UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FABRÍCIO NEGRISOLO DE GODOI

CONTRIBUIÇÃO PARA O AUMENTO DA CONFIABILIDADE NA ENTREGA DE PACOTES EM REDES DE SENSORES SEM FIO MULTISSALTOS

DISSERTAÇÃO

PATO BRANCO
2018

FABRÍCIO NEGRISOLO DE GODOI

CONTRIBUIÇÃO PARA O AUMENTO DA CONFIABILIDADE NA ENTREGA DE PACOTES EM REDES DE SENSORES SEM FIO MULTISSALTOS

Dissertação de mestrado apresentado ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica – Área do conhecimento: Sistemas E Processamento De Energia.

Orientador: Prof. Dr. Gustavo Weber Denardin

PATO BRANCO
2018

G588c Godoi, Fabrício Negrisolo de.
Contribuição para o aumento da confiabilidade na entrega de pacotes em redes de sensores sem fio multissaltos / Fabrício Negrissolo de Godoi. – 2018.
143 f. : il. ; 30 cm.

Orientador: Prof. Dr. Gustavo Weber Denardin
Dissertação (Mestrado) - Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica. l. Pato Branco, PR, 2018.
Bibliografia: f. 131 - 140.

1. Confiabilidade. 2. Comunicação e tecnologia. 3. Serviço - Qualidade. 4. Detectores. 5. Processamento de sinais. I. Denardin, Gustavo Weber, orient. II. Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDD 22. ed. 621.3

Ficha Catalográfica elaborada por
Suélem Belmudes Cardoso CRB9/1630
Biblioteca da UTFPR Campus Pato Branco

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Diretoria de Pesquisa e Pós-Graduação
Programa de Pós-Graduação em Engenharia Elétrica

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

TERMO DE APROVAÇÃO

Título da Dissertação n.° 064

“Contribuição para o Aumento da Confiabilidade na Entrega de Pacotes em Redes de Sensores sem Fio Multissaltos”

por

FABRÍCIO NEGRISOLO DE GODOI

Dissertação apresentada às oito horas e trinta minutos, do dia dez de julho de dois mil e dezoito, como requisito parcial para obtenção do título de MESTRE EM ENGENHARIA ELÉTRICA, do Programa de Pós-Graduação em Engenharia Elétrica - Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho APROVADO.

Banca examinadora:

Prof. Dr. Gustavo Weber Denardin
(orientador) UTFPR/PB

Prof. Dr. Fábio Luiz Bertotti
UTFPR/PB

***Prof. Dr. Carlos Henrique Barriquello**
UFSM/RS
*Participação à distância.

Prof. Dr. Jean Patric da Costa
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica - PPGEE/UTFPR

A versão devidamente assinada desse termo, encontra-se em arquivo na Biblioteca da UTFPR – Câmpus Pato Branco.

À minha mãe, Kátia Torres Negrisolo, por me ensinar a trilhar minhas próprias conquistas, mostrando-me através de suas atitudes e lutas como superar os piores desafios.

AGRADECIMENTOS

Primeiramente, ao apoio e carinho de meus pais, irmãos e toda minha família, que me proporcionaram os meios necessários para realizar este mestrado e todo incentivo para ir sempre adiante.

Ao meu grande amigo João Vitor Sampar, por me guiar, apoiar e incentivar por anos para que eu me torne uma pessoa sempre melhor, em busca de paz e sabedoria.

À Geórgia A. C. Zangaro por me auxiliar e incentivar a continuar seguindo meus objetivos, mesmo nos momentos mais difíceis.

Ao meu orientador Gustavo Weber Denardin e ao professor Carlos H. Barriquello, por me guiarem e auxiliarem durante o trajeto de desenvolvimento deste trabalho.

Ao Programa de Pós-Graduação em Engenharia Elétrica (PPGEE) e todos seus integrantes que me guiaram e proporcionarem as condições necessárias para execução desta pesquisa.

À esta universidade, seu corpo docente, direção e administração que proporcionaram todas as ferramentas necessárias para conclusão desta etapa.

À CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), à Fundação Araucária (FA), à Financiadora de Estudos e Projetos (FINEP), bem como à todo povo brasileiro, que com seus esforços possibilitam bolsas de incentivo à pesquisa em nosso país.

RESUMO

GODOI, Fabrício Negrisolo de. Contribuição para o aumento da confiabilidade na entrega de pacotes em redes de sensores sem fio multissaltos. 2018. 144 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2018.

Redes de Sensores Sem Fio têm sido utilizadas para coletar informações dos mais diversos ambientes e enviá-las aos agregadores de informações, os quais são responsáveis por analisá-las. Entretanto, essas redes utilizam um meio de comunicação que é suscetível a ruídos e interferências, de modo que os pacotes que trafegam pela rede podem ser perdidos em suas rotas até o agregador de dados. Além disso, os dispositivos utilizados nesse tipo de rede possuem restrições quanto a sua capacidade de processamento, armazenamento e de comunicação, o que prejudica ainda mais a confiabilidade de entrega de dados, pois se torna inviável utilizar métodos complexos de correção de erros, ou mesmo utilizar *buffers* de rede de tamanhos adequados. Portanto, esse trabalho descreve um método de comunicação, denominado µNet, que possui objetivo de aumentar a taxa de entrega de informações em rede multissaltos. A abordagem utiliza mensagens de confirmação de transmissão ponto-a-ponto, de modo a garantir que a informação tenha sido encaminhada corretamente. Além disso, o µNet foi desenvolvido para ser compatível com dispositivos de baixa capacidade, ao reduzir o tamanho dos cabeçalhos dos protocolos de comunicação e do *buffer* de rede. Ao utilizar ferramentas de simulação para comparar o µNet com protocolos utilizados em RSSFs, demonstra-se que tal abordagem resulta em maior confiabilidade na entrega de dados fim-a-fim em redes suscetíveis a interferências.

Palavras-chave: *Buffer*. Confiabilidade. Protocolos de Comunicação. Qualidade de Serviço. Rede de Sensores Sem Fio.

ABSTRACT

GODOI, Fabrício Negrisolo de. Contribution to increase the reliability of package delivery in multi-hop wireless sensor network 2018. 144 p. Thesis – Programa de Pós-graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2018.

Wireless Sensor Networks are being used to collect a large amount of data from the most diverse environments and forward it to data collectors, which are also responsible to analyze such data for a particular purpose. However, these networks rely on unreliable communication medium that is susceptible to noise and interference, which results in loss of the packages that are being transported in the routes from the sensors to the data aggregators. In addition, the devices used in these networks have restrictions on their processing, storage, and communication capacity, which diminishes, even more, the reliability of the data delivery, since it becomes impracticable to use complex methods for error correction, or even to use appropriately sized network buffers. Therefore, this work describes a new communication method named µNet, which aims to increase the delivery rate of packets in multi-hop networks. Such approach is based on hop-by-hop control messages to ensure that the packet has been properly forwarded. Besides that, the µNet is designed to be compatible with constrained devices, reducing the size of headers of the protocols and the size of the network buffer. Using simulations tools to compare the µNet with widely used WSNs protocols, it is demonstrated that this approach results in better end-to-end reliability in networks susceptible to interference.

Keywords: Buffer. Communication Protocols. Quality of Service. Reliability. Wireless Sensor Network.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SIGLAS

ACK	*Acknowledgement*
ACKN	*Acknowledgement of the Network*
ACKR	*Acknowledgement of the Radio*
BRTOS	*Brazilian Real-Time Operating System*
CSMA	*Carrier Sense Multiple Access*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
CSMA/CD	*Carrier Sense Multiple Access with Collision Detection*
HTTP	*Hyper Text Transfer Protocol*
ICMP	*Internet Control Message Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IoT	*Internet of Things*
IP	*Internet Protocol*
ISM	*Industrial, Scientific and Medical*
LLC	*Logical Link Control*
LLN	*Low-power and Lossy Network*
OF	*Objective Function*
QoS	*Quality of Service*
RPL	*Routing Protocol for Low-Power and Lossy Network*
RSSF	Rede de Sensores Sem Fio
RUDP	*Reliable User Datagram Protocol*
TCP	*Transmission Control Protocol*
UDGM	*Unit Disk Graph Medium*
UDP	*User Datagram Protocol*
WSN	*Wireless Sensor Network*

LISTA DE ACRÔNIMOS

6LoWPAN	*IPv6 over Low power Wireless Personal Area Networks*
CoAP	*Constrained Application Protocol*
CON	*Confirmable*
DAO	*Destination Advertisement Object*
DIO	DODAG *Information Object*
DIS	DODAG *Information Solicitation*
DODAG	*Destination Oriented Directed Acyclic Graphs*
FIFO	*First In – First Out*
MAC	*Media Access Control*
NON	*Non-confirmable*
OSI	*Open System Interconnection*
PAN	*Personal Area Network*
SAP	*Service Access Point*

LISTA DE SÍMBOLOS

M	Distribuição exponencial (sem memória)
D	Distribuição determinística
G	Distribuição genérica
L_q	Número médio de clientes na fila
L	Número médio de clientes no sistema
λ	Taxa de chegadas à fila
μ	Taxa de atendimentos da fila
K	Capacidade do sistema
λ_L	Taxa de chegadas à fila local
λ_E	Taxa de chegadas à fila por encaminhamento
D_{TX}	Raio de alcance de transmissão
D_{TI}	Raio de alcance da interferência de uma transmissão
P_S	Taxa de sucesso para enviar e receber mensagem entre dispositivos
P_{TX}	Probabilidade de transmitir uma informação corretamente
P_{RX}	Probabilidade de receber uma informação corretamente
$F_{Rádio}$	Razão entre potência configurada e máxima do rádio
P	Período entre transmissões
ID	Identificador único de cada dispositivo
T_R	Tempo para rotear uma mensagem até o destino
T_{ID}	Tempo de espera único para cada dispositivo dentro de um P
qc	Quantidade de clientes na rede
C	Taxa de confiabilidade de entrega de pacotes fim-a-fim
Pkt_{Rec}	Quantidade total de pacotes recebidos
Pkt_{Totais}	Quantidade total de pacotes emitidos
qp	Quantidade de pacotes a serem transmitidos
t_{PP}	Tempo de transmissão ponto-a-ponto de um pacote
h	Quantidade de saltos de um dispositivo até seu destino
P_I	Intervalo ideal entre transmissões de pacotes em uma rede

SUMÁRIO

1 INTRODUÇÃO

Sistemas de sensoriamento com comunicação sem fio são utilizados para os mais diversos propósitos, tais como: detecção de fogo em matas, análise de solo em áreas de cultivo, sensores climáticos para previsões do tempo, detecção de tropas militares (aliadas ou não), entre outras aplicações (STOJMENOVIĆ, 2005). Com o avanço da tecnologia adotada em Rede de Sensores Sem Fio (RSSF), aplicações voltadas para Internet das Coisas (IoT – *Internet of Things*) e cidades inteligentes (*smart cities*), têm se tornado promissoras. A IoT utiliza sensores e programas em dispositivos físicos (veículos, eletrodomésticos, etc.) capazes de comunicar entre si, assim como interagir com usuários. Aplicações de IoT voltadas para cidades são denominadas *smart cities*, cuja principal função é melhorar a qualidade e desempenho dos serviços urbanos, de modo a facilitar e melhorar a vida da população (SHAH; MISHRA, 2016).

Uma das principais aplicações para esse tipo de tecnologia é o sensoriamento e controle dos recursos essenciais, como: água (à qual é essencial para qualquer organismo vivo) e energia elétrica (que é utilizada para realizar diversas atividades na sociedade). Para garantir o melhor uso desses recursos, são realizados estudos sobre métodos para evitar desperdícios. Entre eles, o controle de iluminação pública, que por meio da análise de movimento de pessoas e automóveis, definem quais setores são iluminados, ou não, na cidade. Esse método evita que lugares com baixo fluxo de pessoas tenham constante iluminação, desperdiçando energia elétrica. Para realizar esse controle, são utilizados dispositivos que se comunicam por um meio sem fio, levando as informações a uma central de controle e gerenciamento, conforme representação pela Figura 1. Métodos cabeados poderiam ser utilizados, entretanto, o custo para realizar o cabeamento estruturado, e sua manutenção, é economicamente inviável por causa da quantidade de dispositivos necessários nesse tipo de aplicação (LAVRIC *et al.*, 2014; RAMAKRISHNAN; GAUR, 2016; DAELY *et al.*, 2017; GHARAIBEH *et al.*, 2017; KNOBLOCH; BRAUNSCHWEIG, 2017; MAHOOR *et al.*, 2017).

