UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

RENATO SANTOS CARRIJO

**Proposta de roteamento híbrido para redes de sensores sem fio usando
inteligência Swarm (Ant Colony Optimization) combinada a métricas do
RPL para economia de energia**

São Carlos

2018

RENATO SANTOS CARRIJO

Proposta de roteamento híbrido para redes de sensores sem fio usando inteligência Swarm (Ant Colony Optimization) combinada a métricas do RPL para economia de energia

Tese apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, como requisito para a obtenção do Título de Doutor em Ciências, Programa de Engenharia Elétrica

Orientador: Prof. Dr. Dennis Brandão

São Carlos

2018

Trata-se da versão corrigida da tese. A versão original se encontra disponível na EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica.

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTES TRABALHOS, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

C316p	<p>CARRIJO, RENATO SANTOS</p> <p>Proposta de roteamento híbrido para redes de sensores sem fio usando inteligência Swarm (Ant Colony Optimization) combinada a métricas do RPL para economia de energia / RENATO SANTOS CARRIJO; orientador DENNIS BRANDÃO. São Carlos, 2018.</p> <p>Tese (Doutorado)- Programa de Pós-Graduação em Engenharia Elétrica e Área de Concentração em Sistemas Dinâmicos -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2018.</p> <p>1. Otimização por colônia de formigas. 2. Redes de sensores sem fio. 3 Roteamento. I. Título.</p>
-------	---

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE JULGAMENTO

Candidato: Engenheiro **RENATO SANTOS CARRIJO**.

Título da tese: "Proposta de roteamento híbrido para redes de sensores sem fio usando inteligência Swarm (Ant Colony Optimization) combinada a métricas do RPL para economia de energia".

Data da defesa: 23/10/2018.

Comissão Julgadora:

Resultado:

Prof. Associado **Dennis Brandão**
(Orientador)
(Escola de Engenharia de São Carlos/EESC)

Aprovado

Prof. Associado **Evandro Luis Linhari Rodrigues**
(Escola de Engenharia de São Carlos/EESC)

Aprovado

Prof. Associado **Rogério Andrade Flauzino**
(Escola de Engenharia de São Carlos/EESC)

Aprovado

Prof. Dr. **Eduardo Paciência Godoy**
(Universidade Estadual Paulista "Júlio de Mesquita Filho"/UNESP-Sorocaba)

APROVADO

Prof. Dr. **Ivanovitch Medeiros Dantas da Silva**
(Universidade Federal do Rio Grande do Norte/UFRN)

Aprovado

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica:
Prof. Associado **Luís Fernando Costa Alberto**

Presidente da Comissão de Pós-Graduação:
Prof. Associado **Luís Fernando Costa Alberto**

AGRADECIMENTOS

Agradeço, em primeiro lugar, a Deus: autor da vida, fonte inesgotável de amor, sabedoria e ciência;

Ao professor e amigo Dr. Dennis Brandão, pela oportunidade, incentivo, amizade e motivação que me dedicou ao orientar esse trabalho;

À minha querida esposa, Rita de Cassia, pelo carinho, compreensão, amor e companheirismo em todas as etapas do doutorado;

Aos meus filhos Gabriel e Mariana, por, juntamente com minha esposa, serem a grande motivação dessa jornada;

Aos meus pais, ao meu sogro e à minha sogra, por todo apoio;

Aos meus irmãos;

Ao professor e amigo Renato Fernandes, pelo apoio no desenvolvimento do projeto;

À Universidade de São Paulo, por colocar à disposição sua estrutura;

À Universidade Federal de Uberlândia, especialmente à Faculdade de Engenharia Elétrica, pelo incentivo à pós-graduação;

A todos os amigos, que direta ou indiretamente contribuíram para que esse trabalho chegasse até aqui.

RESUMO

CARRIJO, R. S. **Proposta de roteamento híbrido para redes de sensores sem fio usando inteligência Swarm (Ant Colony Optimization) combinada a métricas do RPL para economia de energia.** 2018. 181 f. Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

As redes de sensores sem fio estão presentes nos mais diversos setores, formando um conjunto de elementos colaborativos que realizam o transporte da informação em cenários urbanos, agricultura de precisão, saúde e automação industrial. Sua utilização enfrenta desafios como necessidade constante de adaptações a alterações de topologia, baixas taxas de comunicação e uso eficiente de energia. Dessa forma, as aplicações dessas redes precisam suportar estas características dinamicamente. Esta tese propõe, então, um novo protocolo de roteamento: o HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*), com base no funcionamento do padrão RPL (*Routing Protocol for Low Power and Lossy Networks*) aplicado à eficiência energética. Este protocolo proposto preserva as interfaces já consolidadas de roteamento e faz uso de uma informação heurística baseada em *swarm intelligence* (inteligência de enxames), por meio do uso de agentes – colônia de formigas –, para a composição híbrida de métricas utilizadas na avaliação do *rank* durante a construção de topologias. Esta proposta foi validada por meio de implementações em simulações e dispositivos reais em duas plataformas distintas: o *Contiki* e o *OpenWSN*. O processo de validação utilizou parâmetros como número de mensagens de controle, alterações de configurações topológicas, latência da rede e taxa de entrega de pacotes comparando-se a proposta a soluções tradicionais. Os resultados obtidos demonstraram um funcionamento compatível com estes parâmetros e uma melhoria em torno de 7,6% no sentido de redução do tempo de estabilização da rede. Adicionalmente, foi feito um teste comparativo de consumo de energia, evidenciando uma melhoria em termos de eficiência energética de até 20,9% da proposta com relação a protocolos tradicionais.

Palavras-chave: Otimização por colônia de formigas. Redes de sensores sem fio. Roteamento.

ABSTRACT

CARRIJO, R. S. Hybrid routing proposal for wireless sensor networks using Swarm intelligence (Ant Colony Optimization) combined with RPL metrics for energy savings. 2018. 181 f. Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

The wireless sensor networks are present in the most diverse sectors, forming a set of collaborative elements that perform the transport of information in urban scenarios, precision agriculture, health and industrial automation. Its use faces challenges as a constant need for adaptations to topology changes, low communication rates and efficient energy use. In this way, the applications of these networks must support these characteristics dynamically. This thesis proposes, therefore, a new routing protocol: the HOFACO (Hybrid Objective Function based on Ant Colony Optimization), based on the RPL (Routing Protocol for Low Power and Lossy Networks) applied to energy efficiency. This proposed protocol preserves the already consolidated routing interfaces and makes use of heuristic information based on swarm intelligence, using ant colony agents for the hybrid composition of metrics used in rank evaluation during the construction of topologies. This proposal was validated through implementations in simulations and real devices in two different platforms: Contiki and OpenWSN. The validation process used parameters such as number of control messages, changes of topological configurations, network latency and packet delivery rate, comparing the proposal to traditional solutions. The results showed a functioning compatible with these parameters and an improvement around 7.6% in the sense of reducing the time of stabilization of the network. In addition, a comparative test of energy consumption was made, evidencing an improvement in terms of energetic efficiency up to 20.9% of the proposal with respect to traditional protocols.

Keywords: Ant Colony Optimization. Wireless sensor networks. Routing.

LISTA DE ILUSTRAÇÕES

Figura 1 – Modelo de uma rede de sensores sem fio.....	35
Figura 2 – Composição do sensor	36
Figura 3 – Pilha de protocolos das redes de sensores sem fio	37
Figura 4 – Bandas de frequência	38
Figura 5 – Frame IEEE 802.15.4e	40
Figura 6 – Frame IEEE 802.15.4e	42
Figura 7 – Superframe do padrão IEEE 802.15.4e.....	43
Figura 8 – Topologias de rede do padrão IEEE 802.15.4e.....	45
Figura 9 – O protocolo RPL	52
Figura 10 – Processo de formação do DODAG	53
Figura 11 – Aplicação da Arquitetura OpenWSN.....	56
Figura 12 – Pilha da arquitetura do <i>OpenWSN</i>	56
Figura 13 – <i>OpenVisualizer Web - OpenWSN</i>	57
Figura 14 – Arquitetura do <i>Contiki</i>	58
Figura 15 – COOJA Network Simulator	59
Figura 16 – Comportamento das formigas	63
Figura 17 – Algoritmo ACO.....	64
Figura 18 – Fluxograma do Algoritmo Proposto baseado em Colônia de Formigas	79
Figura 19 – Envio da mensagem de i para j	81
Figura 20 – Envio da mensagem de i para j (FORWARD ANT).....	81
Figura 21 – Envio da mensagem de j para i	81
Figura 22 – Envio da mensagem de j para i (BACKWARD ANT)	82
Figura 23 – Envio da mensagem de j para i (BACKWARD ANT) – Depósito do feromônio	82
Figura 24 – Depósito do feromônio – Atualização da lista de vizinhos.....	82
Figura 25 – Pilha da arquitetura do OpenWSN.....	86
Figura 26 – Arquitetura do Contiki	87
Figura 27 – Placa eletrônica desenvolvida	88
Figura 28 – Powertrace– Cápsulas de Energia	89
Figura 29 – Powertrace – Cápsulas de Energia	90
Figura 30 – Cenário 1 – Topologia com 6 nós (1 <i>sink</i> e 5 <i>motes</i>).....	93
Figura 31 – Cenário 1 – Visibilidade dos nós – Parte 1	95
Figura 32 – Cenário 1 – Visibilidade dos nós – Parte 2	96
Figura 33 – Cenário 1 – Número de mensagens de Controle RPL	97
Figura 34 – Cenário 1 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo OF0.....	98
Figura 35 – Cenário 1 - Evolução do intervalo do envio de mensagens	98
Figura 36 – Cenário 1 – Evolução do intervalo do envio de mensagens.....	99
Figura 37 – Cenário 1 – Quantidade de alterações de nós <i>parent</i>	99
Figura 38 – Cenário 1 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo	100
Figura 39 – Cenário 1 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo	101
Figura 40 – Cenário 1 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo	102
Figura 41 – Cenário 1 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo	102

Figura 42 – Cenário 2 – Topologia com 1 nós (1 <i>sink</i> e 9 <i>motes</i>)	105
Figura 43 – Cenário 2 – Visibilidade dos nós – Parte 1.....	107
Figura 44 – Cenário 2 – Visibilidade dos nós – Parte 2.....	108
Figura 45 – Cenário 2 – Visibilidade dos nós – Parte 3.....	109
Figura 46 – Cenário 2 – Número de mensagens de Controle RPL.....	110
Figura 47 – Cenário 2 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo OF0	111
Figura 48 – Cenário 2 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo ETX	111
Figura 49 – Cenário 2 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo HOFACO.....	112
Figura 50 – Cenário 2 – Quantidade de alterações de nós <i>parent</i>	112
Figura 51 – Cenário 2 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo.....	113
Figura 52 – Cenário 2 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo.....	114
Figura 53 – Cenário 2 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo.....	115
Figura 54 – Cenário 2 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo.....	115
Figura 55 – Cenário 3 – Topologia com 1 nós (1 <i>sink</i> e 9 <i>motes</i>)	117
Figura 56 – Cenário 3 – Visibilidade dos nós – Parte 1.....	119
Figura 57 – Cenário 3 – Visibilidade dos nós – Parte 2.....	120
Figura 58 – Cenário 3 – Visibilidade dos nós – Parte 3.....	121
Figura 59 – Cenário 3 – Número de mensagens de Controle RPL.....	122
Figura 60 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo OF0	123
Figura 61 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo ETX	123
Figura 62 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo HOFACO.....	124
Figura 63 – Cenário 3 – Quantidade de alterações de nós <i>parent</i>	124
Figura 64 – Cenário 3 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo.....	125
Figura 65 – Cenário 3 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo.....	126
Figura 66 – Cenário 3 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo.....	127
Figura 67 – Cenário 3 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo.....	127
Figura 68 – Cenário 4 – Topologia com 1 nós (1 <i>sink</i> e 9 <i>motes</i>)	129
Figura 69 – Cenário 4 – Visibilidade dos nós – Parte 1.....	131
Figura 70 – Cenário 4 – Visibilidade dos nós – Parte 2.....	132
Figura 71 – Cenário 4 – Visibilidade dos nós – Parte 3.....	133
Figura 72 – Cenário 4 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo.....	134
Figura 73 – Cenário 4 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo.....	136
Figura 74 – Cenário 4 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo.....	138

Figura 75 – Cenário 4 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo	138
Figura 76 – Cenário 5 – Topologia com 10 nós (1 <i>sink</i> e 9 <i>motes</i>)	140
Figura 77 - Cenário 5 – <i>OpenVisualizer Web</i>	141
Figura 78 – Cenário 5 – Número de mensagens de Controle RPL	142
Figura 79 – Cenário 5 – Quantidade de alterações de nós <i>parent</i>	143
Figura 80 – Cenário 5 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo	143
Figura 81 – Cenário 5 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo	144
Figura 82 – Cenário 5 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo	145
Figura 83 – Cenário 5 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo	146
Figura 84 – Cenário 6 – Topologia com 1 nós (1 <i>sink</i> e 9 <i>motes</i>)	147
Figura 85 – Cenário 6 – Número de mensagens de Controle RPL	149
Figura 86 – Cenário 6 – Quantidade de alterações de nós <i>parent</i>	149
Figura 87 – Cenário 6 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo	150
Figura 88 – Cenário 6 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo	151
Figura 89 – Cenário 6 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo	152
Figura 90 – Cenário 6 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo	152
Figura 91 – Cenário 7 – Topologia com 1 nós (1 <i>sink</i> e 9 <i>motes</i>)	154
Figura 92 – Cenário 7 – Número de mensagens de Controle RPL	156
Figura 93 – Cenário 7 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo OF0.....	156
Figura 94 – Cenário 7 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo ETX.....	157
Figura 95 – Cenário 7 – Evolução do intervalo do envio de mensagens DIO (<i>Trickle Timer</i>) para a Função Objetivo HOFACO	157
Figura 96 – Cenário 7 – Quantidade de alterações de nós <i>parent</i>	158
Figura 97 – Cenário 7 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo	158
Figura 98 – Cenário 7 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo	159
Figura 99 – Cenário 7 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo	160
Figura 100 – Cenário 7 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo	161
Figura 101 – Cenário 7 – Consumo de energia acumulado x tempo.....	162
Figura 102 – Extensão da arquitetura do OpenWSN	177
Figura 103 – Formato da Mensagem de Log.....	178
Figura 104 – Formato da Mensagem de Log.....	178
Figura 105 – Captura dos dados de <i>debugging</i> com o Wireshark.....	180

LISTA DE TABELAS

Tabela 1 – Bandas de frequência e taxa de dados	38
Tabela 2 – Fenômenos biológicos e aplicações.....	62
Tabela 3 – Configurações Gerais do Cenário 1 no Simulador <i>COOJA</i>	94
Tabela 4 – Cenário 1 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	100
Tabela 5 – Cenário 1 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes.....	101
Tabela 6 – Configurações Gerais do Cenário 2 no Simulador <i>COOJA</i>	106
Tabela 7 – Cenário 2 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes.....	113
Tabela 8 – Cenário 2 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	114
Tabela 9 – Configurações Gerais do Cenário 3 no Simulador <i>COOJA</i>	118
Tabela 10 – Cenário 3 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	125
Tabela 11 – Cenário 3 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	126
Tabela 12 – Configurações Gerais do Cenário 4 no Simulador <i>COOJA</i>	130
Tabela 13 – Cenário 4 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	135
Tabela 14 – Cenário 4 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	137
Tabela 15 – Cenário 5 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	144
Tabela 16 – Cenário 5 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	145
Tabela 17 – Cenário 6 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	150
Tabela 18 – Cenário 6 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	151
Tabela 19 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes	159
Tabela 20 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes	160

LISTA DE ABREVIATURAS E SIGLAS

ACK	<i>Acknowledgement</i>
ACO	<i>Ant Colony Optimization</i>
ACS	<i>Ant Colony System</i>
AntNET	<i>Ant Network</i>
AODV	<i>Ad-Hoc On Demand Distance Vector</i>
AP-TEEN	<i>Adaptative Threshold sensitive Energy Efficient sensor Network Protocol</i>
AS	<i>Ant System</i>
BER	<i>Bit Error Rate</i>
BI	<i>Beacon Interval</i>
BO	<i>macBeaconOrder</i>
CAP	<i>Contention Access Period</i>
CCA	<i>Clear Channel Assessment</i>
CFP	<i>Content Free Period</i>
CGSR	<i>Clusterhead Gateway Switching Routing</i>
CPU	<i>Central Processing Unit</i>
CRC	<i>Cyclic Redundancy Check</i>
CSMA/CA	<i>Carrier Sense Multiple Access with Collision Avoidance</i>
DAG	<i>Directed Acyclic Graph</i>
DAO	<i>Destination Advertisement Object</i>
DD	<i>Directed Diffusion</i>
DIO	<i>DODAG Information Object</i>
DIS	<i>DODAG Information Solicitation</i>
DODAG	<i>Destination-Oriented Directed Acyclic Graph</i>
DSDV	<i>Destination-Sequenced Distance-Vector Routing</i>
DSR	<i>Dynamic Source Roting</i>
EC	<i>Energia Consumida</i>
ED	<i>Energy Detection</i>
ETX	<i>Expected Transmission Count</i>
FFD	<i>Full Function Device</i>
GAF	<i>Geographic Adaptive Fidelity</i>
GEAR	<i>Geographic and Energy-Aware Routing</i>

GPIO	<i>General Purpose Input Output</i>
GTS	<i>Guaranteed Time Slots</i>
HOFACO	<i>Hybrid Objective Function based on Ant Colony Optimization</i>
ICMPv6	<i>Internet Control Message Protocol version 6</i>
IEEE	<i>Institute of Electrical and Electronic Engineers</i>
IETF	<i>Internet Engineering Task Force</i>
IP	<i>Internet Protocol</i>
IPv4	<i>Internet Protocol version 4</i>
IPv6	<i>Internet Protocol version 6</i>
ISM	<i>Industrial, Scientific and Medical</i>
JTAG	<i>Joint Test Access Group</i>
LEACH	<i>Low-Energy Adaptive Clustering Hierarchy</i>
LQI	<i>Link Quality Indicator</i>
LR-WPAN	<i>Low Rate-Wireless Personal Area Network</i>
MAC	<i>Medium Access Control</i>
MANET	<i>Mobile Ad Hoc Networking</i>
MECN	<i>Minimum Energy Communication Network</i>
MFR	<i>MAC Footer</i>
MHC	<i>MAC Header</i>
MMAS	<i>Max-Min Ant System</i>
MPDU	<i>Mac Payload Data Unit</i>
MRHOF	<i>Minimum Rank With Hysteresis Objective Function</i>
MRM	<i>Multi-path Raytracer Medium</i>
NS-2	<i>Network simulator versão 2</i>
NS-3	<i>Network simulator versão 3</i>
OF	<i>Objective Function</i>
OF0	<i>Objective Function Zero</i>
OSI	<i>Open Systems Interconnection</i>
PAN	<i>Personal Area Network</i>
PER	<i>Packet Error Rate</i>
PHR	<i>Physical Header</i>
PHY	<i>Physical Layer</i>
PPDU	<i>Physical Protocol Data Unit</i>

PSDU	<i>Physical Service Data Unit</i>
PSO	<i>Particle Swarm Optimization</i>
RFD	<i>Reduced Function Device</i>
ROLL	<i>Routing Over Low power and Lossy networks</i>
RPL	<i>Routing Protocol for Low Power and Lossy Networks</i>
RSSF	<i>Redes de sensores sem fio</i>
RSSI	<i>Received Signal Strength Indication</i>
RX	<i>Recepção</i>
SD	<i>Superframe Duration</i>
SHR	<i>Sync Header</i>
SO	<i>macSuperframeOrder</i>
SPIN	<i>Sensor Protocols for Information via Negotiation</i>
TCP	<i>Transmission Control Protocol</i>
TEEN	<i>Threshold sensitive Energy Efficient sensor Network protocol</i>
TSP	<i>Traveling Salesman Problem</i>
TTL	<i>Time to Live</i>
TX	<i>Transmissão</i>
UDGM	<i>Unit Disk Graph Medium</i>
UDP	<i>User Datagram Protocol</i>
UDP OTG	<i>Universal Serial Bus On The Go</i>
WSN	<i>Wireless Sensor Networks</i>
6LoWPAN	<i>IPv6 over Low power Wireless Personal Area Network</i>

SUMÁRIO

1.	INTRODUÇÃO.....	27
1.1.	Justificativa.....	29
1.2.	Objetivos da Tese	30
1.3.	Contribuições e originalidade desta tese	31
1.4.	Estrutura do Trabalho	32
2.	REDES DE SENSORES SEM FIO	33
2.1.	Características Gerais	33
2.2.	A composição do sensor	35
2.3.	O Padrão IEEE 802.15.4e.....	36
2.3.1.	A camada física	37
2.3.2.	A camada de enlace (ou camada MAC)	40
2.3.1.	O formato do frame	41
2.4.	Dispositivos de rede	44
2.5.	Topologias de rede.....	44
2.6.	Consumo de Energia.....	45
2.7.	Roteamento em Redes de Sensores sem Fio	48
2.8.	O Protocolo RPL	49
2.9.	Plataformas de Simulação para Redes de Sensores sem Fio	55
2.10.	Discussão	59
3.	SISTEMAS INTELIGENTES BASEADOS EM COLÔNIA DE FORMIGAS	60
3.1.	Swarm Intelligence	60
3.2.	Sistemas bioinspirados	61
3.3.	Colônia de Formigas.....	62
3.4.	Algoritmo Ant System.....	65
3.5.	Algoritmo MAX-MIN Ant System	66
3.6.	Algoritmo Ant Colony System	67
3.7.	Discussão	69
4.	ROTEAMENTO EM REDES DE SENSORES SEM FIO	70
4.1.	Análise Geral	70
4.2.	Avaliação das características de utilização.....	71
4.3.	Desafios inerentes ao roteamento	72
4.4.	Estado da arte das pesquisas na área de roteamento.....	73

5.	MÉTODO, PROPOSTA DO ALGORITMO, ASPECTOS DE IMPLEMENTAÇÃO E CONSUMO DE ENERGIA.....	77
5.1.	Discussões iniciais e desafios	77
5.2.	Método e Apresentação do Algoritmo	78
5.3.	Aspectos de implementação.....	85
5.3.1.	OpenWSN.....	85
5.3.2.	Contiki.....	86
5.4.	Hardware utilizado.....	87
5.5.	Mecanismo adotado para verificação do consumo de energia.....	88
6.	EXPERIMENTOS REALIZADOS E RESULTADOS.....	92
6.1.	Cenário 1 – Topologia com 6 nós sensores – Simulação com <i>COOJA</i>	93
6.1.1.	Itens analisados e Resultados	96
6.1.2.	Discussões.....	103
6.2.	Cenário 2 – Topologia com 10 nós sensores – Simulação com <i>COOJA</i>	105
6.2.1.	Itens analisados e Resultados	109
6.2.2.	Discussões.....	116
6.3.	Cenário 3 – Topologia com 10 nós sensores – Simulação com <i>COOJA</i>	117
6.3.1.	Itens analisados e Resultados	121
6.3.2.	Discussões.....	128
6.4.	Cenário 4 – Topologia com 10 nós sensores – Simulação com <i>COOJA</i>	129
6.4.1.	Itens analisados e Resultados	133
6.4.2.	Discussões.....	139
6.5.	Cenário 5 – Topologia com 10 nós sensores – Simulação com <i>OpenVisualizer</i> – <i>OpenWSN</i>	140
6.5.1.	Itens analisados e Resultados	141
6.5.1.	Discussões.....	146
6.6.	Cenário 6 – Topologia com 10 nós sensores (hardware real) – Plataforma <i>OpenWSN</i> 147	
6.6.1.	Itens analisados e Resultados	148
6.6.2.	Discussões.....	153
6.7.	Cenário 7 – Topologia com 10 nós sensores (hardware real) – Plataforma <i>Contiki</i> ..	154
6.7.1.	Itens analisados e Resultados	155
6.7.2.	Avaliação do consumo de energia	161
6.7.3.	Discussões.....	163
6.8.	Análise Geral dos Cenários de utilização	164
7.	CONCLUSÕES	166

Trabalhos Futuros	168
8. REFERÊNCIAS BIBLIOGRÁFICAS	169
Apêndice A	176

CAPÍTULO 1

1. INTRODUÇÃO

Nos últimos anos, houve uma grande evolução nas arquiteturas empregadas em redes de comunicação, permitindo a integração entre diversos dispositivos, equipamentos e aplicações. Esta evolução teve seu início a partir de tecnologias de sistemas eletromecânicos em escalas de tamanho reduzido, permitindo o desenvolvimento de nós sensores de baixo custo, capazes de realizar uma comunicação sem fio com equipamentos e sistemas digitais. Estes dispositivos, chamados de nós sensores, quando organizados de acordo com um padrão de comunicação, constituem uma rede de sensores sem fio (RSSF). Estas redes, por sua vez, apresentam seus elementos distribuídos em uma determinada área com a finalidade de monitoramento de grandezas físicas como, por exemplo: temperatura, pressão, umidade, qualidade do ar, etc. (CHHABRA; ARORA, 2017).

Cada componente capaz de transmitir ou receber informações pode, então, possuir uma interface de comunicação com um nó que participa da rede. Uma característica intrínseca de um nó de comunicação em uma rede de sensores sem fio é que, geralmente, ele é alimentado por uma bateria, cuja eficiência deveria ser a melhor possível, uma vez que o processo de substituição da fonte de energia (troca da bateria) acarretaria algum tipo de manutenção, tendo como consequência uma interrupção do funcionamento do nó de comunicação (ÅKERBERG et al., 2011).

As aplicações para as RSSF são bastante abrangentes, podendo contemplar o controle de tráfego em redes urbanas, monitoramento de redes elétricas, aplicações médicas e ambientes industriais. Estas redes situadas em ambientes industriais passaram a ter fundamental importância nos sistemas de automação a partir do momento em que a sua utilização gerou benefícios como eficiência, flexibilidade e agilidade a partir da integração entre instrumentos

sensores, atuadores e sistemas de controle. A troca de informações relevantes entre diversos níveis, permite, assim, uma comunicação mais efetiva entre as diversas camadas de uma solução (PALATTELLA et al., 2013).

