application_Finished', mote_ID));
            end
        else
            for mote_ID=mote_IDs
                feval([app_name, '_application'], make_event(0, 'Application_Stopped', mote_ID));
            end
        end
    catch
    end

    sim_params('set', 'SIMULATION_RUNNING',0);
    real_sim_T=real_sim_T+toc;
    % housekeeping A2: add events to global_event_Q which were not purged from event_Q by the time
    % simulation was stopped
    % see housekeeping A1
    global_event_Q=[global_event_Q, event_Q];
    highlight_offset=length(event_Q)-event_Q_ix; % event monitor highlights the next event to be executed



    print_event(global_event_Q(max(1,end-NUM_EVENTS_STOP):end)) % update event list with a longer list
    print_event(['Stopped. (SimTime=' sprintf('%1.1f', real_sim_T) 's)'], highlight_offset+1)
    drawnow;
    % enable pulldown menus
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    if ~isempty(h_fig)
        a=guihandles(h_fig);
        set([a.Application_def, a.Radio_def, a.Simulation_start, a.Simulation_continue], 'enable', 'on')
    end

elseif strcmpi(command, 'StopSimulation')
    sim_params('set','SIMULATION_RUNNING',0);
    % enable pulldown menus
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    if ~isempty(h_fig)
        a=guihandles(h_fig);
        set([a.Application_def, a.Radio_def, a.Simulation_start, a.Simulation_continue], 'enable', 'on')
    end

elseif strcmpi(command, 'InsertEvents2Q')
    events=varargin{1};
    [event_Q, event_Q_ix]=insert_events(event_Q,event_Q_ix,events,sim_t);

elseif strcmpi(command, 'GetRadioName')
    varargout={radio};
```

```
elseif strcmpi(command, 'GetAnimationName')
    varargout={[app_name '_animation']};
elseif strcmpi(command, 'GetTopologyInfo')
    varargout={topology, mote_IDs, topology_update_stamp};

elseif strcmpi(command, 'RefreshApplicationAndRadioInfo')
    app_name=sim_params('get', 'APP_NAME');
    radio=sim_params('get', 'RADIO_NAME');

    application_name=[app_name '_application'];      % application is implemented in this m-file
    topology_name   =[app_name '_topology'];         % topology and ID info for the application
    animation_name  =[app_name '_animation'];        % topology and ID info for the application
    % check names
    if ~exist(application_name, 'file'), error(['Application file '' '  application_name '''.m is missing']); end
    if ~exist(topology_name, 'file'),    error(['Topology file '' '        topology_name '''.m is missing']); end
    if ~exist(animation_name, 'file'),   error(['Animation file '' '      animation_name '''.m is missing']); end
    if ~exist(radio, 'file'),            error(['Radio definition file '' '         radio '''.m is missing']); end


elseif strcmpi(command, 'RefreshTopologyInfo')
    topology_name=[app_name '_topology'];         % topology and ID info for the application
    if nargin >1; % init
        topology_update_stamp=0;
        [topology, mote_IDs]=feval(topology_name, 'init');
    else
        [topology, mote_IDs]=feval(topology_name);
    end
    topology_update_stamp=topology_update_stamp+1;
    if  topology_update_stamp>1   % not init
        radio=sim_params('get', 'RADIO_NAME');
        feval(radio, 'Prowler!RefreshTopology');  % notify the radio channel, it should update its internal info
    end
elseif strcmpi(command, 'TextMessage')
    plot_event('TextMessage', varargin{1}, varargin{2})
elseif strcmpi(command, 'LED')
    plot_event(command, varargin{1}, varargin{2})

elseif findstr(command, 'Draw')
    if findstr(command, 'Line')
        plot_line('Line', varargin{:})
    elseif findstr(command, 'Arrow')
        plot_line('Arrow', varargin{:})
    elseif findstr(command, 'Delete')
        plot_line('Delete', varargin{:})
    end

elseif strcmpi(command, 'Redraw')
    plot_event(command)

elseif strcmpi(command, 'Gui_Mouse_Axes_Click')  % message from gui; can be used to update topology
    position=varargin{1}; position=position(1,1:2);
    try
        feval([app_name '_topology'], 'Refresh', position);
        prowler('RefreshTopologyInfo');
        prowler('Redraw')
    end
elseif strcmpi(command, 'GuiMouseMoteClick')
    h_clicked_mote=varargin{1};
    clicked_mote_ID=get(h_clicked_mote, 'userdata');
    feval([app_name, '_application'], make_event(sim_t, 'GuiInfoRequest', clicked_mote_ID));


elseif strcmpi(command, 'show_LEDs')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    ch=allchild(h_fig);
    h_cb=findobj(ch, 'flat', 'tag', 'showLEDs');
    show_LEDs=get(h_cb, 'value');
    % find all LED related staff on plot
    %h_ax=findobj(ch, 'flat', 'tag', 'simulation_plot_ax');
    h_ax=findall(0, 'tag', 'simulation_plot_ax');
    %     h_line=findobj(allchild(h_ax), 'flat', 'type', 'line');
    %     h_LEDs=[]; h_frames=[]; i=1; found=1;
    %     while ~isempty(found)
    %         found=[ findobj(h_line, 'flat', 'tag', ['rLED' num2str(i)]); ...
    %                 findobj(h_line, 'flat', 'tag', ['gLED' num2str(i)]); ...
    %                 findobj(h_line, 'flat', 'tag', ['yLED' num2str(i)]); ...
    %                 findobj(h_line, 'flat', 'tag', ['LED_frame' num2str(i)])];
    %         h_LEDs=[h_LEDs; found]; i=i+1;
    %     end
    %     h_LEDs  =[findobj(h_ax, 'tag', 'rLED'); findobj(h_ax, 'tag', 'gLED'); findobj(h_ax, 'tag', 'yLED')];
    %     h_frames=findobj(h_ax, 'tag', 'LED_frame');
    h_LEDs=findobj(allchild(h_ax), 'flat',  'buttondownfcn', '3.1415926;');

    if show_LEDs
        set([h_LEDs], 'visible', 'on')
    else
        set([h_LEDs], 'visible', 'off')
    end


elseif strcmpi(command, 'show_distances')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    ch=allchild(h_fig);
    % h_ax=findobj(ch, 'flat', 'tag', 'simulation_plot_ax');
    h_ax=findall(0, 'tag', 'simulation_plot_ax');
    h_cb=findobj(ch, 'flat', 'tag', 'show_distances');
    if nargin > 1
        show=varargin{1};
        set(h_cb, 'value', show);
    else
        show=get(h_cb, 'value');
    end
```

```
    if show
        set(h_ax, 'xtickmode', 'auto')
        set(h_ax, 'ytickmode', 'auto')
        grid(h_ax, 'on')
    else
        set(h_ax, 'xtickmode', 'manual')
        set(h_ax, 'ytickmode', 'manual')
        set(h_ax, 'xtick', [])
        set(h_ax, 'ytick', [])

        grid(h_ax, 'off')
    end

elseif strcmpi(command, 'show_animation')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    ch=allchild(h_fig);
    h=findobj(ch, 'flat', 'tag', 'show_animation');
    if nargin >1 % value provided
        anim=varargin{1};
        if anim==0; anim=3; end
        set(h,'value', anim);
        sim_params('set_from_gui', 'ANIMATE', mod(anim,3));
    else
        anim=get(h,'value');
        sim_params('set_from_gui', 'ANIMATE', mod(anim,3));
    end

elseif strcmpi(command, 'show_events')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    ch=allchild(h_fig);
    h=findobj(ch, 'flat', 'tag', 'show_events');

    if nargin >1 % value provided
        shw=varargin{1};
        set(h,'value', shw);
        sim_params('set_from_gui', 'PRINT_EVENTS', shw);
    else
        sim_params('set_from_gui', 'PRINT_EVENTS', get(h,'value'));
    end

elseif strcmpi(command, 'ShowApplicationInfo')
    infofile=AdjustTipButton(app_name);
    if ~isempty(infofile)
        feval(infofile);
    end

elseif strcmpi(command, 'ShowApplicationParams')
    appparamw('init', app_name)

elseif strcmpi(command, 'OpenGUI')
    simgui;
elseif strcmpi(command, 'CloseGUI')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    delete(h_fig)
    h_fig=findobj(allchild(0), 'flat', 'tag', 'paramgui_fig');
    close(h_fig)
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Prowler_External_Display_fig');
    delete(h_fig)

elseif strcmpi(command, 'SwitchDisplay')
    h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
    h_cb=findobj(allchild(h_fig), 'flat', 'tag', 'external_display');
    if nargin>1
        mode=varargin{1};
        if strcmp('mode', 'out')
            set(h_cb, 'value', 1)
        else
            set(h_cb, 'value', 0)
        end
    end
    if get(h_cb, 'value')
        mode='out';
    else
        mode='in';
    end
    SwitchDisplay(mode);

elseif strcmpi(command, 'GetDisplayHandle')
        h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
        h_ax=findall(h_fig, 'tag', 'simulation_plot_ax');
        if isempty(h_ax)
            h_fig=findobj(allchild(0), 'flat', 'tag', 'Prowler_External_Display_fig');
            h_ax=findall(h_fig, 'tag', 'simulation_plot_ax');
        end
        varargout={h_ax};

% 'PrintEvent' command is added by LK and YZ
elseif strcmpi(command, 'PrintEvent')
    print_event(varargin{1});

elseif strcmpi(command, 'version')
    % CURRENT VERSION NUMBER
    varargout={'1.25'};
else
    error(['Unknown command: ' command])
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [new_Q, new_Q_ix]=insert_events(old_Q,old_Q_ix,events,last_t);
global global_event_Q

PURGE_LIMIT=50;
```

```
new_Q=[old_Q, events];

L=length(new_Q);
t=zeros(1,L);
for i=1:L
    t(i)=new_Q(i).time;
end
[tmp,ix]=sort(t);

ix1=find(t(ix)>=last_t);
purge_Num=length(ix)-length(ix1);
if purge_Num>PURGE_LIMIT; PURGE=1; else PURGE=0; end
if PURGE % purge old events from Q
    global_event_Q=[global_event_Q, old_Q(1:purge_Num)];
    new_Q=new_Q(ix(ix1));
    new_Q_ix=old_Q_ix-purge_Num;
else
    new_Q=new_Q(ix);
    new_Q_ix=old_Q_ix;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function print_event(e, offset,force)
persistent h_list ct str
if (~sim_params('get', 'PRINT_EVENTS'))&sim_params('get', 'SIMULATION_RUNNING')&nargin<3, return, end
if nargin<2, offset=0; end
if nargin>2, ct=100; end % force event update
MAX_LINES=100;
if ischar(e) % init or finish
    if findstr(e, 'init')
        clear=1; ct=100;
        h_list=findall(0, 'tag', 'message_list');
        disp_str=e; list_str={e};
        %plot_event('init')
    elseif findstr(e, 'Running')
        clear=1; ct=100;
        disp_str=e; list_str={e};
    else % finish
        clear=0; MAX_LINES=inf; % prevent list truncation
        disp_str=e; list_str={e}; ct=100; % ensure to update listbox
    end
else
    len_e=length(e);
    if len_e>1,
        clear=1;
        list_str=[];
    else
        clear=0;
        list_str=[];
    end

    for ev_ix=1:len_e
        [sim_t, event, ID, data]=get_event(e(ev_ix));
        sim_t_sec=sim_t*sim_params('get', 'BIT_TIME');
        list_stri=sprintf('%7d %6.2f  %-26s %5d', floor(sim_t), sim_t_sec, event, ID);
        list_str=[list_str; {list_stri}];
    end
    if len_e >1
        disp_str=[]; ct=100;
    else
        disp_str=sprintf('t:%7d (%6.2fs), %-27s ID:%3d', floor(sim_t), sim_t_sec, [event ','], ID);
        %plot_event(e)
    end

end

if ~isempty(h_list)
    %str=get(h_list, 'string');
    if clear
        str=list_str;
    else
        if length(str)>MAX_LINES-1;
            str=[str(end-MAX_LINES+1:end); list_str];
        else
            str=[str; list_str];
        end
    end
    if ct>10 % do not update list too frequently; it's too slow
        set(h_list, 'string',  str, 'listboxtop', max(1, length(str)-1), 'value', length(str)-offset)
        ct=0;
    end
    ct=ct+1;
else
    % disp(disp_str)
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function plot_event(varargin)
persistent h hr hg hy ht hf ax updatetime
upd=sim_params('get', 'ANIMATE');
if ischar(varargin{1}) % command
    cmd=varargin{1};
    if ~upd&~strcmp(cmd, 'init')
        return
    end
else
    if ~upd
        return
    end
    cmd='update';
    event_struct=varargin{1};
```

```
    [t, event, ID, data]=get_event(event_struct);
end
[topology, mote_IDs]=prowler('GetTopologyInfo');
drw=0; % init draw update flag
switch cmd
case 'Refresh'  % refresh handles which could be changed when display changed
    ax=findall(0, 'tag', 'simulation_plot_ax');
    ch=allchild(ax);
    lines=findobj(ch, 'flat', 'type', 'line');
    texts=findobj(ch, 'flat', 'type', 'text');
    for i=1:length(h)
        h(i) =findobj(lines, 'flat', 'tag', ['DisplayObjMote' num2str(i)]);
        hr(i)=findobj(lines, 'flat', 'tag', ['rLED' num2str(i)]);
        hg(i)=findobj(lines, 'flat', 'tag', ['gLED' num2str(i)]);
        hy(i)=findobj(lines, 'flat', 'tag', ['yLED' num2str(i)]);
        ht(i)=findobj(texts, 'flat', 'tag', ['DisplayObjTxt' num2str(i)]);
        hf(i)=findobj(lines, 'flat', 'tag', ['LED_frame' num2str(i)]);
    end
    plot_line('refresh')

case {'init', 'Redraw'}

    ax=findall(0, 'tag', 'simulation_plot_ax');
    if ~isempty(ax)
        Mx=max(topology(:,1)); mx=min(topology(:,1));
        My=max(topology(:,2)); my=min(topology(:,2));

        if Mx-mx<1, Mx=Mx+.5; mx=mx-.5; end
        if My-my<1, My=My+.5; my=my-.5; end
        %sc=0.001; mx=mx-sc*abs(mx); my=my-sc*abs(my); Mx=Mx+sc*abs(Mx); My=My+sc*abs(My);
        deltaX=Mx-mx;
        deltaY=My-my;
        DX=deltaX/50; DY=-deltaY/50;

        if strcmp(cmd, 'init')
            updatetime=clock;
            delete(allchild(ax));
            set(ax, 'nextplot', 'add')
            axis(ax, [mx-1*DX Mx+6*DX my+4*DY My-3*DY])
            h=[]; hr=[]; hg=[]; hy=[]; ht=[]; hf=[];
            for i=1:length(mote_IDs)
                PX=topology(i, 1); PY=topology(i, 2);
                h(i)=plot(PX,PY,'.', 'parent', ax, ...
                    'userdata', mote_IDs(i), 'buttondownfcn', 'prowler(''GuiMouseMoteClick'', gcbo)', 'tag',
['DisplayObjMote' num2str(i)]);
                hr(i)=plot(PX+1.5*DX,PY+DY,'.r', 'parent', ax, 'tag', ['rLED' num2str(i)], 'userdata', [1   0   0]);
                hg(i)=plot(PX+2.5*DX,PY+DY,'.g', 'parent', ax, 'tag', ['gLED' num2str(i)], 'userdata', [0   1   0]);
                hy(i)=plot(PX+3.5*DX,PY+DY,'.y', 'parent', ax, 'tag', ['yLED' num2str(i)], 'userdata', [1   0.6 0]);
                hf(i)=line(PX+[.7*DX, .7*DX 4*DX 4*DX .7*DX],...
                    PY+[DY/2 DY*3/2 DY*3/2 DY/2 DY/2], 'parent', ax, 'tag', ['LED_frame' num2str(i)]);
                ht(i)=text(PX+.5*DX,PY-DY,' ', 'FontSize', 8, 'clipping', 'on', 'parent', ax, 'tag', ['DisplayObjTxt'
num2str(i)]);
            end
            set([hr hg hy], 'markersize', 6, 'color', [1 1 1])
            set([hr hg hy hf], 'buttondownfcn', '3.1415926;', 'hittest', 'off'); % search purposes
            plot_line('init')
            drawnow
        else
            for i=1:length(mote_IDs)
                PX=topology(i, 1); PY=topology(i, 2);
                set(h(i), 'xdata', PX, 'ydata', PY);
                set(hr(i), 'xdata', PX+1.5*DX, 'ydata', PY+DY);
                set(hg(i), 'xdata', PX+2.5*DX, 'ydata', PY+DY);
                set(hy(i), 'xdata', PX+3.5*DX, 'ydata', PY+DY);
                set(hf(i), 'xdata', PX+[.7*DX, .7*DX 4*DX 4*DX .7*DX], ...
                    'ydata',  PY+[DY/2 DY*3/2 DY*3/2 DY/2 DY/2]);
                set(ht(i), 'position', [PX+.5*DX,PY-DY]);
            end
            plot_line('redraw')
        end
        drw=1;
    end
case 'update'
    if ~isempty(ax)
        ix=find(mote_IDs==ID);
        a=feval(prowler('GetAnimationName'));
        for i=1:length(a)
            if strcmpi(a(i).event, event)
                if a(i).animated
                    switch a(i).animated
                    case 1 % the mote
                        if ~isempty(a(i).color), set(h(ix), 'color', a(i).color); end
                        if ~isempty(a(i).size),  set(h(ix), 'markersize',  a(i).size);  end
                    case {2,3,4} % LEDs
                        if a(i).color(1)
                            mode='on';
                        elseif a(i).color(2)
                            mode='off';
                        else
                            mode='toggle';
                        end
                        switch a(i).animated
                        case 2 % red LED
                            h_LED=hr(ix);
                        case 3 % green LED
                            h_LED=hg(ix);
                        case 4 % yellow LED
                            h_LED=hy(ix);
                        end
                        cur_col=get(h_LED, 'color');
                        on_col =get(h_LED, 'userdata');
                        if strcmp('toggle', mode)
```