Outro exemplo de aplicação é o *Smart Grid*, em que são empregadas redes de comunicação com objetivo de monitorar o consumo e automatizar a transmissão/distribuição de energia elétrica. Nesse contexto, as redes de sensores têm papel importante na leitura automática de medidores (AMR do inglês *Automatic Meter Reading*). Seu funcionamento é dependente da troca de informações entre fornecedores e consumidores de energia, definindo

o melhor método de fornecimento de energia com as informações coletadas dos consumidores. Para realizar as medições são necessários diversos sensores distribuídos, os quais enviam suas informações a uma central de análise. Além disso, a mesma metodologia pode ser empregada para verificar a qualidade estrutural dos equipamentos utilizados, prevenindo que ocorram falhas por mal funcionamento. Portanto, para realizar a troca de informação, são utilizados métodos de comunicação sem fio, os quais provêm custos e consumo energético baixos (AL-ANBAGI *et al.*, 2014; HOSNI; HAMDI, 2016; VILGELM *et al.*, 2016; ULLAH *et al.*, 2017).

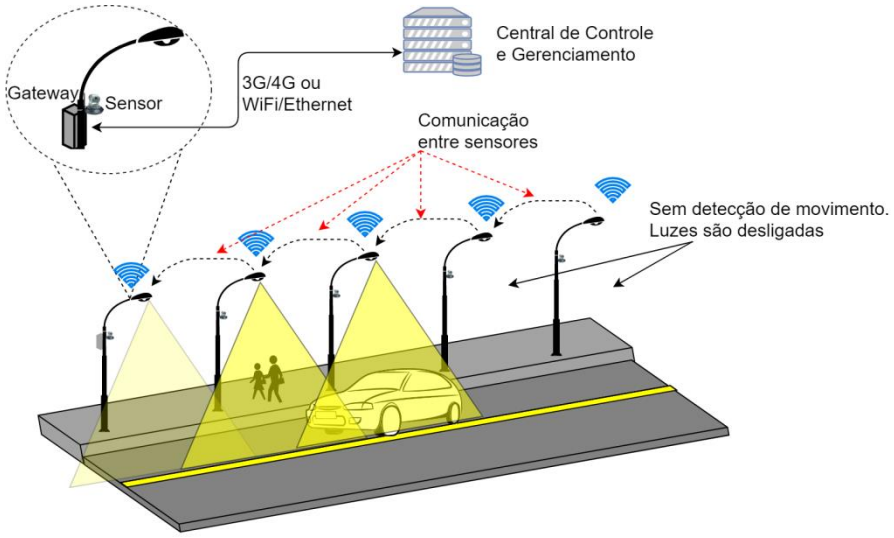


Figura 1 – Representação de iluminação pública controlada.
Fonte: Adaptado de Gharaibeh *et al.* (2017, p. 6).

Outras aplicações que utilizam do sistema de comunicação sem fio são: monitoramento de emissão de gás carbônico, temperatura e umidade (SHAH; MISHRA, 2016); leitura automática de medidores de energia elétrica e água (NHAT-QUANG NHAN *et al.*, 2012; MUDUMBE; ABU-MAHFOUZ, 2015); monitoramento e controle de trânsito (DAGHER *et al.*, 2014; NANDURY; BEGUM, 2016); monitoramento em casas inteligentes (GHAYVAT *et al.*, 2015); entre outras. Apesar de ser possível utilizar essa tecnologia em diversas áreas, há vários problemas de projeto que devem ser levados em consideração. Os dispositivos utilizados para esses propósitos contêm recursos limitados, sejam eles processamento, memória ou energia. Além disso, devem ser tolerantes a falhas, para que suas aplicações funcionem da maneira esperada, pois os ambientes em que essa tecnologia é empregada, em sua maioria, contém fontes de ruídos, poeira, umidade, terra, etc., que agravam falhas em suas transmissões e modos operacionais (GUNGOR; HANCKE, 2009).

Em meios urbanos há problemas mais recorrentes e que afetam a estrutura da comunicação, como bloqueio ou reflexão do sinal de comunicação devido a prédios, estradas, veículos, etc (YOON *et al.*, 2014). Além disso, há problemas referentes ao espectro de rádio livre utilizado (900 MHz, 2,4 GHz e 5,7 GHz), pois esse é utilizado por indústrias, grupos científicos e unidades hospitalares (ISM - *Industrial, Scientific and Medical*), os quais provocam interferências nos canais de comunicação (UNAWONG *et al.*, 1999).

Uma forma de prevenir que as informações sejam perdidas no meio de comunicação de uma rede, seja por causa de interferências externas ou por sobrecarga da rede, é pela utilização de protocolos de comunicação que forneçam algum mecanismo de garantia de entrega. Geralmente, em redes de computadores, utiliza-se o protocolo de comunicação TCP (*Transmission Control Protocol*) para garantir que dados entre emissor e receptor sejam entregues de forma consistente. Entretanto, esse não é o caso para RSSF, pois a latência para se estabelecer uma conexão entre dois dispositivos, mais o tamanho do cabeçalho necessário para manter a conexão, acarreta numa série de eventos nos quais mais pacotes são perdidos e, consequentemente, a taxa de transmissão é significativamente reduzida. Mesmo quando a comunicação é ponto-a-ponto, sem saltos e sem interferência, esse protocolo provoca uma sobrecarga no tamanho do pacote transmitido, o qual, em sua maioria, é composto por pequenos dados de sensoriamento (SOHRABY *et al.*, 2007).

Por causa da sobrecarga na RSSF pelo protocolo TCP, opta-se por utilizar o protocolo UDP (*User Datagram Protocol*), que não possui garantia de entrega de informação, e utilizar outro protocolo em conjunto para garantir a entrega, como o CoAP (*Constrained Application Protocol*). Apesar do CoAP não ser um protocolo exclusivo de garantia de entrega de informações, ele possui métodos simplificados do TCP para RSSF (SHELBY *et al.*, 2014). Além disso, formas alternativas para melhorar a transmissão utilizando o protocolo UDP foram criadas. Entre elas cita-se o RUDP (*Reliable* UDP) (BOVA; KRIVORUCHKA, 1999), o qual busca maneiras diferentes de aperfeiçoar o protocolo UDP existente ao adicionar pequenas informações de controle, de forma a garantir a entrega dos dados. Além de métodos que buscam otimização do protocolo de transmissão utilizado, foram implementados outros algoritmos para melhorar a comunicação da rede. Entre eles, o método de junção de roteamento e escalonamento, os quais têm como objetivo alcançar a taxa de transmissão máxima, como nos trabalhos de: Tassiulas (1992), Giaccone *et al.* (2007), Lin e Shroff (2006), Joo e Shroff (2009), Bui *et al.* (2008) e Kar *et al.* (2008).

A maioria desses trabalhos visa atingir a maior taxa de transmissão com menor comunicação e complexidade computacional, assumindo que os *buffers*[1] de comunicação são infinitos, de forma a nunca ocorrer estouro de memória (*overflow*). Entretanto, não levar em consideração que os dispositivos possuem memória física limitada, ignora um importante problema da engenharia. Além de que, a otimização do tamanho do *buffer* deve ser feita de forma a não perder capacidade de comunicação na rede, caso contrário pode haver perdas significativas na taxa de transmissão (LE *et al.*, 2012). Conforme o trabalho de Baron *et al.* (2009), verifica-se o paradoxo de alocação de capacidade, o qual define que uma rede estável pode se instabilizar conforme o incremento de fluxo de dados, não importando o tamanho de *buffer* utilizado, prejudicando o desempenho da rede. Al-Anbagi *et al.* (2013) e Omondi *et al.* (2015) apresentam outros exemplos de pesquisas que analisam o impacto do tamanho do *buffer* numa rede de baixa capacidade. Todos esses estudos são detalhados no capítulo 4.4.

Portanto, ao considerar a aplicabilidade das RSSF e seus desafios, é proposto um novo método de comunicação para RSSF, com objetivo de aprimorar a confiabilidade na entrega de informações. Para aumentar a confiabilidade, são utilizadas mensagens de confirmação entre dispositivos vizinhos (ponto-a-ponto) ao enviar uma mensagem, o que resulta numa maior confiabilidade em transmissões de múltiplos saltos (fim-a-fim). Além disso, o método leva em consideração que os dispositivos utilizados possuem recursos limitados e, portanto, demonstra como é possível aumentar a confiabilidade na rede com *buffers* reduzidos, com base em algumas características de sistemas de fluxo e teoria de filas.

1.1 OBJETIVOS

O objetivo geral deste trabalho consiste em aprimorar os métodos de comunicação de RSSF, com foco em melhorar a confiabilidade de entrega de informações fim-a-fim (origem ao destino) de redes multissaltos.

Para consolidar o objetivo geral, deve-se atingir os seguintes objetivos específicos:

- Coletar informações do estado da arte, métodos e métricas utilizados em RSSF;

[1] *Buffer*, de acordo com Michaelis (2017), é definido como: “área de armazenamento temporário de dados à espera de processamento”.

- Avaliar e confrontar o método proposto de acordo com a revisão literária, de modo a validar o processo utilizado;
- Definir o método de comparação entre soluções conhecidas (retiradas a partir da análise literária) e a proposta;
- Com base no estado da arte, definir e validar uma ferramenta de simulação de RSSF para comparar as propostas;
- Criar uma aplicação para comparação de desempenho entre soluções;
- Configurar os sistemas operacionais utilizados de forma que possam ser executados na ferramenta de simulação;
- Extrair os dados dos ambientes de simulações, para que sejam feitas as devidas comparações de desempenho;
- Analisar e apresentar os resultados coletados e utilizá-los para validar e concluir se o método proposto atinge os objetivos estipulados.

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho está dividido em oito capítulos. O primeiro capítulo contém a contextualização do assunto abordado, contendo as principais características sobre rede de sensores sem fio, suas vantagens, desvantagens e a motivação deste trabalho. O segundo capítulo aprofunda-se nas características de RSSFs, demonstrando como podem ser aplicadas nos mais diversos campos, e quais os principais desafios que compõem essa tecnologia.

O terceiro capítulo aborda a estrutura das redes de comunicações com ênfase em RSSF, explicando os métodos e protocolos utilizados para obter uma rede com melhor desempenho. O quarto capítulo aprofunda-se no entendimento de sistemas de fluxo, filas e seus impactos no desempenho de RSSF. São demonstrados trabalhos que utilizam as teorias sobre filas e como a quantidade de memória de um dispositivo pode afetar seu funcionamento.

Os capítulos cinco e seis abordam a solução proposta para melhorar a confiabilidade de entrega de pacotes em RSSF, como descrito nos objetivos, assim como os materiais e métodos utilizados para validar e coletar os resultados de avaliação da proposta. Por fim, nos capítulos sete e oito, são apresentadas as análises dos resultados obtidos pela ferramenta de simulação de RSSF, assim como uma discussão detalhada de que cada informação.

2 REDES DE SENSORES SEM FIO

Um sensor de uma RSSF é um dispositivo que contém processador, memória, interfaces de comunicação e mecanismos para coletar dados do meio em que está inserido. Entre os tipos de dados amostrados, os mais comuns são: intensidade luminosa, intensidade sonora, níveis de emissão de gás carbônico, detecção de fumaça/fogo, umidade e temperatura. Consequentemente, uma rede de sensores é composta por um conjunto de dispositivos com sensores, geralmente composta por centenas ou milhares de unidades, densamente implantados em campos, indústrias, hospitais, oceanos, corpo humano, ou em qualquer outro ambiente possível. Por ser uma rede, os sensores possuem a propriedade de trocar informações entre eles, por cabos ou por módulos de comunicação sem fio, que utilizam, em sua maioria, rádios que se comunicam por meio de ondas eletromagnéticas, utilizando o ar como meio de transporte (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007).