A pesquisa e o desenvolvimento na área das redes de sensores sem fio, é, portanto, uma área multidisciplinar e utiliza conceitos ligados a sistemas eletrônicos embarcados, redes de comunicação sem fio, protocolos digitais, sistemas inteligentes e gerência e otimização de recursos. De acordo com Zorzi et al. (2016), são necessários novos paradigmas para que se possa lidar com os modernos sistemas de comunicação, uma vez que vários fatores complexos como tamanho e heterogeneidade se fazem presentes nas arquiteturas de comunicação. Os desafios se encontram na solução de problemas de complexidade, heterogeneidade e escalabilidade resultantes da evolução das próprias arquiteturas de comunicação (LIU et al., 2015).

Todavia, apesar desses desafios serem o resultado da própria evolução das arquiteturas de comunicação, eles já vêm sendo tratados por sistemas da natureza. Geralmente, a dinâmica de sistemas da natureza é composta por um conjunto reduzido e genérico de regras simples, baseadas em padrões colaborativos para gerenciamento dos recursos, distribuição de tarefas, diferenciação social e sincronismo sem a necessidade de uma interferência externa (ZHANG et al., 2018).

Dessa maneira, a evolução mostra vários aspectos oportunos, que são adequados para serem usados em redes de comunicação, como por exemplo: flexibilidade com relação à necessidade de adaptação às influências internas e externas, robustez a falhas, capacidade de auto-organização, capacidade de lidar com comportamentos complexos baseando-se em um conjunto de regras simples, capacidade de aprendizado a partir do surgimento de novas condições, eficiência na gestão de recursos limitados evidenciando-se uma aparente inteligência global e capacidade de colaboração (ZORZI et al., 2016).

Todos esses aspectos conduzem à possibilidade de inspiração em sistemas da natureza para a construção de algoritmos que possam ser aplicados na solução de problemas existentes em redes de comunicação, com o objetivo de torná-las mais eficientes, robustas e confiáveis.

De acordo com Ayaz et al. (2017), vários pesquisadores estão empenhados no desenvolvimento de paradigmas inovadores aplicados a redes de sensores sem fio. Conforme descrito por Zhang et al. (2018), a razão dessas pesquisas é a captura da dinâmica existente nos sistemas biológicos e o entendimento dos seus fundamentos para a criação de novos métodos e ferramentas capazes de projetar e gerir as redes de comunicação adaptáveis a ambientes heterogêneos, dinâmicos, escaláveis, com auto-organização e evolutivos.

1.1. Justificativa

Algumas pesquisas, como a de Gao et al. (2014), Byun et al. (2015) e Albert et al. (2018), afirmam que é possível encontrar soluções para os desafios das RSSF tomando-se como referência soluções bioinspiradas por meio de modelos computacionais baseados em algoritmos de *swarm intelligence* (ou inteligência de enxames). Estes modelos possuem pontos em comum com arquiteturas de comunicação complexas.

Esses pontos podem ser verificados comparando-se algoritmos e estruturas comuns, utilizados na Internet, como, por exemplo, o modelo OSI (*Open Systems Interconnection*), fundamentado em camadas, que formam as bases para a comunicação. A divisão em camadas permite que aplicações de níveis mais altos, possam trabalhar sobre um conjunto menor de protocolos de transporte, que, por sua vez, operam sobre protocolos de rede, que se encontram acima de outros protocolos de enlace e meios físicos diversos, os quais podem ser substituídos por outros meios sem nenhum prejuízo de funcionalidade (LI et al., 2011).

Por um lado, tomando-se como referência a pilha de protocolos da Internet, é possível observar várias similaridades no nível de protocolos individuais. O protocolo TCP (*Transmission Control Protocol*), da camada de transporte, por exemplo, determina a taxa de transmissão com base em um algoritmo de controle de congestionamento padrão. Esse algoritmo utiliza o *feedback* do receptor na forma de confirmações por meio das mensagens para determinar quando a taxa do transmissor deveria ser ajustada. Enquanto o transmissor estiver recebendo as respectivas respostas de reconhecimento (*feedback* positivo), ele aumenta lentamente sua velocidade de transmissão. Quando o transmissor recebe *feedback* negativo, ele reduz rapidamente a sua velocidade de transmissão.

Agregado a isto, este algoritmo permite que a taxa do transmissor controle continuamente a taxa de envio ideal para o receptor, considerando o estado atual da rede (GHARGE; VALANJOO, 2014). No campo da teoria de controle, esse processo pode ser visto na forma de *feedback* integral (DOYLE et al., 2013). Essa característica, comparada com sistemas da natureza, reflete a capacidade de adaptação a condições do próprio meio.

Por outro lado, tomando-se como referência sistemas da natureza, este comportamento pode ser também observado em grupos ou colônias de bactérias que realizam a busca de nutrientes que servirão como alimento. Pesquisas demonstram fortes evidências de que as bactérias utilizam o *feedback* integral para controlar a velocidade e direção de movimento quando estão rastreando a concentração de certas substâncias químicas em seu ambiente (LILLACCI et al., 2017).

Esse comportamento das bactérias segue um padrão de construção hierárquico no qual o processo metabólico é realizado no sentido de se produzir substâncias mais simples, as quais, posteriormente, serão remontadas para a construção de macromoléculas complexas, servindo estas últimas como alimento das colônias de bactérias. De maneira análoga, uma organização hierárquica está presente nos protocolos com múltiplas camadas, como, por exemplo: a Internet; refletindo assim, a utilização de uma abordagem hierárquica em sistemas complexos (LI et al., 2011).

Existem, portanto, vários desafios para a evolução das arquiteturas das redes de comunicação atuais e para a implementação de novas arquiteturas.

Dentre os desafios, um ponto importante a ser considerado é o consumo eficiente de energia - objeto de estudo desta tese - realizado pela comunicação entre os diversos nós de equipamentos interligados por meio de uma rede de sensores sem fio.

Al-Jemeli e Hussin (2015) apresentam um modelo de operação em camadas com foco na eficiência energética aplicado a redes de sensores sem fio que utilizam o padrão IEEE 802.15.4e. Da mesma maneira, a eficiência energética também é foco das pesquisas de Tabus et al. (2015) e Wang et al. (2018). As pesquisas de Saleem et al. (2014) e Wang et al. (2014), por sua vez, apresentam uma proposta de eficiência energética realizada a partir de uma estratégia de roteamento baseada em sistemas bioinspirados.

1.2. Objetivos da Tese

Essa tese tem como objetivo geral propor uma arquitetura de roteamento aplicável a redes de sensores sem fio com a finalidade de se atingir a eficiência energética comparada a modelos já existentes. Para isso é proposto um novo protocolo de roteamento: o HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*), baseado no funcionamento do padrão RPL (*Routing Protocol for Low Power and Lossy Networks*). Este novo protocolo preserva as interfaces de roteamento do RPL e usa uma informação heurística, concebida a partir de um padrão bioinspirado utilizando-se *swarm intelligence* (inteligência de enxames) a partir de agentes – colônia de formigas –, fazendo a composição híbrida de métricas utilizadas na avaliação do *rank* durante a construção de topologias.

Como objetivos específicos são enumerados:

- a) Implementação do protocolo HOFACO em duas plataformas de redes de sensores sem fio distintas capazes de serem utilizadas também em dispositivos reais: o *Contiki* e o *OpenWSN*.

b) Verificação e validação do protocolo HOFACO a partir dos aspectos: números de mensagens de controle, alterações de configurações topológicas, latência da rede e taxa de entrega comparando-se o mesmo aos padrões de avaliação de *rank* no RPL usando-se número de saltos e expectativa de transmissão.

c) Avaliação do consumo de energia do protocolo HOFACO com relação ao RPL tradicional baseado no número de saltos e expectativa de transmissão, aplicado a dispositivos reais usando-se a plataforma *Contiki*.

1.3. Contribuições e originalidade desta tese

Essa tese propõe uma arquitetura de comunicação que contempla o uso de *swarm intelligence* por meio de agentes (colônias de formigas) aplicado ao mecanismo de roteamento para garantir a eficiência energética de redes de sensores sem fio e sem a necessidade de envio de mensagens de controle adicionais para garantir o roteamento.

O presente trabalho apresenta de forma inédita a proposta do protocolo HOFACO, cujo algoritmo é baseado em colônia de formigas, concebido a partir do uso de dois agentes: *Forward Ant* e *Backward Ant*, aplicando o seu resultado na composição de métricas para avaliação do *rank* do protocolo RPL sem a necessidade de envio de mensagens adicionais. O depósito de feromônio é feito por meio da comunicação já existente e os caminhos pelos quais a comunicação acontece são reforçados, tornando a rede mais estável com o passar do tempo.

Este é o primeiro trabalho que apresenta um algoritmo aplicado à rede de sensores sem fio por meio do uso de agentes (colônia de formigas) sem a necessidade de envio de mensagens adicionais de controle. Da mesma forma, são inéditas as implementações realizadas para uso desta arquitetura proposta nas plataformas *Contiki* e *OpenWSN*.

A proposta não apresenta nenhuma quebra de compatibilidade com interfaces já existentes em arquiteturas já consolidadas. Ao invés, propõe uma solução de composição híbrida do *rank*, capaz de melhorar a avaliação da função objetivo dentro do padrão de roteamento do RPL, permitindo, ainda, a convivência desses motes com outros convencionais.

Durante o curso deste trabalho, foi também desenvolvida uma ferramenta específica e uma arquitetura de *debugging* para auxiliar o processo de testes e validações na plataforma *OpenWSN*. Esta ferramenta é descrita no apêndice A.

1.4. Estrutura do Trabalho

No primeiro capítulo é apresentado o contexto da área de conhecimento onde este trabalho está inserido, bem como a justificativa, os objetivos e contribuições.

O capítulo 2 apresenta características das redes de sensores sem fio.

O capítulo 3 descreve os principais conceitos envolvidos de *swarm intelligence* e algoritmos bioinspirados baseados em colônia de formigas.

No capítulo 4 são apresentadas questões sobre as pesquisas relacionadas ao roteamento em redes de sensores sem fio.

O capítulo 5 descreve os detalhes do método utilizado e desenvolvimento.

O capítulo 6 descreve os experimentos e os resultados obtidos.

Finalmente, o capítulo 7 apresenta a conclusão.

CAPÍTULO 2

2. REDES DE SENSORES SEM FIO

Este capítulo apresenta o contexto das redes de sensores sem fio. Nas seções seguintes são apresentadas características gerais das redes de sensores sem fio, a composição de um nó sensor, os detalhes do padrão IEEE 802.15.4e e os tipos de dispositivos de rede.

Além disso, são também abordados aspectos relacionados à topologia, consumo de energia, padrões de roteamento, protocolo RPL e plataformas de simulação.

2.1. Características Gerais

De acordo com Kazienko et al. (2015), uma rede de comunicação é formada por módulos processadores capazes de trocar informações entre si, sendo interligados através de um sistema formado por enlaces físicos – meios de transmissão – e de um conjunto de regras com a finalidade de se organizar a comunicação, denominado protocolos.

As RSSF (Redes de Sensores Sem Fio) ou WSN (*Wireless Sensor Networks*) são constituídas por dispositivos com tamanho reduzido, baixo poder de processamento, pequena quantidade de memória e limitações com relação ao uso de energia.

Estes dispositivos, por sua vez, são conectados entre si por meio de uma comunicação sem fio do tipo *ad hoc*.

O termo *ad hoc* é uma expressão proveniente do latim e que significa “para esta finalidade” ou “para isto”, “para este fim”.

Esta nomenclatura é adequada a redes de sensores com estas características, uma vez que não existe uma infraestrutura pré-definida e a comunicação pode ocorrer de forma temporária, sem, necessariamente, a necessidade de um elemento roteador central.

Neste caso, cada um dos elementos funciona com um roteador, sendo capaz de encaminhar à frente a comunicação proveniente de seus nós vizinhos. Isto significa que em uma rede *ad hoc* os dispositivos trocam dados diretamente entre si. As informações coletadas do ambiente são transmitidas por meio da comunicação sem fio, geralmente por meio de sinais de rádio frequência (IQBAL et al., 2013).

De acordo com este modelo, se algum caminho de comunicação falhar, um outro pode ser criado, de tal forma a se encaminhar a informação por meio de outros nós.

Um exemplo de rede com esta característica, é uma MANET (*Mobile Ad Hoc Network*). Neste modelo de rede, todos os nós podem ser roteadores. A transmissão acontece por meio da difusão de mensagens e cada nó possui apenas conectividade com um conjunto de nós próximos, denominados vizinhos.

Em uma MANET, os nós podem, ainda, se mover de forma aleatória tornando muito dinâmico o processo de comunicação da rede, gerando vários desafios com relação ao roteamento, gasto de energia e possível instabilidade na comunicação. (WATTEYNE et al., 2010).

De acordo com Reddy et al. (2016), do ponto de vista de organização, uma RSSF (rede de sensores sem fio) poderia ser vista como um tipo especial de rede móvel *ad hoc*, assim como uma MANET. Contudo, enquanto uma MANET tem a função principal de prover a comunicação entre os seus elementos computacionais para processamento individual, uma RSSF, ao invés, executa uma função colaborativa na qual os elementos envolvidos (nós sensores) provêm dados que são processados por nós especiais, chamados sorvedouros.

Além disso, uma RSSF se distingue de uma MANET pelas características de natureza aleatória da comunicação, bem como a limitação dos nós em termos de processamento e ainda a necessidade do uso restrito de energia, uma vez que, geralmente, os nós sensores são alimentados por baterias (WATTEYNE et al., 2011).

Dentre o potencial de aplicações disponíveis para uma RSSF, é possível citar os seguintes cenários: a) redes urbanas de comunicação para uso em sistemas de energia; b) aplicações militares de monitoramento; c) redes de sensores para agricultura de precisão; d) monitoramento de condições de saúde em pacientes para a área médica; e) aplicações industriais em ambientes onde há a dificuldade de instalações de cabos físicos, etc.

Basicamente, uma rede de sensores sem fio é constituída por vários nós ou sensores eletrônicos distribuídos em uma determinada região, que realizam a comunicação entre si por meio da troca de mensagens em um meio sem fio (wireless) segundo protocolos de comunicação padronizados, como, por exemplo, o IEEE 802.15.4e (CHHABRA; ARORA, 2017).

De maneira geral, as RSSF são formadas por dispositivos autônomos, os quais são capazes de medir grandezas e enviar esses valores para outros dispositivos ou sistemas.

A Figura 1 ilustra essa característica:

Figura 1 – Modelo de uma rede de sensores sem fio

Fonte: (SYEDALIFATHIMA; SUMITHA, 2014)

2.2. A composição do sensor

Cada um dos elementos participantes de uma rede, como demonstrada na Figura 1, pode ser chamado de nó sensor.

Um sensor pode ser entendido como um hardware eletrônico com recursos limitados. De acordo com Kazienko et al. (2013), um sensor típico é composto de 4 componentes, conforme ilustrado na Figura 2:

- a) Uma unidade de alimentação, responsável pelo fornecimento de energia aos outros componentes.
- b) Uma unidade de sensoriamento, constituída por um sensor capaz de medir alguma grandeza física, como, por exemplo: pressão, temperatura.

- c) Uma unidade de processamento, contendo, por exemplo, um microcontrolador.
- d) Uma unidade de comunicação, capaz de enviar e receber sinais de rádio.

Figura 2 – Composição do sensor

Fonte: KAZIENKO et al. (2013)

2.3. O Padrão IEEE 802.15.4e

O padrão IEEE 802.15.4e foi desenvolvido para aplicações de redes de sensores sem fio, especificamente as LR-WPANs (*Low-Rate Wireless Personal Area Networks*) e apresenta uma arquitetura de comunicação baseada em camadas. Um fator relevante para um projeto uma LR-WPAN é a baixa potência de consumo, com o objetivo de maximizar a vida útil da bateria (DE GUGLIELMO et al., 2014).

O padrão IEEE 802.15.4e define a camada física de baixa potência PHY (*Physical Layer*), e também a camada de controle de acesso ao meio MAC (*Medium Access Control*).

Com referência nesse padrão, outros protocolos fazem uso do IEEE 802.15.4e como camada inferior e definem as suas camadas superiores conforme a aplicação e cenários de uso, a exemplo o WirelessHART e ISA100 no ambiente industrial e o ZigBee em outras aplicações.

O padrão IEEE 802.15.4e apresenta dois tipos de dispositivos: os dispositivos com função completa – FFD (*Full Function Device*) - e os dispositivos com função reduzida – RFD (*Reduced Function Device*). O equipamento FFD possui o conjunto completo dos serviços MAC e pode operar tanto como coordenador de rede quanto como dispositivo simples. Os equipamentos RFD, por sua vez, implementam apenas uma parte dos serviços MAC e só conseguem operar como dispositivos simples na rede.

De acordo com Kazienko et al. (2013), a pilha de protocolos para o padrão IEEE 802.15.4e é apresentada na Figura 3, a seguir.

Figura 3 – Pilha de protocolos das redes de sensores sem fio

Fonte: KAZIENKO et al. (2013)

2.3.1. A camada física

A camada física está relacionada com a frequência de transmissão, detecção de sinal, geração da frequência portadora e modulação dos dados. O padrão utilizado é o ISM (*Industrial, Scientific & Medical*). A Tabela 1 apresenta as faixas de frequência utilizadas pela camada física.

Tabela 1 – Bandas de frequência e taxa de dados

PHY (MHz)	Banda de frequência (MHz)	Modulação	Taxa de bit (kb /s)
868	868-868.6	BPSK	20
915	902-928	BPSK	40
2450	2400-2483.5	O-QPSK	250

Fonte: GUO et al. (2012)

A utilização de cada uma das faixas de frequência implica em taxas de transmissão, número de canais e espectros diferentes, conforme mostrado na Figura 4.

Figura 4 – Bandas de frequência

Fonte: GUO et al. (2012)

A camada física tem como funções:

- realizar o controle de ativar/desativar o *tranceiver* do sensor;
- selecionar a frequência do canal a ser utilizado;
- fazer a avaliação sobre a utilização do canal selecionado usando o CCA (*Clear Channel Assessment*) para suportar o CSMA-CA (*Carrier Sense Multiple Access with Collision Avoidance*);
- Levantar métricas como ED (*Energy Detection*) do canal selecionado; LQI (*Link Quality Indicator*) das mensagens recebidas; BER (*Bit Error Rate*) e PER (*Packet Error Rate*).

A taxa de erro de bit ou BER (*Bit Error Rate*) é uma métrica muito importante e representa a razão entre a quantidade total de bits incorretos – ou com erro – recebidos e a quantidade total de bits transferidos durante um determinado intervalo de tempo. O motivo pelo qual ocorreram os erros podem ter origem em ruídos, problemas de atenuação, interferência de outros sinais, entre outros. O valor da taxa de erro de bit é adimensional e pode ser usado como uma estimativa da probabilidade de ocorrer algum tipo de erro no meio.

A equação 1, a seguir, demonstra essa representação:

$$BER = \frac{N_{\text{erro}}}{N_{\text{total}}} \quad (1)$$

Onde:

- BER representa a taxa de erro de bit – ou *Bit Error Rate*
- N_{erro} representa a quantidade de bits recebidos e com erros
- N_{total} representa a quantidade de bits transmitidos

Já a taxa de erro de pacote ou PER (*Packet Error Rate*) é outra métrica importante e representa a relação entre a quantidade de pacotes com erro recebido e a quantidade de pacotes transferidos. Conforme representa Lee, Jeong e Lee (2010), esta métrica pode ser obtida por meio da seguinte equação 2:

$$PER = 1 - (1 - BER)^N \quad (2)$$

Onde:

- PER representa a taxa de erro de pacote – ou *Packet Error Rate*
- BER representa a taxa de erro de bit – ou *Bit Error Rate*
- N representa a quantidade de bits do pacote

O indicador de qualidade LQI (*Link Quality Indicator*) é uma métrica que mede o erro na modulação de entrada de pacotes recebidos com sucesso que usa o ED (*Energy Detection*), por meio de uma estimativa da relação sinal-ruído. Este indicador ED (*Energy Detection*), por sua vez, é um indicador utilizado para selecionar o canal que se encontra em melhores condições para transmissão e recepção. Estes dois indicadores são obtidos, geralmente, a partir do *tranceiver* do sensor.

Além dos indicadores anteriores, o padrão IEEE802.15.4e define ainda o indicador do RSSI (*Received Signal Strength Indication*), com o objetivo de detecção da potência do sinal recebido.

O formato do quadro IEEE 802.15.4e na camada física é apresentado na Figura 5.

A unidade de dados de nível físico PPDU (*Physical Protocol Data Unit*) pode ter até 127 bytes de *payload* – sem contar os cabeçalhos. No frame é possível identificar o cabeçalho de sincronismo (*Sync Header*), o cabeçalho físico (*Phy Header*) e o *payload* físico (*Phy Payload*), que contém os dados da camada de enlace, ou seja, o PSDU (*Physical Service Data Unit*).

Figura 5 – Frame IEEE 802.15.4e

Fonte: KHANAFER et al. (2014)

2.3.2. A camada de enlace (ou camada MAC)

A camada de controle e acesso ao meio MAC controla o acesso ao canal físico do rádio. Devido ao fato de o canal físico ser compartilhado na vizinhança dos nós, a camada MAC provê um mecanismo para determinar quando o canal do rádio está disponível, de tal forma que seja possível enviar as mensagens de forma segura.

O padrão CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*) é usado para determinar quando é seguro transmitir as mensagens. Caso o nó necessite enviar dados, primeiramente, o mesmo deverá escutar o canal por um período determinado de tempo. A ideia é determinar se um outro nó está ou não transmitindo em um determinado canal. Caso o status

do canal seja "disponível", é permitido ao nó iniciar o processo de transmissão. Caso o canal esteja com o status de "ocupado", o nó espera para transmitir por um período aleatório de tempo.

De acordo com o padrão IEEE 802.15.4e, não é possível verificar a transmissão ou não no canal continuamente, uma vez que isso implica em consumo de energia. É utilizado, portanto, um tempo, chamado *backoff timer*.

Na camada de enlace há dois modos de operação: um chamado *beacon-enabled* e outro *non beacon-enabled*. No modo *non beacon-enabled* o nó procura um canal para realizar a transmissão. Por outro lado, no modo *beacon-enabled*, existe uma estrutura chamada superframe que provê um mecanismo de sincronização para a transmissão, de tal forma a se evitar as colisões. Dessa forma, para que um nó inicie a transmissão, ele deverá receber um quadro de *beacon* (KHANAFER et al., 2014).

2.3.1. O formato do frame

Segundo o padrão IEEE 802.15.4e, a transmissão é feita por meio de frames, que podem ser de quatro tipos:

- Frame de *Beacon*;
- Frame de Dados;
- Frame de ACK (*Acknowledgment*);
- Frame de comando MAC.

Todos eles são estruturados com um SHR (*Sync Header*), um PHR (*Phy Header*) e uma PSDU (*Physical Service Data Unit*), sendo este último composto por uma MPDU (*Mac Payload Data Unit*), que, por sua vez, é composta de um MHC (*MAC Header*), um MSDU (*MAC Service Data Unit*) e um MFR (*MAC Footer*). Há também o CRC (*Cyclic Redundancy Check*). A Figura 6 ilustra o frame, conforme Khanafer et al. (2014).

Figura 6 – Frame IEEE 802.15.4e

Fonte: KHANAFER et al. (2014)

Na camada de enlace há dois modos de operação: um chamado *beacon-enabled* e outro *non beacon-enabled*. No modo *non beacon-enabled* o nó procura um canal para realizar a transmissão. Por outro lado, no modo *beacon-enabled*, existe uma estrutura chamada superframe que provê um mecanismo de sincronização para a transmissão, de tal forma a se evitar as colisões. Dessa forma, para que um nó inicie a transmissão, ele deverá receber um quadro de *beacon* (KHANAFER et al., 2014).

O superframe possui um formato definido pelo coordenador PAN. O seu uso permite a funcionamento adequado da comunicação entre os diversos elementos da rede de sensores sem fio, quando eles precisam de qualidade de serviço e não requerem nenhuma prioridade.

O superframe é sempre iniciado com um quadro *beacon*, seguido de uma parte ativa e uma parte inativa (sendo esta opcional), durante a qual o coordenador pode entrar em um modo de economia de energia. Dentro da parte ativa, há uma divisão de slots de mesmo tamanho, onde o quadro de *beacon* é transmitido no primeiro *slot*, conforme a Figura 7.

Neste padrão, um dispositivo irá transmitir em qualquer momento durante um slot, desde que complete sua transação antes do próximo *beacon*. Há uma parte de contenção CAP (*Contention Access Period*), seguido de um CFP (*Content Free Period*), onde existem slots de tempo garantidos – os GTS (*Guaranteed Time Slots*), formando um período livre de contenção, antes do próximo *beacon*.

O período ativo acontece em um intervalo de tempo chamado SD (*Superframe Duration*), e os *beacons* são transmitidos de acordo com um intervalo chamado BI (*Beacon Interval*).

Estes dois intervalos de tempo são definidos pelos parâmetros BO (*macBeaconOrder*) e SO (*macSuperframeOrder*) da camada MAC.

Figura 7 – Superframe do padrão IEEE 802.15.4e

Fonte: AFONSO et al. (2011)

2.4. Dispositivos de rede

O padrão IEEE 802.15.4e define a existência de três elementos com funções distintas: a) o coordenador PAN (*Personal Area Network*), o dispositivo de função completa FFD (*Full Function Device*) e o dispositivo de função reduzida RFD (*Reduced Function Device*).

O coordenador PAN é um elemento único na rede e possui a funcionalidade de organizar a própria rede, sendo o nó raiz, podendo também realizar a comunicação com outras redes.

Os dispositivos de função completa FFD podem funcionar tanto com papel de coordenador quanto de participante da rede.