```
                            if cur_col==on_col
                                mode='off';
                            else
                                mode='on';
                            end
                        end
                        if strcmp('on', mode)
                            set(h_LED, 'color', on_col);
                        else
                            set(h_LED, 'color', [1 1 1]);
                        end
                    end
                    drw=1;
                end
                break
            end
        end
    end
case 'TextMessage'
    if ~isempty(ax)
        ID=varargin{2};
        txt=varargin{3};
        ix=find(mote_IDs==ID);
        set(ht(ix), 'string', txt)
        drw=1;
    end
case 'LED'
    if ~isempty(ax)
        ID=varargin{2};
        msg=varargin{3};
        ix=find(mote_IDs==ID);
        if findstr(lower(msg), 'red');    h_LED=hr(ix);
        elseif findstr(lower(msg), 'green');  h_LED=hg(ix);
        elseif findstr(lower(msg), 'yellow'); h_LED=hy(ix);
        else error(['Bad LED color in command ' msg]);
        end

        cur_col=get(h_LED, 'color');
        on_col =get(h_LED, 'userdata');

        if findstr(lower(msg), 'on');     mode='on';
        elseif findstr(lower(msg), 'off');    mode='off';
        elseif findstr(lower(msg), 'toggle');
            if cur_col==on_col; mode='off'; else mode='on'; end
        else error(['Bad LED state in command ' msg]);
        end

        if strcmp('on', mode)
            set(h_LED, 'color', on_col);
        else
            set(h_LED, 'color', [1 1 1]);
        end
        drw=1;
    end
end

if drw*0  % drawnow's are managed in the main loop
    if upd==1 % fast update
        % drawnow necessary
        drwnow=1;
    else % slow update
        % check update time
        if etime(clock, updatetime)>5
            drwnow=1;
        else
            drwnow=0;
        end
    end
    if drwnow
        drawnow;
        updatetime=clock;
    end
end

function plot_line(command, ID1, ID2, varargin)
persistent table
% table contains current lines (or arrows) in the following format:
% each line (arrow) has a row in the table: {ID1_i, ID2_i, handle_i, command, varargin}

command=lower(command);

if nargin>3
    style=varargin;
else
    style=[];
end

switch command
case {'line', 'arrow', 'delete'}
    if ~sim_params('get', 'ANIMATE');
        return
    end
    [topology, mote_IDs]=prowler('GetTopologyInfo');

    if strcmp(command, 'delete') & isinf(ID1+ID2)
        ct=1; % special delete all syntax, only ID1 is needed
    else
        ct=0; % two ID's required
    end
    for ix=1:length(mote_IDs)
        if mote_IDs(ix) == ID1
            ix1=ix; ct=ct+1;
```

```
        elseif  mote_IDs(ix) == ID2
            ix2=ix; ct=ct+1;
        end
        if ct>1
            break
        end
    end
    if ct<2 % ID not found
        error(sprintf('Bad ID for line draw: %d, %d', ID1, ID2))
    else
        ax=findall(0, 'tag', 'simulation_plot_ax');
        if ~isempty(ax)
            % search for line in the table; if exists, remove
            len_t=size(table,1);
            for t_ix=1:len_t
                if table{t_ix, 1}==ID1 & table{t_ix, 2}==ID2
                    h=table{t_ix, 3};
                    table(t_ix,:)=[];
                    delete(h);
                    break
                end
            end
            if strcmp(command, 'delete')
                if isinf(ID1+ID2)  % delete all
                    if isinf(ID1)
                        ID_ix=2; ID_val=ID2;
                    else
                        ID_ix=1; ID_val=ID1;
                    end

                    del_h=[]; del_ix=[];
                    for t_ix=1:len_t
                        if table{t_ix, ID_ix}==ID_val
                            del_h=[del_h, table{t_ix, 3}];
                            del_ix=[del_ix, t_ix];
                        end
                    end
                    table(del_ix,:)=[];
                    delete(del_h);
                end
                return
            end

            x1=topology(ix1,1); y1=topology(ix1,2);
            x2=topology(ix2,1); y2=topology(ix2,2);

            if strcmp(command, 'arrow')

                phi=pi/10;  % arrow angle
                L=0.03;     % arrow size constant

                xa = get(ax,'xlim');
                ya = get(ax,'ylim');
                set(ax, 'unit', 'points');
                pos= get(ax,'position');
                xp=pos(3); yp=pos(4); % axis size in figure
                xd = xa(2)-xa(1);     % axis limits
                yd = ya(2)-ya(1);
                scalex = L*xd/xp*yp; % compensate aspect ratio
                scaley = L*yd;

                dx = x1 - x2;
                dy = y1 - y2;

                alphac=atan2(dy/yd*yp, dx/xd*xp); % angle of line on screen

                xx = [x1, x2, x2+scalex*cos(alphac+phi), NaN, ...
                        x2, x2+scalex*cos(alphac-phi)]';
                yy = [y1, y2, y2+scaley*sin(alphac+phi), NaN, ...
                        y2, y2+scaley*sin(alphac-phi)]';

                hl=line(xx,yy, 'parent', ax, 'hittest', 'off', 'tag', 'DisplayArrow', 'userdata', [ID1, ID2]);
                set(ax, 'unit', 'normalized');  % necessary for resizable external plot
            else % line
                hl=line([x1,x2], [y1,y2], 'parent', ax, 'hittest', 'off', 'tag', 'DisplayArrow', 'userdata', [ID1, ID2]);
            end
            if ~isempty(style)
                set(hl,  style{:})
            end
            table=[table; {ID1, ID2, hl, command, varargin}];
        end
    end
case 'init'
    table={};
case 'refresh'
    ax=findall(0, 'tag', 'simulation_plot_ax');
    h_arr=findobj(allchild(ax), 'flat', 'tag', 'DisplayArrow');
    new_table=table;
    for i=1:length(h_arr)
        ix=get(h_arr(i),'userdata');
        for j=1:length(h_arr)
            if table{j,1}==ix(1) & table{j,2}==ix(2)
                new_table{j,3}=h_arr(i);
                break
            end
        end
    end
    table=new_table;

case 'redraw'
    [topology, mote_IDs]=prowler('GetTopologyInfo');
    len_t=size(table,1);
```

```
    table_old=table;
    for t_ix=1:len_t
        ID1=table_old{t_ix,1};
        ID2=table_old{t_ix,2};
        command=table_old{t_ix,4};
        xtra=table_old{t_ix,5};
        plot_line(command, ID1, ID2, xtra{:})
    end
end

function out=AdjustTipButton(app_name)
% Enables Application Info pushbutton if _info file exists for the application,
% disables otherwise. Returns the info file name.
h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
h_tip=findobj(allchild(h_fig), 'flat', 'tag', 'Application_tips');
infofile=[app_name, '_info'];
if exist([infofile '.m'], 'file')
    set(h_tip, 'enable', 'on')
    out=infofile;
else
    set(h_tip, 'enable', 'off')
    out=[];
end

function out=SetApplicationParams(app_name)
% Checks if application parameters are defined.
% If not, set the default.
h_fig=findobj(allchild(0), 'flat', 'tag', 'Simulation_Fig');
h_par=findobj(allchild(h_fig), 'flat', 'tag', 'Application_params');
paramfile=[app_name, '_params'];
if exist([paramfile '.m'], 'file')
    set(h_par, 'enable', 'on')
    p=feval(paramfile);
    for i=1:length(p)
        if isempty(sim_params('get_app', p(i).name))
            if iscell(p(i).default) % popupmenu, the first element is the default
                sim_params('set_app', p(i).name, p(i).default{1});
            else
                sim_params('set_app', p(i).name, p(i).default);
            end
        end
    end
    out=paramfile;
else
    set(h_par, 'enable', 'off')
    out=[];
end

function SwitchDisplay(mode)
% switch between internal and external display modes

switch mode
case 'out'
    h_sim_fig=findall(0, 'tag', 'Simulation_Fig');
    h_ax=findall(h_sim_fig, 'tag', 'simulation_plot_ax');
    if ~isempty(h_ax)
        set(h_ax, 'tag', 'inactive_simulation_plot_ax')
        h=findall(0, 'tag', 'simulation_plot_ax'); delete(h); % just in case...

        h_fig=figure(...
            'name', 'Prowler - Display',...
            'numbertitle', 'off', ...
            'integerhandle', 'off', ...
            'closerequestfcn', 'prowler(''SwitchDisplay'', ''in'')', ...
            'handlevisibility', 'off', ...
            'units', 'pixels', ...
            'tag', 'Prowler_External_Display_fig');

        h_ax_new=copyobj(h_ax, h_fig);
        delete(allchild(h_ax))
        set(h_ax, 'buttondownfcn', 'figure(findall(0,''tag'', ''Prowler_External_Display_fig''))');

        set(h_ax_new, ...
            'unit', 'normalized',...
            'position', [0 0 1 1], ...
            'tag', 'simulation_plot_ax');
        xx=get(h_ax, 'Xlim'); yy=get(h_ax, 'Ylim');
        plot([xx(2) xx(1) nan xx(1) xx(2)], [yy(1) yy(2) nan yy(1) yy(2)], 'parent', h_ax)

    end
case 'in'
    h_sim_fig=findall(0, 'tag', 'Simulation_Fig');
    h_ext_fig=findall(0, 'tag', 'Prowler_External_Display_fig');
    h_ax=findall(h_ext_fig, 'tag', 'simulation_plot_ax');
    if ~isempty(h_ax)
        set(h_ax, 'tag', 'external_simulation_plot_ax');

        h=findall(0, 'tag', 'simulation_plot_ax'); delete(h); % just in case...

        h_ax_old=findall(h_sim_fig, 'tag', 'inactive_simulation_plot_ax');
        h_ax_new=copyobj(h_ax, h_sim_fig);
        set(h_ax_new, ...
            'unit', get(h_ax_old, 'unit'), ...
            'position', get(h_ax_old, 'position'), ...
            'tag', 'simulation_plot_ax');
        delete(h_ax_old)
        delete(h_ext_fig)
    end
end
plot_event('Refresh'); % update handles
```

APÊNDICE IV

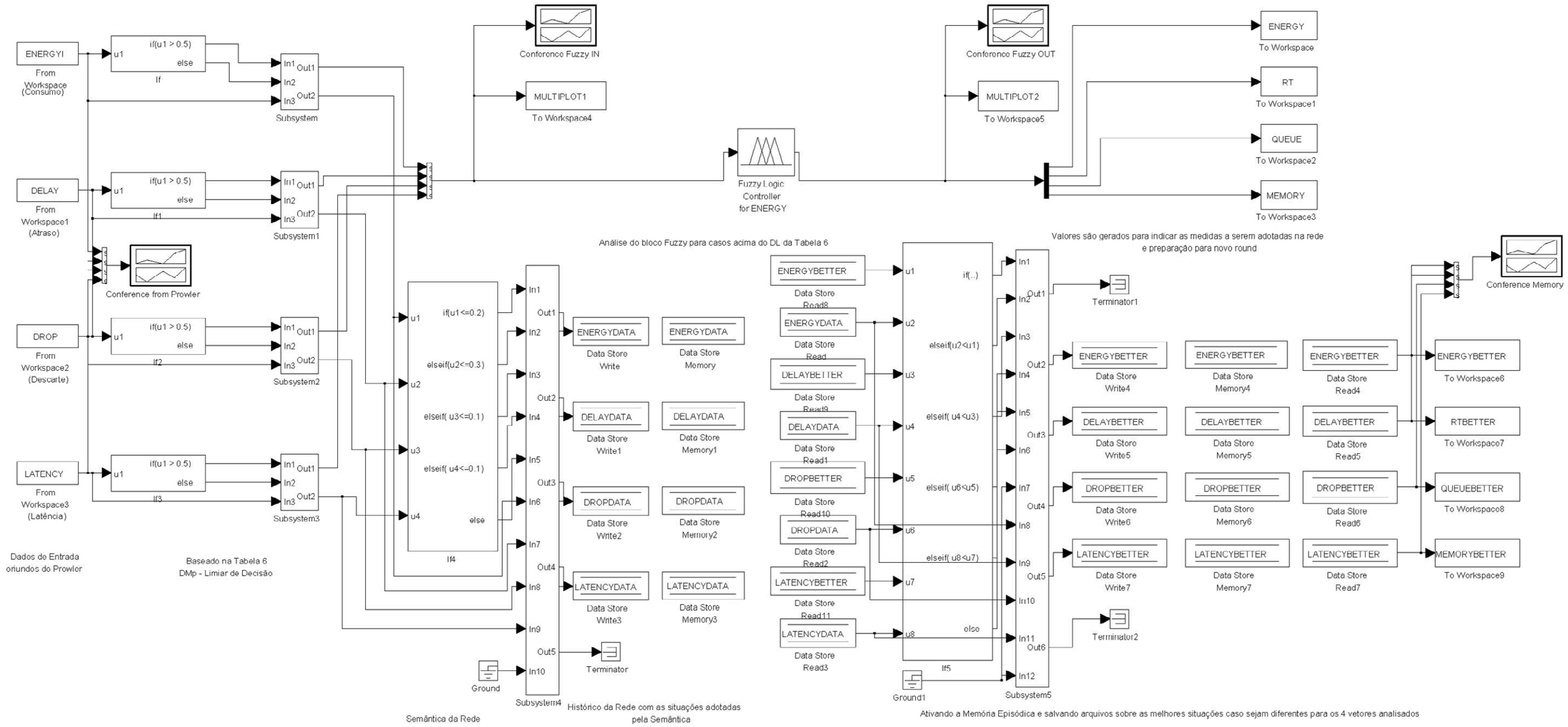
Regras *Fuzzy*

```
[System]
Name='ProcessoCognitivo3a'
Type='mamdani'
Version=2.0
NumInputs=4
NumOutputs=4
NumRules=162
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Error'
Range=[0 1]
NumMFs=3
MF1='Considerable':'trapmf',[0 0 0.2 0.5]
MF2='Verification':'trimf',[0.2 0.5 0.8]
MF3='Problem':'trapmf',[0.5 0.8 1 1]

[Input2]
Name='FN'
Range=[0 1]
NumMFs=6
MF1='VL':'trimf',[0 0 0.2]
MF2='L':'trimf',[0 0.2 0.4]
MF3='M':'trimf',[0.2 0.4 0.6]
MF4='H':'trimf',[0.4 0.6 0.8]
MF5='VH':'trimf',[0.6 0.8 1]
MF6='E':'trimf',[0.8 1 1]

[Input3]
Name='VMq'
Range=[0 1]
NumMFs=3
MF1='Good':'trapmf',[0 0 0.2 0.4]
MF2='Acceptable':'trapmf',[0.2 0.4 0.6 0.8]
MF3='Bad':'trapmf',[0.6 0.8 1 1]

[Input4]
Name='VMh'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.4]
MF2='Considerable':'trapmf',[0.2 0.4 0.6 0.8]
MF3='Full':'trapmf',[0.6 0.8 1 1]

[Output1]
Name='Memory'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.5]
MF2='Good':'trimf',[0.2 0.5 0.8]
MF3='High':'trapmf',[0.5 0.8 1 1]

[Output2]
Name='Queue'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.4]
MF2='Reasonable':'trapmf',[0.2 0.4 0.6 0.8]
MF3='High':'trapmf',[0.6 0.8 1 1]

[Output3]
Name='Route'
Range=[0 1]
NumMFs=3
MF1='Empty':'trimf',[0 0 0.3]
MF2='Normal':'trapmf',[0 0.3 0.6 0.8]
MF3='Overload':'trapmf',[0.6 0.8 1 1]

[Output4]
Name='Energy'
Range=[0 1]
NumMFs=5
MF1='Controlled':'trapmf',[0 0 0.1 0.3]
MF2='Regular':'trimf',[0.1 0.3 0.5]
MF3='Attention':'trimf',[0.3 0.5 0.7]
MF4='Critical':'trimf',[0.5 0.7 0.9]
MF5='Urgent':'trapmf',[0.7 0.9 1 1]

[Rules]
1 1 1 3, 1 1 2 1 (1) : 1
1 2 1 3, 1 1 2 1 (1) : 1
1 3 1 3, 1 2 2 2 (1) : 1
1 4 1 3, 1 2 2 2 (1) : 1
1 5 1 3, 1 3 2 3 (1) : 1
1 6 1 3, 1 3 2 4 (1) : 1
1 1 2 3, 2 1 2 1 (1) : 1
1 2 2 3, 2 1 2 1 (1) : 1
1 3 2 3, 2 2 2 2 (1) : 1
1 4 2 3, 2 2 2 2 (1) : 1
1 5 2 3, 2 3 2 3 (1) : 1
1 6 2 3, 2 3 2 4 (1) : 1
1 1 3 3, 3 1 2 1 (1) : 1
1 2 3 3, 3 1 2 1 (1) : 1
1 3 3 3, 3 2 2 2 (1) : 1
1 4 3 3, 3 2 2 2 (1) : 1
1 5 3 3, 3 3 2 3 (1) : 1
1 6 3 3, 3 3 2 4 (1) : 1
1 1 2 2, 1 1 2 1 (1) : 1
1 2 2 2, 1 1 2 1 (1) : 1
1 3 2 2, 1 2 2 2 (1) : 1
1 4 2 2, 1 2 2 2 (1) : 1
1 5 2 2, 1 2 2 3 (1) : 1
1 6 2 2, 1 2 2 4 (1) : 1
1 1 1 2, 1 1 2 1 (1) : 1
1 2 1 2, 1 1 2 1 (1) : 1
1 3 1 2, 1 1 2 2 (1) : 1
1 4 1 2, 1 1 2 2 (1) : 1
1 5 1 2, 1 2 2 3 (1) : 1
1 6 1 2, 1 2 2 4 (1) : 1
1 1 3 2, 3 2 2 1 (1) : 1
1 2 3 2, 3 2 2 1 (1) : 1
1 3 3 2, 3 2 2 2 (1) : 1
1 4 3 2, 3 2 2 2 (1) : 1
1 5 3 2, 3 3 2 3 (1) : 1
1 6 3 2, 3 3 2 4 (1) : 1
1 1 1 1, 1 1 1 1 (1) : 1
1 2 1 1, 1 1 1 1 (1) : 1
1 3 1 1, 1 2 1 2 (1) : 1
1 4 1 1, 1 2 1 2 (1) : 1
1 5 1 1, 1 2 1 3 (1) : 1
1 6 1 1, 1 2 1 4 (1) : 1
1 1 2 1, 2 1 2 1 (1) : 1
1 2 2 1, 2 1 2 1 (1) : 1
1 3 2 1, 2 2 2 2 (1) : 1
1 4 2 1, 2 2 2 2 (1) : 1
1 5 2 1, 2 2 2 3 (1) : 1
1 6 2 1, 2 2 2 4 (1) : 1
1 1 3 1, 3 1 1 1 (1) : 1
1 2 3 1, 3 2 1 1 (1) : 1
1 3 3 1, 3 2 2 2 (1) : 1
1 4 3 1, 3 2 2 2 (1) : 1
1 5 3 1, 3 2 2 3 (1) : 1
```