Assim, a RSSF pode ser utilizada como uma ferramenta com objetivo de coletar dados e transmiti-los a um coordenador, o qual deve agir de acordo com sua aplicação. Para agregar as informações no coordenador, são utilizadas redes baseadas em múltiplos saltos, de forma que a informação é transmitida de dispositivo a dispositivo, até chegar ao coordenador. A arquitetura de rede multissaltos pode ser classificada em: malha, árvore ou árvore por agrupamento; conforme exemplificado pela Figura 2.

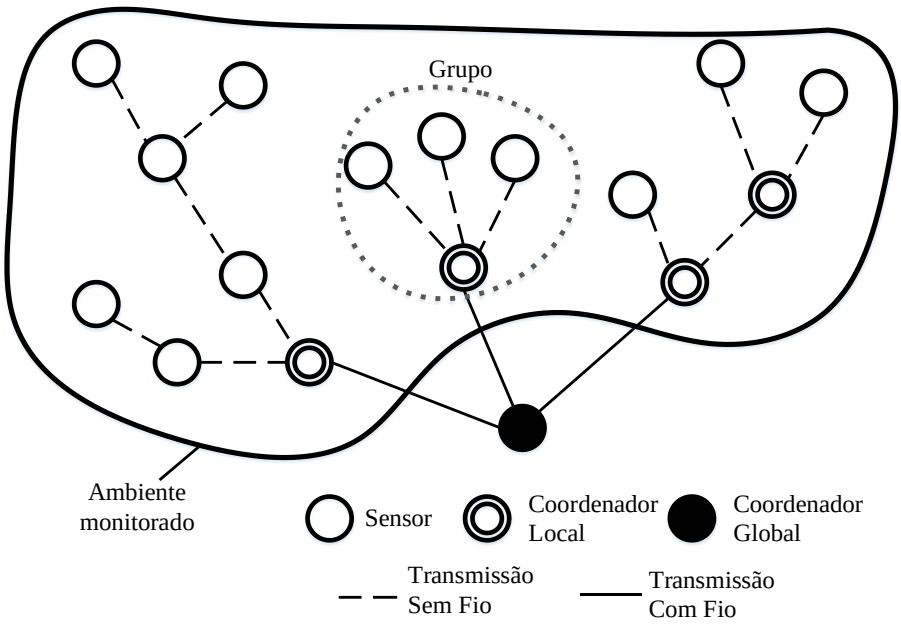


Figura 2 – Exemplo de RSSF por agrupamento.
Fonte: Adaptado de Sohraby *et al.* (2007, p. 16).

A RSSF da Figura 2 utiliza a arquitetura de agrupamento, na qual cada dispositivo está agrupado a um Coordenador Local responsável por gerenciar esse grupo. Cada dispositivo dessa rede obtém suas informações do “Ambiente Monitorado” por seus sensores, e as envia ao próximo dispositivo na cadeia de comunicação, passando pelos Coordenadores Locais, até chegar ao Coordenador Global.

Por causa da capacidade de agregação de dados, aplicações para RSSF têm sido desenvolvidas para as mais diversas áreas. As principais aplicações são agrupadas nas categorias: civil, hospitalar, militar, industrial e ambiental (STOJMENOVIĆ, 2005). No capítulo 2.1 são apresentadas pesquisas que envolvem aplicações para cada grupo específico.

Entretanto, apesar da aplicabilidade das RSSF, tal metodologia é composta por diversas dificuldades de implantação, causados pelos recursos limitados dos sensores (memória, processamento, etc.), ou por estarem situadas em ambientes degradantes ao seu funcionamento (umidade, poeira, temperatura, etc.). A partir disso, no capítulo 2.2 é realizado um detalhamento dos principais desafios encontrados na literatura referentes à RSSF.

2.1 APLICAÇÕES

Conforme descrito na Introdução deste trabalho, existem diversos projetos que têm como objetivo melhorar a vida em cidades, podendo ser pelo simples monitoramento de algum aspecto, ou pelo controle direto de componentes da rede. Pelo foco desses projetos serem voltados às cidades, comumente são nomeados de projetos para cidades inteligentes, ou *smart cities*.

Aplicações de RSSF envolvendo saúde têm por objetivo melhorar a vida de pacientes que dependem de constante monitoramento. Trabalhos como de Puvaneshwari S. e Vijayashaarathi S. (2016) e Toh *et al.* (2008) demonstram técnicas de integração de rede de sensores com celulares, de forma que os dados coletados pelos sensores possam ser transmitidos para uma equipe médica, ou médico responsável. Essa rede de sensores monitora constantemente as condições de cada paciente, avaliando medidas de: temperatura, pressão arterial, condições cardíacas, saturação de oxigênio, níveis de glicose, hemoglobina, entre outras medidas possíveis. Dessa forma, médico e paciente obtêm formas para evitar ou tratar enfermidades.

Outra área que se beneficia do uso de RSSF é a militar, para coletar as informações de tropas inimigas e aliadas. De forma a se preparar contra ataques inimigos, uma RSSF pode fornecer métodos de detecção e de posicionamento de tropas, sendo por vias terrestres, áreas ou navais. Com essas informações pode-se analisar e prever possíveis ataques e utilizar formas defensivas mais efetivas. Por outro lado, também pode ser utilizado para analisar as condições de tropas aliadas, verificar munições, locais e condição física de cada indivíduo (STOJMENOVIĆ, 2005; HUSSAIN *et al.*, 2009; GRUMAZESCU *et al.*, 2016).

Em indústrias, a necessidade de automatização de processos, ou mesmo o monitoramento de máquinas e equipamentos, se mostra cada vez mais necessária. Indústrias em que a segurança do trabalhador está sempre em risco, como em minas de carvão, é essencial ter um método de monitoramento de seus funcionários ou mesmo de possíveis desabamentos (GUNGOR; HANCKE, 2009; LU *et al.*, 2015; CHENG *et al.*, 2016).

Ao utilizar RSSF em áreas ambientais, é possível obter informações localizadas e detalhadas, o que seria impraticável sem esse tipo de tecnologia (STOJMENOVIĆ, 2005). A partir disso, diversas aplicações foram propostas, como: monitoramento de habitat, rastreamento de animais, detecção de incêndios, cultivo de precisão (VISCONTI *et al.*, 2016), monitoramento de emissão de gás carbônico (SUDARSONO *et al.*, 2015) e predição de catástrofes (TORII, YOSHITAKA; OTSUKA, T.; ITO, 2016).

2.2 DESAFIOS NAS RSSF

Pelo fato das RSSF serem compostas de dispositivos de baixo custo, os recursos utilizados devem ser maximizados para seu propósito. Assim, memória, processador, baterias, rádios e outras características devem ser analisadas para cada projeto. As principais pesquisas existentes na literatura têm como objetivo melhorar o desempenho, diminuir a energia gasta e melhorar a qualidade na transmissão de informações. Entre os diversos desafios enfrentados nessas pesquisas, os mais citados são: recursos, ambiente operacional, qualidade de serviço, segurança e sincronização (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007; GUNGOR; HANCKE, 2009).

2.2.1 Recursos

Dispositivos utilizados nas RSSF são limitados por tamanho e ambientes situados, sendo que nem sempre é possível que tenham fontes de energia ou mesmo baterias, suficientemente, duradouras. Para que seja possível otimizar a longevidade de suas aplicações e reduzir custos de manutenção da rede, são utilizados periféricos de processamento, armazenamento e de comunicação que consomem a menor quantidade de energia possível. Além disso, as redes são compostas de centenas ou milhares de sensores, e para que o custo de sua implantação seja realizável, é necessário utilizar a quantidade mínima de recursos, sem que se interfira na operabilidade ao atingir o objetivo desejado (STOJMENOVIĆ, 2005; GUNGOR; HANCKE, 2009).

De forma a maximizar a longevidade da rede, são desejados processadores que consumam baixa quantidade de energia quando em operação e que não necessitem de dissipadores de calor, pois o consumo energético para resfriá-los aumentaria. Para atingir esses requisitos, são utilizadas arquiteturas de unidade de lógica de processamento que variam de 8 *bit*s, até mais recentemente 64 *bit*s. De forma a não aquecer o sistema e reduzir o custo energético, são utilizadas frequências de processamento que variam, geralmente, em dezenas de MHz. Além disso, memórias voláteis, que necessitam de constante fornecimento de energia para o funcionamento, devem energeticamente eficientes e possuir capacidade adequada para cada aplicação. Para exemplificar, os processadores MSP430, que operam dentro dessa faixa de frequência, consomem energia com correntes elétricas em unidades de mA quando em operação, podendo chegar a um consumo em nA quando em espera, e possuem memórias de armazenamento e processamento com algumas dezenas de kbits (STOJMENOVIĆ, 2005; TEXAS..., 2014).

Ao utilizar dispositivos com baixa capacidade de processamento e armazenamento, é necessário utilizar métodos de comunicação mais simples e eficientes a esses dispositivos. Para tal, um dos padrões de comunicação que pode se utilizar é o IEEE 802.15.4, o qual utiliza um cabeçalho reduzido para comunicação, com melhor desempenho em redes densas (STOJMENOVIĆ, 2005; IEEE..., 2015). Esse padrão é descrito no capítulo 3.1.1 com mais detalhes de seu funcionamento. Um dos módulos utilizados para esse propósito é o CC2650, que consome aproximadamente 10 mA para realizar uma transmissão (TEXAS..., 2016a).

2.2.2 Ambiente Operacional

Espera-se que as RSSF operem remotamente em regiões que geralmente emitem algum tipo de interferência na aplicação. Entre essas interferências pode-se citar: interferência no espectro da radiofrequência utilizada, ambientes corrosivos, umidade elevada, vibrações, poeira, etc. (GUNGOR; HANCKE, 2009). Em tais locais (campos, florestas, ou mesmo dentro de indústrias ou casas), há níveis de interferências que dificultam sua aplicabilidade (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007).

Em geral, em sistemas operantes por tecnologia sem fio, o sinal chega ao receptor com potência reduzida devido à distância entre o emissor e o receptor. Essa perda se deve à redução do campo elétrico entre o emissor e o receptor, assim como efeitos de múltiplos caminhos, provenientes da reflexão planar e da difração de borda de sinais em planos de construções, avenidas, veículos, etc. Os sinais propagados por múltiplos caminhos podem chegar, simultaneamente, ao receptor com diferentes amplitudes e fases. Não há como predizer as características ótimas de propagação de sinal em cidades, por causa do ambiente e como cada dispositivo é alocado (diferentes alturas, construções, rodovias, veículos, etc.) (YOON *et al.*, 2014).

As RSSF, em geral, utilizam bandas de operação na faixa ISM (*Industrial, Scientific and Medical*), a qual opera entre 900 MHz e 5.700 MHz. Entretanto, a faixa de operação ISM já é utilizada por outros aparatos, tais como: fornos de micro-ondas, equipamentos de ultrassom, diatermias médicas, etc. Como esses aparatos utilizam radiofrequência para aquecimento, a maioria deles irradia interferência eletromagnética, denominada interferência ISM. Esse tipo de interferência apresenta características impulsivas, às quais são prejudiciais para sistemas que utilizam comunicação sem fio entre 1 GHz e 3 GHz. Além disso, a faixa de comunicação ISM pode ser utilizada sem licenciamento, o que a torna acessível para desenvolvimento. Assim, as RSSF não têm garantias de serem livres de interferências (UNAWONG *et al.*, 1999). Contudo, é necessário considerar as restrições de cada país ao utilizar a ISM, pois, apesar de ser livre, devem-se respeitar as normas de cada órgão regulamentador. No caso do Brasil, a Agência Nacional de Telecomunicações possui a resolução nº 680 de 2017 com normas para certificação (AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES, 2017).

Em cidades é comum utilizar dispositivos que operem na banda ISM, entre eles os computadores e roteadores domésticos, que utilizam o padrão de comunicação IEEE

802.11b/g, mais conhecido como Wi-Fi. Conforme Figura 3, é possível verificar que ambos os padrões de comunicação utilizam as mesmas faixas de comunicação, o que causa a interferência entre as redes. O trabalho de Wagh *et al.* (2015) demonstra como ocorre essa interferência entre os padrões de comunicação e formas que podem ser adotadas para tentar minimizar seus efeitos.