Já os dispositivos de função reduzida RFD são aqueles que apenas participam como elementos da rede e, geralmente, são equipamentos com pouca quantidade de memória, com potência reduzida e ficam em um modo *stand by* (ou de baixo consumo) na maioria do tempo. Estes elementos podem estar acoplados a sensores, atuadores, entre outros.

2.5. Topologias de rede

As redes de sensores sem fio que seguem o padrão IEEE 802.15.4e podem ser configuradas com topologias do tipo estrela, árvore e malha.

Na topologia do tipo estrela há somente um dispositivo que realiza o gerenciamento da rede. Este é o coordenador PAN (*Personal Area Network*) (IEEE 802.15.4e, 2012).

Na topologia do tipo árvore é utilizado um encaminhamento hierárquico, de tal forma que ocorra um encaminhamento a partir do nó coordenador para outros módulos e, por fim, estes enviam mensagens aos elementos finais.

Em uma topologia do tipo malha (ou *mesh*), todos os elementos colaboram entre si, desde que não sejam do tipo RFD. Este tipo de configuração é bastante interessante por permitir algumas vantagens, como, por exemplo, o encaminhamento de mensagens por caminhos alternativos, caso haja uma falha na rede.

As configurações de topologia são ilustradas na Figura 8.

Figura 8 – Topologias de rede do padrão IEEE 802.15.4e

Fonte: IOVA et al. (2017)

Segundo Rao et al. (2018), a informação em uma rede de sensores sem fio atravessa um caminho composto pela colaboração de diversos nós sensores – chamados *motes* –, geralmente, com baixa capacidade computacional e de energia, no sentido de se levar os dados em direção a outros nós – chamados sorvedouros ou *sink node*. Estes últimos – que são, geralmente, o coordenador PAN – possuem algum tipo de ligação com outras redes ou com a internet, de tal forma a conduzir a informação coletada para esta direção.

2.6. Consumo de Energia

Um importante desafio para o projeto de redes de sensores sem fio é o consumo de energia. Conforme mostra a Figura 2, a fonte de energia de um nó sensor é geralmente limitada às baterias internas. De acordo com Rault et al. (2014), o consumo de energia em nós sensores pode ser classificado em dois tipos:

- consumo dinâmico – por meio da ação de ligar/desligar o componente de hardware;
- consumo estático – sempre ocorre, pois faz parte de alimentação mínima necessária quando o hardware não está ativo.

O consumo de energia no padrão IEEE 802.15.4e está diretamente relacionado à corrente drenada nos circuitos eletrônicos que implementam a camada de comunicação, bem como o tempo que o *transceiver* do rádio permanece ativo.

Segundo Zhou et al. (2017), a quantidade de energia consumida é maior na recepção (modo de escuta), sendo um pouco menor na transmissão e mínima quando o dispositivo não está transmitindo e nem recebendo nenhuma informação. Como o consumo de energia é maior no momento da operação (envio/recebimento de dados), este aspecto é particularmente interessante e pode ser objeto de estudo de alguma técnica de otimização que pode aumentar o tempo de vida do nó sensor.

De acordo com Ahmad e Hanzálek (2018), o consumo eficiente de energia é um requisito essencial a ser observado nas redes de sensores sem fio e várias pesquisas estão empenhadas em encontrar soluções com o objetivo de maximizar o tempo de vida dos nós sensores, maximizando, por consequência, o tempo de vida da rede. Para tanto, as abordagens são feitas de forma a minimizar o consumo de energia, especialmente, mantendo os nós sensores na condição de baixo consumo de energia (consumo estático, com alimentação mínima).

Conforme descreve Rault et al. (2014), ainda não existe uma abordagem única, de maneira que possa ser aplicada de maneira universal. As principais abordagens são baseadas nas seguintes características:

- a) Otimização do rádio: são levantadas técnicas de otimização nos parâmetros de modulação de tal forma a se atingir um consumo mínimo de energia pelo rádio. Esta abordagem é eficaz para comunicação envolvendo pequenas distâncias, mas não consegue resolver problemas de longas distâncias.
- b) Controle da potência de transmissão: são investigadas técnicas para garantir uma melhoria na eficiência energética levando-se em conta um ajuste na potência de transmissão. Essa estratégia é interessante, pois a redução da potência de transmissão, além de reduzir o consumo de energia, também gera a redução do risco de interferência. Por outro lado, podem ocorrer maiores atrasos na transmissão.
- c) Antenas direcionais: essa abordagem permite que os sinais sejam enviados e recebidos em uma única direção, aumentando-se o alcance e a taxa de transmissão de dados. As antenas direcionais podem utilizar uma potência menor do que as antenas omnidirecionais, reduzindo, por consequência, o consumo de energia.
- d) Rádios inteligentes: essa opção se baseia em um mecanismo dinâmico capaz de realizar adaptações na seleção de canais e promover ajustes no mecanismo de comunicação. Apesar de ser uma abordagem complexa do ponto de vista

computacional, ela gera bons resultados do ponto de vista de eficiência energética. Esta área possui oportunidades relacionadas a estudos com a camada de acesso ao meio, roteamento e novos protocolos capazes de utilizar um mecanismo de aprendizagem relacionado aos rádios.

- e) Redução do volume de dados: esta técnica envolve mecanismos de agregação de informação, compactação e simplificação na precisão de leituras de sensores, conduzindo, assim, a um menor consumo de energia pelo uso de um menor volume de dados.
- f) Controle de topologia: nesta opção o objetivo é manter uma topologia confiável do ponto de vista a se evitar novas reconfigurações, tentando preservar as comunicações já estabelecidas.
- g) Roteamento utilizando-se energia como métrica: nesta opção, o objetivo é a construção de algoritmos de roteamento capazes de utilizar informações de consumo de energia para seleção de melhores caminhos, evitando-se, ainda, o esgotamento de nós sensores cuja energia residual seja baixa.
- h) Escalonamento baseado na atividade da rede: neste padrão, o funcionamento do rádio depende da atividade de rede no sentido de minimizar os estados de escuta do rádio sem necessariamente haver o recebimento de mensagens. Esta opção exige um funcionamento assíncrono, capaz aumentar a quantidade de tempo em que o nó sensor fica em estado de consumo mínimo de energia, reduzindo, por sua vez, o consumo energético.

More e Raisinghani (2017) afirmam que as soluções aplicadas ao consumo de energia na área de redes de sensores sem fio ainda possuem vários pontos abertos a pesquisas cujo foco seja melhoria, otimização e utilização eficiente de energia.

O aspecto do consumo eficiente de energia é uma área emergente, que apresenta vários desafios, e tem sido o gargalo do desenvolvimento das tecnologias de redes de sensores sem fio. Os desafios estão relacionados ao endereçamento de soluções cujo desenvolvimento apresente um custo reduzido, seja eficiente e também confiável (SHAIKH; ZEADALLY, 2016).

2.7. Roteamento em Redes de Sensores sem Fio

O roteamento em redes de sensores sem fio é um tema bastante amplo e é nessa área que existem oportunidades para melhorias como, por exemplo: redução de falhas, confiabilidade da comunicação e eficiência energética. De acordo com HASAN, et al. (2017), os mecanismos de roteamento são classificados de acordo com a estrutura de redes: centrado em dados, hierárquicos e geográficos.

Como exemplo de protocolos centrados em dados é possível citar: *Flooding* e *Gossiping*, SPIN (*Sensor Protocols for Information via Negotiation*), DD (*Directed Diffusion*) e Roteamento baseado em Gradiente (GARCIA et al., 2018).

Já os protocolos de roteamento hierárquico podem ser exemplificados pelo LEACH (*Low-Energy Adaptive Clustering Hierarchy*), TEEN (*Threshold sensitive Energy Efficient sensor Network Protocol*) e AP-TEEN (*Adaptive Threshold sensitive Energy Efficient sensor Network protocol*) (GEETHA et al., 2017).

Há também os protocolos geográficos, que utilizam a localização dos nós para o cálculo da distância e o consumo de energia necessário para as transmissões. Dentre esses protocolos, é possível se enumerar: MECN (*Minimum Energy Communication Network*), GAF (*Geographic Adaptive Fidelity*) e GEAR (*Geographic and Energy-Aware Routing*) (MITTAL et al., 2018).

Além dessas classificações, há duas outras que levam em consideração as mudanças possíveis na topologia:

- a) Protocolos de roteamento proativos;
- b) Protocolos de roteamento reativos.

Os protocolos de roteamento proativos também são classificados como *Table-Driven Routing Protocol*, uma vez que utilizam tabelas de roteamento para cada nó sensor. Como exemplo, pode-se citar o DSDV (*Destination-Sequenced Distance-Vector Routing*) e o CGSR (*Clusterhead Gateway Switching Routing*).

Os protocolos de roteamento reativos são utilizados para a descobertas de rotas, por meio de mensagens de inundação (*flooding*). Como exemplo, pode-se citar o AODV (*Ad-Hoc On Demand Distance Vector*), o DSR (*Dynamic Source Roting*) e o RPL (*Routing Protocol for Low-Power and Lossy Network*).

2.8. O Protocolo RPL

O IETF (*Internet Engineering Task Force*) possui um grupo de trabalho – chamado ROLL (*Routing Over Low power and Lossy networks*) - envolvido com a especificação de um protocolo baseado em Ipv6 (*Internet Protocol version 6*). Este grupo tem avaliado extensivamente várias opções de roteamento e definiu um protocolo de roteamento chamado RPL (*Routing Protocol for Low Power and Lossy Networks*).

De acordo com a RFC 6550 (2015), o RPL é um protocolo com um mecanismo preferencialmente reativo, tem o objetivo de minimizar a sobrecarga para manter o roteamento e também manter o foco em links que realmente estejam sendo usados.

Segundo Gaddour e Koubâa (2012), é possível rodar mais de uma instância do RPL simultaneamente em um conjunto de nós sensores. Cada instância tem sua própria métrica e a construção de topologia se inicia a partir de um nó raiz, criando-se uma estrutura de árvore, dando origem a um DODAG (*Destination-Oriented Directed Acyclic Graph*).

O RPL é um protocolo do tipo “*distance vector*” e suporta os seguintes padrões de tráfego: a) Multipoint-to-point; b) Point-to-multipoint; c) Point-to-point.

A formação da topologia é baseada na ideia de Grafos Acíclicos Dirigidos – DAG (*Directed Acyclic Graph*). O protocolo organiza os nós da topologia como um conjunto de DODAGs (*Destination Oriented DAGs*).

Nesse contexto, os nós *sink* e *gateway* são os nós raízes dos DAGs.

O RPL possui suporte a diferentes tipos de aplicação por meio de uma Função Objetivo – OF (*Objective Function*).

No funcionamento do RPL, uma OF especifica a maneira como o caminho de comunicação é construído, sendo cada nó representado por um *rank* dentro de um DODAG. O nó raiz possui *rank* 0, enquanto cada um dos próximos nós possui um incremento dependente do número de saltos a partir do nó raiz.

De acordo com Thubert (2011), uma OF especifica como o RPL seleciona os nós pais, ou seja, qual o caminho do DODAG. As OFs definem, então, a maneira pela qual a métrica de roteamento será utilizada. Em resumo, o *rank* representa a distância relativa ao nó raiz do DODAG.

O IETF ROLL definiu, então, dois tipos de função objetivo para assegurar o cálculo do *rank*:

- a) OF0 (*Objective Function Zero*) – Esta função é baseada no número de saltos. O *rank* de um nó específico é calculado tomando-se como base a sua distância relativa ao nó raiz.
- b) MRHOF (*Minimum Rank With Hysteresis Objective Function*) – Já esta função leva em consideração um valor chamado ETX (*Expected Transmission Count*), que utiliza como métrica a quantidade de comunicação realizada com sucesso naquele caminho.

Essas funções objetivo tem como premissa tentar evitar ou reduzir o número de retransmissões de pacotes. A função OF0 tenta minimizar o número de saltos, enquanto MRHOF avalia a utilização de links com melhor qualidade de comunicação.

O mecanismo utilizado para a construção e manutenção do DODAG é feito por meio de mensagens ICMPv6 (*Internet Control Message Protocol version 6*), denominadas DIO (*DODAG Information Object*) para a descoberta de vizinhos e estabelecimento de rotas.

Durante a construção da topologia, cada nó raiz cria um pacote DIO e o envia para todos os seus nós filhos. A mensagem DIO possui um valor do *rank* do nó, que é incrementado a cada vez que um nó filho se junta ao DAG.

A Figura 9 apresenta em 4 passos o processo de formação do DODAG:

Passo 1: Na Figura 9a, inicialmente, o nó raiz (*sink node* ou roteador de borda) realiza uma difusão de mensagens DIO (*DODAG Information Object*). Este nó possui *rank* 0. Todos os outros nós ainda não foram identificados e são classificados com um *rank* muito alto (identificado pelo valor ∞ na figura, uma vez que não são conhecidos ainda).

Passo 2: As mensagens DIO que foram enviadas pelo roteador de borda chegam aos nós próximos do roteador, cujos *rank*s são incrementados em relação ao nó raiz. Estes nós, por sua vez possuem *rank* 1 e enviam para seus vizinhos a mesma difusão de mensagens DIO. Este processo é demonstrado na Figura 9b.

Passo 3: As mensagens DIO atingem outros nós vizinhos, cujos *ranks* também são incrementados em relação à mensagem que receberam. O processo continua novamente, e estes nós com *rank* 2 realizam uma nova difusão de mensagens DIO até que aconteça a formação da topologia, conforme a Figura 9c.

Passo 4: Uma vez que todos os nós foram atingidos, acontece a formação completa do DODAG (*Destination Oriented Directed Acyclic Graph*), descrito na Figura 9d.

Para efeito de diminuição do efeito *flooding*, é utilizado um algoritmo adaptativo, chamado *Trickle Algorithm* (LEVIS et al., 2011), o qual realiza o escalonamento de mensagens DIO. Este algoritmo é baseado em um intervalo de tempo entre dois valores I_{MIN} e I_{MAX} , através dos quais, à medida em que a rede é estabilizada, o número de envio de mensagens DIO é reduzido.

Figura 9 – O protocolo RPL

Fonte: IOVA et al. (2013)

É interessante observar que, no processo de formação do DODAG, caso algum nó receba uma mensagem DIO cujo *rank* é maior que do próprio *rank*, esta mensagem é descartada.

Um nó ao receber uma mensagem DIO, além de enviá-la para seus vizinhos, responde ao nó do qual recebeu a mensagem DIO com uma mensagem DAO (*Destination Advertisement Object*). Este processo é ilustrado na Figura 10. Ao responder à mensagem DIO com um DAO, o nó é adicionado ao DODAG.

Figura 10 – Processo de formação do DODAG

Fonte: Cobarzan et al. (2014)

Após o processo de inicialização, um nó – que não seja o próprio nó raiz – entra em um modo de espera por mensagens DIO. Caso não receba uma mensagem DIO após um limite determinado de tempo, ele envia uma mensagem do tipo DIS (*DODAG Information Solicitation*). Esta mensagem representa um pedido de envio de DIO, que por sua vez, ao ser recebida por um nó vizinho, faz com que este vizinho responda por meio do envio de uma mensagem DIO, conforme a Figura 10.

O processo de estabelecimento de rotas no sentido inverso, em direção de volta ao nó raiz, é completado, então, por meio das mensagens DAO, chegando ao roteador de borda (ou *DAG root*).

O protocolo RPL suporta dois mecanismos para gerenciar as suas rotas de volta ao nó raiz:

- a) o *storing mode*;
- b) *non-storing mode*.

No padrão *storing mode* são enviadas regularmente mensagens de tal maneira a se compor tabelas de rotas nos nós que compõe o caminho. A cada vez que novas rotas são descobertas, estas tabelas precisam ser reconstruídas, causando uma sobrecarga tanto do ponto de vista de envio de mensagens quanto da necessidade de armazenamento de tabelas de rotas – o que se torna um grande desafio, uma vez que os nós possuem capacidade limitada de memória. Este mecanismo do tipo *storing mode* não se torna viável quando se tem uma rede composta por um grande número de saltos.

No padrão *non storing mode*, cada nó contém informações apenas de seus vizinhos locais e o próprio processo do RPL não têm o ônus do armazenamento local de tabelas de rotas.

Nesta tese, todas as abordagens realizadas contemplam apenas o padrão *non-storing mode*, uma vez que o armazenamento de tabelas de rotas torna complexo o processo de transferência da mesmas, causando envio adicional de mensagens, necessidade de armazenamento em um dispositivo já limitado em termos de memória, e, conseqüentemente, um maior consumo de energia – sendo este último, objeto de estudo no qual estamos propondo a realização de uma melhoria em termos de eficiência energética.

2.9. Plataformas de Simulação para Redes de Sensores sem Fio

Com o desenvolvimento e melhoria contínua dos trabalhos da IETF, a validação de novos protocolos em redes de sensores sem fio tem se beneficiado por meio de uso de simuladores computacionais, que proporcionam um ambiente adequado para realização de testes e verificação de limites, os quais podem influenciar de maneira significativa a elaboração de uma nova arquitetura de comunicação (INTANA et al., 2013).

Dentre as plataformas disponíveis para simulação de redes de sensores sem fio pode-se citar: NS-2 e NS-3 (*Network Simulator* nas versões 2 e 3), *OMNet++*, *Cooja Network Simulator* (distribuído na plataforma *Contiki*), *OpenWSN*, entre outros.

As próximas seções irão apresentar alguns detalhes das plataformas *OpenWSN* e *Contiki*.

2.9.1. O OpenWSN

O *OpenWSN* é um projeto aberto, iniciado na Universidade da Califórnia (Berkeley) em 2010 e oferece uma implementação gratuita e de código aberto, com suas bases no padrão IEE802.15.4e (WATTEYNE et al., 2012), apresentando uma implementação da pilha desse protocolo e permitindo a simulação de uma rede de sensores sem fio completa a partir de um computador convencional (executando um sistema operacional Linux ou Windows). O código fonte utiliza "C puro", sem a utilização de extensões da linguagem C, com o objetivo de poder ser utilizado em múltiplas plataformas, independente do sistema operacional. Inclusive, é possível realizar o *deployment* em hardwares físicos.

Essencialmente, ele é baseado em uma arquitetura similar à de um sistema operacional: o *OpenOS*, o qual funciona com um *kernel scheduler* e trata uma lista de tarefas. Essa plataforma permite a simulação de um ambiente real por meio do *OpenSim*, que é uma abstração de hardware construída em software e provida pela linguagem Python. Há ainda, uma interface de visualização e interação com o usuário: o *OpenVisualizer*, onde é possível a interação e troca de dados com outros nós conectados da rede de sensores sem fio.

A aplicação da arquitetura do *OpenWSN* é ilustrada na Figura 11.

Figura 11 – Aplicação da Arquitetura OpenWSN

Fonte: WATTEYNE et al. (2012)

A proposta do *OpenWSN* segue o padrão IEEE802.15.4e, implementando a pilha de comunicação de acordo com a Figura 12.

Figura 12 – Pilha da arquitetura do *OpenWSN*

Fonte: WATTEYNE et al. (2012)

As simulações podem ser realizadas por meio de parâmetros adequados, construídos a partir de linhas de comando e os valores dos resultados são observados em uma interface web, o *OpenVisualizer*, sendo apresentadas dentro de um navegador de internet. Essa interface permite a interação com o usuário e é ilustrada na Figura 13.

Figura 13 – *OpenVisualizer* Web - *OpenWSN*

Fonte: WATTEYNE et al. (2012)

2.9.2. O Contiki

O *Contiki* é um sistema operacional de código aberto com suporte à rede, concebido para sistemas embarcados com limitações de hardware, baixo poder de processamento, pouca memória e pouca banda de comunicação (MCBRIDE et al., 2018).

O funcionamento do *Contiki* é baseado em um sistema de eventos, permitindo diferentes blocos de código executarem dependendo do evento que chega. Esses blocos são responsáveis por executarem alguma tarefa. Uma vez iniciado, o código não será interrompido e os blocos são concebidos para permitir o compartilhamento da mesma pilha.

Um processo do *Contiki* consiste, portanto, em uma função para tratamento de eventos e uma função opcional para operações de varredura. A Figura 14 ilustra a arquitetura do *Contiki*.

Figura 14 – Arquitetura do *Contiki*

Fonte: FAROOQ; KUNZ (2011)

A maioria das funcionalidades adicionais pode ser implementada por meio do mecanismo das *ProtoThreads*, as quais funcionam como um protótipo no qual um trecho de código irá executar no contexto de uma *Thread*, escrita em “C puro”. As *ProtoThreads* possuem a vantagem de serem extremamente leves e foram projetadas especificamente para sistemas embarcados com sérias restrições em termos de CPU e memória.

O *Contiki* possui ainda um ambiente de simulação de sensores virtuais - o *COOJA Simulator*, o qual é construído em Java e permite diversas configurações por meio de parâmetros, os quais irão contemplar a simulação de uma rede de sensores sem fio.

A interface gráfica do *COOJA* permite, ainda, a seguinte configuração do mecanismo de propagação do rádio. Dentre as opções disponíveis, é possível citar:

- UDGM (*Unit Disk Graph Medium*) – *Distance Loss* – para atenuação do sinal baseado em raios de alcance;
- UDGM (*Unit Disk Graph Medium*) – *Constant Loss* – para atenuação constante à medida em que se aumenta a distância do nó sensor;
- MRM (*Multi-path Raytracer Medium*) para propagação no espaço livre.

Figura 15 – COOJA Network Simulator

Fonte: FAROOQ; KUNZ (2011)

2.10. Discussão

Neste capítulo foi apresentada uma visão geral dos aspectos relativos ao funcionamento de uma rede de sensores sem fio, com especial destaque aos detalhes envolvidos no processo de construção de topologia, feito pelo RPL.

O RPL é o padrão de roteamento utilizado por diversas plataformas, inclusive o *Contiki* e o *OpenWSN*.

Um fator relevante do RPL é que ele está associado a um *Trickle Timer*, o qual é responsável pelo controle do envio de mensagens DIO na rede.

É interessante observar que o comportamento do *Trickle Timer* pode refletir situações de estabilidade ou instabilidade na formação da topologia.

O próximo capítulo irá abordar conceitos relativos a heurísticas, sistemas inteligentes e inteligência *swarm*.

CAPÍTULO 3

3. SISTEMAS INTELIGENTES BASEADOS EM COLÔNIA DE FORMIGAS

Este capítulo apresenta uma abordagem de sistemas inteligentes a partir do conceito da meta-heurísticas de colônia de formigas. As seções, a seguir, apresentam: o conceito de *swarm intelligence*, sistemas bioinspirados, a ideia principal de um algoritmo bioinspirado no funcionamento de uma colônia de formigas e, por fim, os três principais algoritmos artificiais baseados em colônia de formigas.

3.1. Swarm Intelligence

O conceito de *swarm intelligence* – ou inteligência de enxames – tem sido objeto de estudo de diversos pesquisadores (Ertenlice, et al. 2018). Este conceito está ligado ao comportamento coletivo de sistemas complexos, com capacidade de auto-organização, de maneira descentralizada e com características de um comportamento social.

Estes sistemas são construídos a partir de uma estrutura baseada em agentes, os quais podem interagir entre si por meio de uma organização de sociedade (ou enxames).

O comportamento resultante da interação entre estes agentes, por sua vez, apresenta evidências de inteligência, capacidade de aprendizagem e adaptação.

Uma das primeiras contribuições interessantes nesta área foi o algoritmo chamado PSO (*Particle Swarm Optimization*), proposto em 1995, por meio da colaboração entre um engenheiro eletricitista – Russel Eberhart – e um psicólogo – James Kennedy.

A ideia apresentada por estes pesquisadores foi no sentido de construção de uma otimização baseada no comportamento social. Essa construção usa um movimento estocástico dentro de um espaço de busca e este movimento é inspirado em uma característica da natureza: o comportamento de um bando de pássaros.

Dentre as possibilidades de algoritmos inspirados na natureza, é possível citar os enxames como: colônia de formigas, cardume de peixes, colônia de abelhas, morcegos, pássaros, entre outros.

A solução de diversos problemas de engenharia se depara, por vezes, com situações que são consideradas bastante complexas, sejam elas do ponto de vista de uma solução matemática analítica ou mesmo do ponto de vista computacional.

De acordo com Dickinson e Gijben (2014), tais problemas fazem parte de uma classificação daqueles que não conseguem ser resolvidos em um tempo polinomial, sendo que sua solução exige uma abordagem exponencial, e tem o nome de NP-difíceis.

Segundo Abdulrahman (2017), uma alternativa para a solução de problemas do tipo NP-difíceis tem sido usada com bastante sucesso por meio de uma abordagem baseada em *swarm intelligence*. O resultado obtido não provém unicamente de um agente, mas da colaboração e interação entre os diversos agentes participantes do sistema na busca da solução.

Este conjunto de soluções bioinspiradas são classificados como meta-heurísticas. O termo meta-heurística é resultado da combinação das palavras gregas “meta + heurística”, onde meta está relacionado a algo que se encontra em um “nível superior”, enquanto o termo heurística é proveniente da palavra “*heurisken*”, cujo significado está relacionado à ação de “descobrir” ou “encontrar”.

3.2. Sistemas bioinspirados

As técnicas de computação bioinspirada e sistemas inteligentes tem sido alvo de muitas pesquisas nos últimos anos. De acordo com Kar (2016), as áreas de pesquisas bioinspiradas podem ser classificadas como: computação bioinspirada, sistemas bioinspirados e redes bioinspiradas.

A computação bioinspirada representa uma classe de algoritmos cujo foco é a eficiência computacional, como, por exemplo, a otimização de processos e o reconhecimento de padrões.

Já os sistemas bioinspirados são uma classe de arquiteturas aplicadas a sistemas distribuídos e colaborativos, com aplicações no sensoriamento e na exploração.

Finalmente, as redes bioinspiradas representam uma classe de estratégias para atingir a eficiência e o escalonamento de redes sob determinadas condições, como por exemplo auto-organização e sistemas distribuídos em larga escala.