```
2 3 1 2, 1 2 2 3 (1) : 1
2 4 1 2, 1 2 2 3 (1) : 1
2 5 1 2, 1 2 2 4 (1) : 1
2 6 1 2, 1 2 2 4 (1) : 1
2 1 1 3, 2 1 1 3 (1) : 1
2 2 1 3, 2 1 2 3 (1) : 1
2 3 1 3, 2 2 2 3 (1) : 1
2 4 1 3, 2 2 2 3 (1) : 1
2 5 1 3, 2 2 2 4 (1) : 1
2 6 1 3, 2 2 2 4 (1) : 1
2 1 2 1, 1 1 1 3 (1) : 1
2 2 2 1, 1 1 1 3 (1) : 1
2 3 2 1, 1 1 2 3 (1) : 1
2 4 2 1, 1 1 2 3 (1) : 1
2 5 2 1, 2 2 2 4 (1) : 1
2 6 2 1, 2 2 2 4 (1) : 1
2 1 3 1, 2 1 2 3 (1) : 1
2 2 3 1, 2 1 2 3 (1) : 1
2 3 3 1, 3 2 2 3 (1) : 1
2 4 3 1, 3 2 2 3 (1) : 1
2 5 3 1, 3 3 2 4 (1) : 1
2 6 3 1, 3 3 2 4 (1) : 1
2 1 2 2, 2 1 2 3 (1) : 1
2 2 2 2, 2 1 2 3 (1) : 1
2 3 2 2, 2 2 2 3 (1) : 1
2 4 2 2, 2 2 2 3 (1) : 1
2 5 2 2, 3 2 2 4 (1) : 1
2 6 2 2, 3 2 2 4 (1) : 1
2 1 2 3, 2 1 1 3 (1) : 1
2 2 2 3, 2 1 2 3 (1) : 1
2 3 2 3, 2 2 2 3 (1) : 1
2 4 2 3, 2 2 2 3 (1) : 1
2 5 2 3, 3 3 2 4 (1) : 1
2 6 2 3, 3 3 2 4 (1) : 1
2 1 3 2, 2 2 2 3 (1) : 1
2 2 3 2, 2 2 2 3 (1) : 1
2 3 3 2, 2 3 2 3 (1) : 1
2 4 3 2, 2 3 2 3 (1) : 1
2 5 3 2, 3 3 2 4 (1) : 1
2 6 3 2, 3 3 2 4 (1) : 1
2 1 3 3, 3 1 2 3 (1) : 1
2 2 3 3, 3 1 2 3 (1) : 1
2 3 3 3, 3 2 2 3 (1) : 1
2 4 3 3, 3 2 2 3 (1) : 1
2 5 3 3, 3 3 2 4 (1) : 1
2 6 3 3, 3 3 2 4 (1) : 1
3 1 1 1, 1 3 1 3 (1) : 1
3 2 1 1, 1 3 1 3 (1) : 1
3 3 1 1, 1 3 2 3 (1) : 1
3 4 1 1, 1 3 2 3 (1) : 1
3 5 1 1, 1 3 2 4 (1) : 1
3 6 1 1, 1 3 3 4 (1) : 1
3 1 1 2, 1 3 2 3 (1) : 1
3 2 1 2, 1 3 2 3 (1) : 1
3 3 1 2, 2 3 2 3 (1) : 1
3 4 1 2, 2 3 2 3 (1) : 1
3 5 1 2, 3 3 2 4 (1) : 1
3 6 1 2, 3 3 2 4 (1) : 1
3 1 1 3, 2 3 2 3 (1) : 1
3 2 1 3, 2 3 2 3 (1) : 1
3 3 1 3, 3 2 2 3 (1) : 1
3 4 1 3, 3 2 2 3 (1) : 1
3 5 1 3, 3 3 2 4 (1) : 1
3 6 1 3, 3 3 2 4 (1) : 1
3 1 2 1, 1 2 2 3 (1) : 1
3 2 2 1, 1 2 2 3 (1) : 1
3 3 2 1, 2 2 2 4 (1) : 1
3 4 2 1, 2 2 2 4 (1) : 1
3 5 2 1, 2 3 2 5 (1) : 1
3 6 2 1, 2 3 2 5 (1) : 1
3 1 2 2, 2 2 2 3 (1) : 1
3 2 2 2, 2 2 2 3 (1) : 1
3 3 2 2, 2 2 2 4 (1) : 1
3 4 2 2, 2 2 2 4 (1) : 1
3 5 2 2, 3 2 2 5 (1) : 1
3 6 2 2, 3 2 2 5 (1) : 1
3 1 2 3, 2 2 2 3 (1) : 1
3 2 2 3, 2 2 2 3 (1) : 1
3 3 2 3, 3 3 2 4 (1) : 1
3 4 2 3, 3 3 2 4 (1) : 1
3 5 2 3, 3 3 3 5 (1) : 1
3 6 2 3, 3 3 3 5 (1) : 1
3 1 3 1, 2 2 2 4 (1) : 1
3 2 3 1, 2 2 2 4 (1) : 1
3 3 3 1, 2 3 2 4 (1) : 1
3 4 3 1, 2 3 2 4 (1) : 1
3 5 3 1, 3 3 2 5 (1) : 1
3 6 3 1, 3 3 2 5 (1) : 1
3 1 3 2, 3 2 2 4 (1) : 1
3 2 3 2, 3 2 2 4 (1) : 1
3 3 3 2, 3 2 2 4 (1) : 1
3 4 3 2, 3 2 2 4 (1) : 1
3 5 3 2, 3 3 3 5 (1) : 1
3 6 3 2, 3 3 3 5 (1) : 1
3 1 3 3, 3 3 2 4 (1) : 1
3 2 3 3, 3 3 2 4 (1) : 1
3 3 3 3, 3 3 2 4 (1) : 1
3 4 3 3, 3 3 3 5 (1) : 1
3 5 3 3, 3 3 3 5 (1) : 1
3 6 3 3, 3 3 3 5 (1) : 1
1 6 3 1, 3 2 2 4 (1) : 1
2 1 1 1, 1 2 1 3 (1) : 1
2 2 1 1, 1 2 1 3 (1) : 1
2 3 1 1, 1 2 1 3 (1) : 1
2 4 1 1, 1 2 1 3 (1) : 1
2 5 1 1, 1 3 2 4 (1) : 1
2 6 1 1, 1 3 2 4 (1) : 1
2 1 1 2, 1 1 2 3 (1) : 1
2 2 1 2, 1 1 2 3 (1) : 1
```

APÊNDICE V

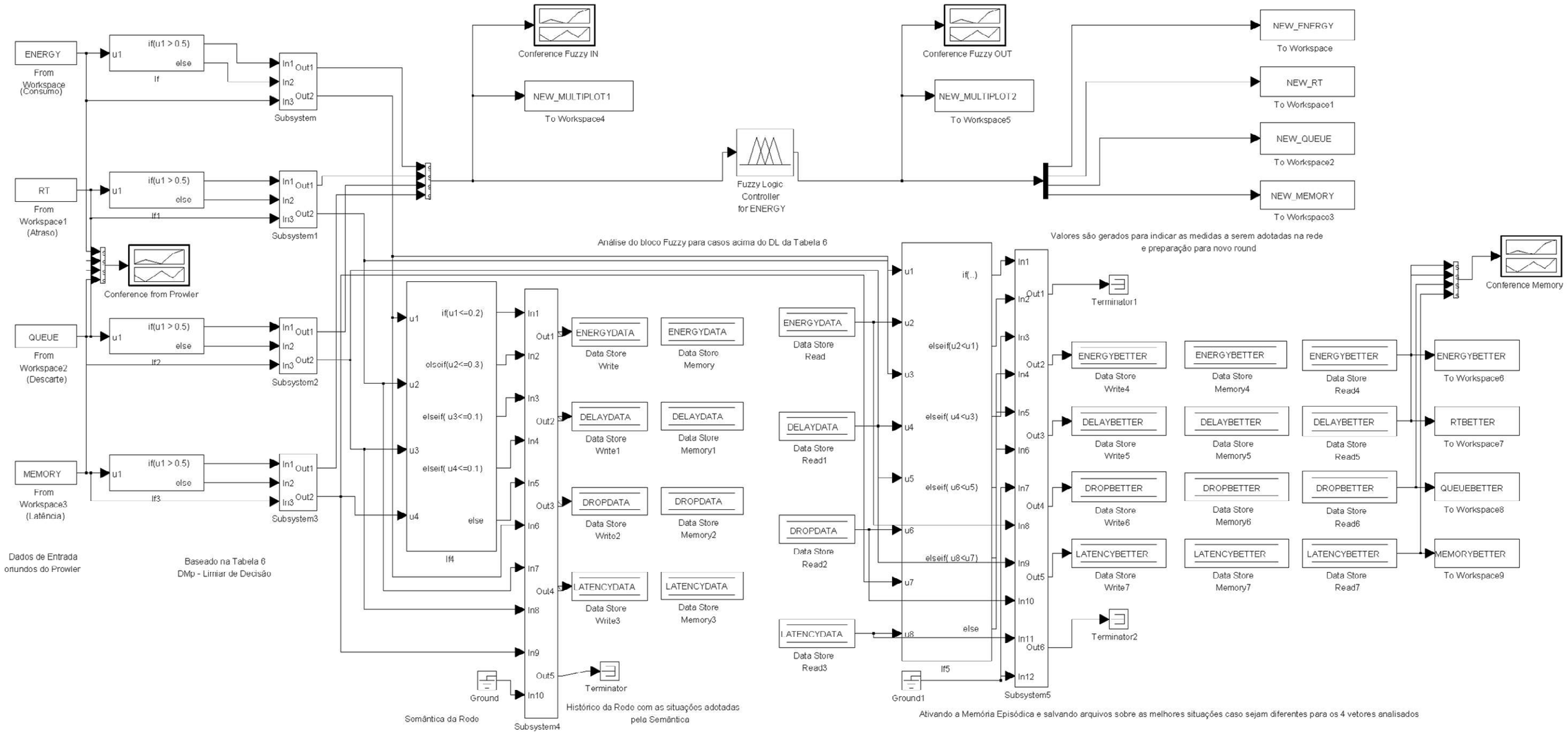
Módulo do CPMod



Escola Politécnica da Universidade de São Paulo	Escola Politécnica da Universidade de São Paulo – EPUSP			
	Estrutura do CPMod para o primeiro *round*.			
Orientador do doutorado	TAMANHO A3	Ano 2016	Elaborado por: Marcel Stefan Wagner	REV 01
Prof. Dr. Miguel Arjona Ramírez	ESCALA 1:100	Data: 08/04/2016	PLANILHA 1 DE 1	

APÊNDICE VI

Módulo de correção do CPMod



Escola Politécnica da Universidade de São Paulo	Escola Politécnica da Universidade de São Paulo – EPUSP			
	Estrutura do CPMod para os *rounds* de correção dos parâmetros da rede.			
Orientador do doutorado	TAMANHO A3	Ano 2016	Elaborado por: Marcel Stefan Wagner	REV 01
Prof. Dr. Miguel Arjona Ramírez	ESCALA 1:100	Data: 08/04/2016	PLANILHA 1 DE 1	

ANEXOS

ANEXO I	–	Artigo apresentado no IEEE CSNT 2015
ANEXO II	–	Artigo publicado no IJCSNT 2016
ANEXO III	–	*Datasheet* do MICAz (família MICA *motes*)

ANEXO I

Artigo apresentado no IEEE CSNT 2015

Adaptive Cognitive System Applied on Wireless Sensor Networks Nodes Decisions

Marcel Stefan Wagner[1], Miguel Arjona Ramirez[2], Wagner Luiz Zucchi[3]
Department of Electronic Systems Engineering
Polytechnic School, University of São Paulo, São Paulo, Brazil
[1]marcel.wagner@usp.br [2]miguel@lps.usp.br [3]wzucchi@lps.usp.br

***Abstract*—The system presented here is based on the concept of cognition applied to Wireless Sensor Networks (WSN) concerning aspects related to memory, history and decision making over network nodes tasks. The use of cognitive features in the WSN scope allows the routing protocols to make better decisions about conflicts or anomalies generated by node or route fails that probably will affect the performance network as a whole. However, with the use of cognitive aspects regarding feedback processes to make decisions, such as adjustments in memory, queue and table routes, it is possible to obtain an improvement in the data transmitted by nodes. As sequence of this work, other rules, metrics and parameters will be added to the analysis aiming at the algorithm enhancement.**

Keywords-WSN; Cognitive Network; Decision Making

I. INTRODUCTION

Wireless Sensor Network (WSN) represents a major network research area for the reason that it fosters different kinds of applications and interests that range from a simple exchange of information to the transmission and reception of significant data. Safety aspects are not only the important parameters to be explored in WSN. Algorithms and protocols focused on the transmission speed, packet delivery increase and node energy economy are well surveyed in this field, thus features related to performance and Quality of Service (QoS) are important for the network success and acceptance.

Focusing on the improving of network performance, this work applies the combination of cognitive features within WSN nodes to introduce certain intelligence based on memory and critical analysis to provide the decision making process, that will be described in the section IV. The combination of cognition with WSN to solve network problems reflects the need for studies encompassing different areas of expertise in search of elaborate alternative solutions.

Cognitive parameters can be inserted in logical mechanisms as the system's intelligent part, allowing it to act on the collected data obtained from network feedback vectors according to history and memory aspects, observing yet the network's policies and purposes. Once the data are gathered, the cognitive process is able to proceed to the decision making stage.

To provide nodes decisions, a decision-threshold analysis is made with system's critical table values collected from the network monitoring vector parameters, as described in next sections. These vectors are essential for system analysis because they are source for critical tables.

Some issues concerning this research area treats the theme focusing on prediction routing scheme, such as Jie Li et al [1], that suggests a routing algorithm based on a traffic prediction model known as Efficient Traffic Aware Multi-path Routing (ETAMR). The ETAMR considers traffic distribution and load to build a multi-path routing using a prediction model. Metrics such as nodes energy consumption or even network throughput are not considered in this model and are the matter of their future works.

In contrast with [1], instead of relying on prediction mechanism to change some aspects of network routing, the present work uses the actual readings of network data, collected in pre-determinated intervals, to execute cognitive process and provide actuation on network aspects such as routing. This provides allowing more coherent alterations, not only in routing aspects, but in the nodes processing schemes and buffer operation.

The algorithm developed here emphasizes the QoS regarding end-to-end packet delivery ratio among network nodes in scenarios with random node's movement. The insertion of nodes' energy data vector and the monitoring of this aspect will be conducted in a sequence study. Therefore, control over the amount of energy consumption in the nodes will be approached as part of the cognitive process as a sequence of this work.

In [10] another prediction model is presented and in [2] and [11] the importance of WSN integration and application in IoT (Internet of Things) meaning are considered. [3] and [5] gives us examples of WSN simulations using NS-2 software. [6] and [7] treat about cognition applied into WSN environments. [4], [8] and [9] have examples of certain difficulties or anomalies concerning sensor networks.

The rest of paper is organized as follows. Section II describes the node's internal architecture and interface into block's diagrams. Section III represents the environment description and important features related to nodes dimension. Section IV corresponds to the system's development and organization. Section V shows the results. Finally, concluding remarks and future survey directions are given in Section VI.

II. NODE DESCRIPTION

It's important to know the node's internal architecture, as well as the modules connections of sensor nodes. Therefore, it can be viewed ahead the internal structure of the sensor studied in this paper.

978-1-4799-1797-6/15 $31.00 © 2015 IEEE
DOI 10.1109/CSNT.2015.65

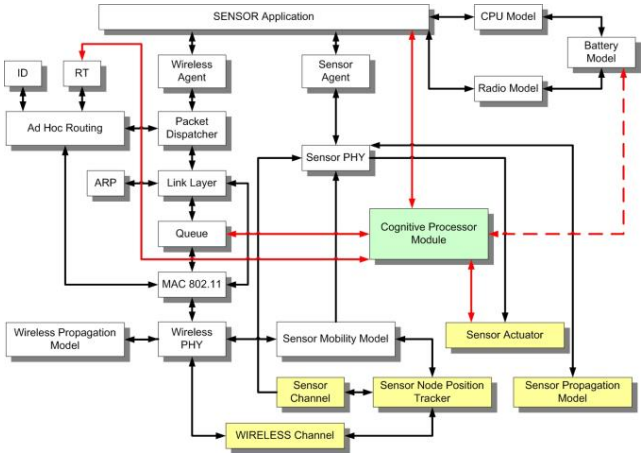


Figure 1. The general node structure with cognitive module.

As can be seen in Fig. 1, the presented structure corresponds to a common sensor node with the insertion of a cognitive module responsible for the execution of cognitive processes. In the scheme, white blocks represent the traditional internal interface of sensor nodes, yellow blocks are the medium representation and green block represents the cognitive module developed in this work.

The Cognitive Processor Module (CPMod) is the intelligent sensor part with cognitive aspects engaged to execute the Decision Making process (DMp) for acting over sensor applications concerning memory and changing queue modeling according to the necessity. In the following, ID corresponds to the node identification and RT to the route tables. ARP is the Address Resolution Protocol and acts according to 802.11 standard for Wireless Local Area Network (WLAN). The continuous red interfaces were implemented here and dashed line corresponds to the next step of this work in the survey about cognitive energy control.

Note that the cognitive module operates in a higher layer concerning the node's internal structure. Moreover, the lower modules will continue to execute their tasks normally, although the decision making running in the cognitive module will affect some internal parameters such as queues rules, buffers allocations and table routes (i.e. ordering to optimize routing protocols operating in the node) to accomplish network performance adjustments.

III. Environment and Node Details

This work considers the application of dynamic or differentiated media. The study comprises networks with random node movement, so it is not limited to a single scenario. The network environment is variable, thus any link between two nodes should be reestablished if any alteration in the environment occur or due to the changes in nodes positions, so the system can adapt to it and to its logical routes configurations.

To identify the node dimensions for the environment applied, the system-related variables are given by (1):

$$F_V = \left(\alpha \times P_V + \beta \times R_V + \gamma \times L_V + \sigma \times N_V\right) \qquad (1)$$

From (1), F_V represents the node variability and it's composed by the following: P_V, accounting for the node position variation; R_V, accounting for the node radius variation; L_V represents the variation in the number of links established by nodes and N_V is the number of nodes variation in the network.

The variables P_V, R_V, L_V and N_V are obtained from (2)-(5) and the relations in the variability detection are presented in pseudo-code part posted bellow, where t represents current time and t-1 represents time in the previous step.

Algorithm for sensor variability detection

$N = Number\ of\ nodes\ into\ network;$

$n = 0;$

$t = 0;$

$WHILE\ \left(n \le N\right)$

$WHILE\ \left(t > 0\right)$

$LET\ \ P_{V(n)} = \phi_0,\ \ \alpha_{(n)} = \theta_0$

$R_{V(n)} = \phi_1,\ \ \beta_{(n)} = \theta_1$

$L_{V(n)} = \phi_2,\ \ \gamma_{(n)} = \theta_2$

$N_{V(n)} = \phi_3,\ \ \sigma_{(n)} = \theta_3$

$\left\{ P_{V(n)},\ R_{V(n)},\ L_{V(n)}\ and\ N_{V(n)}\ are\ readings\ of\ network\ by\ nodes\ measurements. \right\}$

$FOR\quad i = 0\ \ to\ \ 3$

$$DO\quad \theta_i = \frac{1}{4} \cdot \left| \frac{\phi_{i(t)} - \phi_{i(t-1)}}{\arg\max(\phi_{i(t)}, \phi_{i(t-1)})} \right|$$

$$F_{V\,i(n)} = \prod_{i=0}^{3} \theta_i \phi_{i(t)}$$

$$F_{V(n)} = \sum_{i=0}^{3} F_{V\,i(n)}$$

$\left\{ F_{V(n)}\ represents\ the\ percentage\ relative\ to\ the\ variability\ in\ n^{th}\ node. \right\}$

$END\ \ FOR$

$t = t + 1$

$n = n + 1$

$END\ \ WHILE$

$END\ \ WHILE$

The system variability-related variables, P_V, R_V, L_V and N_V are obtained from the following sequence:

$$P_v = \left\{ P_t - P_{t-1} \mid t \ge 1;\ P = \int_C F_{(X,Y)} dS = \int_a^b f_{(x,y)} \cdot \sqrt{1 + \left(y'\right)^2}\, dx \right\} \qquad (2)$$

From the above, P_V is the position analysis result, comparing the present node location with the previous node location. P denotes the node trajectory in a two-dimensional plan (x, y) from point a to b.

$$R_v = \left\{ R_t - R_{t-1} \mid t \ge 1;\ 0 \le r \le \infty;\ r_{t-1} \le R \le \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr \right) \right\} \qquad (3)$$

The prior procedure corresponds to the antenna radius simplified model, where R_V denotes the signal power in the

maximum radius border with connection possibility and r is related to the antenna's reach radius. After that, connections will probably fail with 70% chance and current connections will drop with over 85%.

$$L_v = \left\{ L_t - L_{t-1} \mid t \geq 1;\ 0 \leq L \leq (N-1) \right\} \tag{4}$$

With respect to the L_V parameter, L is associated to the number of links established by nodes.

$$N_v = \left\{ N_t - N_{t-1} \mid t \geq 1 \right\} \tag{5}$$

Equation (5) shows N_V as the current number of nodes in the network. It can be changed according to the insertion or loss of nodes in the network. The proportion weights among P_V, R_V, L_V and N_V are the multiplicative factors α, β, γ and σ respectively, and are given by:

$$\sum(\alpha,\beta,\gamma,\sigma) \leq 1 \begin{cases} \alpha = \frac{1}{4} \cdot \left| \frac{P_t - P_{t-1}}{\arg\max(P_t - P_{t-1})} \right|, \forall \alpha \in \Re \mid 0 \leq \alpha \leq 25\% \\ \beta = \frac{1}{4} \cdot \left| \frac{R_t - R_{t-1}}{\arg\max(R_t - R_{t-1})} \right|, \forall \beta \in \Re \mid 0 \leq \beta \leq 25\% \\ \gamma = \frac{1}{4} \cdot \left| \frac{L_t - L_{t-1}}{\arg\max(L_t - L_{t-1})} \right|, \forall \gamma \in \Re \mid 0 \leq \gamma \leq 25\% \\ \sigma = \frac{1}{4} \cdot \left| \frac{N_t - N_{t-1}}{\arg\max(N_t - N_{t-1})} \right|, \forall \sigma \in \Re \mid 0 \leq \sigma \leq 25\% \end{cases} \tag{6}$$

Equation (6) refers to the proportions related to the system's variability factors: α, the position ratio; β, the node radius ratio; γ, the ratio of the node's possible connections and σ, the ratio of number of nodes in the network. The relationship among system variables P_V, R_V, L_V and N_V is depicted in Fig. 2. The number of nodes in a unique network must be defined (or monitored, if it already exists) based on the monitored/applied environment requirements.

In other words, to accomplish the environment necessities is necessary to know the real application of sensor nodes. If the environment being monitored is small and requires few nodes, the number of nodes N is a known parameter instead of a variable one, constraining the variability feature to three variables and the multiplicative factors assume 1/3 ratio unless 1/4. If the random enrolment of nodes to the network is required, then should be used the complete equation with four variables, as given in (1).

According to (2)-(5), if the subtraction shows result 0 the system variable can assume binary value 0 corresponding to no variation in the network parameter, and if exists another value in subtraction result, it indicates that there is occurrence of variation in the parameter affording binary value 1 (it is for P_V, R_V, L_V and N_V values applied in (1)). The cognitive process proceeds with the monitoring of nodes applying the relationships given in (1) to (6) for each node, and then calculates the variability feature F_N that is the F_V average.