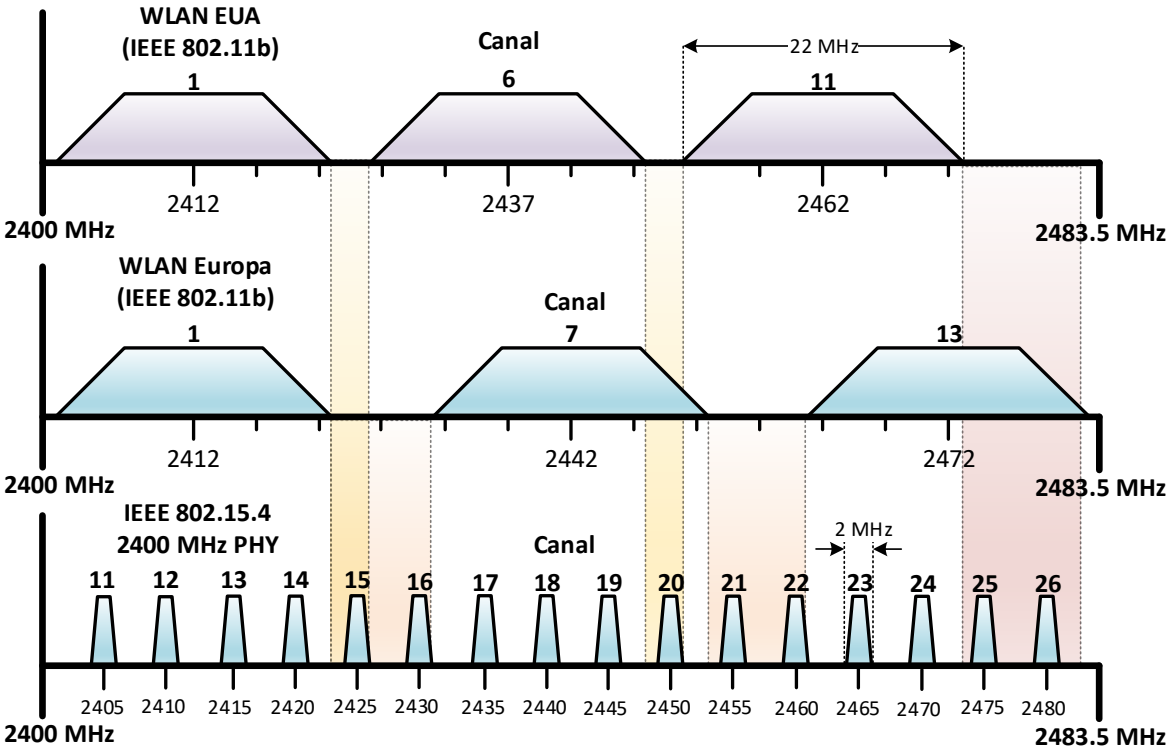


Figura 3 – Espectro de comunicação IEEE 802.11 e IEEE 802.15.4.
Fonte: Adaptado de Wagh *et al.* (2015, p. 746).

2.2.3 Qualidade de Serviço

Qualidade de serviço, ou *Quality of Service* (QoS), refere-se à capacidade dos dispositivos conseguirem transmitir seus dados da origem ao destino conforme as necessidades da aplicação. Dependendo da aplicação, é importante que os dados sejam transmitidos de forma coerente e temporizados, pois podem ser críticos à sua operação. Dados entregues com atrasos, devido à falha na comunicação, podem estar desatualizados, possibilitando que o sistema entre num estado de falha de operação (GUNGOR; HANCKE, 2009).

Uma das métricas utilizadas para avaliar o QoS é a confiabilidade, e em RSSFs ela é geralmente associada com a garantia de transmissão correta de cada pacote. Algumas aplicações de RSSF necessitam somente dos dados de alguns dispositivos de uma determinada área monitorada e não de cada dispositivo que a compreende, ou ainda aceitam que cada dispositivo possua alguma taxa de transmissões bem sucedidas, tolerando até certo ponto uma quantia de pacotes perdidos. Esse conceito de confiabilidade torna os mecanismos de controle de congestão e recuperação de dados perdidos salto-a-salto mais atrativos, pois pode reduzir a quantidade de pacotes perdidos e conservar a energia dos dispositivos. Ao utilizar mecanismos de controle e recuperação a cada salto da rede, reduz a necessidade de *buffer* em dispositivos intermediários, o que os torna úteis para dispositivos de recursos limitados (FAHMY, 2016, p. 61).

Em uma rede de sensores e atuadores, o QoS impacta diretamente no consumo de energia, processamento e memória de cada dispositivo dentro da rede. Conforme a necessidade de precisão e exatidão dos dados mensurados pelos sensores e ações realizadas pelos atuadores, para a aplicação desejada, devem-se agrupar os sensores e os atuadores nos seguintes conjuntos (STOJMENOVIĆ, 2005):

- um conjunto mínimo de atuadores numa região;
- um conjunto mínimo de sensores numa região;
- um conjunto mínimo de atuadores e sensores numa região;
- todo conjunto de atuadores e sensores na região.

Os três primeiros casos têm como foco a redução do nível de redundância, enquanto o último caso foca na cobertura do evento detectado. Cada caso tem seus prós e contras, sendo os três primeiros responsáveis por uma diminuição do consumo de energia, e uma cobertura mais ampla. O último caso permite uma cobertura máxima do evento detectado, com o custo energético mais elevado (STOJMENOVIĆ, 2005).

Outra forma de aprimorar a QoS em redes, é pela modificação dos métodos utilizados para comunicação e controle da rede. Usualmente, são utilizados protocolos de comunicação com confirmação de recebimento para garantir que toda informação transmitida seja recebida adequadamente. Entretanto, utilizar métodos complexos de confirmação, pode levar uma RSSF à inoperância devido à estagnação da rede pelo meio de comunicação (SOHRABY *et al.*, 2007; MASIRAP *et al.*, 2016). Esse assunto é abordado com ênfase no Capítulo 3, ao analisar as métricas utilizadas para aprimorar a qualidade da RSSF.

2.2.4 Segurança na Rede

A segurança na rede deve lidar com a confiabilidade (utilizando encriptação), integridade e disponibilidade do serviço (proteção contra negação de serviço) (SOHRABY *et al.*, 2007). O objetivo fundamental da RSSF é de produzir, por um longo período de tempo, informações obtidas de seus sensores, as quais devem ser consistentes e corretas, tolerantes a ataques e falhas de dispositivos (STOJMENOVIĆ, 2005). Entre as vulnerabilidades de uma RSSF pode-se citar (YI; ZHONGYONG, 2015):

- canal de comunicação aberto: o canal de comunicação utilizado é o meio sem fio, em que qualquer dispositivo que esteja no raio de alcance de comunicação pode obter informações, se utilizar a mesma frequência de comunicação;
- recursos limitados: restrições de tamanhos e custos resultam em limitações em memória, energia, poder de processamento e banda de comunicação, o que implica na impossibilidade de utilizar algoritmos de segurança mais complexos;
- ambiente de implementação autônoma: as RSSF são alocadas em ambientes extremos ou mesmo em territórios inimigos, podendo não haver estrutura, ou proteção, necessária. Uma vez instalada a RSSF, é difícil manter um monitoramento adequado, o que dificulta barrar aquisição dos sensores e obter dados cruciais sobre segurança do dispositivo por atacantes.

A arquitetura computacional dos meios de comunicação utilizam modelos estruturados em camadas para definir os modos operacionais de comunicação. O modelo OSI (*Open System Interconnection Model*) define sete camadas de rede, sendo elas: física, *link* de dados (ou enlace), rede, transporte, sessão, apresentação e aplicação; conforme representado na Figura 4. Cada camada inferior atua como provedora de serviço para camada superior. Por exemplo, a camada de rede provê endereçamento e roteamento para a camada de transporte (TORRES, 2001; SOHRABY *et al.*, 2007).

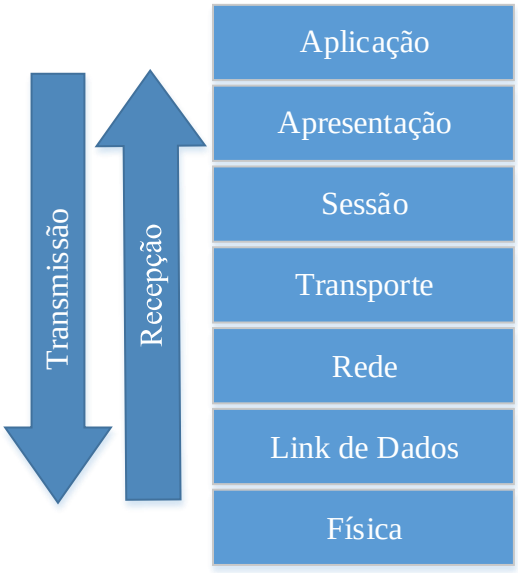


Figura 4 – Modelo OSI.
Fonte: Torres (2001, p. 40).

Os ataques realizados à RSSF podem ocorrer em qualquer camada do modelo OSI, entre os quais cita-se (YI; ZHONGYONG, 2015):

- camada de Aplicação: ataques na camada de aplicação são relativos a aplicações específicas em cada tipo de RSSF. Entre os tipos de ataques pode-se citar ataques de localização e códigos maliciosos;
- camada de Transporte: nesse caso o atacante tem como objetivo, por meio de pedidos múltiplos, estabelecer conexão com nodos vizinhos, esgotamento dos recursos dos nodos vizinhos, fazer com que requerimentos de conexão com outros nodos não sejam atingidos e fazer com que o serviço da RSSF não funcione;
- camada de Rede: ataques à camada de rede têm por objetivo obstruir ou incapacitar que nodos pertencentes à rede consigam comunicação com o coordenador. Métodos como *Wormholes*, *Sinkhole*, *Sybil*, *Hello Food* e encaminhamentos falsificado-alterado-repetido são utilizados para prejudicar a rede;
- camada de Enlace: ataques por esgotamento, competição não justa e colisão são relacionados a essa camada. Nesses casos, os atacantes tentam ocupar o meio de comunicação, por meio de protocolos com maiores prioridades, ou mesmo explorando os métodos de retransmissão que certos protocolos possuem, incapacitando o funcionamento da RSSF;

- camada Física: a camada física é responsável pela modulação do sinal, transmissão, recepção, encriptação dos dados e seleção da banda de frequência. Nesse meio, os atacantes buscam restringir, ou mesmo bloquear, o meio de comunicação e capturar informações transientes da camada física.

Oreku (2013), Jasmin e Velayutham (2014) e Peng *et al.* (2016) apresentam trabalhos focados em melhorar a segurança em RSSF, utilizando modelos matemáticos, algoritmos não lineares ou por determinação da confiabilidade do dado a partir da qualidade do sinal.

2.2.5 Sincronização Temporal

A sincronização temporal é um serviço essencial em RSSF para que seja possível coordenar operações entre dispositivos com propósito de realizar tarefas mais complexas. Temporizadores globais permitem que sensores detectem eventos com cronologia, duração e intervalos corretos. A configuração correta de tempo é essencial para manter o ciclo de tarefas no intervalo mínimo necessário, e entrar em modo de economia de energia se desligando (STOJMENOVIĆ, 2005).

Quando os dispositivos são sincronizados corretamente, é possível acioná-los simultaneamente para envio de mensagens para a estação, e, subsequentemente, entrar novamente em modo de economia de energia. Dispositivos dessincronizados aumentam seu tempo de envio de mensagens, pois têm que esperar os nodos vizinhos estarem preparados para receber a informação, que no pior dos casos pode não ocorrer (STOJMENOVIĆ, 2005).

Protocolos de sincronismo utilizados em redes tradicionais, como o *Network Time Protocol*, não podem ser utilizados diretamente em RSSF por causa de seus recursos limitados. Com isso, protocolos de sincronização otimizados para RSSF têm sido desenvolvidos, como: *Reference-Broadcast Synchronization* (RBS) e *Timing-Sync Protocol for Sensor Networks* (TPSN) (GANERIWAL *et al.*, 2003; SICHITIU; VEERARITTIPHAN, 2003; MUNIR *et al.*, 2015).

2.3 CONCLUSÃO

Neste Capítulo, foram abordados conceitos sobre o que é uma rede de sensores sem fio, como utilizá-la e quais são os principais desafios envolvidos. A arquitetura de RSSF expande as possibilidades de aplicações para análise de informações de campos, em larga escala. Entretanto, esse tipo de rede é composto por desafios que necessitam de análises aprofundadas e métodos de gerenciamento aperfeiçoados para garantir seu funcionamento.