A Tabela 2 apresenta alguns tipos de princípios bioinspirados associados à sua aplicação em redes.

Tabela 2 – Fenômenos biológicos e aplicações

Princípio biológico	Campos de aplicação em redes
<i>Swarm intelligence</i> e colônia de insetos	Busca distribuída e otimização; roteamento em redes de computadores, especialmente MANETs, RSSF; tarefas e alocação de recursos
Sincronização de vaga-lumes	Clock robusto e completamente distribuído
Sistemas imunológicos artificiais	Segurança de redes; anomalias e detecção de intrusão

Fonte: MAVROVOUNIOTIS et al. (2017)

3.3. Colônia de Formigas

O processo busca de alimentos realizado pelas formigas acontece, a princípio, de maneira aleatória. De acordo com Toksari (2016), a partir do momento em que o alimento é encontrado, no caminho de retorno para o ninho, as formigas depositam uma substância, chamada feromônio. Esta substância é responsável por atrair novas formigas que se organizam de tal forma a seguir também pelo caminho deste feromônio depositado. Em se encontrando mais alimentos, estas novas formigas realizam o depósito de mais feromônio por este caminho, reforçando ainda mais a trilha inicial de feromônio criada, atraindo assim, outras formigas para que sigam também por este caminho.

Este processo, gera uma trilha que é seguida por praticamente todas as outras novas formigas que irão fazer a busca do alimento.

Dado que a busca inicial acontece de forma aleatória, é possível que algumas formigas também sigam por outros caminhos, que por sua vez, também levam ao alimento. Estes outros caminhos também recebem o reforço do feromônio, de tal forma que haja mais de um caminho válido. A Figura 16 apresenta esta situação:

Figura 16 – Comportamento das formigas

Fonte: TOKSARI (2016)

É interessante notar, que no caso de haver mais de um caminho válido, os menores caminhos serão reforçados e preferidos, pois um maior caminho leva mais tempo para ser percorrido e neste caso, ele possuirá uma menor quantidade de feromônio, pois além do depósito, acontece o processo de evaporação do mesmo.

Isto significa que caminhos menores serão reforçados e preferidos ao invés de caminhos maiores.

Este padrão de comportamento das formigas foi observado por Marco Dorigo e foi motivador da sua proposta do algoritmo chamado ACO (*Ant Colony Optimization*). Uma das primeiras aplicações para o algoritmo do ACO foi a solução do problema do caixeiro viajante (TSP – *Traveling Salesman Problem*) (LÓPEZ-IBÁÑEZ et al., 2018).

O problema TSP consiste em solucionar, de maneira otimizada, qual a sequência de cidades a serem percorridas por um caixeiro viajante de tal forma que cada cidade seja visitada somente uma vez e se percorra o trajeto, evitando-se rotas redundantes. Ou seja, o problema consiste em se percorrer da melhor forma um conjunto de cidades de tal forma que a distância percorrida seja mínima e não haja passagem em uma cidade por mais de uma vez.

A solução do problema TSP por meio de um algoritmo cuja base é o comportamento de uma colônia de formigas (ACO), consiste em se criar um número de agentes – neste caso, formigas – que se deslocam entre as cidades seguindo as estradas disponíveis entre elas. A cada trajeto entre 2 cidades, é inferido um valor para o feromônio, de tal forma que os agentes possam utilizar este valor para realizar as melhores escolhas.

Assim, a execução do ACO acontece de maneira iterativa e cada iteração utiliza um número N de agentes.

A princípio, os agentes são distribuídos aleatoriamente para que percorram todas as cidades, sendo que todas devem ser visitadas. A esse processo, chamamos de ciclo, e a cada ciclo, os agentes são distribuídos de forma aleatória entre as cidades.

A partir de então, cada agente k constrói uma lista L de cidades a serem visitadas.

Os critérios utilizados para a seleção de uma outra cidade, segundo López-Ibáñez et al. (2018), levam em conta:

- a) se a cidade já foi visitada ou não;
- b) a distância entre as cidades;
- c) a quantidade presente de feromônio no caminho para escolha da cidade.

O algoritmo básico é descrito a seguir:

Figura 17 – Algoritmo ACO

Algoritmo ACO

Iniciar configurações

Inicializar quantidade inicial de feromônios

Início

Distribuir agentes aleatoriamente

Realizar busca para construir soluções

Atualizar feromônios de maneira global (opcional)

Atualizar feromônios de maneira local

Comparar resultados

Teste (atingiu esperado)?

Sim: Eleger melhor resultado

Não: voltar ao Início

Fonte: TOKSARI (2016)

A primeira versão do algoritmo baseado no ACO é conhecida como Sistema de Formigas (AS - *Ant System*) e foi proposto no início da década de 90 (TOKSARI, 2016).

Há também outras variações, baseadas em otimizações do próprio ACO. São elas: o *MAX-MIN Ant System* e o *ACS (Ant Colony System)*.

3.4. Algoritmo Ant System

De acordo com este algoritmo, inicialmente, as formigas são distribuídas aleatoriamente e uma quantidade inicial de feromônio é atribuída a cada caminho. As formigas iniciam sua busca pela sequência de cidades avaliando-se 2 fatores: a) a quantidade depositada de feromônio e b) a distância até chegar a próxima cidade.

O agente – neste caso, a formiga – ao passar por uma cidade, faz o registro daquela passagem em uma memória própria, de tal forma que não visite aquela cidade novamente durante o processo de busca.

Ao final do percurso, onde todas as cidades já tenham sido visitadas, é atribuído uma quantidade de feromônio para cada trecho visitado.

Uma característica específica deste algoritmo é que não ocorre nenhum depósito de feromônio durante a fase de construção dos caminhos.

A realização de cada etapa acontece por meio de uma função onde é verificada a probabilidade de escolha de uma cidade dentre a lista das cidades que ainda não foram visitadas.

O valor da probabilidade é definido pela equação 3:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N_k^i} [\tau_{il}]^\alpha [\eta_{il}]^\beta} \quad (3)$$

Onde:

- p_{ij}^k é a probabilidade de que o agente k , a partir da cidade i , escolha a cidade j
- α é um parâmetro associado ao peso do feromônio
- β é um parâmetro associado à característica heurística
- τ_{ij} é a quantidade de feromônio na aresta ij (ou caminho que liga a cidade i a j)
- η_{ij} é o inverso de d_{ij} , sendo que d_{ij} representa a distância entre as cidades i e j
- N_k^i o conjunto de cidades que ainda não foram percorridas pela pelo agente k

Neste algoritmo específico não acontece nenhuma atualização global de feromônios – mostrada como opcional, no algoritmo da Figura 17.

A atualização dos feromônios acontece dentro da lista de cidades armazenadas por cada um dos agentes – formigas.

Esta atualização leva em conta tanto a evaporação do feromônio como também o seu incremento.

O processo é feito de acordo com a equação 4:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \sum_{k=1}^m \Delta \tau_{ij}^k \quad (4)$$

Onde:

- τ_{ij} é a quantidade de feromônio presente no caminho entre as cidades i e j
- ρ é um valor associado à evaporação do feromônio
- m é a quantidade de agentes – formigas
- $\Delta \tau_{ij}^k$ é definido como sendo:

$$\begin{cases} \frac{1}{L^k} & , \text{ se a aresta } ij \text{ faz parte do caminho da formiga } k \\ 0 & , \text{ caso contrário.} \end{cases}$$

- L^k é o custo do caminho ou trilha seguida pela formiga k .

É interessante notar que alguns itens da equação 4, como, por exemplo, ρ , associado com a evaporação deve possuir um valor limitado (no caso, menor que 1), sob pena de ocorrer um acúmulo de feromônio ilimitado.

De acordo com Dorigo et al. (2016), os valores padrões sugeridos são: $\rho = 0,5$ e $\alpha > 0$. Valores altos para α trazem com consequência um grande peso para a importância do fator feromônio e valores muito baixos fazem o algoritmo funcionar de forma aleatória.

Dorigo et al. (2018), apresenta sugestões de $\alpha=1$ e $\beta=5$. Além disso, ele propõe o uso de um número m de agentes equivalente ao número n de cidades envolvidas.

3.5. Algoritmo MAX-MIN Ant System

O algoritmo *MAX-MIN Ant System* foi proposto por Stützle e Hoss em 1998 como uma melhoria do algoritmo *Ant System*, no sentido de se evitar uma convergência muito rápida (LIAW; CHANG, 2017).

O algoritmo é praticamente o mesmo do AS. A atualização de feromônio também ocorre no final da criação do caminho.

As duas principais diferenças ocorrem:

- a) No valor da atualização do feromônio, que – neste algoritmo – está limitado a valores mínimo e máximo;
- b) Na maneira com o feromônio é atualizado – a atualização acontece somente para o melhor agente (ou melhor formiga).

A equação de atualização do feromônio adotada é:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \tau_{ij}^{melhor} \quad (5)$$

Onde:

- τ_{ij} é a quantidade de feromônio presente no caminho entre as cidades i e j
- ρ é um valor associado à evaporação do feromônio
- τ_{ij}^{melhor} é o incremento devido ao melhor agente – ou melhor formiga.

De acordo com Liaw e Chang (2017), o estabelecimento de limites mínimo e máximo evitam a convergência precoce do algoritmo.

3.6. Algoritmo Ant Colony System

Assim como o algoritmo anterior, o ACS (*Ant Colony System*) também foi proposto como uma melhoria do AS (*Ant System*) (SKINDEROWICZ, 2017).

O ACS foi proposto por Dorigo e Gambardella, e o seu funcionamento prevê o uso de uma nova expressão para decisão do agente. Este parâmetro é o q_0 . Durante a iteração é feito um sorteio de número aleatório, comparando-o ao valor de q_0 .

Se o número aleatório sorteado for menor que q_0 , o valor de probabilidade utilizado é o descrito na equação 6:

$$p_{ij}^k = \tau_{ij} \cdot \eta_{ij}^\beta \quad (6)$$

Onde:

- p_{ij}^k é a probabilidade do agente k , estando na cidade i , escolher a cidade j .
- τ_{ij} é a quantidade de feromônio presente no caminho entre as cidades i e j
- η_{ij} é o inverso de d_{ij} , sendo que d_{ij} representa a distância entre as cidades i e j
- β é um parâmetro associado à característica heurística

Se o número aleatório sorteado for maior que q_0 , utiliza-se a equação tradicional do AS.

É interessante ressaltar que valores de q_0 próximos de 1 são refletidos em uma intensificação e valores de q_0 próximos de 0 são refletidos em uma diversificação.

As atualizações de feromônio, ao invés de acontecer como nos outros dois algoritmos – somente ao final do percurso – ocorrem, neste algoritmo, a cada iteração, especificamente no caminho em que acabou de ser escolhido.

A equação 7, é utilizada para atualização de feromônio:

$$\tau_{ij} = (1 - \varphi) \cdot \tau_{ij} + \varphi \cdot \tau_0 \quad (7)$$

Onde:

- τ_{ij} é a quantidade de feromônio presente no caminho entre as cidades i e j
- φ é um parâmetro definido entre 0 e 1
- τ_0 é o valor inicial dos feromônios das arestas.

Esta atualização de feromônio descrita acontece de forma local. Neste algoritmo também ocorre uma atualização global de feromônios.

Da mesma forma que no algoritmo anterior, a atualização global é feita somente para o melhor agente (ou melhor formiga).

A equação (8) apresenta o valor desta atualização:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \tau_{ij}^{melhor} \quad (8)$$

De acordo com Skinderowicz (2017), as sugestões apresentadas por Dorigo e Gambardella são de valores de $\alpha = 0,1$, $\rho = 0,1$ e $\beta = 2$.

3.7. Discussão

Neste capítulo foram apresentados alguns conceitos relativos a *swarm intelligence* e os três principais algoritmos ligados ao processo de solução de problemas por meio da heurística de colônia de formigas.

Estes algoritmos são interessantes para serem aplicados na solução de problemas, especialmente a maximização ou minimização de uma função.

Nesta tese, onde é proposto um protocolo de roteamento com base no funcionamento do RPL, utilizou-se o conceito do *MAX-MIN Ant System* para minimização da função objetivo do RPL no processo de avaliação do *rank* durante a construção da topologia.

O próximo capítulo irá abordar as pesquisas na área de roteamento.

CAPÍTULO 4

4. ROTEAMENTO EM REDES DE SENSORES SEM FIO

Este capítulo apresenta uma discussão relativa ao roteamento de redes de sensores sem fio, contextualizando as pesquisas na área.

Inicialmente, é feita uma análise geral seguida de uma relação de características de utilização. Os desafios inerentes ao roteamento são apresentados na sequência e, finalmente, a última seção apresenta as principais pesquisas desenvolvidas nesta área.

4.1. Análise Geral

Uma rede de sensores sem fio (RSSF) consiste de vários nós, geralmente distribuídos em uma região com o objetivo do monitoramento de alguma grandeza física.

No cenário de redes urbanas, é possível realizar a coleta de dados como, por exemplo, informações de energia das redes elétricas, tráfego de veículos, qualidade do ar, enfim, vários fatores capazes de proporcionar melhorias na qualidade de vida e conforto do ser humano. Tipicamente, um sensor é um dispositivo reduzido em tamanho e também em recursos, capaz de realizar a coleta de informações por meio do sensoriamento, processar informações e enviá-las a outros sistemas por meio de transmissão de dados em um meio de comunicação sem fio. Todos os elementos participantes da arquitetura do sensor são alimentados geralmente por um módulo de baterias, com uma capacidade de energia que deveria ser utilizada de bastante otimizada.

Uma vez que os sensores são um elemento essencial para a coleta das informações ligadas à sua aplicação, é de fundamental importância que o seu funcionamento ocorra por um longo período de tempo, sem a necessidade de constantes manutenções que exijam a troca de baterias, de tal forma a se prolongar a sua vida útil.

Nesse sentido, uma oportunidade surge no que concerne ao desenvolvimento de pesquisas relacionadas à otimização do uso dos sensores, com relação à eficiência energética.

É nesse contexto que essa tese está inserida, com o objetivo do desenvolvimento de um mecanismo capaz de atuar justamente nessa questão: a eficiência energética.

Estudos experimentais conduzidos por Sandra et al. (2011) demonstram que a transmissão de dados (onde se utiliza o transceptor de rádio) é mais dispendiosa em termos de consumo de energia do que o processamento de dados (onde se utiliza o microcontrolador) do sensor.

De acordo com Abbas e Yoon (2015), em termos quantitativos, o custo de energia utilizado para a transmissão de um único bit de informação é aproximadamente a mesma que a necessária para o processamento de milhares de informações em um sensor típico.

Apesar do consumo de energia ser intrinsicamente dependente do tipo de sensor e também do transceptor, de maneira geral, a energia gasta para o processamento de informações é bem menor quando comparada à energia gasta para a transmissão de dados.

Por esse motivo, é possível se inferir que técnicas baseadas na redução quantidade de informação transmitida poderiam reduzir consideravelmente o consumo de energia, ainda que fosse necessário algum tipo de processamento interno com o objetivo de se reduzir a transmissão pelo transceptor de rádio.

4.2. Avaliação das características de utilização

A preocupação com a eficiência energética também é foco dos trabalhos desenvolvidos pelo IETF dentro do grupo de trabalho ROLL, que realizou a proposta do roteamento por meio do protocolo RPL, que por sua vez, age de forma reativa, levando-se em conta algumas características da rede por meio de uma Função Objetivo (OF).

A abordagem feita por essa tese propõe como foco de estudo uma melhoria no mecanismo de roteamento, já endereçado pelo RPL e funcional em várias implementações disponíveis tanto em simuladores quanto em ambientes que utilizam um hardware real para as redes de sensores sem fio.

4.3. Desafios inerentes ao roteamento

Quando se considera a proposição de algum tipo de modificação em um mecanismo de roteamento, algumas questões emergem, uma vez que as redes de sensores sem fio (RSSF) devem atender aos requisitos de operação em vários cenários distintos, sejam eles: redes urbanas, ambientes industriais, agricultura de precisão, área médica, entre outros.

De acordo com Ovsthus et al. (2014), os desafios estão relacionados com as seguintes características:

- *Tempo-real*: um roteamento adequado é capaz de realizar a comunicação com baixa latência entre dois pontos, particularmente adequado a atender características de funcionamento em tempo-real.

- *Confiabilidade*: uma rota é confiável se provê um caminho pelo qual a informação, necessariamente, conseguirá ser transmitida. Este fator leva em consideração a transmissão e recepção de pacotes com nível baixo de perdas ou até nenhuma perda, uma vez que perdas na transmissão tornariam necessária a adoção de um processo de retransmissão.

- *Eficiência energética*: uma rota é eficiente quando utiliza nós que possuem mais energia que outros nós. Como consequência, uma rota eficiente deveria levar em conta a quantidade de bateria disponível em cada nó, de tal forma a se evitar a utilização intensa de nós que possuem baixos níveis de bateria disponíveis.

Associado a estes desafios, novas propostas de roteamento, precisariam também levar em conta o uso dessas redes em largas escalas, como por exemplo, os cenários de redes urbanas.

Dessa forma, faz necessária a validação de projetos de RSSF com um grande número de nós. A experimentação poderia ser então conduzida de duas maneiras:

- a) utilizando-se dispositivos e implementações físicas;
- b) utilizando-se simulações.

De acordo com Aznoli e Navimipour (2017), a implementação física gera a necessidade de demanda de tempo para a reprogramação de cada um dos nós, além de custos financeiros com os próprios equipamentos.

Dessa forma, por meio da redução dos custos citados anteriormente, as simulações oferecem vantagens ao pesquisador no sentido de que tornam possível a validação de redes de sensores sem fio antes de sua implementação, permitindo, inclusive, a sua operação em ambientes de larga escala.

Conforme os trabalhos de Modieginyane et al. (2018), atualmente, há uma grande contribuição por meio de pesquisas conduzidas com base em resultados de simulações de redes de sensores sem fio.

Levando-se em conta as considerações anteriores, essa tese propõe a construção de um algoritmo capaz de atuar no mecanismo de roteamento das redes de sensores sem fio, de tal forma a se obter uma eficiência energética, com relação ao uso dos tradicionais mecanismos de roteamento.

4.4. Estado da arte das pesquisas na área de roteamento

Devido ao fato do roteamento para redes de sensores sem fio apresentar vários aspectos desafiadores, tanto do ponto de vista de estabilidade – considerando-se por estabilidade a ausência de alterações constantes de topologia –, quanto de economia de energia, várias pesquisas têm sido feitas no sentido de se endereçar estes dois elementos.

A utilização de uma meta-heurística baseada em colônia de formigas, já tem sido explorada com sucesso pelo trabalho de alguns pesquisadores.

No trabalho de Ökdem e Dervi (2009), os autores fazem a proposta de um mecanismo baseado em posição. Neste modelo, o nó raiz é definido como estático e sua posição é conhecida pelos outros nós. Os demais nós têm, ainda, o conhecimento de sua própria posição, por meio de um mecanismo que usa dados de GPS e um algoritmo de posicionamento é aplicado juntamente com tabelas de posições dos nós vizinhos.

Os autores em Yan et al. (2009) propõe um uso combinado do número de saltos com o LQI (*Link Quality Indicator*) para a seleção dos nós pais no processo de construção da topologia.

As pesquisas apresentadas por Ghataoura et al. (2009) propõe uma modificação do mecanismo do RPL para usar uma métrica baseada em um sistema genético-*fuzzy*, capaz de realizar escolhas ótimas na transmissão das mensagens.

Os autores em Kazandjieva et al. (2012) fazem a proposta de uma combinação das duas métricas padrão do RPL, fazendo uma proposta de função objetivo baseada tanto em OF0 (número de saltos) quanto em ETX (expectativa de transmissão).

Barbato et al. (2013) apresentou uma proposta para uso de métricas de energia associado ao funcionamento do RPL. Neste trabalho, a quantidade de energia atual do nó é levada em conta para o processo de decisão de encaminhamento do tráfego de rede para os nós vizinhos.

A técnica apresentada realmente assegura a redução do consumo de energia, contudo, deteriora a qualidade do encaminhamento de mensagens, comprometendo a performance da rede.

Rezaei (2014) propôs um controle na potência de transmissão para reduzir o consumo de energia associado à busca de novas rotas.

No trabalho apresentado por Yang. et al. (2014), os autores realizam uma investigação da estabilidade do RPL e propõe uma modificação para o controle de tráfego. A ideia é interessante, mas pressupõe que os links sejam todos bidirecionais e simétricos, o que do ponto de vista prático, não é aplicável em redes de sensores sem fio do padrão IEEE 802.15.4e.

Em Lee et al. (2014), os autores propõem o uso do RSSI (*Received Signal Strength Indication*) como métrica para o funcionamento do RPL.

Os autores em Kamgueu et al (2015) propõe um mecanismo de clusterização para construção hierárquica da topologia. O objetivo é melhorar um roteamento baseado em árvores por meio da redução do número de saltos, tentando resolver, assim, problemas com falhas em links. Além da proposta de construção hierárquica da topologia, é usada uma função objetivo com seu *rank* baseado em lógica *fuzzy* para construção e manutenção destas topologias em árvores.

Em Fotouhi e Moreira (2015), os autores fazem a proposta de uso de um mecanismo baseado no RPL, onde os dados de informações geográficas são utilizados como métrica para composição da avaliação da função objetivo.

Em Ko e Chang (2015), os pesquisadores propõem a utilização de uma função chamada *MoMoRo*, situada em um ponto intermediário entre as camadas de enlace e camada de rede, com o objetivo de se aumentar a entrega de pacotes, evitando que a rede sofra alterações em sua topologia causada por algum nó móvel.

A partir do trabalho proposto por Kamgueu et al (2011), alguns anos mais tarde, o trabalho é estendido por Kamgueu. et al. (2015), propondo o uso de uma métrica combinada, que leva também em conta o uso de lógica *fuzzy* para construção da topologia.

O trabalho proposto por Alvi et al. (2017) propõe uma modificação do RPL por meio da função objetivo fazendo uma métrica combinada das duas funções objetivo básicas propostas pelo ROOL: OF0 e ETX.

No trabalho de Mustary et al. (2017), é feita uma análise do algoritmo AntNET, desenvolvido especificamente para redes MANET, cujo fundamento está no uso de dois agentes (*forward ant* e *backward ant*). Nessa proposta um agente *forward ant* tenta realizar um caminho completo de uma origem até um determinado destino, guardando a lista de todos os pontos percorridos. No caso de sucesso em se alcançar o destino proposto, esta *forward ant* se

transforma em uma *backward ant* e por meio da lista de nós percorridos, realiza o caminho inverso fazendo o depósito de feromônio neste caminho. De acordo com esta proposta, existe a necessidade de armazenamento desta informação de feromônio em cada um dos nós percorridos, os quais seriam usados pelos próximos agentes *forward ant*.

No trabalho de Park et al. (2017), é apresentada uma proposta de utilização do *Trickle Timer* de forma reversa para os nós móveis. Dessa forma, o algoritmo responsável pela regulação do envio de mensagens do tipo DIO reduziria o tempo de envio das mensagens, ao detectar alguma instabilidade na rede. De certa forma, essa condição poderia “acelerar” a reintegração de um nó instável ao DODAG.

O trabalho apresentado por Sousa et al. (2017) propõe a uma otimização do protocolo RPL por meio de uma combinação de métricas da função objetivo baseado em dois parâmetros: ETX (expectativa de transmissão), associada à EC (que é a energia consumida pelo nó atual) na composição do rank.

Em Thapar e Batra (2018), os autores propõem o uso de um algoritmo baseado em colônia de formigas, como métrica para composição da função objetivo dentro do protocolo RPL. A presente tese diferencia-se deste trabalho nos seguintes pontos:

- a) A proposta de Thapar e Batra (2018) realiza o armazenamento de rotas com todos os nós para a composição de um caminho percorrido pelas formigas, enquanto a presente tese não faz uso de tabelas de rotas.
- b) A implementação proposta por esta tese, ao invés de realizar um percurso de um determinado nó até o nó raiz, faz o uso de apenas dois passos: passo 1 com a *forward ant* e passo 2 com a *backward ant*.
- c) A implementação realizada por Thapar e Batra (2018) exige um envio de mensagens adicionais para o controle do roteamento. Ao invés, a presente tese não usa nenhum tipo de envio de mensagens adicionais.
- d) A implementação realizada por Thapar e Batra (2018) exige um ambiente de simulação onde acontecem atualizações globais de feromônio, inviabilizando a sua implementação física, uma vez que seria necessário um mecanismo de sincronização entre os nós. Por outro lado, a presente tese realiza apenas atualizações locais de feromônio, e a contabilização não é armazenada de maneira global, mas localmente, e pela lista de vizinhos que já conseguiram sucesso na comunicação.

O trabalho apresentado por Guo et. al (2018), propõe um mecanismo de clusterização e um algoritmo de decomposição do tráfego de dados com o objetivo de economia de energia.

Liu e Li (2018) apresentam uma proposta baseada em múltiplas antenas para um acesso multicanal coerente associado a um algoritmo para controle da taxa de transmissão, com o objetivo de se atingir a eficiência energética.

Wang et al. (2018), propõe o uso de um algoritmo heurístico a redes de sensores sem fio para realização da distribuição de carga de roteamento com a finalidade de economia de energia, tomando-se como base um roteamento não mínimo, desviando o tráfego para pontos de acesso específicos.

É interessante notar que várias pesquisas assumiram algum tipo de modificação em padrões de roteamento com a finalidade de alcançar uma eficiência energética.

Muitas propostas, no entanto, fazem o uso de metodologias específicas, as quais são soluções particulares, que se limitam em prova de conceito, sem, necessariamente, se preocupar com o funcionamento em ambientes reais.

É neste contexto, portanto, que esta tese está inserida, fazendo a proposta de um algoritmo de roteamento comprometido com a compatibilidade e funcionamento em arquiteturas já consolidadas, como o *Contiki* e o *OpenWSN*, sendo possível de ser aplicados a dispositivos reais.