$$F_N = \overline{X}_{(F_{V(n)})} = \frac{\sum_{n=1}^{N} F_{V(n)}}{N} \tag{7}$$

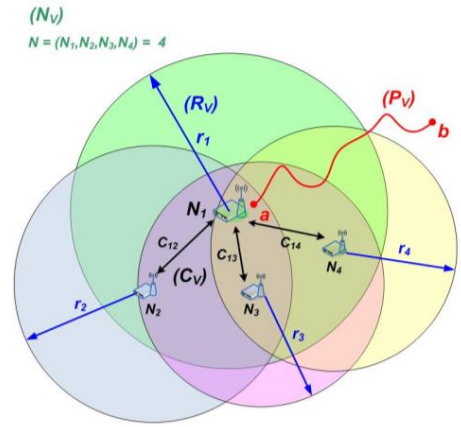


Figure 2. The general features of nodes connection in the network.

In pseudo-code described before, the calculation round find out the parameters and respective ratios to facilitate the F_N achievement. F_N is one of the parameters used by the cognitive process running into CPMod to provide Decision Making process (DMp) and it will be described in next section.

IV. System Features and Development

The cognitive process uses the memory vector V_M, the variability feature F_N (described in previous section) and the input signal S_I to generate the output signal S_O that controls the system with cognitive process. In the cognitive process the comparator result is the error (ε) between the received signal S_I and the generated signal S_O and it is used to system adequacy.

For a given input signal S_I composed by network variability feature and memory vector (the memory vector will be described further, but it comprises data obtained from the network's nodes regarding connections successes or failures in addition to route information), the cognitive process uses F_N, V_M and ε to identify possible problems in the network and to act on the output signal S_O with a control vector V_C.

Fig. 3 illustrates the interfaces and connections taking place in the cognitive process regarding internal development of CPMod. The signal S_I derives from the combination of F_N and V_M, while V_C acts on S_O. F_N and V_M are indicators of system environment situation. With the analysis of these two parameters, S_I signal can be composed for error calculation. The results applying these indicators are shown in the section V.

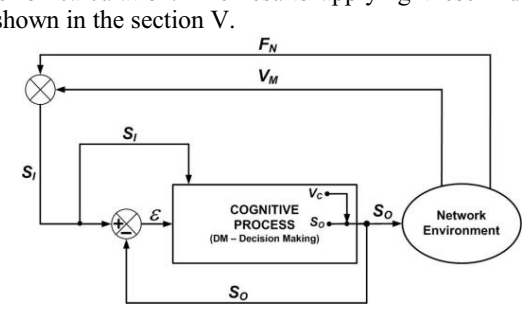


Figure 3. Block diagram of the cognitive process interconnection.

A. Memory Vector

The Memory Vector V_M is in the core of the adaptive cognitive module studied here since it allows the cognitive process to store information regarding network and its nodes, establishing the history of the network situations and memory. This vector results from the compilation of instant data regarding the monitored environment within time slots after each 5 seconds is recommended. With V_M it is possible to act on the environment using the DMp to generate *DM* vectors.

$$V_M = (V_S, V_F, V_R) \tag{8}$$

Equation (8) introduces V_M matching to the memory vector composed by: V_S as the vector containing the successes of nodes connections; V_F as the vector containing the failures in the nodes and V_R as a vector of valid routes collected from RT. The components of V_S, V_F and V_R are described in Table I.

TABLE I. MEMORY VECTOR COMPONENTS

Vector Type	Components and Description		
	Description	***Components***	***Decision***
V_S	Success Vector	Successful end-to-end connection, End-to-end delay, Rate of effectively delivered data.	By the routes with a higher probability of end-to-end packet delivery success.
V_F	Failure Vector	Node disconnected three times in a row, Unresponsive node with reply message (RREP).	Avoiding these nodes for a given period of time.
V_R	Route Vector	Valid routes, Origin node, Destination node, Number of hops to destination.	By the best route according to the valid routes history.

The V_M application for the *DM* classification will be described ahead and this feature is present on the CPMod as an instance of *DM* analysis, as in (10).

B. Error Vector

$$\text{Parameters} \begin{cases} S_I = (F_N, V_M) \\ S_O = (V_C, V_M) \end{cases} \tag{9}$$

$$\therefore \ \varepsilon = \frac{|S_I - S_O|}{S_O} \cdot 100 \ (\%) \ , \ 0 \le \varepsilon \le 100\%$$

Basically, the error vector ε works with the relative error of the cognitive module measurements input S_I and output S_O signals. S_I is composed by the variability feature and memory vector, and S_O works with control vector (described in *D* division of this section) and memory vector. The composition of this feature appears in the third parameter for *DM* analysis, as shown in (10).

C. Decision Making Vector

The procedure for Decision Making process (DMp) is conducted by the cognitive process and takes into account the environment variability feature F_N, the memory vector V_M and the error ε, seeking for the optimal network configuration and the adjustment of the employed routing protocol.

DM is assigned according to the expected quality of the task and its action is directly related to the Decision Limit (*DL*). The threshold *DL* corresponds to a probabilistic value for the adjustment of the system resulting from the combination of F_N, V_M and ε according to the range entrance (*DM* description) given in Table II.

$$DM = (F_N, V_M, \varepsilon) \tag{10}$$

Five abstraction levels were adopted since they represent the network conditions and allow V_C actuation. *DL* is calculated using the inverse of the average of *DM* composition as described in (11).

TABLE II. DECISION MAKING ACCORDING TO THE THRESHOLD

***DM* Description**	***DL* Configuration**	
	Decision Limit (%) Range [a]	***Action about DM (with DL)***
Urgent	80,0 → 100	Reinitiates the routing table in the nodes. The process evaluates the index (F_N, V_M, ε) originating the urgent situation. V_C is sent to nodes.
Critical	70,0 → 79,9	The process evaluates memory (V_M) and error (ε). Adjusts signal S_O and reads *DL* again. If the situation is maintained a control vector V_C is sent to the node.
Attention	50,0 → 69,9	Adjusts S_O with signal correction according to detected error ε.
Regular	30,0 → 49,9	No need for changes but decisions may be taken to improve QoS. The system is operating normally.
Controlled	0 → 29,9	Network without need for changes. The system is operating in better case.

Measurement level regarding the network situation.

As mentioned before, *DL* is calculated based on the inverse of *DM* average, according to (11). The graphical relationship for *DL* is depicted in Fig. 4.

$$\overline{X}_{(F_N, V_M, \varepsilon)} = \frac{\sum_{i=1}^{N} (F_{N_i}, V_{M_i}, \varepsilon_i)}{N} \Rightarrow \tag{11}$$

$$\Rightarrow \begin{cases} \overline{X}_{(F_N, V_M, \varepsilon)} \ge 1, \ DL_{(\%)} = \dfrac{1}{\overline{X}_{(F_N, V_M, \varepsilon)}} \times 100 \ (\%) \\ \overline{X}_{(F_N, V_M, \varepsilon)} < 1, \ DL_{(\%)} = 100 \ \% \end{cases}$$

The smaller *DL*, in percentage, the better is the network condition and, similarly, the greater *DL* the worst is the network situation, therefore more significant actions should be taken by the CPMod. Furthermore, in order to keep the network with controlled or regular *DM* level, $\overline{X}$ must be high.

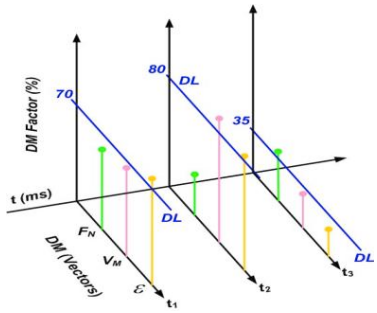


Figure 4. Example of relationship between *DL* and *DM*.

In other case, the lower is the average $\overline{x}$ in (11) for memory, variability and error vectors, more critical the network condition will be. It can be found in Fig. 4 an example of a *DM* data reading composed by F_N, V_M and ε vectors, distributed in times t_1, t_2 and t_3.

D. Control Vector

The CPMod with *DM* sends a control data vector in S_O to define the nodes alterations according to the necessity of change by using a control vector V_C. This vector is defined in (12) and can be adapted to other actions according to the network's needs and to alterations in the settings (such as the insertion of energy parameter in the future development).

$$V_C = (Memory, Queue, RT) \tag{12}$$

With the settlements of V_C on the output signal S_O it is possible to organize the nodes. The nodes receive and make alterations regarding memory amount sizing through the final control vector to execute tasks. Queues and Route Tables (RT) can be addressed in this time, correcting the issues related with these features. The cognitive processing allows the system (in a general way) to readjust and, at the same time, execute the network monitoring task. This cognitive system also allows adaptation regarding monitored aspects (e.g., the insertion of a monitoring vector of energy levels at the nodes for a possible control by the cognitive process and a raise in each node's yield according to *DL* adequacy) as well as *DM* and V_C routing in S_O.

V. Results and Comments

Added either as vectors or only as another variable to the network data reading, the mechanism gains robustness but tends to lose processing response time since the cognitive process needs to conduct a larger number of analysis with the insertion of new monitoring elements, however this case occur when the system is running in the initial stage.

In the study scenario were used 0-100 sensor nodes with possibility to each node establishing 0-9 connections, and each node has antenna radius variability of 1-5 meters and displacement of 1-10 meters in random trajectory. For simplicity, was adopted a basic antenna model. The data collected from nodes were generated randomly. As mentioned before, the CPMod operates according with network/nodes situation. Nevertheless, the procedure to execute data analysis (described in previous section) is based on Table II and it is explained as follows.

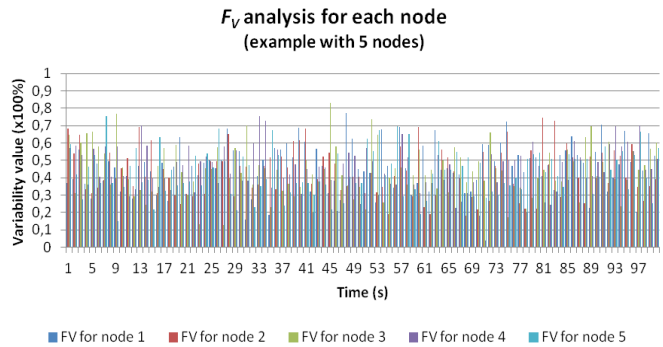


Figure 5. Graphical analysis concerning F_V variability for 5 nodes.

The graphs were obtained from the random nodes statistics samples, applying the expressions mentioned in previous sections. From the Fig. 5, it is possible to identify the relation between Table II and variability values in the graph. As can be seen, the most of data collected about variability is in the range of 20% to 50%, so according to the Table II, these 5 nodes are operating normally in the situation is controlled and regular. *DL* is present here according to each peak signal.

There are yet a big amount of circumstances concerning attention condition and few cases in that nodes require action with V_C intervention, because they achieve the critical level. In this sampling, approximately in 45 seconds occurred an urgent condition. Therefore, in this case, a V_C was sent to the respective node (node 3) to perform the adjustment task (queue scheme modification from First-In Last-Out (FILO) to First-In First-Out (FIFO)).

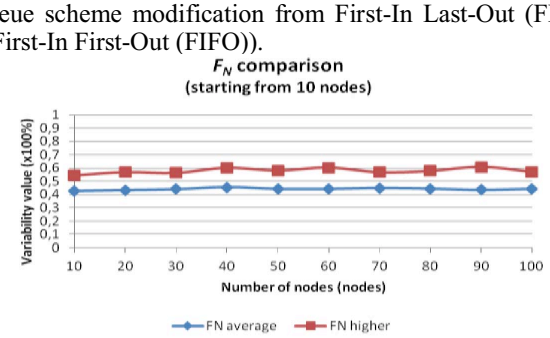


Figure 6. Graphic representing the F_N comparison for variability.

This graph shows us the occurrence of stable condition in the system, this means that the nodes are operating in approximately 45% of variability, according to the position, antenna radius, possible links and number of nodes. The higher situations are appearing over 50% and less than 70%, so in the attention range according to Table II.

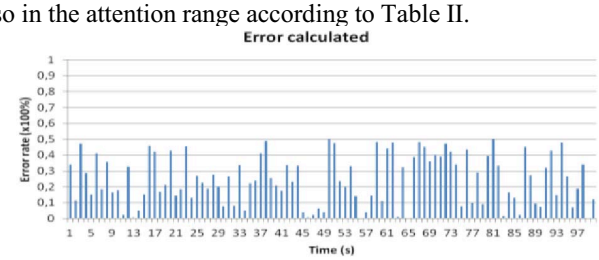


Figure 7. Error rate concerning S_I and S_O.

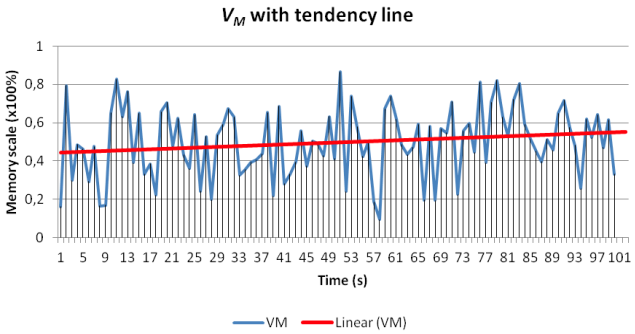


Figure 8. V_M with tendency line according to the memory status.

From the above graph, is understood that with the network time running, the memory usage tends to be higher according to the tendency line. It can be seen that exists 6 peaks surrounding 80% memory usage, and in that cases V_C is sent to the respective nodes to try corrections in memory, queue or route tables. The idea that this mechanism will overload the network with unnecessary data packets seems reasonable but, when the sensors/actuators are synchronized within the nodes and respective features after the initial setting stage, the system pre-stabilizes and acts according to the alterations done by the CPMod.

Finally, using F_N, V_M and ε parameters, the DMp can be executed according to the Table II to regularize the nodes situations regarding memory, queue and route adjustments. *DL* is obtained too from these parameters and tends to be near to their higher values, concerning the average relation described in (11). The real strictly direct correlation between CPMod process and QoS assurance is not depicted from the above graphs, but it will be treated in a sequence work.

VI. Conclusions

The cognitive process adds intelligence aspects to the Wireless Sensor Networks control mechanism, attaching memory, history and decision making. These features reflect some inherent aspects of human beings that represent a strong tool to the improvement of QoS in WSN. With the use of measurement, monitoring and analysis mechanisms by nodes where the cognitive process occur, it is possible to improve the network metrics with the actuation of Decision Making process. The *DL* verification represents a relation of the vectors that integrates the *DM* to provide the system with the action rules sent by V_C. This procedure guarantees the establishment of actions directly in the nodes and, therefore, an improvement in the performance regarding to the data packet delivery. In fact, with CPMod insertion into sensor nodes, more internal tasks in processing are needed, but this control results in a great gain concerning error corrections in the nodes mechanisms, queue and routing tables adjustments and in the node memory usage for sensor applications.

For future researches some issues should be treated as follows: more nodes can be inserted into *DM* analysis, the feature concerning energy usage can be explored expanding F_N, V_M and V_C, another control scheme can be defined as complementary solution such as the use of Fuzzy logic in the DMp, and comparisons with other WSN protocols concerning control and metrics improvement in the network point of view.

Acknowledgment

M.S.W. thanks all his coworkers and friends and especially his wife for participating in the accomplishment of this work that is a part of his PhD thesis.

This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under Grant 307633/2011-0 and by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) under Grant 2012/24789-0.

References

[1] LI, J. et al. Efficient Traffic Aware Multipath Routing Algorithm in Cognitive Networks. Fifth International Conference on Genetic and Evolutionary Computing (ICGEC). Xiamen, September, 2011.

[2] ALCARAZ, C. et al. Wireless Sensor Networks and the Internet of Things: Do We Need a Complete Integration? Computer Science Department of University of Malaga. 1st International Workshop on the Security of the Internet of Things (SecIoT'10). Tokyo, Japan, 2010.

[3] EZREIK, A. and GHERYANI, A. Design and Simulation of Wireless Network using NS-2. 2nd International Conference on Computer Science and Information Technology (ICCSIT'2012). Singapore, April, 2012.

[4] INTANAGONWIWAT, S.; GOVINDAN, R. and ESTRIN, D. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. MobiCOM 2000. Boston, August, 2000.

[5] SINGH, N.; LAL DUA, R. and MATHUR, V. Network Simulator NS2-2.35. International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE). vol. 2, issue 5, May, 2012.

[6] SOMOV, A.; DUPONT, C. and GIAFFREDA, R. Supporting Smart-city Mobility with Cognitive Internet of Things. Future Network & MobileSummit 2013. International Information Management Corporation (IIMC). Portugal, Lisboa, July, 2013.

[7] VIJAY, G.; BDIRA, E. B. A. and IBNKAHLA, M. Cognition in Wireless Sensor Networks: A Perspective. IEEE Sensors Journal. vol. 11, n. 3, March, 2011.

[8] WANG, Q. Traffic Analysis & Modeling in Wireless Sensor Networks and Their Applications on Network Optimization and Anomaly Detection. Macrothink Institute. Network Protocols and Algorithms. vol. 2, n. 1, Sundsvall, Sweden, 2010.

[9] WEI YE et al. Evaluating Control Strategies for Wireless-Networked Robots Using an Integrated Robot and Network Simulation. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2001). Seoul, Korea, May, 2001.

[10] ZHANG, N.; GUAN, J. and XU, C. Traffic Prediction Model for Cognitive Networks. Proceedings of International Conference on Advanced Intelligence and Awareness Internet (AIAI 2011). China, Shenzhen, October, 2011.

[11] ZHANG, M. et al. Cognitive Internet of Things: Concepts and Application Example. International Journal of Computer Science Issues (IJCSI). vol. 9, issue 6, n. 3, November, 2012.

ANEXO III