Entre os desafios mencionados, garantir a qualidade de serviço nessas redes é um dos tópicos mais complicados, pois é diretamente relacionado com recursos disponíveis e o ambiente no qual está inserido. Nota-se que em ambientes reais, é improvável que uma rede opere com absoluta garantia de entrega de informação, pois além da interferência da própria rede, há interferências provenientes do meio inserido. Por isso, numa RSSF é natural perder informações, entretanto deve ser mínima para o funcionamento da aplicação, independente qual seja seu propósito.

Portanto, o propósito deste trabalho é garantir que as aplicações de RSSF possuam garantia na entrega de informação, ou seja, melhorar sua QoS. Para tal, são estudados métodos que utilizem a menor quantidade de recursos possíveis, de forma que as aplicações situadas em ambientes ruidosos tenham um desempenho aceitável. A partir disso, os Capítulos 3 e 4 apresentam informações necessárias para o entendimento de RSSF e como seu processo pode ser melhorado.

3 ESTRUTURA DE REDE PARA RSSF

Há diversos protocolos desenvolvidos para RSSFs com intuito de adequar o sistema de comunicação à aplicação de sensoriamento, de modo que sejam de baixa complexidade computacional e energeticamente eficiente. Por causa disso, o modelo de rede das RSSFs geralmente é descrito por cinco camadas: aplicação, transporte, rede, enlace e física (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007; YICK *et al.*, 2008; FAHMY, 2016). Além dessas camadas, é comum encontrar em trabalhos com camadas específicas no modelo de rede, como o caso do protocolo 6LoWPAN, que se encaixa melhor em uma camada de adaptação, entre as camadas de rede e enlace (OLSSON, 2014; GARDASEVIC et al., 2015). A Figura 5 é uma representação do modelo de rede com conjunto de protocolos específicos para RSSFs (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007; YICK *et al.*, 2008; HUI; THUBERT, 2011; WINTER *et al.*, 2012; OLSSON, 2014; SHELBY *et al.*, 2014; IEEE..., 2015).

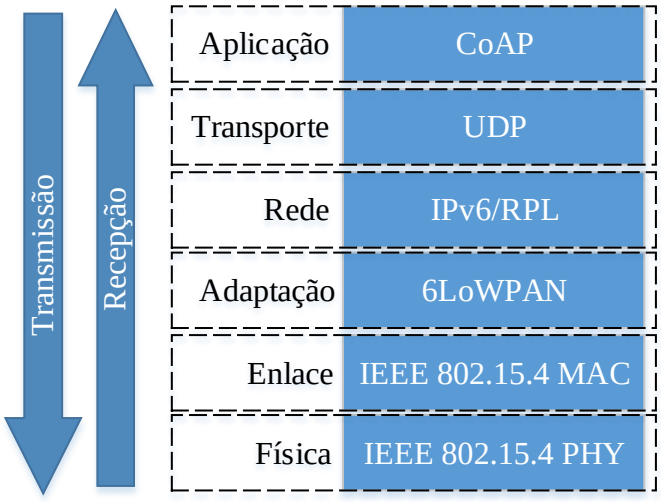


Figura 5 – Modelo de rede para RSSFs.
Fonte: Autoria própria.

Nesse Capítulo são abordados os protocolos utilizados nas camadas do modelo de rede para RSSFs, os protocolos convencionais, como o IPv6, UDP e TCP, e os protocolos específicos para RSSF como o IEEE 802.15.4 (IEEE..., 2015), o RPL (WINTER *et al.*, 2012), o 6LoWPAN (HUI; THUBERT, 2011) e o CoAP (SHELBY *et al.*, 2014). Além disso, são apresentados protocolos e padrões que podem ser utilizados em conjunto com RSSFs, como o RUDP e o LLC (*Logical Link Control*).

3.1 CAMADAS FÍSICA E DE ENLACE

As camadas Física e de Enlace (ou *Link* de Dados) são responsáveis pela forma na qual a informação será transmitida (pulsos elétricos, ondas de radiofrequência, sinais ópticos, etc.) e o modo de controle de acesso ao meio, respectivamente. Nessa camada pode-se citar como mais utilizados para comunicação sem fio os padrões IEEE 802.11, conhecido comercialmente como Wi-Fi, e o IEEE 802.15.1, denominado comercialmente como Bluetooth (TORRES, 2001; JAVVIN TECHNOLOGIES, 2005; SOHRABY *et al.*, 2007). Além desses padrões, o protocolo IEEE 802.15.4 foi desenvolvido para RSSFs, de modo que atenda as necessidades de redes que utilizem dispositivos com recursos limitados, conforme descrito na seção 3.1.1 (JAVVIN TECHNOLOGIES, 2005; SOHRABY *et al.*, 2007).

Em uma rede de padrão IEEE 802, a camada de Enlace se divide em duas subcamadas, a de Controle de Acesso ao Meio (MAC – *Medium Access Control*) e a camada de Controle do *Link* Lógico (LLC – *Logical Link Control*) (SOARES *et al.*, 1995; TORRES, 2001). Portanto, nas seções 3.1.2 e 3.1.3 é realizada uma descrição dessas camadas e dos métodos utilizados para redes de comunicação sem fio.

3.1.1 Padrão IEEE 802.15.4

O padrão de comunicação IEEE 802.15.4, desenvolvido pela IEEE, tem como propósito prover uma norma técnica de comunicação sem fio para dispositivos com baixa: capacidade, custo, consumo de energia e transmissão de dados (IEEE..., 2015). Para tal, esse padrão define as especificações das camadas física (PHY - *physical*) e do MAC para uma conexão de baixa taxa de transmissão de dados, que pode conter restrições energéticas. Além disso, esse protocolo provê que a camada física utilize a faixa de banda livre nas maiorias das regiões mundiais, podendo variar entre, aproximadamente, 470 MHz e 2450 MHz para dispositivos de baixo consumo energético (IEEE..., 2015).

Esse padrão foi desenvolvido de forma a atender dois modelos de redes: estrela e ponto-a-ponto, conforme demonstrado na Figura 6. Os dispositivos de função completa são aqueles que têm a capacidade de atuar como coordenador, enquanto o de função reduzida tem o proposito de associar-se a somente um de função completa por vez. Em ambos os modelos

de rede, a comunicação é estabelecida entre os dispositivos e uma única central de controle, denominado coordenador da PAN (*Personal Area Network* – Rede de Área Pessoal), sendo que cada dispositivo possui uma identificação única de endereço. Geralmente, o coordenador da PAN será o dispositivo conectado a uma fonte de energia, enquanto os outros poderão estar conectados há uma fonte de energia limitada, como uma bateria. Em cada rede independente há um único identificador (ID), o qual permite comunicação dentro da rede utilizando um endereço curto, e permite que tenha transmissões entre redes independentes. Entretanto, o mecanismo de escolha do ID e a forma que é realizada a formação da rede, não pertencem ao escopo desse padrão (IEEE..., 2015).

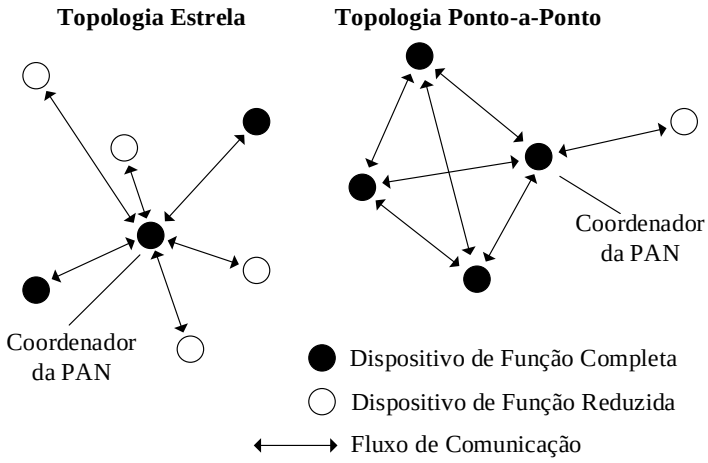


Figura 6 – Exemplos de topologias estrela e ponto-a-ponto.
Fonte: Adaptado de IEEE (2015, p. 46).

Analogamente à topologia ponto-a-ponto, há a topologia de *clusters* (agrupamentos), em que há agregação de duas ou mais redes ponto-a-ponto, conforme demonstrado na Figura 7. Dessa forma, é possível ter um melhor gerenciamento da rede, ou mesmo agregar redes distintas numa só. Nesse tipo de rede, cada *cluster* é composto por um único coordenador, que é responsável pela comunicação fora do *cluster*, com outros *clusters*, ou com um dispositivo de roteamento. Além disso, apesar de haver vários coordenadores distribuídos, há somente um coordenador geral, que é responsável pelo controle total da rede (IEEE..., 2015).

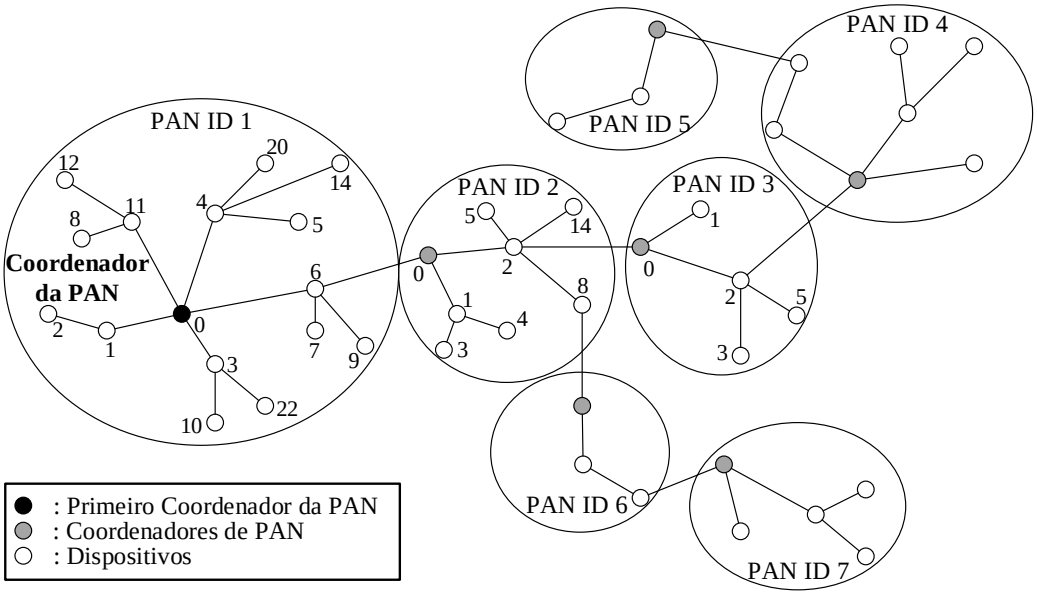


Figura 7 – Exemplo de rede por *cluster*.
Fonte: Adaptado de IEEE (2015, p. 48).

De forma a realizar o controle de acesso ao meio, o padrão IEEE 802.15.4 utiliza o formato de cabeçalho conforme a Figura 8, nos quais: MHR e MFR são o cabeçalho e rodapé do MAC, respectivamente, e FCS é o campo de Confirmação de Sequência de Quadro (*Frame Check Sequence*).

<table>
<tr><td>*Bytes*: 1/2</td><td>0/1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>-</td><td colspan="2">-</td><td>-</td><td>2/4</td></tr>
<tr><td rowspan="2">Quadro de Controle</td><td rowspan="2">Número da Sequência</td><td>PAN ID de Destino</td><td>Endereço de Destino</td><td>PAN ID de Origem</td><td>Endereço de Origem</td><td rowspan="2">Cabeçalho de Segurança Auxiliar</td><td colspan="2" rowspan="2">Elemento de Informação</td><td rowspan="2">Dados</td><td rowspan="2">FCS</td></tr>
<tr><td colspan="4">Campos de Endereçamento</td></tr>
<tr><td colspan="7">MHR</td><td>Cabeçalho</td><td>Dados</td><td>Dados MAC</td><td>MFR</td></tr>
</table>

Figura 8 – Formato geral do MAC do protocolo IEEE 802.15.4.
Fonte: Adaptado de IEEE (2015, p. 151).