Para isso, foi desenvolvido um método tomando-se como base o padrão de roteamento RPL – presente nestas arquiteturas, associando-se ao mesmo, características heurísticas, baseadas em colônias de formigas, com o objetivo de verificar a viabilidade de uma eficiência energética. Esta proposta é apresentada no próximo capítulo.

CAPÍTULO 5

5. MÉTODO, PROPOSTA DO ALGORITMO, ASPECTOS DE IMPLEMENTAÇÃO E CONSUMO DE ENERGIA

Este capítulo apresenta discussões sobre o método utilizado, proposta do algoritmo, aspectos de implementação, hardware utilizado e consumo de energia.

A seção 5.1 faz uma discussão a respeito dos desafios encontrados. A seção 5.2 apresenta a metodologia e o algoritmo proposto. Na seção 5.3 e suas subseções são abordados alguns aspectos de implementação utilizados nas plataformas escolhidas para validação. A seção 5.4 apresenta a placa eletrônica utilizada para os testes físicos. Finalmente, a seção 5.4 apresenta o método utilizado para verificação do consumo de energia.

5.1. Discussões iniciais e desafios

De acordo com Gadour (2012), o RPL é o protocolo padrão, especificamente concebido para o funcionamento de redes de sensores sem fio com baixa potências e perdas, e é compatível com o padrão 6LoWPAN (*IPv6 over Low power Wireless Personal Area Network*), que permite o uso do IPv6 (*Internet Protocol* versão 6) para endereçamento dos nós pertencentes a estas redes.

Conforme descrito por Abuein et al. (2016), desde que foi criado, o RPL adquiriu maturidade e tem se tornado o padrão de roteamento para redes de sensores sem fio no padrão 6LoWPAN, sendo usado largamente em implementações reais.

Dessa forma, as contribuições no sentido de roteamento necessitam seguir estes padrões já consolidados e normatizados, de tal forma a agregar um valor às arquiteturas existentes e permitir que outras propostas possam também ser construídas sob esta perspectiva.

A presente tese propõe, portanto, uma melhoria no padrão de roteamento de uma rede de sensores sem fio, com a finalidade de se atingir a eficiência energética, fazendo o uso do RPL, sem, no entanto, permitir nenhuma quebra de compatibilidade de interfaces. Esta opção por manter as interfaces torna a contribuição mais efetiva do ponto de vista de utilização em sistemas reais e em novas implementações.

De acordo com Aljarrah et al. (2016), a construção da topologia feita pelo processo DODAG do protocolo RPL é um dos pontos onde se encontra o maior desafio do ponto de vista de uso eficiente de energia.

O desafio está ligado à manutenção da estabilidade da rede, uma vez que se houver alguma falha nos links, novas configurações de topologia irão exigir o envio de novas mensagens, que, por consequência, levarão a um aumento no consumo de energia.

Uma hipótese que surge, neste momento, é de que a manutenção da estabilidade da rede, levaria, portanto, a um menor envio de mensagens de controle RPL, que por sua vez, conduziria a uma redução no consumo de energia – principalmente devido à redução de comunicação por meio dessas mensagens.

Esta hipótese é confirmada e validada, portanto, por meio dos testes e apresentada no próximo capítulo nas discussões dos resultados.

5.2. Método e Apresentação do Algoritmo

Tomando-se como base o protocolo RPL – descrito no capítulo 2 – é possível levantar que o seu algoritmo de funcionamento depende, essencialmente, do envio e recebimento adequado das mensagens DIO, DAO e DIS.

A Figura 18, apresenta um fluxograma com este algoritmo.

Dessa forma, o estabelecimento de um *link* (ou caminho) a ser consolidado, passa, necessariamente, pelo tratamento da mensagem DIO recebida, a qual deverá satisfazer alguns critérios para, então, ser tratada por um mecanismo que irá realizar a construção da topologia.

No momento em que um nó sensor recebe uma mensagem do tipo DIO, acontece o processo de avaliação do *rank*, onde são utilizadas métricas adequadas para a construção de

uma função objetivo. Esta função objetivo será responsável por atribuir uma nota ou *rank* àquele nó que enviou a mensagem DIO e este processo dará origem à construção da topologia.

Após o processo de avaliação do *rank*, o nó que recebeu a mensagem DIO irá atualizar uma tabela interna contendo uma lista de nós vizinhos com os quais já realizou algum tipo de comunicação, e irá definir a sua posição hierárquica no DODAG, definindo, assim, qual será o seu nó pai.

Com o intuito de preservar a compatibilidade de interfaces, esta tese propõe uma modificação na função objetivo, de tal forma, que o processo do roteamento RPL continue transparente, sem nenhum tipo de alteração no formato de suas mensagens e nem no processo de envio de mensagens DIO, DAO ou DIS.

A Figura 18, a seguir, apresenta o fluxograma do algoritmo proposto.

Figura 18 – Fluxograma do Algoritmo Proposto baseado em Colônia de Formigas

O fluxograma anterior apresenta, portanto, uma modificação no processamento da função objetivo. Em termos de avaliação do *rank*, a RFC 6550 (2015) propõe o uso de dois tipos de função objetivo: a) OF0 (baseada no número de saltos) e b) MRHOF (baseada no valor de ETX que é uma métrica que leva em conta a expectativa de transmissão).

O modelo proposto por este trabalho é baseado em uma adaptação do algoritmo *MAX-MIN Ant System*, apresentado Marco Dorigo, com a diferença de não armazenar o caminho completo para solução de problemas como os do tipo TSP (*Traveling Salesman Problem*) (DORIGO et al., 2018).

O uso do ACO tradicional, primeiro algoritmo proposto por Dorigo exige o conhecimento de caminhos completos da origem ao destino, tendo várias complexidades envolvidas, sendo de difícil aplicação a redes de sensores sem fio.

Estas dificuldades se encontram na necessidade do envio de caminhos completos, manutenções de tabelas de rotas e necessidade de um gerenciamento global de atualizações.

Ao invés disso, esta tese tomou como base um modelo aplicado em roteamento de redes MANET: o modelo AntNET, sendo aplicadas várias modificações.

O modelo desenvolvido utiliza dois tipos de agentes: **FORWARD ANT** e **BACKWARD ANT**. Tomando-se como base dois nós vizinhos quaisquer, o envio de uma mensagem de um nó para seu vizinho é representado por uma formiga que caminha no sentido de descoberta de um nó vizinho. Durante o percurso de busca, ela é chamada FORWARD ANT e ao retornar, ela é chamada de BACKWARD ANT. Se o caminho for funcional – isto significa mensagem enviada com sucesso, acontece o depósito de feromônio no nó de onde ela veio. A principal diferença com relação a outros algoritmos baseados em colônia de formigas é que este tipo de movimento acontece somente com um salto. Nenhum agente avança mais do que um nó por vez.

As etapas do algoritmo proposto para o protocolo HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) são descritas a seguir:

Condições Iniciais: Não existe topologia formada. O processo será iniciado a partir do roteador de borda (nó *sink*), que irá enviar a primeira mensagem DIO, a qual terá seu processamento iniciado pelo primeiro bloco do fluxograma (Recebimento do DIO) apresentado na Figura 18.

Passo 1: Sejam dois nós vizinhos *i* e *j*, representados na Figura 19. Um nó *i* envia uma mensagem TX ao seu vizinho *j*.

Figura 19 – Envio da mensagem de i para j

Conforme descrito na Figura 20, a mensagem enviada de i para j é considerada a **FORWARD ANT**.

Figura 20 – Envio da mensagem de i para j (FORWARD ANT)

Passo 2: O nó j envia uma mensagem de reconhecimento ACK, respondendo à mensagem recebida de seu vizinho i , conforme a Figura 21.

Figura 21 – Envio da mensagem de j para i

Conforme descrito na Figura 22, a mensagem enviada de j para i é considerada a **BACKWARD ANT**.

Figura 22 – Envio da mensagem de j para i (BACKWARD ANT)

Passo 3: Ao receber a mensagem de reconhecimento ACK, o nó i contabiliza um certo depósito de feromônio na lista do seu vizinho j , conforme a Figura 23.

Figura 23 – Envio da mensagem de j para i (BACKWARD ANT) – Depósito do feromônio

A contabilização do feromônio depositado em j não é feita no próprio nó j , mas sim em i , o qual iniciou o mecanismo de envio do agente FORWARD ANT.

O nó i que possui uma lista de vizinhos, irá associar, portanto, um valor de depósito na sua lista de vizinhos, conforme mostra a Figura 24.

Figura 24 – Depósito do feromônio – Atualização da lista de vizinhos

O modo do RPL usado é *non-storing mode* (ou seja, não há tabela de rotas, mas apenas uma lista de vizinhos diretos “visíveis” para a comunicação).

Este processo acontece para todas as mensagens transmitidas e que recebem um reconhecimento ACK.

Com relação ao processo de criação e remoção dos agentes, a abordagem é realizada da seguinte forma:

- 1) No momento em que uma mensagem será transmitida, é criado um agente FORWARD ANT que, ao chegar em um nó vizinho, se transforma em um agente BACKWARD ANT.
- 2) Ao chegar de volta ao nó de origem, por meio do recebimento do ACK, este agente BACKWARD ANT é removido.
- 3) Os agentes criados por meio de mensagens enviadas e que não receberam o ACK são desconsiderados ou removidos.

De maneira distinta dos outros algoritmos baseados em colônia de formigas, os agentes apresentados na presente proposta não carregam informações de rotas ou listas de nós que atravessaram. Esta característica poderia impor o envio de mensagens adicionais ou modificação no padrão de funcionamento do protocolo RPL.

A contabilização do feromônio depositado não acontece no nó onde ele se encontra, mas nas tabelas dos nós vizinhos, os quais possuem um mecanismo interno pelo qual também realizam o processo de evaporação desse feromônio.

A função objetivo utilizada foi modificada no sentido de se utilizar um padrão híbrido por meio da combinação de métricas baseadas em dois fatores:

- a) Quantidade de feromônio no nó;
- b) Um valor mínimo capaz de garantir o funcionamento do RPL caso não haja nenhum depósito de feromônio. Este valor se baseia na função objetivo ETX pura.

Além disso, foi definido um valor máximo de depósito de feromônio, a partir do qual não ocorrerão mais acréscimos, de tal forma que a função objetivo tenha limiares mínimo e máximo.

$$f_{OBJ} = f_{HOFACO} + f_{ETX} \quad (9)$$

Onde:

- f_{OBJ} é a função objetivo.
- f_{HOFACO} é a contribuição da função objetivo baseada no algoritmo HOFACO proposto.
- f_{ETX} é a contribuição da função baseada em ETX (expectativa de transmissão).

A contribuição devido ao algoritmo HOFACO proposto utiliza a seguinte equação:

$$f_{HOFACO} = \tau_{ij} \quad (10)$$

Onde:

- f_{HOFACO} é a contribuição da função objetivo baseada no algoritmo HOFACO proposto.
- τ_{ij} é quantidade de feromônio presente no nó j, visto a partir do nó i.

A taxa de depósito de feromônio foi definida como:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \tau_{STEP} \quad (11)$$

Onde:

- τ_{ij} é a quantidade de feromônio presente no nó j, visto a partir do nó i.
- ρ é um parâmetro para regular a taxa de evaporação do feromônio.
- τ_{STEP} é a quantidade de incremento para a cada depósito, definido como 0,01.

Além da equação 11, após o depósito ou evaporação foi aplicado um filtro definindo valores mínimos e máximos, da seguinte forma (equação 12):

$$\tau_{MIN} < \tau_{ij} < \tau_{MAX} \quad (12)$$

Onde:

- τ_{MIN} é a quantidade mínima, que foi definida como 0,01.
- τ_{ij} é a quantidade de feromônio presente no nó j, visto a partir do nó i.
- τ_{MAX} é a quantidade máxima, que foi definida como 0,99.

O valor de τ_{STEP} (equação 11) foi definido como 0,01 para efeito de normalização, uma vez que a função HOFACO foi delimitada entre os limiares τ_{MIN} e τ_{MAX} para evitar a convergência precoce do algoritmo, conforme Liaw e Chang (2017).

5.3. Aspectos de implementação

Para efeito de validação, o algoritmo do protocolo HOFACO proposto foi implementado em duas plataformas distintas: *OpenWSN* e *Contiki*.

A escolha das plataformas *OpenWSN* e *Contiki* se deve ao fato de que as mesmas são plataformas de ambientes abertos, ambos compatíveis com o padrão IEEE802.15.4e, com sua arquitetura dividida em camadas e possibilitam, além de ambientes de simulação, a sua utilização em dispositivos físicos.

Com relação às características do *OpenWSN* e *Contiki*, pode-se enumerar:

- a) Possuem código fonte aberto, escrito em linguagem C, compatíveis com diversas arquiteturas computacionais e já estão consolidadas como plataformas de testes em pesquisas;
- b) Apresentam um alto grau de maturidade e robustez;
- c) Permitem tanto uma abordagem em simulação computacional, quanto a possibilidade do *deployment* em dispositivos físicos.

Outras plataformas como por exemplo o NS-2, NS-3 e OMNET++ não foram escolhidas por apresentarem o foco em testes apenas simulados.

As implementações foram feitas no código principal de cada uma das plataformas (*OpenWSN* e *Contiki*) e são apresentadas na seções seguintes. Os resultados dos testes são apresentados no próximo capítulo.

5.3.1. OpenWSN

Na plataforma *OpenWSN* o algoritmo do protocolo HOFACO, descrito na seção 5.2 foi implementado nos módulos *forwarding*, *rpl*, *neighbors*, *forward* e *schedule*, conforme representado na Figura 25.

Figura 25 – Pilha da arquitetura do OpenWSN

Fonte: WATTEYNE et al. (2012), com adaptações

A função objetivo original disponível no *OpenWSN* realiza o cálculo do *rank* baseado na métrica ETX (expectativa de transmissão). Neste sentido, ela foi modificada para contemplar a composição das métricas. Uma vez que a arquitetura do *OpenWSN* está toda baseada mecanismo no schedule, foi implementado uma tarefa para realizar a evaporação do feromônio.

5.3.2. Contiki

Na plataforma *Contiki* o algoritmo do protocolo HOFACO foi implementado nos módulos de gerenciamento de rede, especificamente no módulo RPL.

O *Contiki* utiliza, por padrão, duas possibilidades de funções objetivo: a) OF0 (função objetivo zero), com métrica baseada em número de saltos e b) MRHOF (*Minimum Rank with Hysteresis Objective Function*), com métrica baseada em ETX (expectativa de transmissão).

Para efeito de evaporação do feromônio, foi criada uma entrada na aplicação principal, que faz o uso de uma *protothread*, que, por sua vez, realiza periodicamente a evaporação do feromônio. A Figura 26 ilustra os módulos utilizados.

Figura 26 – Arquitetura do Contiki

Fonte: FAROOQ; KUNZ (2011), com adaptações

5.4. Hardware utilizado

A implementações realizadas tanto na plataforma *Contiki* quanto *OpenWSN* contemplaram, além da simulação, os testes feitos em dispositivos físicos. Para estes últimos, foi desenvolvida pela equipe de pesquisa do LAI (Laboratório de Automação Industrial) da EESC-USP uma placa eletrônica baseada em um microcontrolador da *Texas Instruments* – o C2538 do tipo ARM Cortex M3, demonstrada na Figura 27.

Figura 27 – Placa eletrônica desenvolvida

Na figura 27 são demonstrados: os conectores GPIO (*General Purpose Input Output*) – os quais podem ser conectados a outros dispositivos externos; uma interface USB OTG (*Universal Serial Bus – On The Go*) – possível de ser usada tanto para a comunicação quanto para a alimentação do circuito; uma interface JTAG (*Joint Test Access Group*) – padrão utilizado para realizar a operação de download de firmware e depuração em microcontroladores ARM; a CPU (*Central Processor Unit*) – baseado em um microcontrolador ARM Cortex M3 da Texas Instruments: o cc2538 e a antena on-board.

5.5. Mecanismo adotado para verificação do consumo de energia

Como um dos objetivos desta proposta é avaliar o consumo de energia do protocolo HOFACO, foi necessário levantar um mecanismo suficientemente confiável, capaz de evidenciar os consumos de energia pelos nós sensores durante a utilização deste novo protocolo, comparando-o a outras implementações (OF0 e ETX).

Inicialmente, uma primeira possibilidade seria a realização uma medida de consumo de energia por meio de uma medição indireta de corrente do dispositivo, utilizando-se um osciloscópio. Essa abordagem, portanto, apresenta alguns inconvenientes, pois é necessário que todos os nós sensores sejam avaliados, uma vez que o algoritmo utiliza a comunicação entre os

elementos da rede e cada nó pode ter um comportamento distinto, em termos de consumo de energia. É necessário, portanto, uma estimativa de energia para todo o conjunto.

Por outro lado, foi avaliada uma proposta muito interessante, apresentada por Dunkels et al. (2011), capaz de atender a esta demanda de medição de energia. No referido trabalho, os autores desenvolveram um mecanismo dentro do próprio *firmware* do *Contiki* capaz de realizar de maneira automática a contabilização de energia utilizada, conforme as Figuras 28 e 29.

Figura 28 – Powertrace– Cápsulas de Energia

Fonte: Dunkels et al., (2011), com adaptações

De acordo com Dunkels et al. (2011) a energia total gasta por um sensor deve levar em conta o seu consumo de energia em quatro estados:

- a) CPU em modo de execução.
- b) Nó sensor em modo TX para transmissão de dados.
- c) Nó sensor em modo RX para recepção de dados.
- d) CPU em modo *low power* (de economia de energia).

Figura 29 – Powertrace – Cápsulas de Energia

Fonte: Dunkels et al., (2011), com adaptações

A Figura 29 apresenta a contabilização da energia gasta durante um determinado tempo (t_A, t_B, t_C e t_D) para realização de diferentes atividades (A, B, C e D). Ao final, por meio dos valores acumulados nas cápsulas de energia, é possível realizar a estimativa da quantidade total de energia gasta por uma determinada atividade.

Sendo, assim, a energia total consumida é o resultado da soma das energias consumidas em cada estado, conforme a equação 13:

$$E_{TOTAL} = E_{CPU\ run} + E_{CPU\ low\ power} + E_{TX} + E_{RX} \quad (13)$$

Onde:

- E_{TOTAL} é a energia total consumida pelo nó sensor.
- $E_{CPU\ run}$ é a energia consumida devido ao tempo da CPU em modo de execução.
- $E_{CPU\ low\ power}$ é a energia consumida devido ao tempo da CPU em modo de *low power* (economia de energia).
- E_{TX} é a energia consumida pelo nó sensor em modo TX para transmissão de dados.
- E_{RX} é a energia consumida pelo nó sensor em modo RX para recepção de dados.

Dunkels et al. (2011) propôs um mecanismo baseado em *timers* de tempo real, de altíssima precisão, capazes de capturar os momentos exatos em que um nó sensor esteja em cada um dos quatro estados descritos anteriormente.

Este conceito é utilizado para a composição de cápsulas de energia, capazes de contabilizar cada um dos estados, de tal forma que o valor total do consumo de um nó durante um intervalo de tempo é o resultado da soma algébrica dos consumos individuais de cada um dos estados durante aquele determinado intervalo de tempo.

Os autores em Dunkels et al. (2011), realizaram a verificação da proposta e demonstraram um valor de precisão em torno de 94% para utilização deste módulo *powertrace*.

É importante ressaltar que este mecanismo do módulo *powertrace* está incluso apenas no código fonte do *Contiki*, não sendo ainda aplicável ao *OpenWSN*.

Devido à facilidade de utilização e características de precisão em termos de custo x benefício, este módulo foi escolhido para ser usado nos cenários de testes com o *Contiki* descritos no próximo capítulo.

CAPÍTULO 6

6. EXPERIMENTOS REALIZADOS E RESULTADOS

Para efeito de validação do método proposto e levantamento de limites de utilização foram feitos testes utilizando cenários baseados em simulação e também em hardware físico (estes últimos utilizando-se as placas eletrônicas desenvolvidas apresentadas no capítulo anterior).

Os experimentos apresentados nos cenários 1, 2, 3 e 4 foram realizados por meio do simulador *COOJA* dentro do ambiente *Contiki*.

O experimento apresentado no cenário 5 foi realizado com simulação no *OpenWSN*.

O experimento descrito no cenário 6 foi realizado com hardware físico no *OpenWSN*.

Finalmente, o experimento apresentado no cenário 7 foi realizado com hardware físico no *Contiki*.

O objetivo destes testes é avaliar e comparar o comportamento do roteamento com relação à utilização das seguintes métricas para a função objetivo do protocolo RPL:

- a) Roteamento com Função Objetivo OF0 (*Objective Function Zero*). Esta opção tem como base o cálculo do *rank* baseado apenas no número de saltos.
- b) Roteamento com a Função Objetivo MRHOF (*Minimum Rank with Hysteresis Objective Function*). Esta opção tem como base o cálculo do *rank* baseado no valor de ETX (Expectativa de Transmissão), que é obtido a partir da avaliação de uma relação entre a quantidade de mensagens enviadas e a sua realização com sucesso.
- c) Roteamento com a Função Objetivo Híbrida Proposta HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*).

Como métricas de avaliação foram utilizadas:

- 1) A observação dos valores do intervalo do *Trickle Algorithm* – algoritmo presente no roteamento baseado no RPL que realiza o gerenciamento dos tempos de envio das mensagens de controle do RPL (DIO e DAO).
- 2) O acompanhamento da quantidade de alteração de escolha de nós *parent* a partir da observação de alterações ocorridas.
- 3) Latência da rede a partir do envio de mensagens a algum dos nós a partir do nó *sink*.
- 4) Observação e comparação da quantidade de energia gasta na rede como um todo por meio da utilização da técnica de cápsulas de energia, a partir da ferramenta *powertrace*, sendo este último aplicado ao cenário 7 (hardware real) no ambiente *Contiki*.

6.1. Cenário 1 – Topologia com 6 nós sensores – Simulação com *COOJA*

No cenário 1, construiu-se uma topologia baseada em 6 nós sensores (1 *sink* e 5 *motes*) de tal forma a observar as características de roteamento descritas anteriormente. A Figura 30 ilustra este cenário.

Figura 30 – Cenário 1 – Topologia com 6 nós (1 *sink* e 5 *motes*)

Os nós foram distribuídos em um mesmo plano, com a característica do meio UDGM (*Unit Disk Graph Medium*) - *Distance Loss*, disponível no simulador *COOJA* do *Contiki*.

O modelo *UDGM Distance Loss* do simulador utiliza a configuração de propagação do sinal no meio de acordo com dois discos, conforme a Figura 31: um primeiro (sinalizado na cor verde) onde o alcance da transmissão é efetivo e um segundo (sinalizado na cor cinza) onde acontece degradação do sinal e possível interferência do mesmo com outros sinais.

Os nós sensores utilizados para simulação foram do tipo Z1, pois compartilham a mesma base de tempo da placa eletrônica desenvolvida.

Os firmwares utilizados para os nós foram os distribuídos com os padrões do *Contiki*: *rpl-border-router.c* (para o nó *sink*) e *websensor.c* (para os demais nós sensores) com as devidas modificações para contemplar as implementações realizadas.

De maneira geral, as configurações deste cenário no simulador são apresentadas na tabela 3.

Tabela 3 – Configurações Gerais do Cenário 1 no Simulador *COOJA*

Configuração	Valor
Meio físico do Rádio	UDGM (Unit Disk Graph Medium) - Distance Loss
Mote	Z1
Firmwares	<i>rpl-border-router.c</i> e <i>websensor.c</i> , com modificações
Interface do Rádio	IEEE 802.15.4e
Número de nós	6 (1 <i>sink</i>) e (5 sensores)

As coordenadas, a seguir, representam os valores das posições dos nós sensores, cujas distâncias estão em metros.

As coordenadas dos pontos são:

- Nó *sink*: ID 1 (x = 100.0; y = -95.0; z = 0.0).
- Nós sensores: ID 2(x=70.0; y= -70.0; z = 0.0); ID 3 (x=100.0; y= -70.0; z = 0.0);
ID 4 (x=130.0; y= -70.0; z = 0.0); ID 5 (x=100.0; y= -40.0; z = 0.0) e
ID 6 (x=100.0; y= 0.0; z = 0.0).

A visibilidade dos nós é representada nas figuras 31 e 32, a seguir.

Figura 31 – Cenário 1 – Visibilidade dos nós – Parte 1

A figura 31, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 1 – possui como vizinhos os nós 2, 3 e 4.
- (b) Nó 2 – possui como vizinhos os nós 1, 3 e 5.
- (c) Nó 3 – possui como vizinhos os nós 1, 2, 4 e 5.
- (d) Nó 4 – possui como vizinhos os nós 1, 3 e 5.

Figura 32 – Cenário 1 – Visibilidade dos nós – Parte 2

A figura 32, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 5 – possui como vizinhos os nós 2, 3, 4 e 6.
- (b) Nó 6 – possui como vizinho apenas o nó 1.

6.1.1. Itens analisados e Resultados

Neste cenário 1, foram executadas 3 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada somente na função OF0 (*Objective Function Zero*), que usa número de saltos;
- b) Baseada na função MRHOF (*Minimum Rank with Hysteresis Objective Function*), que usa a métrica ETX (expectativa de transmissão);
- c) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

O tempo de cada uma das rodadas foi de 5 minutos, suficiente para a estabilização da rede (define-se aqui a estabilização da rede como a construção completa da topologia e ausência de alterações na mesma evidenciada pela ausência de mudanças de nós *parent*).

Os itens analisados para cada uma das rodadas neste cenário 1 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Tempo de Intervalo do envio de mensagens DIO (acompanhamento do *Trickle Timer*).
- 3) Número de alterações de nós *parent*.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 5) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 7) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A seguir, a Figura 33 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 33 – Cenário 1 – Número de mensagens de Controle RPL

As Figuras de 34 a 36 apresentam a evolução do intervalo de envio de mensagens DIO, controlada pelo algoritmo *Trickle Timer*, que está diretamente relacionado à estabilização da rede.