Artigo publicado no IJCSNT 2016

IJCSNT Vol.5, No.1, 2016
***DOI*- 10.18486/ijcsnt.2016.5.1.01**

Adaptive Cognitive System Applied to WSN Decisions at Nodes with a Fuzzy Logic Approach

Marcel Stefan Wagner
Department of Electronic Systems Engineering, Polytechnic School, University of São Paulo, São Paulo, Brazil
marcel.wagner@usp.br

Miguel Arjona Ramírez
Department of Electronic Systems Engineering, Polytechnic School, University of São Paulo, São Paulo, Brazil
miguel@lps.usp.br

Wagner Luiz Zucchi
Department of Electronic Systems Engineering, Polytechnic School, University of São Paulo, São Paulo, Brazil
wzucchi@lps.usp.br

Abstract
The Adaptive Cognitive System (ACS) presented here is based on the concept of cognition applied to Wireless Sensor Networks (WSN) concerning aspects related to memory, history and decision making over network node tasks. The use of cognitive features in the WSN scope allows the nodes to make better decisions about conflicts or anomalies arising from node or route failures that probably will affect the performance network as a whole. Moreover, with the use of cognitive aspects in feedback processes to make decisions in a multilayer approach, it is possible to obtain an improvement in the data transmitted end-to-end by the nodes. The decision process consists of adjustments in memory, queue, route protocols and energy consumption. A Fuzzy Inference System (FIS) is proposed for decision making from the vector collected from the network, and this logic determines the adjustments to be applied to the network. This inference system is expandable, allowing other rules, metrics and parameters to be added to the analysis for more flexibility and improved performance.

Keywords – Wireless; WSN; Cognitive Network; Fuzzy Logic.

I. INTRODUCTION

Wireless Sensor Networks (WSN) represent a major network research area since it fosters different kinds of applications and suits interests that range from a simple exchange of information to the transmission and reception of significant data. Safety aspects are not only the important parameters to be explored in WSN.

Algorithms and protocols focused on the transmission speed, packet delivery increase and node energy economy are well surveyed in this field. But features related to performance and Quality of Service (QoS) are important for evaluating whether the network meets its goals and for rating its acceptance.

Focusing on the improvement of network performance, this work applies a number of cognitive processes within WSN nodes to introduce intelligence based on memory and critical analysis which provide the means for the decision making process, that will be described in the section IV. The combination of cognition with WSN to solve network problems underlines the need for studies encompassing different areas of expertise in search of elaborate alternative solutions.

Cognitive parameters can be inserted in logical mechanisms as the system's intelligent part, allowing it to act on the collected data obtained from network feedback vectors according to history and memory aspects, observing yet the network's policies and purposes. Once the data are gathered, the cognitive process is able to proceed to the decision making stage.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

In order to provide the means for node decisions, an analysis based on a decision-threshold is made with system's critical table values collected from the network monitoring vector parameters, as described in next sections. These vectors are essential for system analysis because they are the source of the critical tables. Specifically, these vectors are obtained from *Prowler* simulator running over MATLAB platform.

Some researchers in this area treat the theme focusing on prediction routing schemes, such as Jie Li et al [1], who suggest a routing algorithm based on a traffic prediction model known as Efficient Traffic Aware Multi-path Routing (ETAMR). The ETAMR algorithm uses traffic distribution and load to build a multi-path routing using a prediction model. Metrics such as node energy consumption or even network throughput are not considered in this model and are the matter of their future works.

In contrast with [1], instead of relying on a prediction mechanism to change some aspects of network routing, the present work uses the actual readings of network data, collected in predetermined intervals, to execute cognitive processes and provide actuation on network aspects such as routing. This allows for more coherent alterations, not only in routing aspects, but in the nodes processing schemes, buffer operation and energy consumption economy.

The algorithm developed here emphasizes the QoS regarding end-to-end packet delivery ratio among network nodes in scenarios with random node's movement. Therefore, control over the amount of energy consumption in the nodes will be approached as part of the cognitive process.

In [10] another prediction model is presented and in [2] and [11] the importance of WSN integration and application in Internet of Things (IoT) meaning are considered. [3] and [5] gives us examples of WSN simulations using NS-2 software. [6] and [7] treat about cognition applied into WSN environments. [4], [8] and [9] have examples of certain difficulties or anomalies concerning sensor networks.

The rest of paper is organized as follows. Section II describes the node's internal architecture and interface into block's diagrams. Section III represents the environment description and important features related to nodes dimension. Section IV corresponds to the system's development and organization. Section V describes the Fuzzy Logic implementation. Section VI shows the results. Finally, concluding remarks and future survey directions are given in Section VII.

II. NODE DESCRIPTION

It is important to know the node's internal architecture, as well as the modules connections of sensor nodes. Therefore, it can be viewed ahead the internal structure of the sensor studied in this paper.

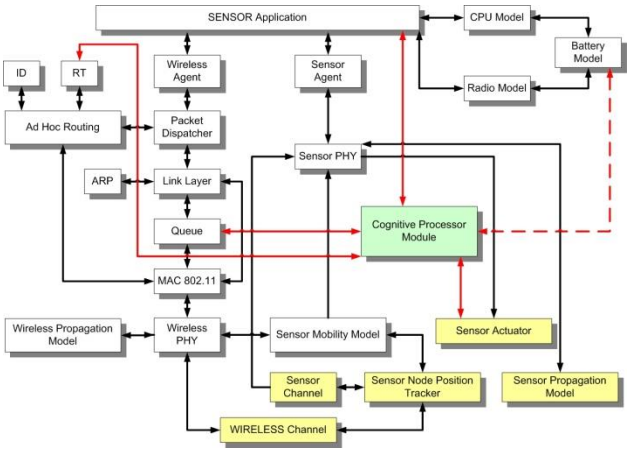


Figure 1. The general node structure with cognitive module.

As can be seen in Fig. 1, the presented structure corresponds to a common sensor node with the insertion of a cognitive module responsible for the execution of cognitive processes.

In the scheme, white blocks represent the traditional internal interface of sensor nodes, yellow blocks are the medium representation and green block represents the cognitive module developed in this work.

The Cognitive Processor Module (CPMod) is the intelligent sensor part with cognitive aspects engaged to execute the Decision Making process (DMp) for acting over sensor layers concerning memory and changing queue modeling according to the necessity. In the following, ID corresponds to the node identification

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

and RT to the route tables. ARP is the Address Resolution Protocol and acts according to 802.11 standard for Wireless Local Area Network (WLAN). The continuous red interfaces were implemented here and dashed line corresponds to the cognitive energy control.

Note that the cognitive module operates in a higher layer concerning the node's internal structure. Moreover, the lower modules will continue to execute their tasks normally, although the decision making running in the cognitive module will affect some internal parameters such as queues rules, buffers allocations, table routes (i.e. ordering to optimize routing protocols operating in the node) and antenna power gain to accomplish network performance adjustments.

III. ENVIRONMENT AND NODE DETAILS

This work considers the application of dynamic or differentiated media. The study comprises networks with random node movement, so it is not limited to a single scenario. The network environment is variable, thus any link between two nodes should be reestablished if any alteration in the environment occur or due to the changes in nodes positions, so the system can adapt to it and to its logical routes configurations.

To identify the node dimensions for the environment applied, the system-related variable is given by (1):

$$F_V = (\alpha.P_V + \beta.R_V + \gamma.L_V + \sigma.N_V) \quad (1)$$

From (1), F_V represents the node variability number between [0,1] for a single trajectory and it is composed by the following: P_V, accounting for the node position variation; R_V, accounting for the node radius variation; L_V represents the variation in the number of links established by nodes and N_V is the number of nodes variation in the network.

The system variability-related variables, P_V, R_V, L_V and N_V are obtained from the following sequence. To define the position variable P_V, were used the random motion feature of the N_1 node with the displacement from *a* to *b*, as is shown in Fig. 2.

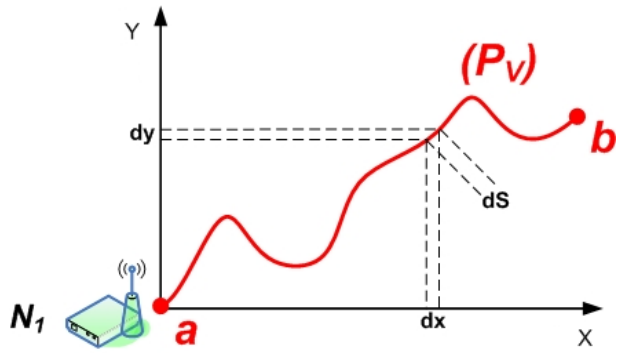


Figure 2. The trajectory from *a* to *b* in a N_1 random movement.

Assuming that *dS* is the differential of the curved line segment representing the trajectory of node N_1 from *a* to *b* and its delimited by *dx* and *dy*, then we have = $\sqrt{dx^2 + dy^2}$. The trajectory from *a* to *b* is referred to as curve *C* in the following. Thus, denoting the continuous trajectory of the node by $f_{(X,Y)}$, the node displacement from *a* to *b* is:

$$P = \int_C f_{(X,Y)} dS \quad (2)$$

In this way, the node displacement is described by (3), where $f_{(x,y)}$ is the function of *a* to *b* segments, which can be a concatenation of several basic functions, such as straight line segments, joined to cover the whole path from *a* to *b*.

$$P_V = \left\{ P_t - P_{t-1} \mid t \geq 1;\ P = \int_C f_{(X,Y)} dS = \right. \\ \left. = \int_a^b f_{(x,y)} . \sqrt{1 + (y')^2} dx \ \mid y' = \frac{dy}{dx} \right\} \quad (3)$$

From the above, P_V is the position analysis result, comparing the present node location with the previous node location. *P* denotes the node trajectory in the two-dimensional plane (x,y) from a point *a* to *b*, as shown in Fig. 2. The antenna radius is given by:

$$R = r_{t-1} + \int_{r_{t-1}}^{r_t} dr \mid r_t \neq r_{t-1} \quad (4)$$

in the simplified model in Fig. 3, where R_V denotes the signal power in the maximum radius border with connection possibility and *r* is related to the antenna range in *t* or *t-1* intervals.

In Fig. 3, two formats can be observed about the condition radius *r* of node coverage signal. In the

Copyright @ Computing Publishing UK, ISSN 2053-6283

DOI- 10.18486/ijcsnt.2016.5.1.01

arrows, the solid red line represents a decrease in range of the node and the dotted red line shows a gain in the range of the antenna signal. With the variation between the current r_t and the earlier r_{t-1}, and considering yet a difference *dr* between them, then the value of the range for the current time is obtained as:

Therefore, the variation of the R_V range shall be subject to *R*, which may be equal to r_{t-1} if there is no change in the node coverage radius. It is considered in (5) a variation *r* tending to the infinity only to demonstrate that the antenna does not have a predefined limit, which is correct from the mathematical point of view, however, inconsistent from the network point of view due to the real range of the sensors antennas, which are mostly in the tens of meters.

$$R_V = \left\{ R_t - R_{t-1} \mid t \geq 1;\ 0 \leq r \ll \infty;\ r_{t-1} \leq R \leq \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr \right) \right\} \quad (5)$$

The range of the antenna signal is also directly linked to the signal fading effect in the transmission medium, which in wireless communications is Rayleigh, i.e. the greater the distance between the nodes, the larger is the signal loss according to the Rayleigh curve $f_{(x)}$ of the Probability Density Function (PDF), which is directly related to distance from the source node, and is presented in Fig. 4 and having the notation R ~ Rayleigh(σ), where σ is the scaling factor for the signal fading.

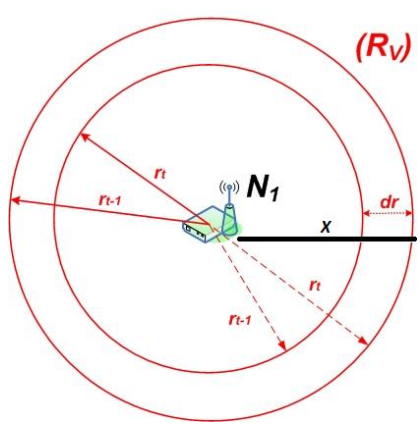


Figure 3. Node signal coverage feature in WSN.

Equation (6) represents the PDF of *x*. As can be seen in Fig. 4, *x* is the distance and σ is the fading factor for the curve.

Fig. 4a shows the PDF for a sequence of 5 simulations of σ and Fig. 4b shows the distribution function for the same 5 simulations of σ changes.

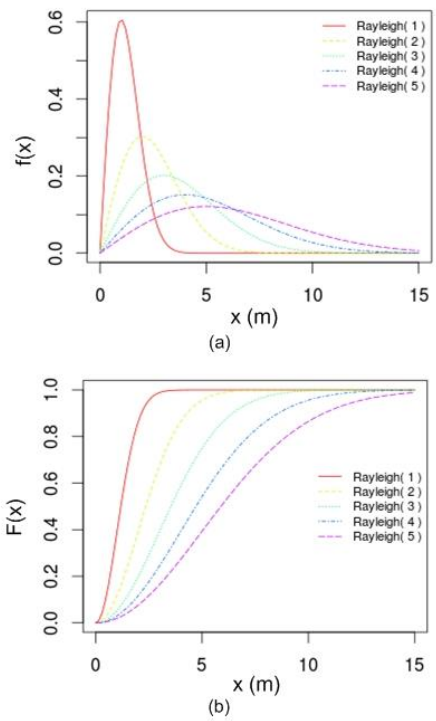


Figure 4. PDF and distribution function of Rayleigh for communication channel.

It is also noted in Fig. 4a that the larger σ gets the straighter the curve. Also, in Fig. 4b, as σ increases, the function spreads wider.

$$f_{(x)} = \frac{x}{\sigma^2} e^{\left\{\frac{-x^2}{2\sigma^2}\right\}}, x \geq 0 \quad (6)$$

Equation (7) shows the distribution function for the presented Fig. 4b.

$$F_{(x)} = 1 - e^{\left\{\frac{-x^2}{2\sigma^2}\right\}} \quad (7)$$

Therefore, considering the fading effect of (6) and applying $F_{(x)}$ from the (7) in the position equation (5), it has:

$$R_V = \left\{ R_t - R_{t-1}\ ;\ r_{t-1}.\left(1 - F_{(x)}\right) \leq R \leq \left(r_{t-1} + \int_{r_{t-1}}^{r_t} dr \right).\left(1 - F_{(x)}\right) \right\}, \quad (8)$$

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

$$1 \le \sigma \le 2;\ 0 \le r \ll \infty; x \ge 0$$

After that, connections will probably will fail with 70% chance and current connections will drop with over 85%.

$$L_v = \left\{ L_t - L_{t-1} \mid t \ge 1;\ 0 \le L \le (N-1); - N_{neig} \subset R_{neig(S)} \right\} \quad (9)$$

With respect to the L_V parameter, L is associated to the number of links established by nodes. N_{neig} corresponds to the number of neighboring nodes that should be inside the radius area of source node.

$$N_v = \left\{ N_t - N_{t-1} \mid t \ge 1 \right\} \quad (10)$$

Equation (10) shows N_V as the current number of nodes in the network. It can be changed according to the insertion or loss of nodes in the network. The ratios among P_V, R_V, L_V and N_V are the factors α, β, γ and σ respectively, and are given by:

$$\sum_{i=1}^{N} (\alpha + \beta + \gamma + \sigma)_i = 1 \Rightarrow$$

$$\Rightarrow \begin{cases} \alpha = \frac{1}{4} \cdot \left| \frac{P_t - P_{t-1}}{\text{argmax}(P_t - P_{t-1})} \right|, \forall \alpha \in \mathbb{R} \mid 0 \le \alpha \le 25\% \\ \beta = \frac{1}{4} \cdot \left| \frac{R_t - R_{t-1}}{\text{argmax}(R_t - R_{t-1})} \right|, \forall \beta \in \mathbb{R} \mid 0 \le \beta \le 25\% \\ \gamma = \frac{1}{4} \cdot \left| \frac{L_t - L_{t-1}}{\text{argmax}(L_t - L_{t-1})} \right|, \forall \gamma \in \mathbb{R} \mid 0 \le \gamma \le 25\% \\ \sigma = \frac{1}{4} \cdot \left| \frac{N_t - N_{t-1}}{\text{argmax}(N_t - N_{t-1})} \right|, \forall \sigma \in \mathbb{R} \mid 0 \le \sigma \le 25\% \end{cases} \quad (11)$$

Equation (11) refers to the proportions related to the system's variability factors: α, the position ratio; β, the node radius ratio; γ, the ratio of the node's possible connections and σ, the ratio of number of nodes in the network.

The variables P_V, R_V, L_V and N_V are obtained from (2)-(10) and the relations in the variability detection are presented in pseudo-code part posted ahead, where t represents current time and t-1 represents time in the previous step.

The relationship among system variables P_V, R_V, L_V and N_V is depicted in Fig. 5. The number of nodes in a unique network must be defined (or monitored, if it already exists) based on the monitored/applied environment requirements. In other words, to accomplish the environment necessities is necessary to know the real application of sensor nodes.

Algorithm for sensor variability detection
N = Number of nodes in the network;
$n = 0$;
$t = 0$;
WHILE $(n \le N)$
WHILE $(t > 0)$
LET $P_{V(n)} = \phi_0,\quad \alpha_{(n)} = \theta_0$
$R_{V(n)} = \phi_1,\quad \beta_{(n)} = \theta_1$
$L_{V(n)} = \phi_2,\quad \gamma_{(n)} = \theta_2$
$N_{V(n)} = \phi_3,\quad \sigma_{(n)} = \theta_3$
{ $P_{V(n)}$, $R_{V(n)}$, $L_{V(n)}$ and $N_{V(n)}$ are readings of network by nodes measurements. }
FOR $i = 0$ *to* 3
DO $\theta_i = \frac{1}{4} \cdot \left| \frac{\phi_{i(t)} - \phi_{i(t-1)}}{\text{argmax}(\phi_{i(t)}, \phi_{i(t-1)})} \right|$