O Quadro de Controle é composto por 1 ou 2 *byte*s, em que cada *bit* é representado de acordo com a Figura 9 (IEEE..., 2015).

Bits: 0-2	3	4	5	6	7	8	9	10-11	12-13	14-15
Tipo de Quadro	Habilitar Segurança	Quadro Pendente	Pedido de Confirmação	Compressão do PAN ID	Reservado	Suprimir Número de Sequência	Elemento de Informação Presente	Modo de Endereçamento de Destino	Versão do Quadro	Modo de Endereçamento de Origem

Figura 9 – Formato do Quadro de Controle do protocolo IEEE 802.15.4.
Fonte: Adaptado de IEEE (2015, p. 151).

As descrições de cada campo do Quadro de Controle mostrado na Figura 9 são (IEEE..., 2015):

- o Tipo de Quadro admite valores de 0 a 7, sendo eles designados, em ordem crescente, conforme: *Beacon*, Dados, Confirmação, Comando MAC, Reservado, Propósito Múltiplo, Fragmento e Estendido;
- o *bit* de segurança deve ser configurado para 1, se o dispositivo emissor do quadro tiver mais dados para o destinatário, caso contrário, deverá ser configurado como 0;
- o Quadro Pendente somente deve ser utilizado em quadros de *Beacon* ou quadros transmitidos durante o período de acesso ao conteúdo. Quando operando em modo de baixa energia com modo de amostragem coordenada, o quadro pendente pode ser utilizado para indicar que o dispositivo emissor contém quadros pendentes para ser enviado ao mesmo destinatário, de forma a manter o rádio ligado até que o *frame* seja configurado para 0. Quando operando em modo de salto de canal temporizado, o *bit* de quadro pendente deve ser configurado para 1, indicando que o destinatário deveria ficar ligado no próximo intervalo de tempo no mesmo canal de comunicação, caso não haja outra conexão programada;
- o Pedido de Confirmação especifica quando uma confirmação é necessária do destinatário, ao receber um quadro de dados ou comando MAC. Se esse campo é configurado para 1, o receptor deverá enviar o quadro de confirmação, caso contrário, não deverá enviar a confirmação de recebimento;
- o campo de compressão do PAN ID é utilizado para indicar a presença do campo de PAN ID. Sua configuração pode ser verificada com mais detalhes no capítulo “7.2.1.5 PAN ID Compression field” no documento descrito pela IEEE (2015, p. 153);

- o campo Suprimir Número de Sequência é configurado para 1, quando se deseja a omissão do número de sequência do pacote. Quando configurado para 0 o número de sequência esta presente no pacote. Se o quadro de versão for 0 ou 1, o número de sequência deverá ser 0;
- caso o Elemento de Informação esteja presente no pacote, esse campo deve ser configurado para 1, caso contrário 0. Se o Quadro de Versão for 0 ou 1, o campo de Elemento de Informação deverá ser 0;
- o Modo de Endereçamento do Destino pode ter quatro configurações, sendo uma delas reservada. Quando configurado para 0, os campos PAN ID e endereço não estão presentes. Quando configurado para 2, os campos utilizam o endereço curto de 16 *bit*s, e quando utilizado 3, é utilizado o endereço longo de 64*bit*s;
- a Versão do Quadro é um valor inteiro sem sinal, o qual especifica o número da versão do quadro correspondente. Para todos os tipos de quadros, esse campo deve ser configurado conforme tabela "7-4 – *Frame Version field values*" descrito por IEEE (2015, p. 155);
- o Modo de Endereçamento de Origem deve ser configurado conforme o Modo de Endereçamento de Destino. Se o campo é igual a 0 e o Tipo de Quadro especifica algum quadro de Dados ou de comando MAC, e a Versão do Quadro é 0 ou 1, o Modo de Endereçamento de Destino não deverá ser 0, implicando que o quadro foi originado de um coordenador PAN com PAN ID especificado no campo de PAN ID de Destino.

O campo de Número de Sequência na Figura 8 possui valores entre 0 e 255, o qual designará o identificador do quadro ao qual pertence. Os Campos de Endereçamento são utilizados para especificar a origem e destino de cada quadro, e devem ser configurados conforme especificações descritas pelo Quadro de Controle (IEEE..., 2015).

O Cabeçalho de Segurança Auxiliar especifica as informações necessárias para o processo de segurança do pacote. Esse campo deverá estar presente, somente se o campo de Habilitar Segurança estiver configurado corretamente no campo de controle. Para maiores informações verificar a capítulo "9.4 *Auxiliary security header*" documentado por IEEE (2015, p. 372).

O campo de Elemento de Informação (EI) não possui tamanho fixo, e pode ser composto por um conjunto de EI, não limitado por esse padrão. Esse campo deverá estar

presente somente se o Quadro de Controle estiver configurado adequadamente. Cada EI é composto por um decriptador e um dado opcional (IEEE..., 2015).

O Quadro de Dados contém informações específicas de cada tipo de quadro, e pode ser protegido criptograficamente, se configurado. Por fim, o campo Quadro de Confirmação de Sequência armazena o cálculo realizado sobre o pacote, conforme descrito em IEEE (2015, p. 156), de forma que se tenha uma confirmação matemática que as informações foram transmitidas corretamente. (IEEE..., 2015, p. 156).

3.1.2 Métodos de Controle de Acesso ao Meio

Devido à natureza das RSSF em utilizar um meio físico em comum entre os dispositivos para comunicação, é necessário realizar o controle de acesso ao meio para evitar colisões. O objetivo do protocolo MAC é regular o acesso ao meio sem fio compartilhado, de modo a satisfazer os requisitos de aplicação (SOHRABY *et al.*, 2007). Por estar situado acima da camada física, o MAC suporta as seguintes funções:

- montar dados em *frames* para transmissão, adicionando um cabeçalho com informações a respeito de endereçamento e detecção de erro;
- decodificar o *frame*, durante a recepção, e fazer o reconhecimento de endereço e detecção e recuperação de falhas;
- controlar o acesso ao meio compartilhado de acordo com as necessidades da aplicação.

Uma das maiores dificuldades ao desenvolver um protocolo MAC provém da distribuição espacial dos nodos transmissores. Para chegar a um consenso de qual nodo utilizará o meio, informações de coordenação devem ser trocadas entre eles. Entretanto, a troca de informações de coordenação utiliza o próprio meio a ser coordenado, o que incrementa a complexidade, sobrecarregando o método de regularização (SOHRABY *et al.*, 2007).

Entre os métodos mais utilizados para controle do meio, tem-se o *Carrier Sense Multiple Access* (CSMA) e suas variações, como: *Carrier Sense Multiple Access with Collision Detection* (CSMA/CD) e *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA), provenientes do protocolo ALOHA (KLEINROCK; TOBAGI, 1975). O CSMA

procura identificar quando o canal não está sendo utilizado durante um determinado intervalo de tempo. Caso não seja detectada uma comunicação, o meio está disponível para envio de mensagens. Apesar da verificação do meio, ainda é possível que ocorram transmissões simultâneas, podendo ocasionar erro no envio da informação desejada (SOHRABY *et al.*, 2007).

3.1.3 *Logical Link Control*

O *Logical Link Control* (LLC) é uma subcamada superior da camada de *Link* de Dados responsável por realizar multiplexação, controle de erro e de fluxo e definição de diferentes classes de serviços (SOARES *et al.*, 1995). Ele é regido pela norma ISO/IEC 8802-2:1998 e descontinuado em 2010 pelo padrão IEEE 802.2 (IEEE..., 1994; KEEN, 2011).

O LLC possibilita a multiplexação de protocolos nas camadas superiores a ele, como a camada de Rede do modelo OSI, ao utilizar pontos de comunicação entre o transmissor e o receptor, denominado SAP (*Service Access Point* – Ponto de Acesso a Serviços) (SOARES *et al.*, 1995; TORRES, 2001). De modo a evitar que cada protocolo da camada superior possua um SAP único, o que resultaria em manutenções extensivas do padrão LLC, optou-se por distribuí-los por fabricante, no qual um grupo de *bits* é destinado a representação da empresa e outro conjuntos de *bits* o protocolo. Desse modo, o órgão responsável pela manutenção do padrão não necessita cadastrar cada protocolo individualmente, possibilitando cadastrar um conjunto de protocolos por fabricante (TORRES, 2001).

O pacote LLC é formado por três campos, o DSAP (*Destination* SAP – SAP de destino) e o SSAP (*Source* SAP – SAP de origem) que são campos de um *byte* cada e o Controle, que varia entre um e dois *bytes* (SOARES *et al.*, 1995; TORRES, 2001). O tamanho de um *byte*, utilizado originalmente para o SAP, foi considerado insuficiente para cadastrar todos os protocolos existentes e os que ainda seriam criados. Portanto, foi criado um novo campo denominado SNAP (*Sub Network Access Protocol* – Protocolo de Acesso a sub-rede) de cinco *bytes* para endereçar os protocolos (TORRES, 2001). A Figura 10 apresenta a estrutura do quadro de controle LLC.

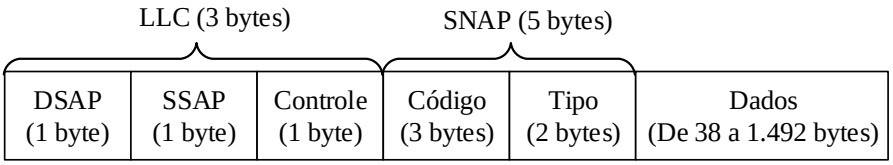


Figura 10 – Representação do quadro de controle LLC.
Fonte: Torres (2001, p. 57).

Para identificar que o SNAP está sendo utilizado, os campos DSAP e SSAP devem possuir o valor binário 10101010. O campo Controle pode conter três tipos de informações: UI (*Unnumbered Information*) utilizado quando se transmite dados; XID (*eXchange IDentification*) para realizar trocas de dados de identificação entre emissor e receptor; e Teste com finalidade de testar a comunicação entre transmissor e receptor. O campo Código armazena a informação do fabricante e o campo Tipo é o código fornecido pelo fabricante ao protocolo que está sendo utilizado (TORRES, 2001).

3.2 CAMADA DE REDE

Há vários protocolos que podem operar nessa camada, entre eles os clássicos: IP (*Internet Protocol*), ICMP (*Internet Control Message Protocol*), ARP (*Address Resolution Protocol*) e RARP (*Reverse Address Resolution Protocol*). Essa camada tem como propósito o gerenciamento das rotas entre os dispositivos (TORRES, 2001). Para as RSSF foram estudados meios de gerenciamento de rotas de baixo custo computacional, entre eles o RPL (*Routing Protocol for Low-Power and Lossy Network*) apresenta um desempenho aceitável para esse tipo de rede (WINTER *et al.*, 2012; ANCILLOTTI *et al.*, 2014). Além disso, outros métodos de otimização de comunicação são utilizados para reduzir tamanhos de cabeçalhos, sendo o 6LoWPAN um método específico para ser utilizado em redes que operem com protocolos IPv6 e UDP (OLSSON, 2014). Os protocolos IPv6, RPL e 6LoWPAN são explicados nas seções a seguir, pois são amplamente utilizados e otimizados para RSSFs.

3.2.1 *Internet Protocol v6* e 6LoWPAN

O *Internet Protocol* (IP) é um protocolo com intuito de dar nome aos dispositivos de uma rede a partir de endereços numerados, denominados endereços IP. Desse modo, é possível identificar cada dispositivo e a rede na qual está inserido. A partir dos endereços IP, o protocolo de roteamento é capaz de definir as rotas que podem ser utilizadas para entregar uma informação de um dispositivo a outro (TORRES, 2001).

Com a expansão de dispositivos conectados à rede global de comunicação, necessitou-se atualizar as especificações do protocolo IP, que atualmente utiliza-se duas versões, a IPv4 e IPv6. O endereçamento IPv4 possibilita a conexão de até 2^{32} ($4{,}2x10^{9}$) dispositivos únicos na rede e, até a data deste documento, ainda está em uso em redes empresariais e domésticas. Apesar de ter sido declarado o esgotamento dos endereços IPv4 em Fevereiro de 2011, foi possível adiar a data final ao utilizar técnicas de tradução de endereços de rede (NAT – *Network Address Translation*) (OLSSON, 2014).