Figura 34 – Cenário 1 – Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo OF0

Figura 35 – Cenário 1 - Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo ETX

Figura 36 – Cenário 1 – Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo HOFACO

A Figura 37 apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos três algoritmos.

Figura 37 – Cenário 1 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2 e 3 é apresentada nas Figuras 38 e 39. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 38 – Cenário 1 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 4, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2 e 3 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 38, acima.

Tabela 4 – Cenário 1 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	137,29	38,34
	2	156,93	28,32
	3	251,32	32,14
ETX	1	127,19	32,34
	2	166,28	23,42
	3	251,32	33,32
HOFACO	1	137,25	29,27
	2	172,42	28,42
	3	231,38	38,33

Figura 39 – Cenário 1 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 5, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2 e 3 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 39, acima.

Tabela 5 – Cenário 1 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	832,49	138,34
	2	1156,93	238,38
	3	1551,32	323,15
ETX	1	951,35	218,41
	2	1421,35	321,47
	3	1620,23	415,53
HOFACO	1	613,97	121,42
	2	1006,33	221,83
	3	1216,24	233,15

Fazendo-se a variação de saltos em 1, 2 e 3 nós sensores. Foi levantada a taxa de entrega de pacotes em porcentagem, para cada um dos três algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 40 e 41.

Figura 40 – Cenário 1 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 41 – Cenário 1 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.1.2. Discussões

A partir desta seção iremos denominar os protocolos analisados da seguinte forma:

- a) OF0 – usado para função objetivo baseada em número de saltos.
- b) ETX – usado para a função objetivo baseada em expectativa de transmissão.
- c) HOFACO – usado para função objetivo híbrida proposta com base no ACO.

Uma análise da quantidade de mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 33 demonstra que no protocolo HOFACO acontece uma ligeira diminuição do envio das mensagens de controle, com relação aos outros protocolos.

Há também uma visível diminuição na quantidade de mensagens DIS. Quando a rede se encontra com instabilidade, os nós que ainda não se juntaram aos demais enviam mensagens DIS, solicitando um DIO. A quantidade menor de mensagens DIS deixa evidências de que há uma melhor estabilidade na rede.

Para comprovar esta ideia, foi também verificado a evolução do *Trickle Timer*, que é o algoritmo faz a regulação do envio de mensagens DIO. À medida que a rede vai se estabilizando, o valor do intervalo vai sendo aumentado. Isto pode ser verificado nas figuras 34, 35 e 36. Estes gráficos demonstram que a estabilidade foi atingida na seguinte ordem: a) em aproximadamente 60 segundos para o protocolo HOFACO; b) em aproximadamente 70 segundos para o protocolo OF0 e c) em aproximadamente 105 segundos para o protocolo ETX.

Uma outra característica de estabilidade seria a menor troca de nós *parent*, fazendo com que um nó esteja estável em sua posição. O gráfico da Figura 37 mostra que ocorreram 2 alterações de *parent* para o protocolo OF0, 3 para o protocolo HOFACO e 4 para o protocolo ETX.

Os gráficos de latência em função do número de saltos com um tráfego pequeno (10 bytes/segundo) são praticamente equivalentes para os três protocolos, conforme mostra a Figura 38.

Ao provocarmos maior tráfego na rede (50 bytes/segundo), notamos que há uma latência um pouco maior para os protocolos OF0 e ETX em relação ao HOFACO, conforme a Figura 39.

Com relação à taxa de entrega de pacotes (apresentadas nas Figuras 40 e 41), notamos que ela é praticamente equivalente para um número menor de saltos.

Entretanto, é interessante observar – a partir dos gráficos – que a taxa de decaimento (ou derivada da função) apresenta valores distintos para cada um dos protocolos ao se aumentar tanto o número de saltos quanto o tráfego na rede.

Os gráficos apresentados nas Figuras 40 e 41 demonstram que com um número maior de saltos o protocolo HOFACO apresenta uma melhor performance do que do ponto de vista de sua taxa de decaimento.

Este cenário contemplou uma configuração de apenas 6 nós, nos quais apenas o nó 5 (mostrado na Figura 30), possui três opções de rotas diferentes a serem escolhidas para seu nó *parent*, em função da visibilidade de seus vizinhos.

Os próximos cenários irão demonstrar outras análises, levando-se em conta outros fatores com um maior número de nós e outras possibilidades de escolhas de rotas diferentes.

É possível concluir – portanto – que do ponto de vista dos itens analisados, para este cenário, o protocolo HOFACO se comportou melhor que os outros dois tradicionais OF0 e ETX.

6.2. Cenário 2 – Topologia com 10 nós sensores – Simulação com *COOJA*

No cenário 2, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente. A Figura 42 ilustra este cenário.

Figura 42 – Cenário 2 – Topologia com 10 nós (1 *sink* e 9 *motes*)

De forma similar ao apresentado no cenário 1, as configurações deste cenário 2 no simulador são apresentadas na tabela 6.

Tabela 6 – Configurações Gerais do Cenário 2 no Simulador *COOJA*

Configuração	Valor
Meio físico do Rádio Mote	UDGM (Unit Disk Graph Medium) - Distance Loss Z1
Firmwares Interface do Rádio	<i>rpl-border-router.c e websensor.c, com modificações</i>
Número de nós	IEEE 802.15.4e 10 (1 <i>sink</i>) e (9 sensores)

As coordenadas representadas, a seguir, representam valores cujas distâncias estão em metros.

As coordenadas dos pontos são:

- Nó *sink*: ID 1 (x = 100.0; y = -95.0; z = 0.0).
- Nós sensores: ID 2(x=70.0; y= -65.0; z = 0.0); ID 3 (x=100.0; y= -80.0; z = 0.0);
ID 4 (x=130.0; y= -65.0; z = 0.0); ID 5 (x=70.0; y= -30.0; z = 0.0);
ID 6 (x=130.0; y= -30.0; z = 0.0); ID 7 (x= 100.0; y= 0.0; z = 0.0);
ID 8 (x=70.0; y= 30.0; z = 0.0); ID 9 (x=130.0; y= 30.0; z = 0.0) e
ID 10 (x=100.0; y= 60.0; z = 0.0).

A visibilidade dos nós é representada nas figuras 43, 44 e 45, a seguir.

Figura 43 – Cenário 2 – Visibilidade dos nós – Parte 1

A figura 43 representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 1 – possui como vizinhos os nós 2, 3 e 4.
- (b) Nó 2 – possui como vizinhos os nós 1, 3 e 4.
- (c) Nó 3 – possui como vizinhos os nós 1, 2 e 4.
- (d) Nó 4 – possui como vizinhos os nós 1, 3 e 6.

Figura 44 – Cenário 2 – Visibilidade dos nós – Parte 2

A figura 44 representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 5 – possui como vizinhos os nós 2 e 7.
- (b) Nó 6 – possui como vizinhos os nós 4 e 7.
- (c) Nó 7 – possui como vizinhos os nós 5, 6, 8 e 9.
- (d) Nó 8 – possui como vizinhos os nós 7 e 10.

Figura 45 – Cenário 2 – Visibilidade dos nós – Parte 3

A figura 45, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 9 – possui como vizinhos os nós 7 e 10.
- (b) Nó 10 – possui como vizinhos os nós 8 e 9.

6.2.1. Itens analisados e Resultados

Neste cenário 2, foram executadas 3 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada somente na função OF0 (*Objective Function Zero*), que usa número de saltos;
- b) Baseada na função MRHOF (*Minimum Rank with Hysteresis Objective Function*), que usa a métrica ETX (expectativa de transmissão);
- c) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

O tempo de cada uma das rodadas foi de 5 minutos, suficiente para a estabilização da rede.

Os itens analisados para cada uma das rodadas neste cenário 2 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Tempo de Intervalo do envio de mensagens DIO (acompanhamento do *Trickle Timer*).
- 3) Número de alterações de nós *parent*.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 5) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 7) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A seguir, a Figura 46 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 46 – Cenário 2 – Número de mensagens de Controle RPL

As Figuras de 47 a 49 apresentam a evolução do intervalo de envio de mensagens DIO, controlada pelo algoritmo *Trickle Timer*, que está diretamente relacionado à estabilização da rede.

Figura 47 – Cenário 2 – Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo OF0

Figura 48 – Cenário 2 - Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo ETX

Figura 49 – Cenário 2 – - Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo HOFACO

A Figura 50 apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos três algoritmos.

Figura 50 – Cenário 2 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2, 3, 4 e 5 é apresentada nas Figuras 51 e 52. Foram utilizadas taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 51 – Cenário 2– Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 7, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 4 e 5 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 51.

Tabela 7 – Cenário 2 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	172,92	41,32
	2	236,91	48,32
	3	352,78	63,43
	4	441,23	70,34
	5	851,32	62,42
ETX	1	143,92	36,67
	2	228,93	54,31
	3	323,23	56,18
	4	418,76	60,49
	5	792,23	72,19
HOFACO	1	132,96	26,79
	2	298,74	63,13
	3	383,45	66,97
	4	458,62	63,43
	5	752,35	90,19

Figura 52 – Cenário 2 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 8, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 4 e 5 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 52.

Tabela 8 – Cenário 2 - Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	783,92	98,42
	2	1251,34	183,87
	3	1498,79	338,52
	4	1852,38	303,46
	5	2141,22	345,56
ETX	1	918,25	112,14
	2	1321,51	298,72
	3	1517,32	401,94
	4	1622,18	523,46
	5	2542,22	442,69
HOFACO	1	796,74	87,25
	2	824,16	122,31
	3	1292,42	252,19
	4	1721,81	322,63
	5	2012,25	392,91

Fazendo-se a variação de saltos em 1, 2, 3, 4 e 5 nós sensores. Foi levantada a taxa de entrega de pacotes em porcentagem, para cada um dos três algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 53 e 54.

Figura 53 – Cenário 2 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 54 – Cenário 2 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.2.2. Discussões

A análise das mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 46, mostra uma menor quantidade de mensagens DIO e DAO no protocolo HOFACO do que nos outros protocolos. Neste cenário, com um número maior de nós, o protocolo que menos enviou mensagens DIS foi o ETX.

A verificação da evolução do *Trickle Timer*, apresentada nos gráficos das Figuras 47, 48 e 49 mostra que a estabilidade foi atingida na seguinte ordem: a) em aproximadamente 120 segundos para protocolo HOFACO; b) em aproximadamente 125 segundos para o protocolo OF0 e c) em aproximadamente 135 segundos para o protocolo ETX.

Com relação a alterações de nós *parent*, mostrados na Figura 50, ocorreram 3 para o protocolo OF0 e HOFACO e 4 para o protocolo ETX.

Os gráficos de latência em função do número de saltos com um tráfego pequeno (10 bytes/segundo) também são praticamente equivalentes para os três protocolos, conforme mostra a Figura 51.

Ao provocarmos maior tráfego na rede (50 bytes/segundo), é possível notar uma maior latência, mas praticamente os três protocolos se comportam de maneira equivalente, conforme a Figura 52.

A análise da taxa de entrega de pacotes com relação ao número de saltos, mostra que para tráfegos menores, há uma melhor performance do protocolo HOFACO com relação aos demais.

Quando se aumenta o tráfego na rede, a relação da taxa de entrega em função do número de saltos mostra que os três protocolos se comportam de maneira equivalente, sendo que para um número de 4 saltos foi observado uma maior taxa de decaimento do protocolo HOFACO do que os demais.

Este cenário contemplou uma configuração de 10 nós, nos quais apenas os nós 7 e 10, devido à sua configuração, teriam visibilidade de escolha de mais de um nó *parent*.

É possível concluir – portanto – que do ponto de vista dos itens analisados, houve uma melhoria um pouco menos expressiva do que no cenário anterior do protocolo HOFACO com relação aos demais.

6.3. Cenário 3 – Topologia com 10 nós sensores – Simulação com *COOJA*

No cenário 3, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente. A Figura 55 ilustra este cenário.

Figura 55 – Cenário 3 – Topologia com 10 nós (1 *sink* e 9 *motes*)

De forma similar ao apresentado nos cenários 1 e 2, as configurações deste cenário 3 no simulador são apresentadas na tabela 9.

Tabela 9 – Configurações Gerais do Cenário 3 no Simulador *COOJA*

Configuração	Valor
Meio físico do Rádio Mote	UDGM (Unit Disk Graph Medium) - Distance Loss Z1
Firmwares Interface do Rádio	<i>rpl-border-router.c e websensor.c, com modificações</i>
Número de nós	IEEE 802.15.4e 10 (1 <i>sink</i>) e (9 sensores)

As coordenadas representadas, a seguir, representam valores cujas distâncias estão em metros.

As coordenadas dos pontos são:

- Nó *sink*: ID 1 (x = 100.0; y = -95.0; z = 0.0).
- Nós sensores: ID 2(x=70.0; y= -65.0; z = 0.0); ID 3 (x=100.0; y= -80.0; z = 0.0);
ID 4 (x=130.0; y= -65.0; z = 0.0); ID 5 (x=70.0; y= -30.0; z = 0.0);
ID 6 (x=130.0; y= -30.0; z = 0.0); ID 7 (x= 100.0; y= -35.0; z = 0.0);
ID 8 (x=100.0; y= 0.0; z = 0.0); ID 9 (x=100.0; y= 30.0; z = 0.0) e
ID 10 (x=100.0; y= 60.0; z = 0.0).

A visibilidade dos nós é representada nas figuras 56, 57 e 58, na sequência.

Figura 56 – Cenário 3 – Visibilidade dos nós – Parte 1

A figura 56 representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 1 – possui como vizinhos os nós 2, 3 e 4.
- (b) Nó 2 – possui como vizinhos os nós 1, 3, 5 e 7.
- (c) Nó 3 – possui como vizinhos os nós 1, 2, 4 e 7.
- (d) Nó 4 – possui como vizinhos os nós 1, 3, 6 e 7.

Figura 57 – Cenário 3 – Visibilidade dos nós – Parte 2

A figura 57 representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 5 – possui como vizinhos os nós 2, 7 e 8.
- (b) Nó 6 – possui como vizinhos os nós 4, 7 e 8.
- (c) Nó 7 – possui como vizinhos os nós 2, 3, 4, 5, 6 e 8.
- (d) Nó 8 – possui como vizinhos os nós 5, 6 e 7.

Figura 58 – Cenário 3 – Visibilidade dos nós – Parte 3

A figura 58, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 9 – possui como vizinhos os nós 8 e 10.
- (b) Nó 10 – possui como vizinho apenas o nó 9.

6.3.1. Itens analisados e Resultados

Neste cenário 3, foram executadas 3 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada somente na função OF0 (*Objective Function Zero*), que usa número de saltos;
- b) Baseada na função MRHOF (*Minimum Rank with Hysteresis Objective Function*), que usa a métrica ETX (expectativa de transmissão);
- c) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

O tempo de cada uma das rodadas foi de 5 minutos, suficiente para a estabilização da rede.

Os itens analisados para cada uma das rodadas neste cenário 3 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Tempo de Intervalo do envio de mensagens DIO (acompanhamento do *Trickle Timer*).
- 3) Número de alterações de nós *parent*.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 5) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 7) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A seguir, a Figura 59 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 59 – Cenário 3 – Número de mensagens de Controle RPL

As Figuras de 60 a 62 apresentam a evolução do intervalo de envio de mensagens DIO, controlada pelo algoritmo *Trickle Timer*, que está diretamente relacionado à estabilização da rede.

Figura 60 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo OF0

Figura 61 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo ETX

Figura 62 – Cenário 3 – Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo HOFACO

A Figura 63 apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos três algoritmos.

Figura 63 – Cenário 3 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2, 3, 4 e 5 é apresentada nas Figuras 64 e 65. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 64 – Cenário 3 – Latência da rede em função do número de saltos
com tráfego de 10 bytes/segundo

A tabela 10, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3, 4 e 5 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 64.

Tabela 10 – Cenário 3 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	196,22	42,31
	2	386,12	59,29
	3	402,69	32,37
	4	482,23	104,19
	5	725,24	108,22
ETX	1	169,21	16,67
	2	262,31	42,15
	3	379,21	62,84
	4	457,62	70,29
	5	815,22	104,37
HOFACO	1	141,62	67,91
	2	301,44	78,93
	3	432,57	90,23
	4	497,15	103,43
	5	822,35	91,35

Figura 65 – Cenário 3 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 11, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3, 4 e 5 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 65.

Tabela 11 – Cenário 3 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	592,92	79,13
	2	903,34	147,82
	3	1134,79	233,29
	4	1965,87	351,74
	5	2511,25	395,68
ETX	1	618,25	161,21
	2	1314,18	176,22
	3	1912,42	501,94
	4	2021,83	431,98
	5	2627,14	397,92
HOFACO	1	623,42	73,51
	2	698,27	113,45
	3	1023,31	229,29
	4	1965,25	332,14
	5	2321,51	483,43

Fazendo-se a variação de saltos em 1, 2, 3, 4 e 5 nós sensores. Foi levantada a taxa de entrega de pacotes em porcentagem, para cada um dos três algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 66 e 67.

Figura 66 – Cenário 3 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 67 – Cenário 3 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.3.2. Discussões

A análise das mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 59, mostra, mais uma vez, uma menor quantidade de mensagens DIO e DAO no protocolo HOFACO do que nos outros protocolos. Neste cenário, um número maior de nós, o protocolo que menos enviou mensagens DIS foi o protocolo HOFACO.

A verificação da evolução do *Trickle Timer*, apresentada nos gráficos das Figuras 60, 61 e 62 mostram que a estabilidade foi atingida na seguinte ordem: a) em aproximadamente 110 segundos para protocolo HOFACO; b) em aproximadamente 120 segundos para o protocolo OF0 e c) em aproximadamente 150 segundos para o protocolo ETX.

Neste ponto, é interessante notar, pelo gráfico da Figura 61, que, logo após os 50 segundos, ocorreu um *reset* do *Trickle Timer* – evidenciado e sinalizado neste gráfico. Este *reset* evidencia uma instabilidade na configuração da rede. É possível inferir que a configuração do nó 7 com visibilidade de 6 nós vizinhos, levou a alterações de topologia, que por consequência tornaram a rede menos instável.

A observação do gráfico da Figura 63, permite confirmar esta evidência da instabilidade, pois o protocolo ETX sofreu 5 alterações de *parent*, enquanto os protocolos OF0 e HOFACO sofreram 4 e 3 alterações, respectivamente.

Os gráficos de latência em função do número de saltos com um tráfego pequeno (10 bytes/segundo), assim como no cenário anterior, são praticamente equivalentes para os três protocolos, conforme mostra a Figura 64.

Ao provocarmos maior tráfego na rede (50 bytes/segundo), é possível notar uma maior latência, mas praticamente os três protocolos se comportam de maneira equivalente, conforme mostra a Figura 65.

A análise da taxa de entrega de pacotes com relação ao número de saltos, mostra que para tráfegos menores, há também uma equivalência para os protocolos neste cenário. Esta equivalência também foi observada para um tráfego maior, conforme as Figuras 66 e 67.

É possível concluir, portanto, que do ponto de vista da estabilidade houve uma menor performance do protocolo ETX e, novamente, o protocolo melhor avaliado foi o HOFACO.

6.4. Cenário 4 – Topologia com 10 nós sensores – Simulação com *COOJA*

No cenário 4, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente e o número de saltos. A Figura 68 ilustra este cenário.

Figura 68 – Cenário 4 – Topologia com 10 nós (1 *sink* e 9 *motes*)

De forma similar ao apresentado nos cenários 1, 2 e 3, as configurações deste cenário 4 no simulador são apresentadas na tabela 12.

Tabela 12 – Configurações Gerais do Cenário 4 no Simulador *COOJA*

Configuração	Valor
Meio físico do Rádio Mote	UDGM (Unit Disk Graph Medium) - Distance Loss Z1
Firmwares Interface do Rádio	<i>rpl-border-router.c e websensor.c, com modificações</i>
Número de nós	IEEE 802.15.4e 10 (1 <i>sink</i>) e (9 sensores)

As coordenadas dos pontos são:

- Nó *sink*: ID 1 (x = 200.0; y = -100.0; z = 0.0).
- Nós sensores: ID 2(x=160.0; y= -100.0; z = 0.0); ID 3 (x=120.0; y= -100.0; z = 0.0);
ID 4 (x=120.0; y= -60.0; z = 0.0); ID 5 (x=120.0; y= -20.0; z = 0.0);
ID 6 (x=120.0; y= 20.0; z = 0.0); ID 7 (x= 160.0; y= 20.0; z = 0.0);
ID 8 (x=200.0; y= 20.0; z = 0.0); ID 9 (x=200.0; y= -20.0; z = 0.0) e
ID 10 (x=180.0; y= -40.0; z = 0.0).

A visibilidade dos nós é representada nas figuras 69, 70 e 71, na sequência.

Figura 69 – Cenário 4 – Visibilidade dos nós – Parte 1

A figura 69, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 1 – possui como vizinhos apenas o nó 2.
- (b) Nó 2 – possui como vizinhos os nós 1 e 3.
- (c) Nó 3 – possui como vizinhos os nós 2 e 4.
- (d) Nó 4 – possui como vizinhos os nós 3 e 5.

Figura 70 – Cenário 4 – Visibilidade dos nós – Parte 2

A figura 70, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 5 – possui como vizinhos os nós 4 e 6.
- (b) Nó 6 – possui como vizinhos os nós 5 e 7.
- (c) Nó 7 – possui como vizinhos os nós 6 e 8.
- (d) Nó 8 – possui como vizinhos os nós 7 e 9.

Figura 71 – Cenário 4 – Visibilidade dos nós – Parte 3

A figura 71, acima, representa a visibilidade dos nós vizinhos a partir das seguintes referências:

- (a) Nó 9 – possui como vizinhos os nós 8 e 10.
- (b) Nó 10 – possui como vizinho apenas o nó 9.

6.4.1. Itens analisados e Resultados

Neste cenário 4, foram executadas 3 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada somente na função OF0 (*Objective Function Zero*), que usa número de saltos;
- b) Baseada na função MRHOF (*Minimum Rank with Hysteresis Objective Function*), que usa a métrica ETX (expectativa de transmissão);
- c) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

O tempo de cada uma das rodadas foi de 5 minutos, suficiente para a estabilização da rede.

Os itens analisados para cada uma das rodadas neste cenário 4 foram:

- 1) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 2) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 3) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 4) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A latência de comunicação variando-se o número de saltos em 1, 2, 3, 4, 5, 6, 7, 8 e 9 é apresentada nas Figuras 72 e 73. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 72 – Cenário 4 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 13, a seguir, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3, 4, 5, 6, 7, 8 e 9 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 72.

Tabela 13 – Cenário 4 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OFO	1	213,43	41,11
	2	401,23	53,91
	3	572,32	58,23
	4	752,21	90,23
	5	918,13	120,21
	6	1130,32	245,23
	7	1203,21	298,24
	8	2093,18	389,34
	9	2987,31	422,31
ETX	1	198,17	32,34
	2	441,29	65,38
	3	527,22	83,23
	4	731,83	103,35
	5	1032,31	142,17
	6	1302,86	223,47
	7	1409,23	301,24
	8	2189,12	332,49
	9	3021,98	413,14
HOFACO	1	193,24	72,13
	2	381,36	59,21
	3	527,24	78,32
	4	724,19	103,93
	5	1012,15	142,17
	6	1132,24	245,23
	7	1318,98	357,82
	8	2293,83	328,42
	9	3042,21	441,19

Figura 73 – Cenário 4 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 14, na sequência, apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3, 4, 5, 6, 7, 8 e 9 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 73.

Tabela 14 – Cenário 4 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	242,18	82,45
	2	380,12	63,92
	3	540,23	98,12
	4	850,21	97,22
	5	1140,98	142,15
	6	3001,32	395,38
	7	3593,23	547,23
	8	6023,18	689,98
	9	7903,23	598,23
ETX	1	287,22	86,78
	2	612,23	33,92
	3	549,37	89,21
	4	915,23	109,45
	5	1238,82	182,53
	6	3731,24	258,83
	7	3992,21	442,39
	8	5823,18	498,85
	9	8003,39	476,24
HOFACO	1	207,13	63,97
	2	297,12	74,92
	3	524,74	83,22
	4	695,23	120,42
	5	1188,22	178,32
	6	3053,48	225,33
	7	3837,15	323,94
	8	6372,82	229,33
	9	7302,92	402,43

Fazendo-se a variação de saltos em 1, 2, 3, 4, 5, 6, 7, 8 e 9 nós sensores, foi levantada a taxa de entrega de pacotes em porcentagem, para cada um dos três algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Os resultados são apresentados nas Figuras 74 e 75.

Figura 74 – Cenário 4 –Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 75 – Cenário 4 –Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.4.2. Discussões

O principal objetivo deste cenário apresentado é analisar o comportamento do protocolo HOFACO com relação aos outros protocolos OF0 e ETX com relação ao número de saltos.

A partir dos gráficos das Figuras 72 e 73, é possível verificar com um tráfego pequeno (10 bytes/segundo), há uma equivalência entre os protocolos em termos da latência da rede com relação ao número de saltos. Neste cenário específico foram testados até 9 saltos.

O mesmo foi observado termos de equivalência quando se aumento o tráfego para uma taxa de 50 bytes/segundo.

A análise da taxa de entrega de pacotes com relação ao número de saltos, mostra que para tráfegos menores, há também uma equivalência para os protocolos neste cenário. Esta equivalência também foi observada para um tráfego maior, conforme as Figuras 74 e 75.

Este fenômeno em termos de taxa de entrega também foi observado nos cenários anteriores, onde o número de saltos era menor. Esta característica apresenta evidências de que este protocolo proposto também pode ser usado para um número maior de saltos.

Dessa forma, podemos concluir que para um número de até 9 saltos, o comportamento do protocolo HOFACO é equivalente aos demais avaliados (OF0 e ETX) em termos de latência e taxa de entrega.

6.5. Cenário 5 – Topologia com 10 nós sensores – Simulação com *OpenVisualizer* – *OpenWSN*

No cenário 5, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente. A Figura 76 ilustra este cenário.