$F_{V\,i(n)} = \prod_{i=0}^{3} \theta_i \phi_{i(t)}$
$F_{V(n)} = \sum_{i=0}^{3} F_{V\,i(n)}$
{ $F_{V(n)}$ represents the percentage relative to the variability in n^{th} node. }
END FOR
$t = t + 1$
$n = n + 1$
END WHILE
END WHILE

If the environment being monitored is small and requires few nodes, the number of nodes N is a known parameter instead of a variable one, constraining the variability feature to three variables and the multiplicative factors assume 1/3 ratio unless 1/4. If the random enrolment of nodes to the network is required, then should be used the complete equation with four variables, as given in (1).

According to (2)-(10), if the subtraction shows result 0 the system variable can assume binary value 0 corresponding to no variation in the network parameter, and if exists another value in subtraction result, it indicates that there is occurrence of variation in the parameter affording binary value 1 (it is for P_V, R_V, L_V and N_V values applied in (1)).

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

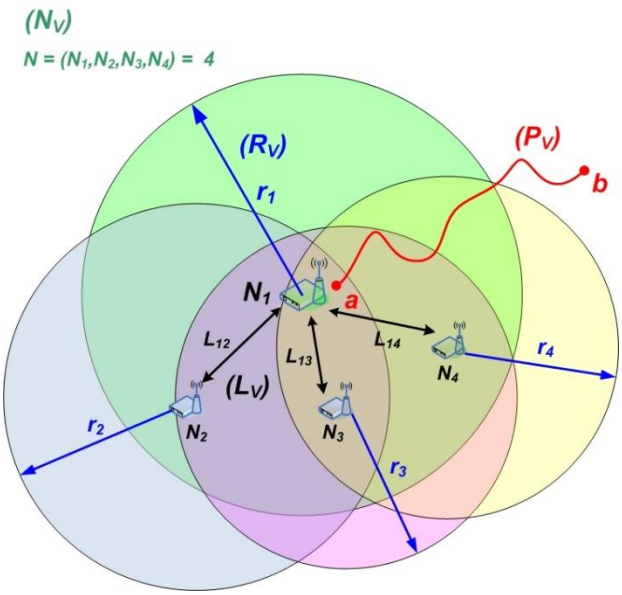


Figure 5. The general features of nodes connection in the network.

The cognitive process proceeds with the monitoring of nodes applying the relationships given in (1) to (11) for each node, and then calculates the variability feature F_N that is the F_V average of an amount of trajectories as shown in (12).

$$F_N = \overline{X}\left(F_{V(n)}\right) = \frac{\sum_{n=1}^{N} F_{V(n)}}{N} \qquad (12)$$

In pseudo-code described before, the calculation round find out the parameters and respective ratios to facilitate the F_N achievement. F_N is one of the parameters used by the cognitive process running into CPMod to provide Decision Making process (DMp) and it will be described in next section.

IV. SYSTEM FEATURES AND DEVELOPMENT

The cognitive process uses the memory vector V_M, the variability feature F_N (described in previous section) and the input signal $S_I(n)$ to generate the output signal $S_O(n)$ that controls the system with cognitive process.

In the cognitive process the comparator result is the mean squared error (MSE) $\mathcal{E}$ between the received signal $S_I(n)$ and the output signal $S_O(n)$ and it is used for system adequacy.

For a given input signal $S_I(n)$ composed by network variability feature and memory vector (the memory vector will be described further, but it comprises data obtained from the network's nodes regarding connections successes or failures in addition to route information), the cognitive process uses F_N, V_M and $\mathcal{E}$ to identify possible problems in the network and to act on the output signal $S_O(n)$ with a control vector V_C.

Fig. 6 illustrates the interfaces and connections taking place in the cognitive process regarding internal development of initial CPMod (iCPMod).

The signal $S_I(n)$ derives from the combination of F_N and V_M, while V_C acts on $S_O(n)$. F_N and V_M are indicators of system environment situation. With the analysis of these two parameters, $S_I(n)$ signal can be composed for error calculation. The results applying these indicators are shown in the section VI.

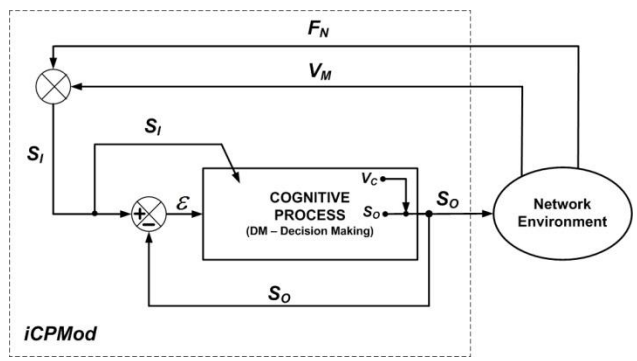


Figure 6. Block diagram of the cognitive process interconnection.

A. Memory Vector

The Memory Vector V_M is in the core of the adaptive cognitive module studied here since it allows the cognitive process to store information regarding network and its nodes, establishing the history of the network situations and memory.

This vector results from the compilation of instant data regarding the monitored environment within time slots after each 5 seconds is recommended. With V_M it is possible to act on the environment using the DMp to generate DM vectors.

$$V_M = \left(V_S, V_F, V_R\right) \qquad (13)$$

Equation (13) introduces V_M matching to the memory vector composed by: V_S as the vector containing the

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

successes of nodes connections; V_F as the vector containing the failures in the nodes and V_R as a vector of valid routes collected from RT. The components of V_S, V_F and V_R are described briefly in Table I.

TABLE I. MEMORY VECTOR COMPONENTS

Vector Type	Components and Description		
	Description	***Components***	***Decision***
V_S	Success Vector	Successful end-to-end connection, End-to-end delay, Rate of effectively delivered data.	By the routes with a higher probability of end-to-end packet delivery success.
V_F	Failure Vector	Node disconnected three times in a row, Unresponsive node with reply message (RREP).	Avoiding these nodes for a given period of time.
V_R	Route Vector	Valid routes, Origin node, Destination node, Number of hops to destination.	By the best route according to the valid routes history.

The V_M application for the DM classification will be described ahead and this feature is present on the CPMod as an instance of DM analysis, as in (15).

B. Error Vector

Basically, the error vector $\mathcal{E}$ works with the relative error MSE of the cognitive module measurements input $S_I(n)$ and output $S_O(n)$ signals. $S_I(n)$ is composed by the variability feature and memory vector, and $S_O(n)$ works with control vector (described in D division of this section) and memory vector.

$$\begin{cases} S_I = (F_N, V_M) \rightarrow \zeta S_I = S_I \text{ estimate} \\ S_O = (V_C, V_M) \rightarrow \xi S_O = S_O \text{ real value} \end{cases}$$
$$\varepsilon_{(\%)} = \sum_{i=1}^{N} \frac{(\zeta S_I - \xi S_O)_i^2}{S_{O_i}} . 100 \mid 0 \leq \varepsilon \leq 100\% \tag{14}$$

The composition of this feature appears in the third parameter for DM analysis, as shown in (15).

C. Decision Making Vector

The procedure for Decision Making process (DMp) is conducted by the cognitive process and it takes into the account of environment variability feature F_N, the memory vector V_M and the error $\mathcal{E}$, seeking for the optimal network configuration and the adjustment of the employed routing protocol.

DM is assigned according to the expected quality of the task and its action is directly related to the Decision Limit (DL). The threshold DL corresponds to a probabilistic value for the adjustment of the system resulting from the combination of F_N, V_M and $\mathcal{E}$ according to the range entrance (DM description) given in Table II.

$$DM = (F_N, V_M, \varepsilon) \tag{15}$$

Five abstraction levels were adopted based on Quality of Experience (QoE) since they represent the network conditions and allow V_C actuation. DL is calculated using the inverse of the average of DM composition as described in (16).

For the measurement level regarding the network situation, as mentioned before, DL is calculated based on the inverse of DM average, according to (16). The graphical relationship for DL is depicted in Fig. 7.

$$\overline{X}_{(F_N, V_M, \varepsilon)} = \frac{\sum_{i=1}^{N} (F_{N_i}, V_{M_i}, \varepsilon_i)}{N} \Rightarrow$$
$$\Rightarrow \begin{cases} \overline{X}_{(F_N, V_M, \varepsilon)} \geq 1, \ DL_{(\%)} = \dfrac{1}{\overline{X}_{(F_N, V_M, \varepsilon)}} \times 100 \ (\%) \\ \overline{X}_{(F_N, V_M, \varepsilon)} < 1, \ DL_{(\%)} = 100 \ \% \end{cases} \tag{16}$$

The smaller DL, in percentage, the better is the network condition and, similarly, the greater DL the worst is the network situation, therefore more significant actions should be taken by the CPMod. Furthermore, in order to keep the network with controlled or regular situation, DL level must be under 49.9% range.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

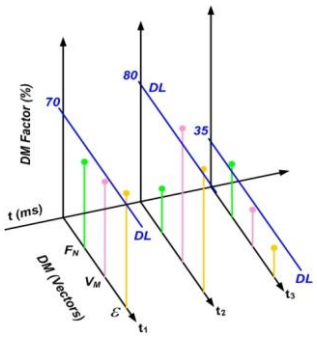


Figure 7. Example of relationship between DL and DM.

TABLE II. DECISION MAKING ACCORDING TO THE DL THRESHOLD

***DM* Description**	***DL* Configuration**	
	Decision Limit (%) Range [a]	***Action about DM (with DL)***
Urgent	80.0 → 100	Reinitiates the routing table in the nodes. The process evaluates the index (F_N, V_M, ε) originating the urgent situation. V_C is sent to nodes.
Critical	70.0 → 79.9	The process evaluates memory (V_M) and error (ε). Adjusts signal S_O and reads *DL* again. If the situation is maintained a control vector V_C is sent to the node.
Attention	50.0 → 69.9	Adjusts $S_O(n)$ with signal correction according to detected error ε.
Regular	30.0 → 49.9	No need for changes but decisions may be taken to improve QoS. The system is operating normally.
Controlled	0 → 29.9	Network without need for changes. The system is operating in better case.

[a] *Measurement level regarding the network situation.*

In other case, the lower is the average in (16) for memory, variability and error vectors, more critical the network condition will be. It can be found in Fig. 7 an example of a DM data reading composed by F_N, V_M and $\mathcal{E}$ vectors, distributed in times t_1, t_2 and t_3.

D. Control Vector

The iCPMod with DM sends a control data vector in S_O to define the nodes alterations according to the necessity of change by using a control vector V_C. This vector is defined in (17) and it can be adapted to other actions according to the network's needs and to alterations in the settings (such as the insertion of other parameters in the future development).

$$V_C = (Energy, Route, Queue, Memory) \qquad (17)$$

With the settlements of V_C on the output signal $S_O(n)$ it is possible to organize the nodes. The nodes receive and make alterations regarding memory amount sizing through the final control vector to execute tasks. Queues and route protocols (described as RT) can be addressed in this time, correcting the issues related with these features, and energy can be controlled by iCPMod. The cognitive processing allows the system (in a general way) to readjust and, at the same time, execute the network monitoring task. This cognitive system also allows adaptation regarding monitored aspects (e.g., the insertion of a monitoring vector of energy levels at the nodes for a possible control by the cognitive process and a raise in each node's yield according to DL adequacy) as well as DM and V_C routing in $S_O(n)$.

For the optimization of ACS, a development with Fuzzy Logic was realized for CPMod interaction, concerning the DMp with respect to V_C defined aspects in (17).

V. FUZZY LOGIC APPROACH

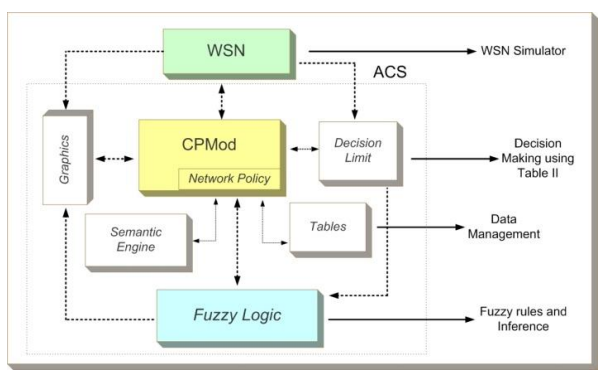


Figure 8. ACS general architecture.

For the Fuzzy Logic implementation, it was defined the ACS architecture as in the Fig. 8. It can be seen that CPMod have interaction with the WSN and Fuzzy Logic blocks. It was used the iCPMod for the algorithm derivation and the CPMod for the block connected into ACS.

Copyright @ Computing Publishing UK, ISSN 2053-6283

DOI- 10.18486/ijcsnt.2016.5.1.01

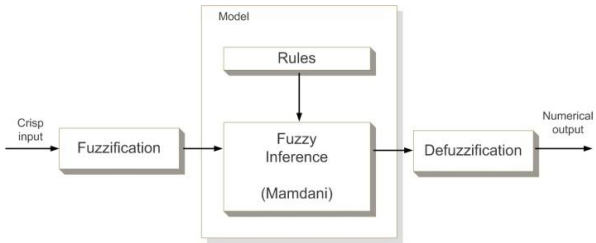


Figure 9. Inference Fuzzy diagram block.

The FIS is intended to be used by the CPMod as an alternative solution for decision making. The inference mechanism aggregates more coherent and fast network analysis using the Fuzzy control and pertinence functions. Considering the provisions of (18), the membership functions mf are shown for the inference and to be applied on the fuzzification and defuzzification steps.

$$A = \{(x, \mu_A(x)) | x \in X\} \text{ and } B = \{(x, \mu_B(x)) | x \in X\},$$

$$\mu_A(x) = mf \Rightarrow \begin{cases} \mu_{A \cap B} = \text{argmin}(\mu_A(x), \mu_B(x)), \forall x \in X \\ \mu_{A \cup B} = \text{argmax}(\mu_A(x), \mu_B(x)), \forall x \in X \end{cases} \quad (18)$$

On behalf of the membership functions mentioned, the triangular and trapezoidal functions were considered to be part of pertinence functions, according to (19) and (20), respectively.

$$\text{Trim}f_{(x;a,b,c)} = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \quad (19)$$

The pertinence functions for the fuzzification are presented in Figs. 10-13.

$$\text{Trapm}f_{(x;a,b,c,d)} = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \quad (20)$$

Equations (22)-(25) assume values from the (16), which relates to five-fold the fuzzy membership degree of x in A.

$$f_A(x) = (x, T(x), G(x), M(x), X) \quad (21)$$

From (21), x represents the vector name data, $T_{(x)}$ vector meanings set, $G_{(x)}$ the set of syntactic rules applied to x, $M_{(x)}$ the semantic rule that assigns to each value generated by $G_{(x)}$ Fuzzy set in x and X the universe of discourse that is the X axis.

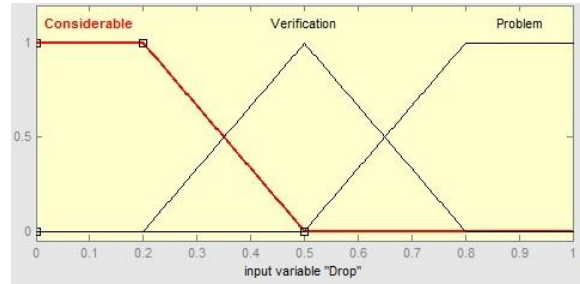


Figure 10. Percentage of input variable drop of packets.

$$f_A(x) = \begin{cases} x\text{: Drop} \\ T(x)\text{: \{Considerable, Verification, Problem\}} \\ X\text{: Number of packets} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (22)$$

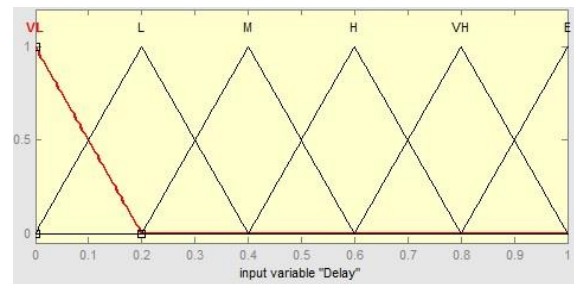


Figure 11. Percentage of input variable delay of packets.

$$f_A(x) = \begin{cases} x\text{: Delay} \\ T(x)\text{: \{}VL, L, M, H, VH, E\text{\}} \\ X\text{: Time in miliseconds} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (23)$$

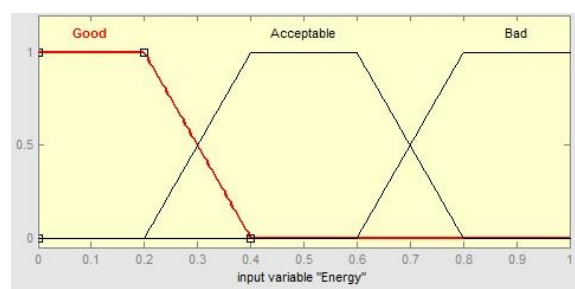
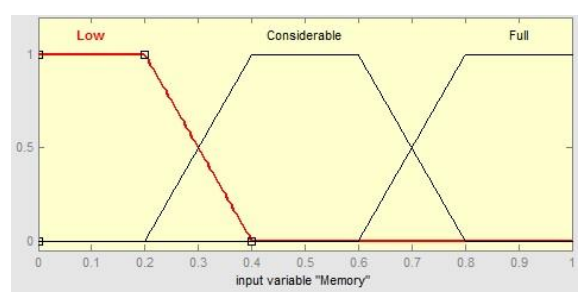


Figure 12. Percentage of input variable energy consumed by nodes.

$$f_A(x) = \begin{cases} x\text{: Energy} \\ T(x)\text{: \{Good, Acceptable, Bad\}} \\ X\text{: Quantity of consumed energy} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (24)$$



Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

Figure 13. Percentage of input variable memory used by nodes.

$$f_A(x) = \begin{cases} x\text{: Memory} \\ T(x)\text{: \{Low, Considerable, Full\}} \\ X\text{: Quantity of used memory} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (25)$$

For the insertion of Fuzzy Logic into simulation scenario, it was implemented the pertinence functions in MATLAB software. The home screen of Fuzzy Logic is shown in Fig. 14 and represents the initial structure data analysis model in software.

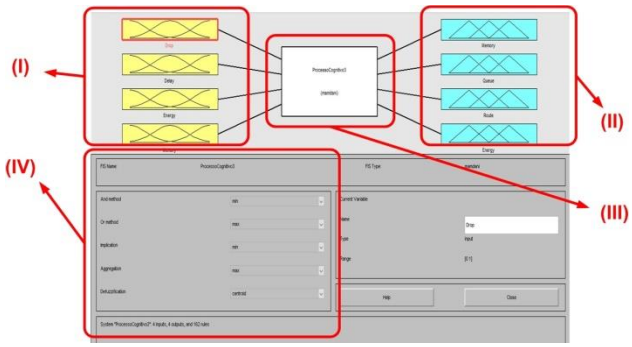


Figure 14. Home screen of Fuzzy Logic in MATLAB.

In screen, were created 4 fuzzification functions to model the crisp data collected from the WSN (I) and 4 corresponding defuzzification pertinence functions to 4 elements present in V_C (II). The item (III) is the use of chosen Mamdani model for inference of the fuzzification membership functions and (IV) is connected to operators and defuzzification method used, which in this case is based on the centroid of defuzzification membership functions.

The pertinence functions for the defuzzification are presented in Figs. 15-18.

Equations (26)-(29) assume values from the (21), which relates to five-fold the fuzzy membership degree of x in A.

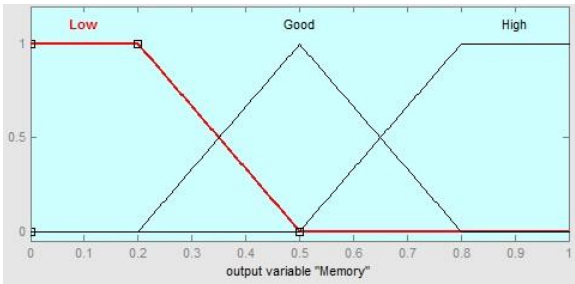


Figure 15. Percentage of output variable Memory.

$$f_A(x) = \begin{cases} x\text{: Memory} \\ T(x)\text{: \{Low, Good, High\}} \\ X\text{: Quantity of available memory} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (26)$$

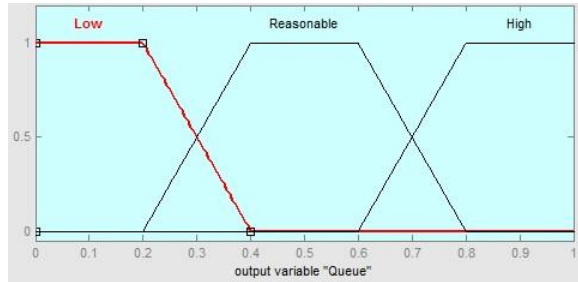


Figure 16. Percentage of output variable Queue.

$$f_A(x) = \begin{cases} x\text{: Queue} \\ T(x)\text{: \{Low, Reasonable, High\}} \\ X\text{: Packet flow} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (27)$$

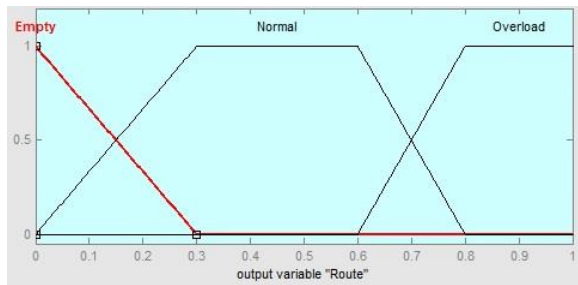


Figure 17. Percentage of output variable Route.

$$f_A(x) = \begin{cases} x\text{: Route} \\ T(x)\text{: \{Empty, Normal, Overload\}} \\ X\text{: Delay in delivering packets} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (28)$$

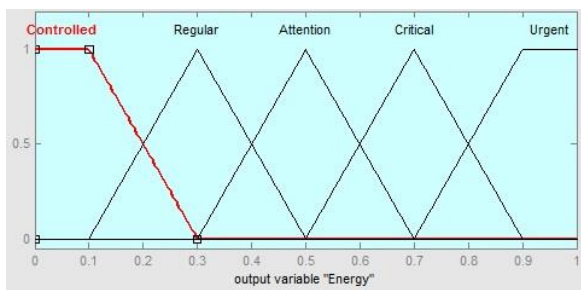


Figure 18. Percentage of output variable Energy.

$$f_A(x) = \begin{cases} x\text{: Energy} \\ T(x)\text{: \{Controlled, Regular, Attention, Critical, Urgent\}} \\ X\text{: Quantity of consumed energy} \\ G(x)\text{: \{rules\}} \\ M(x)\text{: Significate\{rules\}} \end{cases} \quad (29)$$

The Center of Area (CoA) for calculating the centroid is given by (30), which $\mu_{out}(u_i)$ is the area of a membership function and u_i is the centroid position of the individual membership function.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

***DOI*- 10.18486/ijcsnt.2016.5.1.01**

$$u_{defuzzy} = \frac{\sum_{i=1}^{N} u_i \mu_{out}(u_i)}{\sum_{i=1}^{N} \mu_{out}(u_i)} \tag{30}$$

The rules are presented in the MATLAB formatting, representing 162 lines of Fuzzy rules that comprises the overall input-output surfaces present in results.