A limitação do IPv4 exigiu o desenvolvimento do IPv6 na década de 90 e tem sido utilizado comercialmente desde 2006. O IPv6 possibilita o endereçamento de até 2^{128} ($3{,}4x10^{38}$) dispositivos, o que deve ser suficiente para as próximas décadas, mesmo com o crescimento de dispositivos utilizados em IoT (OLSSON, 2014). A Figura 11 apresenta o formato do cabeçalho IPv6.

0 15 31

Versão (4 *bits*)	Classe de Tráfego (8 *bits*)	Rótulo de Fluxo (20 *bits*)	
Tamanho da Carga (16 *bits*)		Próximo Cabeçalho (8 *bits*)	Limite de Saltos (8 *bits*)
Endereço de Origem (128 *bits*)			
Endereço de Destino (128 *bits*)			

Figura 11 – Cabeçalho IPv6.
Fonte: Adaptado de Deering e Hinden (1998, p. 4).

O campo Versão especifica qual a versão do cabeçalho, de valor 6 para o protocolo IPv6. O campo Classe de Tráfego permite identificar e distinguir pacotes IPv6 em classes ou prioridades. O Rótulo de Fluxo é utilizado pelo emissor para categorizar sequências de pacotes com requisições especiais aos roteadores, como qualidade de serviços não padrões ou

serviços de tempo real. Os campos Próximo Cabeçalho e Tamanho de Carga é utilizado para especificar qual o próximo cabeçalho da sequencia e qual seu tamanho, respectivamente. O campo Limite de Saltos especifica quantos saltos o pacote pode ser roteado, sendo decrementado a cada salto até atingir zero e ser descartado. Os endereços de Origem e Destino são utilizados para especificar quem está enviando a informação e para quem ela é destinada, respectivamente (DEERING; HINDEN, 1998).

Entretanto, ao aumentar a quantidade de endereços no protocolo IPv6, necessitou-se de modificações nas tecnologias utilizadas na camada de enlace, as quais foram possíveis com o avanço das tecnologias de transmissões Ethernet e Wi-Fi. Por outro lado, redes que utilizam o padrão IEEE 802.15.4 são afetadas por esse padrão, de modo a necessitar de técnicas para reduzir seus impactos (OLSSON, 2014).

O 6LoWPAN (IPv6 *over Low power Wireless Personal Area Networks*) é uma tecnologia de rede que possibilita enviar pacotes IPv6 de modo eficiente em redes de baixa capacidade, como a definida pelo padrão IEEE 802.15.4. Ele é um padrão especificado no documento RFC 6282 pela *Internet Engineering Task Force* (IETF), situando-se na camada de adaptação, conforme demonstrado na Figura 5. Ele é melhor descrito em uma nova camada. pois esse método prove uma adaptação do protocolo IPv6 para as camadas inferiores da rede (HUI; THUBERT, 2011; OLSSON, 2014; GARDASEVIC *et al.*, 2015).

O objetivo do 6LoWPAN é fornecer métodos para otimizar a transmissão de pacotes IPv6 em redes que utilizem o padrão IEEE 802.15.4 e similares. Para isso, implementa-se métodos para comprimir os cabeçalhos, fragmentar/desfragmentar as informações e de configuração automática de endereço de rede para os dispositivos (HUI; THUBERT, 2011; OLSSON, 2014).

3.2.2 RPL

O *Routing Protocol for Low-Power and Lossy Network* (RPL) é um protocolo de roteamento baseado no IPv6, para redes ruidosas e com dispositivos de recursos limitados, tais como: fonte de energia, potência para transmissões, processamento, memória, etc.. Ele foi investigado e padronizado pela IETF, *Internet Engineering Task Force*, em 2012. Essas redes, denominadas *Low-power and Lossy Networks* (LLNs), geralmente suportam baixo

tráfego de dados, podendo ser do tipo ponto a ponto, ponto para multipontos e até mesmo multipontos para ponto, geralmente com milhares de nodos (WINTER *et al.*, 2012).

As LLNs geralmente não contêm topologias pré-definidas, portanto o RPL tem que descobrir as ligações da rede e definir as rotas com objetivo de obter maior tráfego. Para definir as rotas, o RPL propõem Grafos Acíclicos Dirigidos com Destino Orientado (*Destination Oriented Directed Acyclic Graphs* - DODAGs), conforme demonstrado na Figura 12. Nessa topologia, há um dispositivo denominado DODAG *root* (raiz DODAG), que possui a função de agregar as rotas do DODAG e de operar como ponte entre redes distintas. Por simplicidade, o DODAG é representado por uma topologia de árvore, contudo, o RPL possibilita que os dispositivos contenham mais de uma rota. As rotas adotadas pelos dispositivos da DODAG podem se alterar dependendo das condições da rede, na qual cada mudança resulta em uma nova versão da DODAG (WINTER *et al.*, 2012).

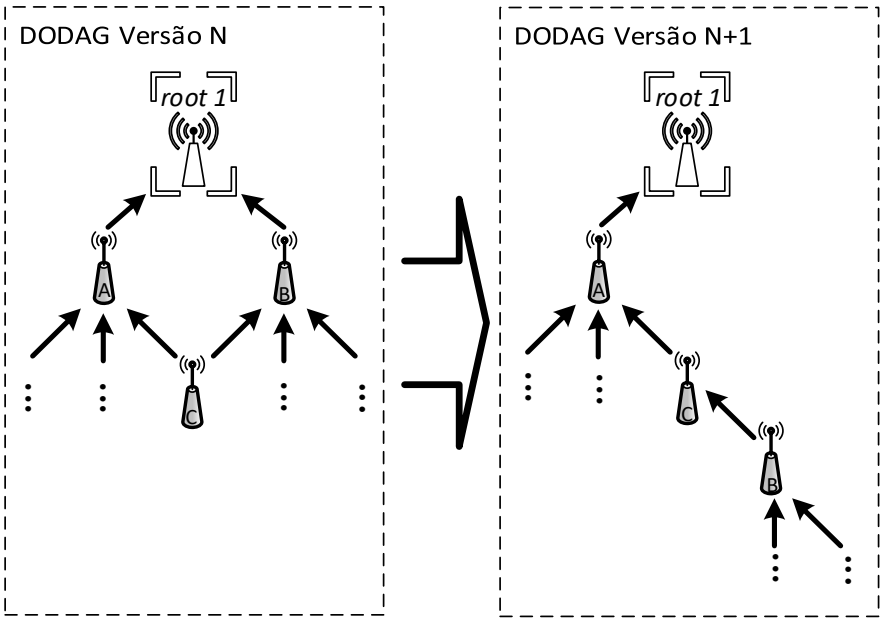


Figura 12 – Representação do DODAG. Comutação do nodo pai do dispositivo B, de *root* 1 para C, resultando em uma nova versão do DODAG.
Fonte: Adaptado de Winter *et al.* (2012, p. 115).

Entre os dispositivos é atribuída uma classificação por peso, ou custo, denominada *rank*, na qual representa a posição relativa do nodo em relação ao DODAG *root*. Para enviar uma mensagem ao nodo raiz, basta selecionar o caminho com menor *rank*. A Função Objetivo, *Objective Funtion* (OF), define como o RPL realiza o cálculo do *rank* e como é selecionado o nodo pai (WINTER *et al.*, 2012; ANCILLOTTI *et al.*, 2014). Entre as OFs, há duas funções padrões definidas pela IETF, a *Objective Function Zero* (OF0) e *Minimum Rank*

with Hysteresis Objective Function (MRHOF) (GNAWALI; LEVIS, 2012; THUBERT, 2012; WINTER *et al.*, 2012).

O protocolo RPL utiliza troca de mensagens entre nodos para estabelecer as rotas, pelo uso de um novo tipo de mensagem ICMP, denominado de mensagem de controle RPL (RPL *control message*). A construção do DODAG se inicia a partir do nodo raiz, que envia mensagens de informações DIO (DODAG *Information Object*) para anunciar o *rank* mínimo. Ao receber a mensagem DIO, o nodo estabelece seu *rank* de acordo com o valor mínimo, e seleciona o melhor caminho para alcançar o nodo raiz. Após se juntar ao DODAG, o nodo RPL inicia o envio de mensagens DIOs para continuar a construção da DODAG (WINTER *et al.*, 2012; ANCILLOTTI *et al.*, 2014). A Figura 13 apresenta os possíveis tratamentos de um DIO num roteador qualquer.

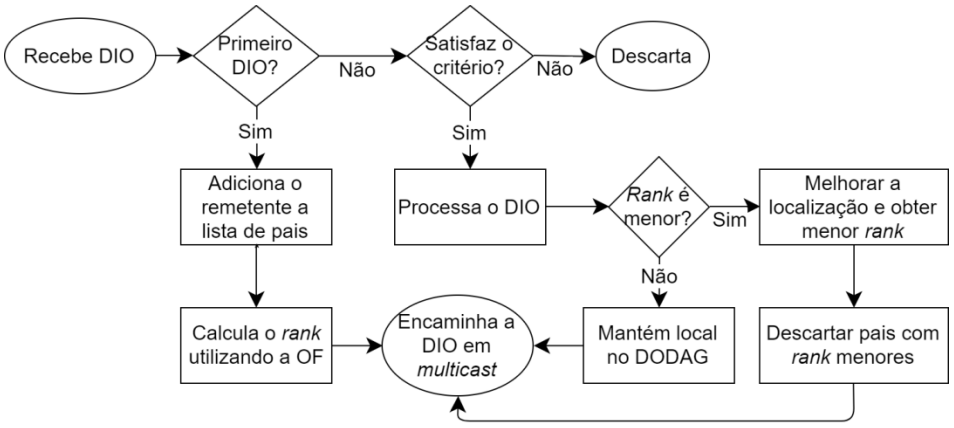


Figura 13 – Tratamento de um DIO numa DODAG.
Fonte: Adaptado de Gaddour e Koubâa (2012, p. 3167).

Nodos podem enviar mensagens aos seus vizinhos solicitando informação da DODAG por meio da DIS (DODAG *Information Solicitation*) e mensagens de anúncio de destino, DAO (*Destination Advertisement Object*), para notificar quais são seus pais. As DAOs são utilizadas para construir rotas descendentes (da raiz até o nodo filho) entre pares de nodos. Por fim, o algoritmo *Trickle* é utilizado para validar a consistência das informações sobre as rotas, aumentando e reduzindo a quantidade de mensagens DIOs emitidas (WINTER *et al.*, 2012; ANCILLOTTI *et al.*, 2014).

3.2.2.1 *Objective Function Zero*

Para um nodo fazer parte da DODAG com a OF0, é necessário garantir que ele possua boa conexão com um determinado conjunto de nodos. Embora não tenha garantias que a rota seja otimizada de acordo com a métrica especificada. Entretanto, o processo para realizar a conexão, sua validação, implementação e dependências não pertencem ao escopo de definição desta função (THUBERT, 2012). Para calcular o *rank* do nodo, é utilizada a equação (1).

$$R_N = R_P + (Rf \times Sp + Sr) \times MHRI \quad (1)$$

De forma que:

- R_N é o *rank* do nodo atual;
- R_P é o *rank* do nodo pai principal;
- Rf é o fator do *rank*, o qual multiplica o efeito das propriedades de conexão;
- Sp é o cálculo intermediário baseado nas propriedades de conexão entre vizinhos;
- Sr é a faixa de classificação, a qual atribui um valor máximo para selecionar um sucessor;
- MHRI é a constante mínima de incremento do *rank* por salto.

Portanto, o cálculo do *rank* de cada nodo é baseado somente pelo *rank* do nodo pai e pela qualidade da conexão entre ele e seu vizinho, o que o torna análogo a uma função de cálculo de rota por saltos (THUBERT, 2012).

3.2.2.2 *Minimum Rank with Hysteresis Objective Function*

A MRHOF foi desenvolvida para localizar as rotas com menor custo, enquanto tenta prevenir ações desnecessárias na rede. Para tal, o processo é dividido em duas etapas: I- localizar a rota com menor custo (ou seja, menor *rank*); II- trocas de rotas ocorrem somente se

o custo da rota for menor que determinado intervalo (faixa de histerese) (GNAWALI; LEVIS, 2012).