Figura 76 – Cenário 5 – Topologia com 10 nós (1 *sink* e 9 *motes*)

Para este cenário, os nós sensores foram distribuídos em um mesmo plano, de acordo com as distâncias destacadas na Figura 76.

A representação na interface de simulação é apresentada na Figura 77.

Figura 77 - Cenário 5 – *OpenVisualizer Web*

Os nós sensores utilizados são nós virtualizados por meio da interface concebida em linguagem Python, e de maneira distinta do *COOJA Contiki* – abordado nos cenários anteriores – não há a opção de configuração do meio de propagação.

Os firmwares utilizados nos nós de 2 a 10 consistem no firmware padrão do *OpenWSN*. A única alteração feita foi para o nó 1, através da alteração de uma diretiva para que ele funcionasse como nó *sink* (DAG root).

6.5.1. Itens analisados e Resultados

Neste cenário 5, foram executadas 2 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada na função ETX, que usa a métrica expectativa de transmissão;
- b) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

Neste cenário de simulação com o *OpenWSN* não foi utilizada a função objetivo OF0 – baseada em número de saltos. Isto ocorreu porque a implementação padrão original do *OpenWSN* faz uso apenas da métrica ETX para a composição da função objetivo.

Assim, como nos cenários anteriores, o tempo utilizado para cada uma dessas rodadas foi de 5 minutos, suficiente para a estabilização da rede.

Os itens analisados para cada uma das rodadas neste cenário 5 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Número de alterações de nós *parent*.
- 3) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 5) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A seguir, a Figura 78 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 78 – Cenário 5 – Número de mensagens de Controle RPL

A Figura 79, a seguir apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos dois algoritmos.

Figura 79 – Cenário 5 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2, 3 e 4 é apresentada nas Figuras 80 e 81. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 80 – Cenário 5 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 15, a seguir apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 e 4 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 80.

Tabela 15 – Cenário 5 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
ETX	1	85,23	9,38
	2	97,69	22,12
	3	207,79	40,14
	4	354,32	50,34
HOFACO	1	83,59	9,67
	2	103,29	19,13
	3	180,23	38,24
	4	290,29	46,28

Figura 81 – Cenário 5 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 16, a seguir apresenta dos dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 e 4 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 81.

Tabela 16 – Cenário 5 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
ETX	1	855,59	43,65
	2	979,69	52,12
	3	2007,79	140,92
	4	2804,32	150,46
HOFACO	1	1255,91	39,53
	2	1079,61	48,23
	3	1807,72	120,74
	4	2804,58	130,89

Fazendo-se a variação de saltos em 1, 2, 3 e 4 nós, foi levantada a taxa de entrega de pacotes em porcentagem para cada um dos dois algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 82 e 83.

Figura 82 – Cenário 5 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 83 – Cenário 5 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.5.1. Discussões

A análise das mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 78, mostra uma menor quantidade de mensagens DIO e DIS no protocolo HOFACO que no protocolo ETX e a quantidade de mensagens DAO são equivalentes.

Com relação a alterações de nós *parent*, mostrados na figura 79, ocorreram 3 para o protocolo HOFACO e 5 para o protocolo ETX.

Os gráficos de latência em função do número de saltos com um tráfego pequeno (10 bytes/segundo) também são praticamente equivalentes para os 2 primeiros saltos e o protocolo HOFACO obteve uma melhor resposta para 3 e 4 saltos, conforme a Figura 80.

Ao provocarmos maior tráfego na rede (50 bytes/segundo), é possível notar uma maior latência, para o primeiro salto no protocolo HOFACO, mas praticamente os dois protocolos se comportam de maneira equivalente, conforme a Figura 81.

A análise da taxa de entrega de pacotes com relação ao número de saltos, mostra a equivalência destes protocolos tanto para tráfegos menores (10 bytes/segundo) com para tráfego um pouco maiores (50 bytes/segundo).

Para este cenário é possível concluir a equivalência entre os protocolos, com uma leve vantagem do protocolo HOFACO com relação ao número de mensagens de controle do RPL.

6.6. Cenário 6 – Topologia com 10 nós sensores (hardware real) – Plataforma *OpenWSN*

No cenário 6, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente.

Para este cenário, os nós sensores foram distribuídos em um mesmo plano, de acordo com as distâncias destacadas na Figura 84.

Figura 84 – Cenário 6 – Topologia com 10 nós (1 *sink* e 9 *motes*)

Os nós sensores utilizados são baseados na placa eletrônica desenvolvida, conforme mostra a Figura 27.

Os firmwares utilizados nos nós de 2 a 10 consistem no firmware padrão do *OpenWSN*. A única alteração feita foi para o nó 1, através da alteração de uma diretiva para que ele funcionasse como nó *sink* (DAG *root*).

6.6.1. Itens analisados e Resultados

Neste cenário 6, foram executadas 2 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada na função ETX, que usa a métrica expectativa de transmissão;
- b) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

Assim, como descrito no cenário anterior, a plataforma *OpenWSN* não utiliza a função objetivo OF0 – baseada em número de saltos. A implementação padrão original do *OpenWSN* faz uso apenas da métrica ETX para a composição da função objetivo.

Da mesma maneira que nos cenários anteriores, o tempo utilizado para cada uma dessas rodadas foi de 5 minutos, suficiente para a estabilização da rede.

Os itens analisados para cada uma das rodadas neste cenário 6 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Número de alterações de nós *parent*.
- 3) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 5) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.

A seguir, a Figura 85 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 85 – Cenário 6 – Número de mensagens de Controle RPL

A Figura 86, a seguir apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos dois algoritmos.

Figura 86 – Cenário 6 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2, 3 e 4 é apresentada nas Figuras 87 e 88. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente.

Figura 87 – Cenário 6 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 17, a seguir apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 e 4 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 87 acima.

Tabela 17 – Cenário 6 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
ETX	1	125,39	13,16
	2	197,69	22,32
	3	307,29	44,79
	4	454,32	53,13
HOFACO	1	155,13	13,61
	2	187,55	18,12
	3	287,92	34,14
	4	414,19	43,34

Figura 88 – Cenário 6 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 18, a seguir apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2, 3 e 4 saltos.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 88 acima.

Tabela 18 – Cenário 6 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
ETX	1	1355,16	53,16
	2	1472,26	62,22
	3	2602,71	160,34
	4	3404,37	170,13
HOFACO	1	1105,93	33,67
	2	1139,61	48,29
	3	2707,78	112,24
	4	3204,29	113,13

Fazendo-se a variação de saltos em 1, 2, 3 e 4 nós, foi levantada a taxa de entrega de pacotes em porcentagem para cada um dos dois algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 89 e 90.

Figura 89 – Cenário 6 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 90 – Cenário 6 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.6.2. Discussões

Este cenário contemplou o uso de sensores físicos reais – foram utilizadas as placas eletrônicas desenvolvidas, conforme Figura 27.

A análise das mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 85, mostra uma menor quantidade de mensagens DIO e DIS no protocolo HOFACO e uma maior quantidade de mensagens DAO, quando comparado ao protocolo ETX.

Com relação a alterações de nós *parent*, mostrados na figura 86, ocorreram 2 para o protocolo HOFACO e 3 para o protocolo ETX.

Os gráficos de latência em função do número de saltos são equivalentes tanto com um tráfego pequeno (10 bytes/segundo) quanto com um tráfego maior (50 bytes/segundo).

Por fim, a análise da taxa de entrega de pacotes com relação ao número de saltos, ressalta mais uma vez que estes protocolos se comportam de maneira equivalente.

Para este cenário é possível concluir a equivalência entre os protocolos, com uma leve vantagem do protocolo HOFACO com relação ao número de mensagens de controle do RPL.

Este comportamento já era esperado, pois foi também verificado no ambiente de simulação.

6.7. Cenário 7 – Topologia com 10 nós sensores (hardware real) – Plataforma Contiki

No cenário 7, construiu-se uma topologia baseada em 10 nós sensores (1 *sink* e 9 *motes*) de tal forma a observar as características de roteamento descritas anteriormente.

Para este cenário, os nós sensores foram distribuídos em um mesmo plano, de acordo com as distâncias destacadas na Figura 91.

Figura 91 – Cenário 7 – Topologia com 10 nós (1 *sink* e 9 *motes*)

Os nós sensores utilizados são baseados na placa eletrônica desenvolvida, conforme mostra a Figura 27.

Os firmwares utilizados nos nós de 2 a 10 foram o *websensor.c*. Para o nó 1 foi utilizado o firmware *rpl-border-router.c*. Os firmwares originais foram modificados para contemplar as implementações realizadas.

6.7.1. Itens analisados e Resultados

Neste cenário 7, foram executadas 3 rodadas de simulação com um intervalo de confiança de 95%, cada uma fazendo o uso distinto de uma função objetivo:

- a) Baseada somente na função OF0 (*Objective Function Zero*), que usa número de saltos;
- b) Baseada na função MRHOF (*Minimum Rank with Hysteresis Objective Function*), que usa a métrica ETX (expectativa de transmissão);
- c) Baseada na função HOFACO (*Hybrid Objective Function based on Ant Colony Optimization*) proposta.

Assim, como nos cenários anteriores, o tempo utilizado para cada rodada de testes foi de 5 minutos.

Os itens analisados para cada uma das rodadas neste cenário 7 foram:

- 1) Número de mensagens de controle do RPL (DIO, DAO e DIS).
- 2) Tempo de Intervalo do envio de mensagens DIO (acompanhamento do *Trickle Timer*).
- 3) Número de alterações de nós *parent*.
- 4) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 5) Latência da rede em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 6) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 10 bytes/segundo.
- 7) Taxa de entrega de pacotes em função do número de saltos, com a presença de um tráfego de comunicação da ordem de 50 bytes/segundo.
- 8) Avaliação do consumo de energia por meio do uso do *powertrace*.

A seguir, a Figura 92 apresenta uma comparação da quantidade total de mensagens de controle do protocolo RPL.

Figura 92 – Cenário 7 – Número de mensagens de Controle RPL

As Figuras de 93 a 95 apresentam a evolução do intervalo de envio de mensagens DIO, controlada pelo algoritmo *Trickle Timer*, que está diretamente relacionado à estabilização da rede.

Figura 93 – Cenário 7 – Evolução do intervalo do envio de mensagens DIO (*Trickle Timer*) para a Função Objetivo OF0

Figura 94 – Cenário 7 – Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo ETX

Figura 95 – Cenário 7 – Evolução do intervalo do envio de mensagens
DIO (*Trickle Timer*) para a Função Objetivo HOFACO

A Figura 96 apresenta uma comparação entre a quantidade de alterações de nós *parent* para cada um dos três algoritmos.

Figura 96 – Cenário 7 – Quantidade de alterações de nós *parent*

A latência de comunicação variando-se o número de saltos em 1, 2, 3 e 4 é apresentada nas Figuras 97 e 98. Foram utilizadas as taxas de 10 bytes/segundo e 50 bytes/segundo, respectivamente

Figura 97 – Cenário 7 – Latência da rede em função do número de saltos com tráfego de 10 bytes/segundo

A tabela 19, a seguir apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 10 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2 e 3.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 97.

Tabela 19 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 10 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	132,87	38,23
	2	192,13	59,24
	3	521,89	78,31
ETX	1	171,78	98,33
	2	324,94	86,21
	3	693,16	103,23
HOFACO	1	167,13	79,24
	2	202,98	110,15
	3	482,25	142,87

Figura 98 – Cenário 7 – Latência da rede em função do número de saltos com tráfego de 50 bytes/segundo

A tabela 20, a seguir apresenta os dados estatísticos resultantes do envio de 100 pacotes a uma taxa de 50 bytes/segundo.

Foram feitas variações no número de saltos contemplando 1, 2 e 3.

Os resultados obtidos foram utilizados para a composição do gráfico demonstrado na Figura 98.

Tabela 20 – Estatística do envio de 100 pacotes – 1 pacote por segundo de tamanho 50 bytes

Função Objetivo	Número de saltos	Média (ms)	Desvio Padrão
OF0	1	824,13	134,23
	2	1303,18	192,42
	3	1529,13	297,16
ETX	1	1015,32	96,24
	2	1237,63	145,98
	3	1712,11	201,31
HOFACO	1	681,45	82,24
	2	987,41	108,16
	3	1522,78	254,13

Fazendo-se a variação de saltos em 1, 2 e 3 nós sensores, foi levantada a taxa de entrega de pacotes em porcentagem, para cada um dos três algoritmos, utilizando-se taxas de comunicação de 10 bytes/segundo e 50 bytes/segundo, respectivamente. Os resultados são apresentados nas Figuras 99 e 100.

Figura 99 – Cenário 7 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 10 bytes/segundo

Figura 100 – Cenário 7 – Taxa de entrega de pacotes em função do número de saltos com tráfego de 50 bytes/segundo

6.7.2. Avaliação do consumo de energia

Neste cenário 7, com 10 sensores físicos, foi também realizada uma captura de consumo de energia, feita pelo próprio firmware de cada um dos nós, por meio do uso da ferramenta *powertrace* do Contiki – apresentada no capítulo 5.

Foram realizados 3 testes: um para cada um dos protocolos envolvidos, de duração de 5 minutos com a presença de um tráfego de dados de 50 bytes/segundo.

Os nós selecionados para contabilização de energia foram os sensores de 2 a 10, os quais passam pelos quatro estados:

- a) CPU em modo de execução;
- b) CPU em modo de economia de energia;
- c) Rádio em modo TX (transmitindo mensagens);
- d) Rádio em modo RX (à espera de mensagens).

Devido ao seu papel de nó *sink*, o sensor 1 possui uma configuração de firmware na qual o seu rádio fica ligado em modo de escuta em praticamente 100% do tempo. Dessa forma, para nenhum dos protocolos ele participou da contabilização de energia.

O mecanismo de funcionamento do *powertrace* é realizado por meio do conceito de cápsulas de energia, na qual são contabilizados em *ticks* a quantidade de tempo em que o nó sensor permaneceu em cada estado.

A partir da folha de dados do nó sensor utilizado CC2538 (2013), usando-se uma tensão de 3.3 V e com os valores típicos de consumo de corrente para estes estados, foram obtidos os seguintes resultados, apresentados no gráfico da Figura 101.

Figura 101 – Cenário 7 – Consumo de energia acumulado x tempo

Durante os testes o consumo de energia contabilizado foi o seguinte:

- a) Protocolo OF0 = 403,06 mJ
- b) Protocolo ETX = 384,89 mJ
- c) Protocolo HOFACO = 318,12 mJ

6.7.3. Discussões

Este cenário, assim como o anterior, contemplou o uso de sensores físicos reais – foram utilizadas as placas eletrônicas desenvolvidas, conforme Figura 27.

A análise das mensagens de controle enviadas (DIO, DAO e DIS), representadas na Figura 92, mostra uma menor quantidade de mensagens DIO e DAO no protocolo HOFACO do que nos outros protocolos. Neste cenário, o protocolo que menos enviou mensagens DIS foi o ETX. Este mesmo comportamento foi observado em ambiente de simulação.

A verificação da evolução do *Trickle Timer*, apresentada nos gráficos das Figuras 93, 94 e 95 mostra que a estabilidade foi atingida na seguinte ordem: a) em aproximadamente 120 segundos para protocolo HOFACO; b) em aproximadamente 130 segundos para o protocolo OF0 e c) em aproximadamente 130 segundos para o protocolo ETX. Isso representa uma melhoria em torno de 7,6% em termos de redução do tempo de estabilização da rede, comparando-se o protocolo HOFACO com os demais.

Com relação a alterações de nós *parent*, mostrados na figura 101, ocorreram 2 para o protocolo OF0, 3 para o HOFACO e 4 para o protocolo ETX.

Os gráficos de latência em função do número de saltos com um tráfego pequeno (10 bytes/segundo) também são equivalentes para o primeiro salto, mas apresentam um maior valor para o protocolo ETX do que os demais, com o aumento do número de saltos, conforme a Figura 96.

Ao provocarmos maior tráfego na rede (50 bytes/segundo), é possível notar uma maior latência, mas praticamente os três protocolos se comportam de maneira equivalente.

A análise da taxa de entrega de pacotes com relação ao número de saltos, mostra que para tráfegos menores, há uma melhor performance equivalente entre os protocolos OF0 e HOFACO com uma tendência de decaimento maior do protocolo HOFACO com relação aos demais à medida que se aumenta o número de saltos.

Quando se aumenta o tráfego na rede, a relação da taxa de entrega em função do número de saltos mostra que os três protocolos se comportam de maneira equivalente, para um pequeno número de saltos e o protocolo HOFACO se sobressai em relação aos demais ao se aumentar o número de saltos.

Considerando-se o gráfico apresentado na Figura 101, obtido a partir da ferramenta *powertrace*, o consumo de energia do protocolo HOFACO foi menor tanto em relação ao protocolo ETX quanto em relação ao protocolo OF0.

Em termos percentuais, foi possível verificar que o protocolo HOFACO apresenta uma melhoria, em termos de eficiência energética em torno de 17,4% quando comparado com o ETX e em torno de 20,9 % quando comparado com o OF0.

Estes resultados comprovam e validam a hipótese apresentada no capítulo 5.

Ainda para efeito de validação estatística, foi conduzido um teste de verificação de normalidade para os dados de consumo de energia adquiridos a partir do *powertrace*.

Foi feito um teste de Shapiro-Wilk e encontrou-se os seguintes valores p-normal:

- a) OF0 – p-normal: 0,17
- b) ETX – p-normal: 0,14
- c) HOFACO – p-normal: 0,25

Confirmada a normalidade dos valores, prosseguiu-se com uma análise de variância, seguida de um teste de Tukey com nível de confiança de 95%, onde foi possível confirmar que os três protocolos são similares e o protocolo HOFACO apresenta comportamento mais próximo de ETX do que OF0.

Sendo assim, é possível concluir que o protocolo HOFACO proposto foi validado e apresenta uma melhoria em termos de eficiência energética quando comparado aos tradicionais protocolos OF0 e ETX.

6.8. Análise Geral dos Cenários de utilização

Os resultados obtidos a partir dos cenários de testes apresentados demonstraram uma viabilidade da utilização do protocolo HOFACO proposto, evidenciando uma característica similar aos demais protocolos tradicionais OF0 e ETX, sendo observado uma melhoria no tempo de estabilização da rede.

Esta melhoria se encontra na redução do tempo de estabilização do *Trickle Timer*. Os cenários de testes que permitiram esta observação foram os que utilizaram o *Contiki* como plataforma de validação.

A plataforma *OpenWSN* não possui esta implementação de *Trickle Timer*, utilizando, ao invés um valor de tempo fixo.

Em ambas plataformas, foi possível evidenciar um comportamento equivalente em termos de número de mensagens, latência de rede e taxa de entrega de pacotes.

Com relação ao número de saltos, o cenário 4 demonstrou a viabilidade de utilização do protocolo tendo em vista um número maior de saltos.

O cenário 3 apresenta uma condição com um grande número de vizinhos onde foi possível confirmar as operações de *reset* do *Trickle Timer* no modo ETX, o que não aconteceu no protocolo HOFACO proposto.

Uma vez que o protocolo proposto utiliza as interfaces já consolidadas pelo RPL para redes com baixas taxas de transmissão, várias intermitências e compatível com redes maiores, a proposta também pode ser estendida para redes com um número maior de nós.

Os testes utilizados não fizeram uso de simuladores simplesmente virtualizados. Ao invés, foram escolhidas plataformas compatíveis com o *deployment* em dispositivos reais físicos – as plataformas Contiki e OpenWSN – trazendo uma maior confiabilidade do ponto de vista de execução em ambientes reais, os quais foram verificados por meio dos resultados apresentados.

Portanto, estes cenários validam a proposta apresentada.

CAPÍTULO 7

7. CONCLUSÕES

Nesta tese, foi proposto um novo protocolo de roteamento, o HOFACO, com base no RPL, para a economia de energia. Para isto foram feitas implementações com simulações e dispositivos reais nas plataformas *Contiki* e *OpenWSN*.

O processo de verificação e validação utilizou uma comparação deste protocolo com outros tradicionais tomando-se como métricas os parâmetros: número de mensagens de controle (DIO, DAO e DIS), alterações de configurações na topologia, latência de rede e taxa de entrega de pacotes comparando-se o modelo proposto de função objetiva híbrida com modelos tradicionais cujas funções objetivo são baseadas em número de saltos e expectativa de transmissão.

Os resultados obtidos demonstraram a equivalência deste novo protocolo com os protocolos tradicionais com relação às métricas avaliadas.

A principal diferença ocorreu por meio de uma melhoria na redução do tempo de estabilização da rede, em torno de 7,6%, refletida por meio do acompanhamento do *Trickle Timer*, fazendo com que a rede chegasse a um estado de estabilidade em um tempo menor, quando do uso do protocolo proposto.

A estratégia utilizada tomou como base uma estabilização do funcionamento do RPL, a qual foi verificada por meio do acompanhamento do *Trickle Timer* e das alterações de configuração de topologia demonstrados pelas alterações de nós *parent* durante os testes realizados com a plataforma *Contiki*.

Na plataforma *OpenWSN* o acompanhamento da estabilização do RPL foi feito por meio das alterações de topologia (alterações de nós *parent*), uma vez que o *OpenWSN* utiliza o *Trickle Timer* de maneira fixa.

A principal contribuição desta tese se encontra na proposta de uma modificação na função objetivo de tal forma a se compor uma métrica combinada para avaliação do *rank*. Esta

métrica usa como princípio básico a informação de um caminho eficaz a partir de dois agentes: *forward ant* e *backward ant*, os quais são responsáveis por realizar o depósito de feromônio em comunicações realizadas com sucesso.

De maneira distinta de outros trabalhos, a presente proposta não exige o envio de mensagens adicionais de controle para o funcionamento do algoritmo e preserva as interfaces de comunicação entre as camadas, permitindo a compatibilidade de funcionamento com outros nós que não estejam usando o algoritmo.

O mecanismo de estimativa de energia utilizado – o *powertrace* – disponível apenas na plataforma *Contiki*, possibilitou a verificação da economia de energia quando comparado a outros padrões como função objetivo do RPL baseada em apenas número de saltos (OF0) ou expectativa de transmissão (ETX).

Adicionalmente, foram conduzidos testes de consumo de energia comparando-se o protocolo HOFACO proposto com os outros protocolos tradicionais usando-se uma rede de sensores física real. Os resultados obtidos em termos percentuais demonstraram uma melhoria de eficiência energética em torno de 17,4% quando comparado com o ETX e em torno de 20,9% quando comparado com o OF0.

Finalmente, o protocolo desenvolvido nesta tese é adequado para ser utilizado em cenários reais para uma solução econômica de energia.

Trabalhos Futuros

Nesta seção são apresentados itens relevantes para continuidade desta pesquisa.

Durante o desenvolvimento desta pesquisa, optou-se por utilizar um mecanismo de roteamento baseado em protocolos padrões para redes de sensores sem fio como o RPL (*Routing Protocol for Low Power and Lossy Networks*), tomando-se como premissa não haver quebra de compatibilidade de interfaces com arquiteturas já consolidadas.

O HOFACO faz uso do funcionamento do RPL na avaliação do *rank* da função objetivo adicionando um mecanismo híbrido baseado em colônia de formigas (*Ant Colony Optimization* – ACO). Seguindo um padrão semelhante, novas abordagens podem ser desenvolvidos a partir de outros sistemas bioinspirados como, por exemplo: colônia de abelhas artificiais (*Artificial Bee Colony* – ABC) e colônia de vaga-lumes (*Firefly Algorithm* – FA).

Da mesma maneira que o HOFACO, estas novas abordagens que preservam a compatibilidade de interfaces podem obter o benefício da convivência entre nós sensores que não possuam o algoritmo de roteamento proposto.

Nas plataformas utilizadas – *Contiki* e *OpenWSN* – o padrão de rede sem fio utilizado é o IEEE 802.15.4e. Como o RPL atua na camada de rede, a presente proposta pode ser estendida para outros tipos de rede como, por exemplo *LoRaWAN* (*Long Range Wide Area Network*).

O mecanismo utilizado para avaliação do consumo de energia – *powertrace* – está disponível apenas na plataforma *Contiki*. Novos trabalhos poderão fazer uso destes conceitos para a sua implementação e aplicação em outras plataformas, como, por exemplo, o *OpenWSN*.

Novos estudos poderão, portanto, ser conduzidos no sentido de utilização dessa arquitetura de roteamento para melhoria de outros requisitos aderentes a novos cenários de utilização das redes de sensores sem fio.