```
[System]
Name='ProcessoCognitivo3a'
Type='mamdani'
Version=2.0
NumInputs=4
NumOutputs=4
NumRules=162
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Drop'
Range=[0 1]
NumMFs=3
MF1='Considerable':'trapmf',[0 0
0.2 0.5]
MF2='Verification':'trimf',[0.2
0.5 0.8]
MF3='Problem':'trapmf',[0.5 0.8
1 1]

[Input2]
Name='Delay'
Range=[0 1]
NumMFs=6
MF1='VL':'trimf',[0 0 0.2]
MF2='L':'trimf',[0 0.2 0.4]
MF3='M':'trimf',[0.2 0.4 0.6]
MF4='H':'trimf',[0.4 0.6 0.8]
MF5='VH':'trimf',[0.6 0.8 1]
MF6='E':'trimf',[0.8 1 1]

[Input3]
Name='Energy'
Range=[0 1]
NumMFs=3
MF1='Good':'trapmf',[0 0 0.2
0.4]
MF2='Acceptable':'trapmf',[0.2
0.4 0.6 0.8]
MF3='Bad':'trapmf',[0.6 0.8 1 1]

[Input4]
Name='Memory'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.4]
MF2='Considerable':'trapmf',[0.2
0.4 0.6 0.8]
MF3='Full':'trapmf',[0.6 0.8 1
1]

[Output1]
Name='Memory'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.5]
MF2='Good':'trimf',[0.2 0.5 0.8]
MF3='High':'trapmf',[0.5 0.8 1
1]

[Output2]
Name='Queue'
Range=[0 1]
NumMFs=3
MF1='Low':'trapmf',[0 0 0.2 0.4]
MF2='Reasonable':'trapmf',[0.2
0.4 0.6 0.8]
MF3='High':'trapmf',[0.6 0.8 1
1]

[Output3]
Name='Route'
Range=[0 1]
NumMFs=3
MF1='Empty':'trimf',[0 0 0.3]
MF2='Normal':'trapmf',[0 0.3 0.6
0.8]
MF3='Overload':'trapmf',[0.6 0.8
1 1]

[Output4]
Name='Energy'
Range=[0 1]
NumMFs=5
MF1='Controlled':'trapmf',[0 0
0.1 0.3]
MF2='Regular':'trimf',[0.1 0.3
0.5]
MF3='Attention':'trimf',[0.3 0.5
0.7]
MF4='Critical':'trimf',[0.5 0.7
0.9]
MF5='Urgent':'trapmf',[0.7 0.9 1
1]