8. REFERÊNCIAS BIBLIOGRÁFICAS

- ABBAS, Zeeshan; YOON, Wonyong. A survey on energy conserving mechanisms for the internet of things: Wireless networking aspects. **Sensors**, v. 15, n. 10, p. 24818-24847, 2015.
- ABDULRAHMAN, S. M. Using swarm intelligence for solving np-hard problems. **Academic Journal of Nawroz University**, 2017. v. 6, n. 3, p. 46–50, 2017.
- ABUEIN, Q. Q. et al. Performance evaluation of routing protocol (rpl) for internet of things. **Performance Evaluation**, 2016. v. 7, n. 7, 2016.
- AFONSO, J. A. et al. An enhanced reservation-based mac protocol for ieee 802.15. 4 networks. **Sensors**, 2011. Molecular Diversity Preservation International, v. 11, n. 4, p. 3852–3873, 2011.
- AHMAD, Aasem; HANZÁLEK, Zdeněk. An energy efficient schedule for IEEE 802.15. 4/zigbee cluster tree WSN with multiple collision domains and period crossing constraint. **IEEE Transactions on Industrial Informatics**, v. 14, n. 1, p. 12-23, 2018.
- ÅKERBERG, J.; GIDLUND, M.; BJÖRKMAN, M. Future research challenges in wireless sensor and actuator networks targeting industrial automation. In: IEEE. **Industrial Informatics (INDIN), 2011 9th IEEE International Conference on**. [S.l.], 2011. p. 410–415.
- ALBERT, Réka; BAILLIEUL, John; MOTTER, Adilson E. Introduction to the Special Issue on Approaches to Control Biological and Biologically Inspired Networks. **IEEE Transactions on Control of Network Systems**, v. 5, n. 2, p. 690-693, 2018.
- AL-JEMELI, Marwan; HUSSIN, Fawnizu A. An energy efficient cross-layer network operation model for IEEE 802.15. 4-based mobile wireless sensor networks. **IEEE sensors journal**, v. 15, n. 2, p. 684-692, 2015.
- ALJARRAH, E.; YASSEIN, M. B.; ALJAWARNEH, S. Routing protocol of low-power and lossy network: Survey and open issues. In: IEEE. **Engineering & MIS (ICEMIS), International Conference on**. [S.l.], 2016. p. 1–6.
- ALVI, S. A.; HASSAN, F. ul; MIAN, A. N. On the energy efficiency and stability of rpl routing protocol. In: IEEE. **Wireless Communications and Mobile Computing Conference (IWCMC), 2017 13th International**. [S.l.], 2017. p. 1927–1932.
- ANTUNES, V. B. **Roteamento Sensível ao Contexto em Redes de Sensores sem Fio: Uma abordagem baseada em Regras de Aplicação para o Protocolo RPL**. Dissertação (Mestrado), 2014.
- AYAZ, Muhammad et al. Wireless Sensor's Civil Applications, Prototypes, and Future Integration Possibilities: A Review. **IEEE Sensors Journal**, v. 18, n. 1, p. 4-30, 2017.
- AZNOLI, Fariba; NAVIMIPOUR, Nima Jafari. Deployment strategies in the wireless sensor networks: systematic literature review, classification, and current trends. **Wireless Personal Communications**, v. 95, n. 2, p. 819-846, 2017.

BARBATO, A. et al. Resource oriented and energy efficient routing protocol for ipv6 wireless sensor networks. In: IEEE. **Online Conference on Green Communications (GreenCom), 2013 IEEE**. [S.l.], 2013. p. 163–168.

BYUN, Heejung; SON, Sugook; YANG, Soomi. Biologically inspired node scheduling control for wireless sensor networks. **Journal of Communications and Networks**, v. 17, n. 5, p. 506–516, 2015.

CC2538 Datasheet (2013). **A Powerful System-On-Chip for 2.4-GHz IEEE 802.15.4, 6LoWPAN and ZigBee Applications**. Disponível em: <http://www.ti.com/lit/ds/symlink/cc2538.pdf>>. Acesso em: 4 maio. 2018.

CHHABRA, Shikha; ARORA, Vishal. A review on General Self-Organized tree-based Energy-balance routing Protocol for Wireless Sensor Network. **INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY**, v. 16, n. 2, p. 7591–7595, 2017.

COBARZAN, C.; MONTAVONT, J.; NOEL, T. Analysis and performance evaluation of rpl under mobility. In: IEEE. **Computers and Communication (ISCC), 2014 IEEE Symposium on**. [S.l.], 2014. p. 1–6.

DE GUGLIELMO, Domenico; ANASTASI, Giuseppe; SEGHETTI, Alessio. From iee 802.15. 4 to iee 802.15. 4e: A step towards the internet of things. In: **Advances onto the Internet of Things**. Springer, Cham, 2014. p. 135–152.

DICKINSON, P. J.; GIJEN, L. On the computational complexity of membership problems for the completely positive cone and its dual. **Computational optimization and applications**, 2014. Springer, v. 57, n. 2, p. 403–415, 2014.

DORIGO, Marco. Swarm Intelligence: 11th **International Conference, ANTS 2018**, Rome, Italy, October 29–31, 2018, Proceedings. Springer.

DOYLE, J. C.; FRANCIS, B. A.; TANNENBAUM, A. R. **Feedback control theory**. [S.l.]: Courier Corporation, 2013.

DUNKELS, A. et al. **Powertrace: Network-level power profiling for low-power wireless networks**. [S.l.]: Swedish Institute of Computer Science, 2011.

ERTENLICE, O.; KALAYCI, C. B. A survey of swarm intelligence for portfolio optimization: Algorithms and applications. **Swarm and evolutionary computation**, 2018. Elsevier, v. 39, p. 36–52, 2018.

FAROOQ, M. O.; KUNZ, T. Operating systems for wireless sensor networks: A survey. **Sensors**, 2011. Molecular Diversity Preservation International, v. 11, n. 6, p. 5900–5930, 2011.

FOTOUHI, H.; MOREIRA, D.; ALVES, M. mrpl: Boosting mobility in the internet of things. **Ad Hoc Networks**, 2015. Elsevier, v. 26, p. 17–35, 2015.

GADDOUR, O.; KOUBÂA, A. Rpl in a nutshell: A survey. **Computer Networks**, 2012. Elsevier, v. 56, n. 14, p. 3163–3178, 2012.

GAO, Cai et al. A bio-inspired algorithm for route selection in wireless sensor networks. **IEEE Communications Letters**, v. 18, n. 11, p. 2019-2022, 2014.

GARCIA, Juliette; PIROVANO, Alain; ROYER, Mickaël. Routing in Wireless Sensor Networks for Surveillance of Airport Surface Area. In: Nets4, **13th International Workshop on Communication Technologies for Vehicles**, Nets4Aircraft and UAV session. 2018.

GHATAOURA, Darminder Singh; YANG, Yang; MATICH, George. GAFO: genetic adaptive fuzzy hop selection scheme for wireless sensor networks. In: **Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly**. ACM, 2009. p. 376-380.

GEETHA, B. T. et al. A study of different routing protocols and encryption algorithms in wireless networks. In: 2017 **IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)**. IEEE, 2017. p. 1213-1217.

GHARGE, S.; VALANJOO, A. Simulation based performance evaluation of tcp variants and routing protocols in mobile ad-hoc networks. In: **Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on**. [S.l.: s.n.], 2014. p. 1-8. ISSN 2347-9337.

GHATAOURA, D. S.; YANG, Y.; MATICH, G. Gafo: genetic adaptive fuzzy hop selection scheme for wireless sensor networks. In: ACM. **Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly**. [S.l.], 2009. p. 376-380.

GUO, Wenqi; HEALY, William M.; ZHOU, MengChu. Impacts of 2.4-GHz ISM band interference on IEEE 802.15. 4 wireless sensor network reliability in buildings. **IEEE Transactions on Instrumentation and Measurement**, v. 61, n. 9, p. 2533-2544, 2012.

HASAN, Mohammed Zaki; AL-RIZZO, Hussain; AL-TURJMAN, Fadi. A survey on multipath routing protocols for QoS assurances in real-time wireless multimedia sensor networks. **IEEE Communications Surveys & Tutorials**, v. 19, n. 3, p. 1424-1456, 2017.

IEEE 802.15.4 (2006). **Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)**, IEEE Computer Society, Revision of IEEE Std 802.15.4-2003, 2006.

IEEE802.15.4e. "IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer." IEEE Std 802.15.4e-2012 (**Amendment to IEEE Std 802.15.4-2011**), 4 2012: 1-225.

INTANA, A.; POPPLETON, M. R.; MERRETT, G. V. Adding value to wsn simulation through formal modelling and analysis. In: IEEE. **Software Engineering for Sensor Network Applications (SESENA), 2013 4th International Workshop on**. [S.l.], 2013. p. 24-29.

IOVA, O.; THEOLEYRE, F.; NOEL, T. Stability and efficiency of rpl under realistic conditions in wireless sensor networks. In: IEEE. **Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on**. [S.l.], 2013. p. 2098-2102.

IOVA, Oana et al. The Love-Hate Relationship between IEEE 802.15. 4 and RPL. **IEEE Communications Magazine**, v. 55, n. 1, p. 188-194, 2017.

IQBAL, Muhamad S.; AL-RAWESHIDY, Hamed S. Performance evaluation of IEEE 802.15.4 standard for low data rate ad hoc wireless sensor networks. In: **Control, Automation and Information Sciences (ICCAIS), 2013 International Conference on**. IEEE, 2013. p. 300-304.

KAMGUEU, P. O. et al. Fuzzy-based routing metrics combination for rpl. 2014. 2014.

KAMGUEU, P.-O.; NATAF, E.; DJOTIO, T. N. On design and deployment of fuzzy-based metric for routing in low-power and lossy networks. In: IEEE. **Local Computer Networks Conference Workshops (LCN Workshops), 2015 IEEE 40th**. [S.l.], 2015. p. 789–795.

KAR, Arpan Kumar. Bio inspired computing—A review of algorithms and scope of applications. **Expert Systems with Applications**, v. 59, p. 20-32, 2016.

KAZANDJIEVA, M. et al. Green enterprise computing data: Assumptions and realities. In: IEEE. **Green Computing Conference (IGCC), 2012 International**. [S.l.], 2012. p. 1–10.

KAZIENKO, Juliano F. et al. SENSORLock: a lightweight key management scheme for wireless sensor networks. **Security and Communication Networks**, v. 6, n. 10, p. 1198-1210, 2013.

KAZIENKO, Juliano F. et al. On the performance of a secure storage mechanism for key distribution architectures in wireless sensor networks. **International Journal of Distributed Sensor Networks**, v. 11, n. 5, p. 392495, 2015.

KHANAFER, Mounib; GUENNOUN, Mouhcine; MOUFTAH, Hussein T. A survey of beacon-enabled IEEE 802.15. 4 MAC protocols in wireless sensor networks. **IEEE Communications Surveys & Tutorials**, v. 16, n. 2, p. 856-876, 2014.

KO, J.; CHANG, M. Momoro: Providing mobility support for low-power wireless applications. **IEEE Systems Journal**, 2015. v. 9, n. 2, p. 585–594, 2015.

LEE, T.-H.; XIE, X.-S.; CHANG, L.-H. Rssi-based ipv6 routing metrics for rpl in low-power and lossy networks. In: IEEE. **Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on**. [S.l.], 2014. p. 1714–1719.

LEVIS, P. et al. **The trickle algorithm**. [S.l.], 2011.

LI, Y. et al. Research based on osi model. In: **Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on**. [S.l.: s.n.], 2011. p. 554–557.

LIAW, Rung-Tzuo; CHANG, Yu-Wei; TING, Chuan-Kang. Solving the selective pickup and delivery problem using max-min ant system. In: **International Conference in Swarm Intelligence**. Springer, Cham, 2017. p. 293-300.

LILLACCI, Gabriele et al. A synthetic integral feedback controller for robust tunable regulation in bacteria. **bioRxiv**, p. 170951, 2017.

LIU, L. et al. Schedule dynamic multiple parallel jobs with precedence-constrained tasks on heterogeneous distributed computing systems. In: **Parallel and Distributed Computing (ISPDC), 2015 14th International Symposium on**. [S.l.: s.n.], 2015. p. 130–137.

LÓPEZ-IBÁÑEZ, Manuel; STÜTZLE, Thomas; DORIGO, Marco. Ant Colony Optimization: A Component-Wise Overview. **Handbook of Heuristics**, p. 371-407, 2018.

MAVROVOUNIOTIS, Michalis; LI, Changhe; YANG, Shengxiang. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. **Swarm and Evolutionary Computation**, v. 33, p. 1-17, 2017.

MCBRIDE, Jack; ARIEF, Budi; HERNANDEZ-CASTRO, Julio. Security Analysis of Contiki IoT Operating System. In: Proceedings of the 2018 **International Conference on Embedded Wireless Systems and Networks**. Junction Publishing, 2018. p. 278-283.

MITTAL, Vaishali et al. LOCATION BASED PROTOCOLS IN WSN: A REVIEW. **IIOAB JOURNAL**, v. 9, p. 67-77, 2018.

MODIEGINYANE, Kgotlaetsile Mathews et al. Software defined wireless sensor networks application opportunities for efficient network management: A survey. **Computers & Electrical Engineering**, v. 66, p. 274-287, 2018.

MORE, Avinash; RAISINGHANI, Vijay. A survey on energy efficient coverage protocols in wireless sensor networks. **Journal of King Saud University-Computer and Information Sciences**, v. 29, n. 4, p. 428-448, 2017.

MUSTARY MR, Nareshkumar R.; PHANIKUMAR, S. Self-Adaptive Intelligent Routing in Dynamic WSN using Natural Inspired Computing. In: **Materials Science and Engineering Conference Series**. 2017. p. 012123.

Ökdem S, Dervi K (2009). “Routing in Wireless Sensor Networks Using an Ant Colony Optimization (ACO) Router Chip”, **Sensors**, 9(2): 909- 921

OVSTHUS, Knut et al. An industrial perspective on wireless sensor networks—a survey of requirements, protocols, and challenges. **IEEE communications surveys & tutorials**, v. 16, n. 3, p. 1391-1412, 2014.

PALATTELLA, M. R. et al. Standardized protocol stack for the internet of (important) things. **Communications Surveys & Tutorials, IEEE**, 2013. IEEE, v. 15, n. 3, p. 1389–1406, 2013.

PARK, J.; KIM, K.-H.; KIM, K. An algorithm for timely transmission of solicitation messages in rpl for energy-efficient node mobility. **Sensors**, 2017. Multidisciplinary Digital Publishing Institute, v. 17, n. 4, p. 899, 2017.

RAO, P. Manohara; RAO, Y. Chalapathi; KUMAR, M. Ashok. Performance analysis of ZigBee wireless sensor networks using Riverbed simulation modeler. In: **2018 2nd International Conference on Inventive Systems and Control (ICISC)**. IEEE, 2018. p. 1272-1277.

RAULT, Tifenn; BOUABDALLAH, Abdelmadjid; CHALLAL, Yacine. Energy efficiency in wireless sensor networks: A top-down survey. **Computer Networks**, v. 67, p. 104-122, 2014.

REDDY, Vijender Busi; NEGI, Atul; VENKATARAMAN, S. A Comparison of Trust in MANETs and WSNs. In: **Advanced Computing (IACC), 2016 IEEE 6th International Conference on**. IEEE, 2016. p. 577-581.

REZAEI, E. **Energy efficient RPL routing protocol in smart buildings**. Dissertação (Mestrado) — University of Waterloo, 2014.

RFC 6550 (2015). RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. Disponível em < <https://datatracker.ietf.org/doc/rfc6550/> >. Acesso em: 30 jan. 2018.

SALEEM, Kashif; FISAL, Norsheila; AL-MUHTADI, Jalal. Empirical studies of bio-inspired self-organized secure autonomous routing protocol. **IEEE Sensors Journal**, v. 14, n. 7, p. 2232-2239, 2014.

S. Guo, Y. Shi, Y. Yang and B. Xiao, **Energy Efficiency Maximization in Mobile Wireless Energy Harvesting Sensor Networks**, in IEEE Transactions on Mobile Computing, vol. 17, no. 7, pp. 1524-1537, July 1 2018.

SANDRA, Sendra et al. Power saving and energy optimization techniques for wireless sensor networks (invited paper). **Journal of communications**, v. 6, n. 6, p. 439-459, 2011.

SHAIKH, Faisal Karim; ZEADALLY, Sherali. Energy harvesting in wireless sensor networks: A comprehensive review. **Renewable and Sustainable Energy Reviews**, v. 55, p. 1041-1054, 2016.

SKINDEROWICZ, Rafał. An improved ant colony system for the sequential ordering problem. **Computers & Operations Research**, v. 86, p. 1-17, 2017.

SOUSA, N. et al. Eraof: A new rpl protocol objective function for internet of things applications. In: IEEE. **Computer and Energy Science (SpliTech), 2017 2nd International Multidisciplinary Conference on**. [S.l.], 2017. p. 1-5.

SYEDALIFATHIMA, K.; SUMITHA, T. To Enhance the Lifetime of WSN Network using PSO. In: **Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)**. 2014.

TABUS, Vlad et al. Energy efficient wireless sensor networks using linear-programming optimization of the communication schedule. **Journal of communications and networks**, v. 17, n. 2, p. 184-197, 2015.

THAPAR, Prateek; BATRA, Usha. **Implementation of Ant Colony Optimization in Routing Protocol for Internet of Things**. In: Innovations in Computational Intelligence. Springer, Singapore, 2018. p. 151-164.

THUBERT, P. Rpl objective function zero. **draft-ietf-roll-of0-20 (work in progress)**, 2011. 2011.

TOKSARI, M. Duran. A hybrid algorithm of ant colony optimization (ACO) and iterated local search (ILS) for estimating electricity domestic consumption: case of Turkey. **International Journal of Electrical Power & Energy Systems**, v. 78, p. 776-782, 2016.

WANG, Cong et al. Combining solar energy harvesting with wireless charging for hybrid wireless sensor networks. **IEEE Transactions on Mobile Computing**, n. 1, p. 1-1, 2018.

WANG, Yong et al. An energy-efficient and swarm intelligence-based routing protocol for next-generation sensor networks. **IEEE Intelligent Systems**, v. 29, n. 5, p. 74-77, 2014.

WATTEYNE, T. et al. From manet to ietf roll standardization: A paradigm shift in wsn routing protocols. **IEEE Communications Surveys and Tutorials**, 2011. v. 13, n. 4, p. 688–707, 2011.

_____. Openwsn: a standards-based low-power wireless development environment. **Transactions on Emerging Telecommunications Technologies**, 2012. Wiley Online Library, 23, n. 5, p. 480–493, 2012.

YAN, C. et al. Rplre: A routing protocol based on lqi and residual energy for wireless sensor networks. In: IEEE. **Information Science and Engineering (ICISE), 2009 1st International Conference on**. [S.l.], 2009. p. 2714–2717.

YANG, X. et al. Stability metric based routing protocol for low-power and lossy networks. In: IEEE. **Communications (ICC), 2014 IEEE International Conference on**. [S.l.], 2014. p. 3688–3693.

Y. Liu and J. Li, **Linear Precoding to Optimize Throughput, Power Consumption and Energy Efficiency in MIMO Wireless Sensor Networks**, in IEEE Transactions on Communications, vol. 66, no. 5, pp. 2122-2136, May 2018.

Zhang Q., Wang X., Huang M., Lv J. (2018) Information-Centric Networking Routing Challenges and Bio/ACO-Inspired Solution: A Review. In: **Tan Y., Shi Y., Tang Q. (eds) Advances in Swarm Intelligence. ICSI 2018**. Lecture Notes in Computer Science, vol 10941. Springer, Cham.

ZHOU, Jun et al. Energy-efficient digital and wireless IC design for wireless smart sensing. **Journal of Semiconductors**, v. 38, n. 10, p. 105005, 2017.

ZORZI, Michele et al. COBANETS: A new paradigm for cognitive communications systems. In: **2016 International Conference on Computing, Networking and Communications (ICNC)**. IEEE, 2016. p. 1-7.

Apêndice A

Arquitetura de Debugging Remoto em Simulações com o Ambiente OpenWSN

Tendo em vista que a arquitetura atual do *OpenWSN* permite o processo de *debugging* por meio do uso de uma interface serial, e essa opção exige a necessidade de uma porta serial para cada nó sensor envolvido, é possível deduzir que o processo de *debug* apresenta um custo: a disponibilidade de tantas portas seriais quantos forem os nós envolvidos na depuração.

Para solução deste problema, o presente trabalho apresentou a seguinte contribuição:

- a) Proposta de uma arquitetura, capaz de contornar o problema da necessidade de várias portas seriais;
- b) Sugestão de um padrão de mensagem de *log/debugging* a ser utilizada pela arquitetura sugerida.
- c) Uso de uma ferramenta padrão – o Wireshark – para análise do *log/debugging*.

A.1 Proposta de Arquitetura

Tomando-se como referência a arquitetura em camadas, com base no padrão IEEE 802.15.4, no qual se baseia o *OpenWSN*, é possível notar, conforme a Figura 12, que existe uma região chamada *crosslayers*, onde é possível o compartilhamento de uma interface de comunicação capaz de interagir com todas as outras camadas.

Esse local é bastante estratégico, uma vez que possibilita a troca de mensagens com todas as outras camadas.

É exatamente nesse ponto que o presente trabalho propõe a extensão da arquitetura do *OpenWSN*: inserindo-se um novo componente chamado *Dispatcher*, capaz de receber notificações de qualquer outra camada da arquitetura original e enviá-las a uma próxima etapa, responsável pelo *debugging*. Essa abordagem é ilustrada na Figura 102.

Figura 102 – Extensão da arquitetura do OpenWSN

Fonte: WATTEYNE et al., (2012)

Considerando-se que o ambiente de desenvolvimento faz uso de uma plataforma de simulação, esse novo componente: o *Dispatcher* é executado como parte do processo Python no computador que hospeda a simulação.

Dessa forma, é possível o envio de mensagens de log a partir de qualquer trecho do código fonte do *OpenWSN*, localizado em qualquer camada, diretamente para esse componente *Dispatcher*, o qual, sendo parte do processo *Python*, é capaz de encaminhar a mensagem para outros computadores presentes o segmento de rede do computador onde acontece a simulação.

A mensagem foi definida com um tamanho máximo de 255 bytes, sendo dividida em 2 partes: cabeçalho e dados; seguindo o formato descrito na Figura 103, onde:

- O byte 0 representa a versão do protocolo;
- O byte 1 representa o tipo da mensagem;
- Os bytes 2 e 3 armazenam o código da mensagem;
- O byte 4 guarda um contador informando a quantidade de bytes na parte de dados;
- Os bytes de 5 em diante representam uma mensagem contendo uma versão textual.

Figura 103 – Formato da Mensagem de Log

* n máximo = 255

O tamanho máximo de 255 bytes foi escolhido com o propósito de uma mensagem de log ter um tamanho limitado.

A arquitetura pela qual essas mensagens são enviadas contempla a utilização do envio de pacotes no formato UDP (*User Datagram Protocol*), com a utilização de endereços *multicast* com protocolo IPv4 (*Internet Protocol* versão 4). A Figura 104 ilustra essa arquitetura.

Figura 104 – Formato da Mensagem de Log

Tendo em vista que o mecanismo utiliza o protocolo UDP, não há a necessidade de confirmação do recebimento da mensagem, tornando o protocolo mais leve, quando comparado à abordagem original, onde era necessário o *handshake* da comunicação pela porta serial.

Além disso, com essa nova arquitetura, não há mais a necessidade de múltiplas portas seriais, tendo como resultado um mecanismo menos oneroso, pois, no modelo original, eram necessárias tantas portas seriais quantos fossem os nós envolvidos.

Com relação à velocidade de comunicação, ou seja, à capacidade de envio de logs, há um ganho relativamente considerável, pois a interface serial estava limitada a apenas 115200 bps, enquanto uma interface *ethernet* utiliza a ordem de grandeza de 100 Mbps.

Ainda que não haja garantias com relação ao determinismo da comunicação no padrão ethernet, a capacidade de comunicação dessa nova abordagem é bastante superior.

A escolha de endereços de rede *multicast* (IPv4) foi realizada com o objetivo de se permitir algumas características interessantes, como, por exemplo:

a) a independência da configuração do endereçamento da rede onde se encontra o computador de simulação (Computador X – conforme a Figura 104).

b) a possibilidade de controle do nível de propagação da mensagem para outras redes, avançando ou não para mais roteadores, através do controle do flag TTL (*time to live*) da mensagem *multicast*.

c) a possibilidade de realização de filtros específicos em um ou mais computadores, com a finalidade de verificação de uma mensagem de nó em especial ou outras configurações de filtros.

A.2 Estudo de Caso

Para o estudo de caso, foi utilizada uma simulação de uma rede de sensores sem fio com 10 nós distribuídos de forma aleatória, com o objetivo de realizar o acompanhamento (*debugging*) do processo de roteamento nessa rede.

Adotou-se, para efeito de observação, o envio de log para todas as chamadas de funções responsáveis pela utilização do protocolo RPL.

O *Dispatcher* foi desenvolvido com um módulo dentro da arquitetura do *OpenWSN*, na camada *crosslayers*, conforme a Figura 102.

Para efeito a visualização das mensagens, adotou-se a escolha de uma ferramenta padrão e de código aberto para análise de redes: o Wireshark.

Para visualização do protocolo proposto, diretamente no Wireshark, foi desenvolvido um plugin do tipo *dissector*, responsável por realizar a análise de protocolos, utilizando-se a linguagem Lua.

A.3 Resultados e Análise

A simulação foi realizada por meio de dois computadores padrões de mercado, ambos executando o sistema operacional Windows. Esses computadores, por sua vez, foram conectados a um switch de 100 Mbps. Em uma das máquinas realizou-se a simulação do estudo de caso por meio do simulador *OpenWSN*, enquanto, no outro, foi realizada captura dos dados com o *Wireshark*, implementando-se a arquitetura proposta, de acordo com a Figura 105.

O ambiente foi configurado para permanecer em funcionamento por 5 minutos.

Figura 105 – Captura dos dados de *debugging* com o Wireshark

O estudo do log realizado pela ferramenta *Wireshark* permitiu o acompanhamento detalhado do processo de roteamento do RPL.

Foi possível, portanto, realizar a prova de conceito do funcionamento da arquitetura proposta.

Nesse caso, comparando-se a presente proposta de arquitetura com relação ao modelo original, é possível verificar que o mecanismo de log *UDP* permite uma melhoria em relação à possibilidade de análise de vários nós simultaneamente.

Além disso, é possível citar que a operação de *debugging*, inicialmente realizada de forma local, através de uma portal serial, passa a ter a possibilidade de ser executada remotamente, trazendo uma facilidade para novos utilizadores da plataforma, que podem acompanhar de um ou mais computadores remotos, devido à característica da utilização de mensagens *multicast*.

A.4 Conclusão

A arquitetura aqui proposta apresentou uma nova abordagem com relação à extensão da plataforma de simulação de redes de sensores sem fio - *OpenWSN* – por meio de um mecanismo capaz de realizar *debugging* remoto desse ambiente.

O uso de ferramentas de código aberto para análises de rede, como o *Wireshark*, escolhido para validação do estudo de caso, demonstrou a possibilidade de criação de novos *dissectors*, capazes de auxiliar na operação de simulação de ambientes de testes com redes de sensores sem fio.

Novos trabalhos poderão, portanto, fazer uso deste mecanismo para melhorar seu processo de *debugging*.