[Rules]
2 3 1 2, 1 2 2 3 (1) : 1
2 4 1 2, 1 2 2 3 (1) : 1
2 5 1 2, 1 2 2 4 (1) : 1
2 6 1 2, 1 2 2 4 (1) : 1
2 1 1 3, 2 1 1 3 (1) : 1
2 2 1 3, 2 1 2 3 (1) : 1
2 3 1 3, 2 2 2 3 (1) : 1
2 4 1 3, 2 2 2 3 (1) : 1
2 5 1 3, 2 2 2 4 (1) : 1
2 6 1 3, 2 2 2 4 (1) : 1
2 1 2 1, 1 1 1 3 (1) : 1
2 2 2 1, 1 1 1 3 (1) : 1
2 3 2 1, 1 1 2 3 (1) : 1
2 4 2 1, 1 1 2 3 (1) : 1
2 5 2 1, 2 2 2 4 (1) : 1
2 6 2 1, 2 2 2 4 (1) : 1
2 1 3 1, 2 1 2 3 (1) : 1
2 2 3 1, 2 1 2 3 (1) : 1
2 3 3 1, 3 2 2 3 (1) : 1
2 4 3 1, 3 2 2 3 (1) : 1
2 5 3 1, 3 3 2 4 (1) : 1
2 6 3 1, 3 3 2 4 (1) : 1
2 1 2 2, 2 1 2 3 (1) : 1
2 2 2 2, 2 1 2 3 (1) : 1
2 3 2 2, 2 2 2 3 (1) : 1
2 4 2 2, 2 2 2 3 (1) : 1
2 5 2 2, 3 2 2 4 (1) : 1
2 6 2 2, 3 2 2 4 (1) : 1
2 1 2 3, 2 1 1 3 (1) : 1
2 2 2 3, 2 1 2 3 (1) : 1
2 3 2 3, 2 2 2 3 (1) : 1
2 4 2 3, 2 2 2 3 (1) : 1
3 5 3 2, 3 3 3 5 (1) : 1
3 2 1 2, 1 3 2 3 (1) : 1
3 3 1 2, 2 3 2 3 (1) : 1
3 4 1 2, 2 3 2 3 (1) : 1
3 5 1 2, 3 3 2 4 (1) : 1
3 6 1 2, 3 3 2 4 (1) : 1
3 1 1 3, 2 3 2 3 (1) : 1
3 2 1 3, 2 3 2 3 (1) : 1
3 3 1 3, 3 2 2 3 (1) : 1
3 4 1 3, 3 2 2 3 (1) : 1
3 5 1 3, 3 3 2 4 (1) : 1
3 6 1 3, 3 3 2 4 (1) : 1
3 1 2 1, 1 2 2 3 (1) : 1
3 2 2 1, 1 2 2 3 (1) : 1
3 3 2 1, 2 2 2 4 (1) : 1
2 4 3 3, 3 2 2 3 (1) : 1
2 5 3 3, 3 3 2 4 (1) : 1
2 6 3 3, 3 3 2 4 (1) : 1
3 1 1 1, 1 3 1 3 (1) : 1
3 2 1 1, 1 3 1 3 (1) : 1
3 3 1 1, 1 3 2 3 (1) : 1
3 4 2 1, 2 2 2 4 (1) : 1
3 5 2 1, 2 3 2 5 (1) : 1
3 6 3 2, 3 3 3 5 (1) : 1
3 1 3 3, 3 3 2 4 (1) : 1
3 2 3 3, 3 3 2 4 (1) : 1
3 3 3 3, 3 3 2 4 (1) : 1
3 4 3 3, 3 3 3 5 (1) : 1
2 5 2 3, 3 3 2 4 (1) : 1
2 6 2 3, 3 3 2 4 (1) : 1
2 1 3 2, 2 2 2 3 (1) : 1
2 2 3 2, 2 2 2 3 (1) : 1
2 3 3 2, 2 3 2 3 (1) : 1
2 4 3 2, 2 3 2 3 (1) : 1
2 5 3 2, 3 3 2 4 (1) : 1
2 6 3 2, 3 3 2 4 (1) : 1
2 1 3 3, 3 1 2 3 (1) : 1
2 2 3 3, 3 1 2 3 (1) : 1
2 3 3 3, 3 2 2 3 (1) : 1
3 4 1 1, 1 3 2 3 (1) : 1
3 5 1 1, 1 3 2 4 (1) : 1
3 6 1 1, 1 3 3 4 (1) : 1
3 1 1 2, 1 3 2 3 (1) : 1
3 5 3 3, 3 3 3 5 (1) : 1
3 6 3 3, 3 3 3 5 (1) : 1
1 6 3 1, 3 2 2 4 (1) : 1
2 1 1 1, 1 2 1 3 (1) : 1
2 2 1 1, 1 2 1 3 (1) : 1
2 3 1 1, 1 2 1 3 (1) : 1
2 4 1 1, 1 2 1 3 (1) : 1
2 5 1 1, 1 3 2 4 (1) : 1
2 6 1 1, 1 3 2 4 (1) : 1
2 1 1 2, 1 1 2 3 (1) : 1
2 2 1 2, 1 1 2 3 (1) : 1
```

The weight of all rules is the same, so the system can be in the balanced state when inference runs over data collected from WSN, in support of providing the results to CPMod realize the DM.

VI. RESULTS AND COMMENTS

Added either as vectors or only as another variable to the network data reading, the mechanism gains robustness but tends to lose processing response time since the cognitive process needs to conduct a larger number of analysis with the insertion of new monitoring elements, however this case occur when the system is running in the initial stage.

In the study scenario were used 0-100 sensor nodes with possibility to each node establishing 0-9 connections, and each node has antenna radius variability of 3-5 meters and displacement of 1-10

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

meters in random trajectory. For simplicity, was adopted a basic antenna model. The data collected from nodes were generated by *Prowler* simulation over MATLAB.

As mentioned before, the CPMod operates according with network/nodes situation. Nevertheless, the procedure to execute data analysis (described in previous section) is based on Table II and it is explained as follows.

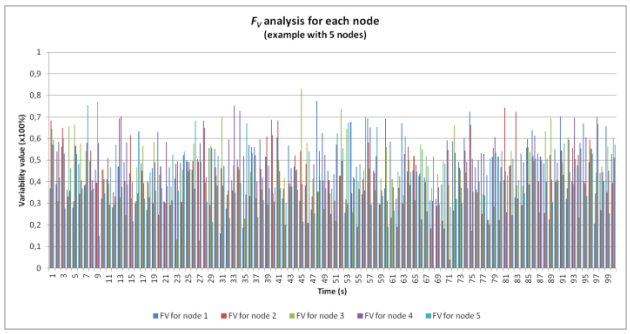


Figure 19. Graphical analysis concerning F_V variability for 5 nodes.

The graphs were obtained from the random nodes statistics samples, applying the expressions mentioned in previous sections.

From the Fig. 19, it is possible to identify the relation between Table II and variability values in the graph. As can be seen, the most of data collected about variability is in the range of 20% to 50%, so according to the Table II, these 5 nodes are operating normally and the situation is controlled and regular. DL is present here according to each peak signal.

There are yet a big amount of circumstances demanding attention and few cases where nodes require action with V_C intervention, because they reach the critical level. In this sampling, at approximately 45 seconds an urgent condition occurred. Therefore, in this case, a V_C was sent to the respective node (node 3) to perform the adjustment task (queue scheme modification).

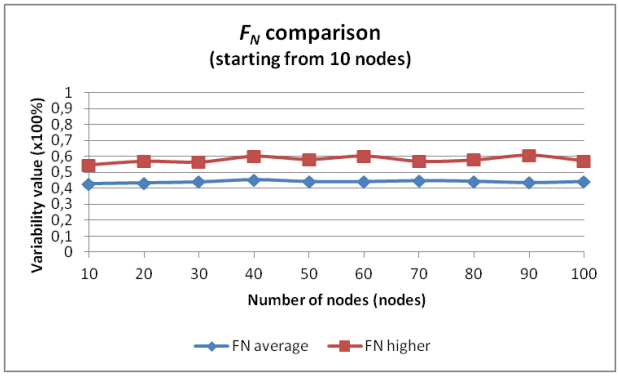


Figure 20. Graphic representing the F_N comparison for variability.

This graph shows us the occurrence of stable condition in the system, this means that the nodes are operating in approximately 45% of variability, according to the position, antenna radius, possible links and number of nodes. The higher situations are appearing over 50% and less than 70%, so in the attention range according to Table II.

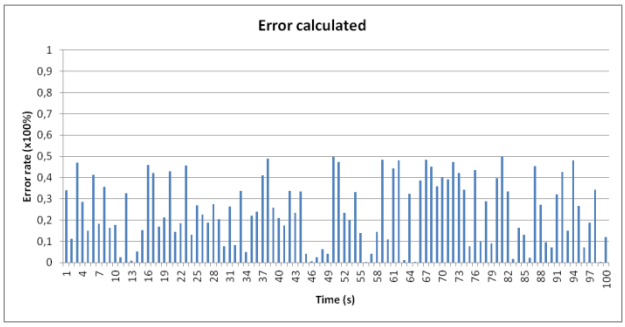


Figure 21. Error rate concerning S_I and S_O.

In the Fig. 21, it can be seen that the error calculated over network data resulted in the Controlled and Regular levels of Table II.

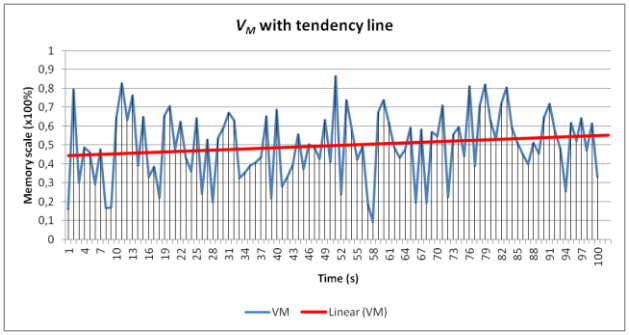


Figure 22. V_M with tendency line according to the memory status.

From the Fig. 22, is understood that with the network time running, the memory usage tends to be higher according to the tendency line. It can be seen that

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

exists 6 peaks surrounding 80% memory usage, and in that cases V_C is sent to the respective nodes to try corrections in memory, queue or route tables.

The idea that this mechanism will overload the network with unnecessary data packets seems reasonable but, when the sensors/actuators are synchronized within the nodes and respective features after the initial setting stage, the system pre-stabilizes and acts according to the alterations done by the CPMod.

Using F_N, V_M and $\mathcal{E}$ parameters, the DMp can be executed according to the Table II to regularize the nodes situations regarding memory, queue, energy and route adjustments. DL is obtained too from these parameters and tends to be near to their higher values, concerning the average relation described in (16).

The real strictly direct correlation between CPMod process and QoS assurance is not depicted from the above graphs, but it will be treated in a sequence work.

With respect to the Fuzzy Logic applied to DM in ACS architecture, the surfaces of Figs. 23-32 present the specific analysis surrounding each input-output element, according to (X, Y, Z) coordinates. In the graphs, X and Y represent input elements and Z represents the output of Fig. 9.

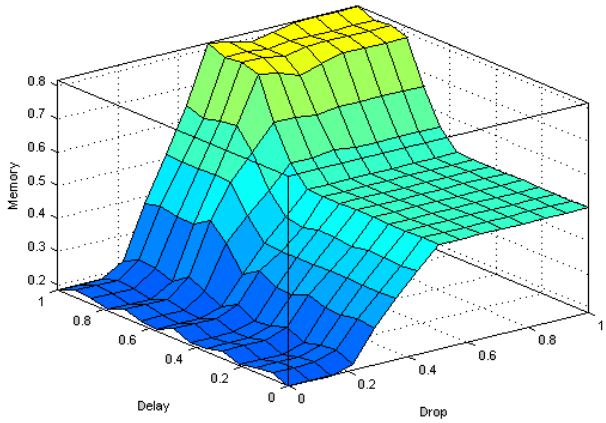


Figure 23. Surface of Drop x Delay x Memory.

From the Fig. 23, it can be seen that with growing of packets dropped in network in conjunction with delay to transmit data end-to-end indicates an elevated necessity for memory available in nodes, or in other words, the memory will be a bottleneck for the network soon.

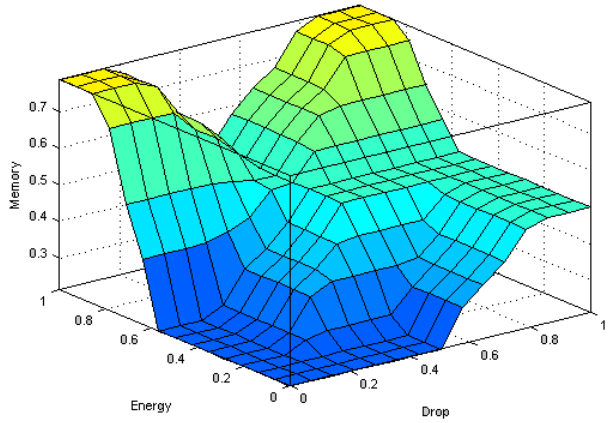


Figure 24. Surface of Drop x Energy x Memory.

The Fig. 24 shows that even with low drop rate of packets, if energy consumed by nodes raises over 60%, the memory is also committed.

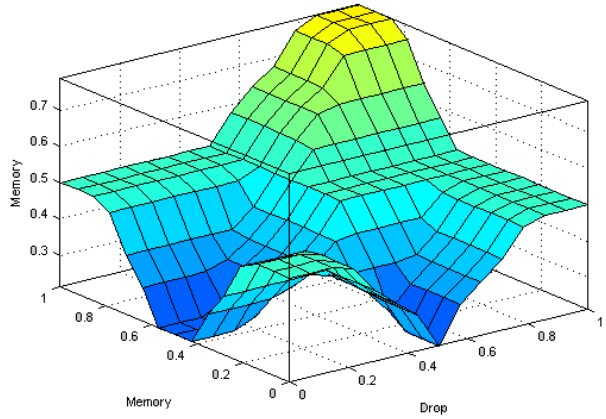


Figure 25. Surface of Drop x Memory x Memory.

In the Fig. 25, when the memory usage by nodes according to the packets dropped raises, the memory available turns a bottleneck too.

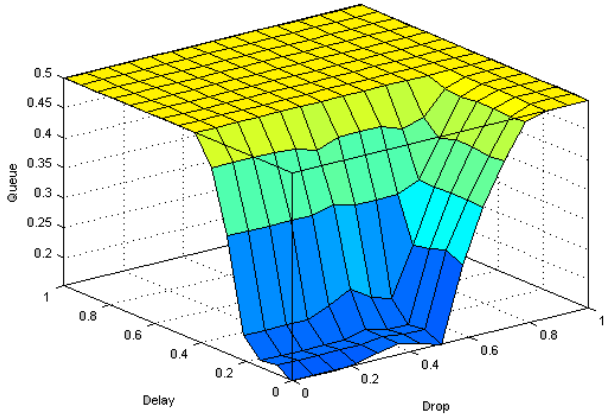


Figure 26. Surface of Drop x Delay x Queue.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

***DOI*- 10.18486/ijcsnt.2016.5.1.01**

It can be seen in the above graph, that the relation between drop and delay does not affect seriously the network in the queue issue.

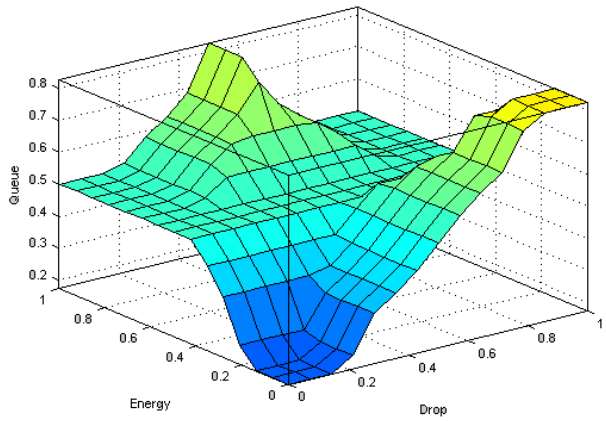


Figure 27. Surface of Drop x Energy x Queue.

Even with low energy usage by nodes, if drop raises the queue is affected directly.

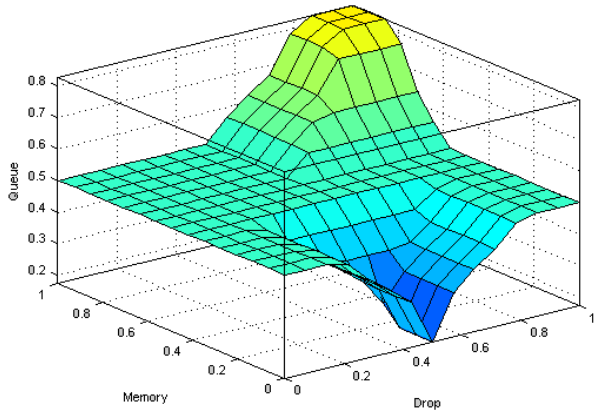


Figure 28. Surface of Drop x Memory x Queue.

The Fig. 27 indicates that with a high rate of packets dropped in network, the Critical and Urgent levels of Table II are meeting, forcing the CPMod to act on Link layer.

From the Fig. 28, with high usage of memory and packets dropped excessively, the queue tends to raise fast over 50% of both.

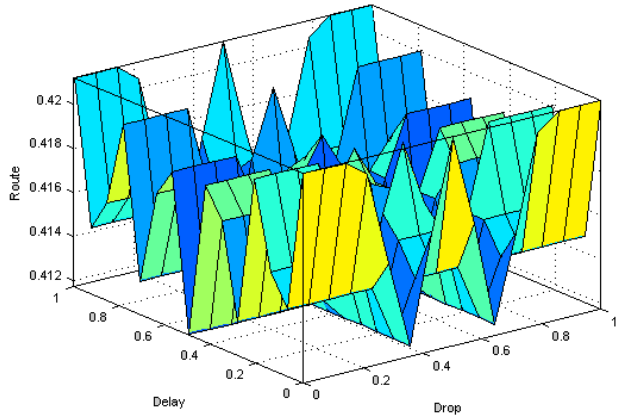


Figure 29. Surface of Drop x Delay x Route.

From the Fig. 29, can be verified that the route or routing scheme is oscillating in the range of 41 to 43% for the relation between rates of packets dropped and delay end-to-end. This means the route protocol is affected by drop in three instances and delay can occur due to the links between nodes reestablishments, nodes position modifying, and other variable features. The results for (Drop x Energy) and (Drop x Memory) related to Route do not presented significant implications to changes in routing protocols on Network layer.

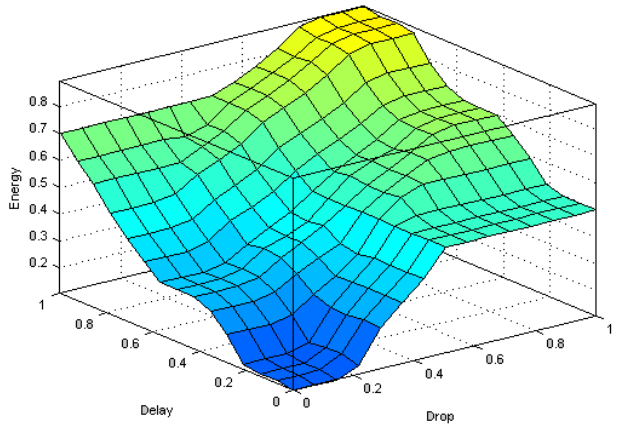


Figure 30. Surface of Drop x Delay x Energy.

Finally, according to Energy output, the drop and delay ongoing affect gently the energy as they raises. With high drop and delay, the energy consumed by nodes is too high, because more processing and routing of packets are needed.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

DOI- 10.18486/ijcsnt.2016.5.1.01

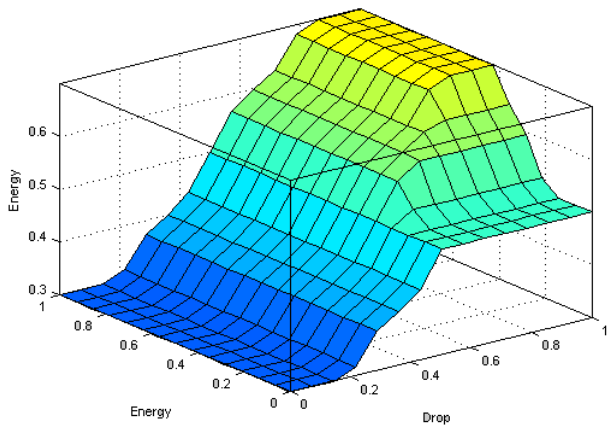


Figure 31. Surface of Drop x Energy x Energy.

The same occurs when Energy > 20% and the drop of packets raises too, until the queue does not support more packets and all of new received packets falls into dropped packets by nodes.

In Fig. 32 is shown that the evolution of memory used by nodes does not affect directly the Energy output; however the combination of memory usage with drop of packets results in an energy spent by nodes. The energy spent is directly associated with the growing of rate of data packets dropped.

As can be seen, the elements that comprises the network features do not work alone and their interactions allows the control of WSN, trying to keep the network balanced into rules defined in Fuzzy Logic and in the DL of Table II.

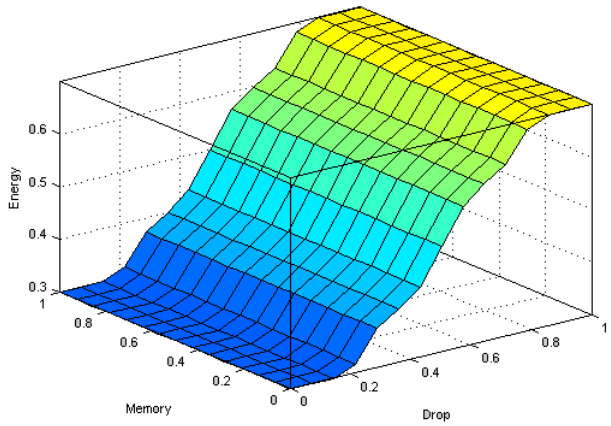


Figure 32. Surface of Drop x Memory x Energy.

The implemented FIS block permits the increase of elements, which can be added for future observation and control of network. As well as the rules and pertinence functions applied for fuzzification and defuzzification, that can be modified and increased.

VII. CONCLUSIONS

The cognitive process adds intelligence aspects to the Wireless Sensor Networks control mechanism, attaching memory, history and decision making. These features reflect some inherent aspects of human beings that represent a strong tool to the improvement of QoS in WSN. With the use of measurement, monitoring and analysis mechanisms by nodes where the cognitive processes occur, it is possible to improve the network metrics with the actuation of Decision Making process. The DL verification represents a relation of the vectors that integrates the DM to provide the system with the action rules sent by V_C. This procedure guarantees the establishment of actions directly in the nodes and, therefore, an improvement in the performance regarding to the data packet delivery. In fact, with CPMod insertion into sensor nodes, more internal tasks in processing are needed, but this control results in a great gain concerning error corrections in the nodes mechanisms, covering queue, routing protocols adjustments, energy consumed by nodes and in the node memory usage for sensor applications. The use of Fuzzy Logic to identify the network situation turns the DMp more robust and dynamic. The V_C with Fuzzy Logic is a strong tool to help the CPMod in the decision making, setting up the CPMod to apply changes in multiple layers of the network.

For future researches some issues should be treated as follows: more nodes can be inserted into DM analysis, the feature concerning energy usage can be explored expanding F_N, V_M and V_C, another control scheme can be defined as complementary solution for DMp, add more elements in Fuzzy block for observation and control of network, the rules and pertinence functions can be modified and increased, and comparisons with other WSN protocols concerning control and metrics improvement in the network point of view.

ACKNOWLEDGMENT

Copyright @ Computing Publishing UK, ISSN 2053-6283

M.S.W. thanks all his coworkers and friends and especially his wife for participating in the accomplishment of this work that is a part of his PhD thesis.

This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under Grant 307633/2011-0 and by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) under Grant 2012/24789-0.

REFERENCES

[1] LI, J. et al., "Efficient Traffic Aware Multipath Routing Algorithm in Cognitive Networks", Fifth International Conference on Genetic and Evolutionary Computing (ICGEC), Xiamen, **(2011)** September.

[2] ALCARAZ, C. et al., "Wireless Sensor Networks and the Internet of Things: Do We Need a Complete Integration?", Computer Science Department of University of Malaga, 1st International Workshop on the Security of the Internet of Things (SecIoT'10), Tokyo, Japan, **(2010)** December.

[3] EZREIK, A. and GHERYANI, A., "Design and Simulation of Wireless Network using NS-2", 2nd International Conference on Computer Science and Information Technology (ICCSIT'2012), Singapore, **(2012)** April.

[4] INTANAGONWIWAT, S.; GOVINDAN, R. and ESTRIN, D., "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", Proceedings of the 6th annual International Conference on Mobile Computing and Networking (MobiCom'2000), Boston, USA, **(2000)** August 6-11.

[5] SINGH, N.; LAL DUA, R. and MATHUR, V., "Network Simulator NS2-2.35", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), vol. 2, issue 5, **(2012)** May, pp. 224-228.

[6] SOMOV, A.; DUPONT, C. and GIAFFREDA, R., "Supporting Smart-city Mobility with Cognitive Internet of Things", Conference of Future Network & MobileSummit'2013, International Information Management Corporation (IIMC), Portugal, Lisboa, **(2013)** July.

[7] VIJAY, G.; BDIRA, E. B. A. and IBNKAHLA, M., "Cognition in Wireless Sensor Networks: A Perspective", IEEE Sensors Journal, vol. 11, n. 3, **(2011)** March, pp. 582-592.

[8] WANG, Q., "Traffic Analysis & Modeling in Wireless Sensor Networks and Their Applications on Network Optimization and Anomaly Detection", Macrothink Institute, Network Protocols and Algorithms, vol. 2, n. 1, Sundsvall, Sweden, **(2010)**, pp. 74-92.

[9] WEI YE et al., "Evaluating Control Strategies for Wireless-Networked Robots Using an Integrated Robot and Network Simulation", Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2001), Seoul, Korea, **(2001)** May.

[10] ZHANG, N.; GUAN, J. and XU, C., "Traffic Prediction Model for Cognitive Networks", Proceedings of International Conference on Advanced Intelligence and Awareness Internet (AIAI'2011), Shenzhen, China, **(2011)** October.

[11] ZHANG, M. et al., "Cognitive Internet of Things: Concepts and Application Example", International Journal of Computer Science Issues (IJCSI), vol. 9, issue 6, n. 3, November, **(2012)**, pp. 151-158.

[12] GS Tomar, Tripti Sharma, Brijesh Kumar, "Fuzzy based Ant colony Optimization Approach for Wireless Sensor Network", Wireless Personal Communication, Vol.84, No.1, pp.361-375, May 2015.

Copyright @ Computing Publishing UK, *ISSN 2053-6283*

ANEXO III

Datasheet do MICAz (família MICA *motes*)

MICAz

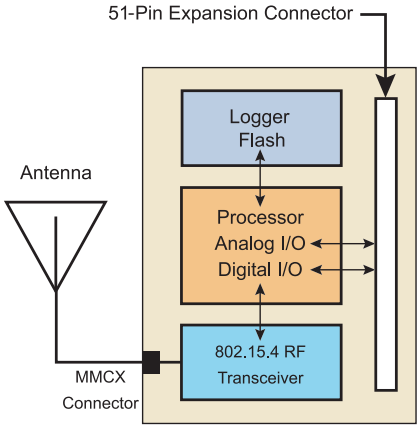
WIRELESS MEASUREMENT SYSTEM

- 2.4 GHz IEEE 802.15.4, Tiny Wireless Measurement System
- Designed Specifically for Deeply Embedded Sensor Networks
- 250 kbps, High Data Rate Radio
- Wireless Communications with Every Node as Router Capability
- Expansion Connector for Light, Temperature, RH, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic and other MEMSIC Sensor Boards

Applications

- Indoor Building Monitoring and Security
- Acoustic, Video, Vibration and Other High Speed Sensor Data
- Large Scale Sensor Networks (1000+ Points)

ZigBee™ Alliance



MPR2400 Block Diagram

MICAz

The MICAz is a 2.4 GHz Mote module used for enabling low-power, wireless sensor networks.

Product features include:

- IEEE 802.15.4 compliant RF transceiver
- 2.4 to 2.48 GHz, a globally compatible ISM band
- Direct sequence spread spectrum radio which is resistant to RF interference and provides inherent data security
- 250 kbps data rate
- Supported by MoteWorks™ wireless sensor network platform for reliable, ad-hoc mesh networking
- Plug and play with MEMSIC's sensor boards, data acquisition boards, gateways, and software

MoteWorks™ enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and a configuration.

Processor & Radio Platform (MPR2400CA)

The MPR2400 is based on the Atmel ATmega128L. The ATmega128L is a low-power microcontroller which runs MoteWorks from its internal flash memory. A single processor board (MPR2400) can be configured to run your sensor application/processing and the network/radio communications stack simultaneously. The 51-pin expansion connector supports Analog Inputs, Digital I/O, I2C, SPI and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals. The MICAz (MPR2400) IEEE 802.15.4 radio offers both high speed (250 kbps) and hardware security (AES-128).

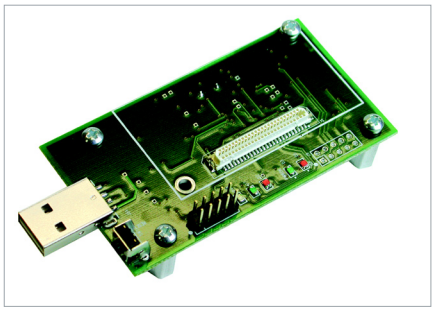
Sensor Boards

MEMSIC offers a variety of sensor and data acquisition boards for the MICAz Mote. All of these boards connect to the MICAz via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available. Please contact MEMSIC for additional information.

Phone: 408.964.9700 | Fax: 408.854.7702 | E-mail: infoca@memsic.com | www.memsic.com

Processor/Radio Board	MPR2400CA	Remarks
Processor Performance		
Program Flash Memory	128K bytes	
Measurement (Serial) Flash	512K bytes	> 100,000 Measurements
Configuration EEPROM	4K bytes	
Serial Communications	UART	0-3V transmission levels
Analog to Digital Converter	10 bit ADC	8 channel, 0-3V input
Other Interfaces	Digital I/O,I2C,SPI	
Current Draw	8 mA	Active mode
	< 15 µA	Sleep mode
RF Transceiver		
Frequency band[1]	2400 MHz to 2483.5 MHz	ISM band, programmable in 1 MHz steps
Transmit (TX) data rate	250 kbps	
RF power	-24 dBm to 0 dBm	
Receive Sensitivity	-90 dBm (min), -94 dBm (typ)	
Adjacent channel rejection	47 dB	+ 5 MHz channel spacing
	38 dB	- 5 MHz channel spacing
Outdoor Range	75 m to 100 m	1/2 wave dipole antenna, LOS
Indoor Range	20 m to 30 m	1/2 wave dipole antenna
Current Draw	19.7 mA	Receive mode
	11 mA	TX, -10 dBm
	14 mA	TX, -5 dBm
	17.4 mA	TX, 0 dBm
	20 µA	Idle mode, voltage regular on
	1 µA	Sleep mode, voltage regulator off
Electromechanical		
Battery	2X AA batteries	Attached pack
External Power	2.7 V - 3.3 V	Molex connector provided
User Interface	3 LEDs	Red, green and yellow
Size (in)	2.25 x 1.25 x 0.25	Excluding battery pack
(mm)	58 x 32 x 7	Excluding battery pack
Weight (oz)	0.7	Excluding batteries
(grams)	18	Excluding batteries
Expansion Connector	51-pin	All major I/O signals

Notes
[1]5 MHz steps for compliance with IEEE 802.15.4/D18-2003.
Specifications subject to change without notice



MIB520CB Mote Interface Board

Base Stations

A base station allows the aggregation of sensor network data onto a PC or other computer platform. Any MICAz Mote can function as a base station when it is connected to a standard PC interface or gateway board. The MIB510 or MIB520 provides a serial/USB interface for both programming and data communications. MEMSIC also offers a stand-alone gateway solution, the MIB600 for TCP/IP-based Ethernet networks.

Ordering Information

Model	Description
MPR2400CA	2.4 GHz MICAz Processor/Radio Board
WSN-START2400CA	2.4 GHz MICAz Starter Kit
WSN-PRO2400CA	2.4 GHz MICAz Professional Kit

Contato com autor:
marcel-wagner@hotmail.com
marcel.s.wagner@gmail.com