O cálculo do *rank* por essa OF pode ser obtido pelas seguintes métricas pré-definidas pela IETF: contador de salto, latência ou ETX (*Expected Transmission Count*). Apesar de serem pré-definidas, a MRHOF pode utilizar outras métricas que não sejam essa, contanto que sejam métricas (GNAWALI; LEVIS, 2012).

O ETX é uma métrica que consiste no número de transmissões esperadas para entregar com sucesso um pacote, definido pela equação (2).

$$ETX = \frac{1}{Df - Dr} \tag{2}$$

De forma que Df é a probabilidade que um pacote seja recebido por seu vizinho, e que Dr é a probabilidade de que o pacote de confirmação seja recebido. As normas sobre o ETX não especificam como devem ser realizados os cálculos de Df e Dr (VASSEUR *et al.*, 2012).

3.3 CAMADA DE TRANSPORTE

A camada de transporte fornece métodos de entregas de mensagens ponto-a-ponto, as quais são segmentadas na origem e montadas no destino. Essa camada não se preocupa em como a mensagem é entregue ou os mecanismos de transmissão utilizados. Os protocolos mais utilizados nessa camada para Internet, em arquiteturas convencionais de computadores, são: TCP (*Transmission Control Protocol*), descrito na seção 3.3.1 e UDP (*User Datagram Protocol*), descrito na seção 3.3.2 (SOHRABY *et al.*, 2007).

Por ser um protocolo orientado a conexão, o TCP tem a vantagem de detectar perda de informações durante as transmissões fim-a-fim. Entretanto, isso torna a transmissão mais lenta, ocasionando congestionamento em toda a rede. Portanto, esse protocolo é inadequado para muitas aplicações de RSSF, as quais necessitam de monitoramento em tempo real. Em contrapartida, o protocolo UDP possui a vantagem de proporcionar uma transmissão ágil. Protocolos com intuito de serem ágeis e robustos foram desenvolvidos para que sejam

melhores agregados à RSSF, como o RUDP exemplificado na seção 3.3.3 (YUNUS *et al.*, 2011; MASIRAP *et al.*, 2016).

3.3.1 *Transmission Control Protocol*

O *Transmission Control Protocol* (TCP) é um protocolo orientado à conexão, usualmente utilizado na Internet. Algumas das aplicações que utilizam o TCP para gerenciar suas funcionalidades, são o *File Transfer Protocol* (FTP) e o *Hypertext Transfer Protocol* (HTTP), para transferência de informações pela Internet. O TCP utiliza serviços fornecidos pelo protocolo IP (*Internet Protocol*) da camada de rede, com o objetivo de oferecer uma transmissão confiável, ordenada, controlável e com velocidade de transmissão adaptável entre emissor e receptor. Apesar das informações particionadas poderem chegar em ordem aleatória, o receptor contém as informações necessárias para realizar a ordenação das partições para formar a informação original (SOHRABY *et al.*, 2007). O TCP é composto por três fases de operações, sendo elas (SOHRABY *et al.*, 2007):

1. estabelecimento de conexão: o emissor envia uma mensagem de solicitação ao receptor para estabelecer uma conexão. Se o receptor estiver disponível e houver um caminho entre origem e destino, a conexão será estabelecida utilizando um enlace virtual;
2. transmissão de dados: após a conexão ter sido estabelecida, inicia-se a transmissão de dados entre transmissor e receptor. Durante a troca de informações, a taxa de transmissão em ambos os lados pode ser ajustada de acordo com congestionamentos. Como dados podem ser perdidos durante a transmissão, métodos de detecção de perda e recuperação de dados perdidos, podem ser utilizados;
3. desconexão: ao finalizar a transmissão de dados, a conexão é desabilitada. Em alguns casos, o receptor pode se tornar indisponível no meio da transmissão, o que ocasiona numa falha de conexão.

Para realizar o controle de fluxo e congestionamento, são realizados em três fases no protocolo TCP (SOHRABY *et al.*, 2007):

1. começo lento: por padrão, todas as transmissões iniciam lentamente, e conforme verifica-se sua estabilidade, incrementa-se a taxa de transmissão;
2. prevenção de congestionamento: após atingir uma determinada taxa de transmissão, o sistema entra em modo de prevenção de congestionamento, reduzindo o incremento da taxa de transmissão. Se ocorrer perda de dados, o protocolo retorna a sua fase de "começo lento". Confirmada a perda de pacote, o sistema entra em um estado de *Fast Recovery Fast Transmission* (FRFT);
3. FRFT: quando o sistema entra nesse estado, a taxa de transmissão é atualizada da mesma forma que no modo de prevenção de congestionamento. A razão desse estado é pelas falhas esporádicas de segmentos, às quais nem sempre indicam um elevado nível de congestionamento, e, portanto, não há necessidade de recomeçar do zero a verificação de congestionamento.

De acordo com o padrão estabelecido por Postel (1981), o cabeçalho TCP é composto por 24 *byte*s de informação para controle, conforme apresentado na Figura 14.

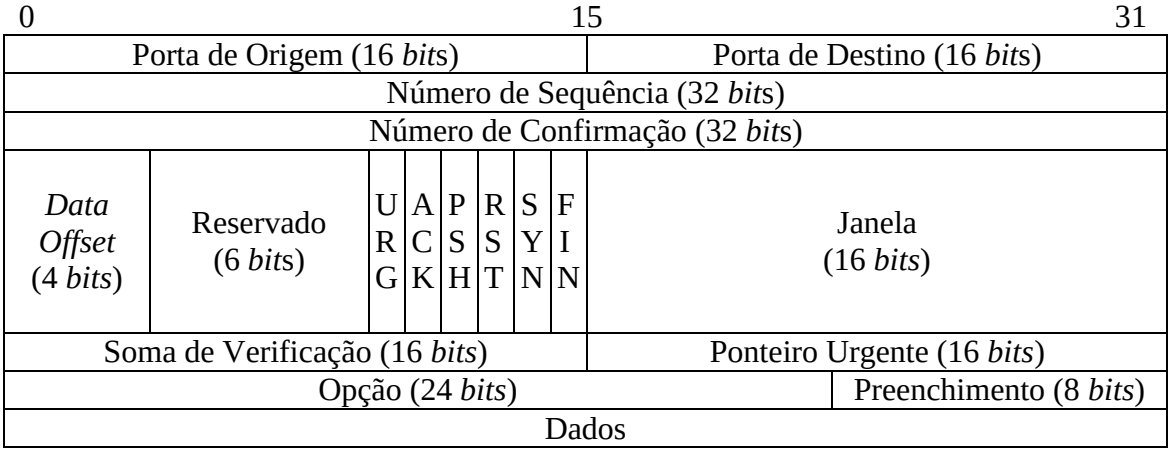


Figura 14 – Cabeçalho TCP.
Fonte: Adaptado de Postel (1981, p. 15).

As portas de origem e destino são identificadores, utilizados em conjunto com endereços de origem e destino, para especificar a qual processo o pacote pertence. Número de Sequência é o valor que representa o primeiro *byte* de um segmento (com exceção de quando o sinal SYN está ativo). Se o sinal SYN estiver ativo, o Número de Sequência é o número inicial da sequência (ISN), e o primeiro *byte* de dados é ISN+1 (POSTEL, 1981).

O Número de Confirmação, quando o campo ACK está ativo, contém o valor do próximo Número de Sequência que o emissor está esperando receber. *Data Offset*, ou

deslocamento de dado, indica o local da memória que inicia a sequência de dados. O campo reservado não possui designação e deve ser sempre zero. Em sequência, têm-se os *bit*s de controle, em ordem da esquerda para direita: ponteiro urgente; confirmação; função de *push*; reiniciar conexão; sincronizar números de sequência; fim dos dados do emissor (POSTEL, 1981).

O campo Janela indica a quantidade de *byte*s que o remetente desse segmento está disposto a aceitar. A Soma de Verificação é calculada pelo método de complemento de um com outras informações do cabeçalho IP para validar a integridade das informações do cabeçalho e dados. Ponteiro Urgente é utilizado para indicar quantos *bytes* são urgentes, com necessidade de serem processados com prioridade. Esse campo só é utilizado se o *bit* de controle URG estiver ativo. O campo Preenchimento é utilizado para garantir que o campo Opção termine em 32 *bits*, preenchendo com zeros caso for menor (POSTEL, 1981).

Apesar de possuir controle de fluxo, prevenção de congestionamento e meios de recuperar dados perdidos, o TCP não é recomendado para uso em RSSF pelos seguintes motivos (SOHRABY *et al.*, 2007):

- devido o tamanho de dados utilizados em RSSF serem geralmente pequenos, o processo de estabelecimento de conexão demanda muita memória e tempo para ser realizado para um volume de dados tão pequeno;
- perda de segmentos no protocolo TCP pode acionar os processos de controle de congestionamento, quando na realidade o segmento pode ser perdido por falha da conexão e não por congestionamento. Esse comportamento irá reduzir significativamente a taxa de transmissão, especialmente em redes que utilizam múltiplos saltos, como as RSSFs;
- o TCP utiliza um processo de controle de congestionamento fim-a-fim, o que, geralmente, resulta em longas esperas de respostas, ocasionando perda de segmentos. Com isso, há necessidade de retransmissões de dados, o que causa maior perda energética e menor taxa efetiva de transmissão;
- a confirmação de recebimento é realizada fim-a-fim, e quando necessário são realizadas transmissões do mesmo modo. Isso reduz a taxa efetiva de comunicação e aumenta o tempo necessário para o envio de dados;
- em RSSF, os sensores podem estar situados a alguns saltos de distância do seu destino. Nesse cenário, sensores próximos ao coletor de dados devem receber

mais oportunidades de transmissão, ocasionando uma perda de energia maior. Além disso, nodos mais distantes podem ser afetados por desconexões da rede.

3.3.2 *User Datagram Protocol*

O *User Datagram Protocol* (UDP) é um protocolo que não é dependente de conexão e não possui velocidade de transmissão adaptável entre emissor e receptor. Ele troca dados sem informações de sequenciamento, ou seja, não há garantia de entrega de dados na ordem correta. Caso um pacote seja perdido, o protocolo não possui métodos de recuperação. Além disso, não há formas para controle de congestionamento, nem controle de fluxo (SOHRABY *et al.*, 2007).

O cabeçalho do protocolo UDP é formado por 8 *byte*s de cabeçalho de controle, composto por: porta de origem, porta de destino, tamanho e soma de verificação, conforme apresentado na Figura 15. Assim como no TCP, as portas de origem e destino são identificadores, utilizados em conjunto com endereços de origem e destino, para especificar à qual processo o pacote pertence. O campo Tamanho especifica o tamanho em *bytes* do cabeçalho UDP mais a quantidade de dados que está transportando. A Soma de Verificação é calculada pelo método de complemento de um com outras informações do cabeçalho IP para validar a integridade das informações do cabeçalho e dados (POSTEL, 1980).

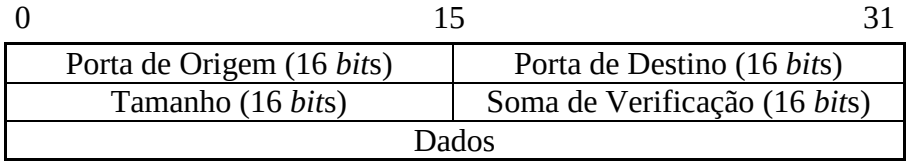


Figura 15 – Cabeçalho UDP.
Fonte: Adaptado de Postel (1980).

Pela falta de controle de fluxo e congestionamento, assim como pela falta de métodos de confirmação de entrega de informações, não é recomendado utilizar o protocolo UDP em RSSF. Entretanto, por ser o mais simples, foi utilizado como base para criar outros protocolos com maior confiabilidade e de baixa latência, tais como: *Reliable* UDP (RUDP), *Performance Adaptative* UDP (PA-UDP), *Pump Slowly, Fetch Quickly* (PSFQ), *Enhanced-*

RUDP (E-RUDP), UDP-*based Data Transfer* (UDT) e *Reliable Dynamic Buffer* UDP (RDBUDP) (YUNUS *et al.*, 2011; MASIRAP *et al.*, 2016).

3.3.3 *Reliable* UDP

O protocolo *Reliable* UDP (RUDP) é uma versão do protocolo UDP, o qual foca em melhorar a confiança de entrega de mensagens, utilizando um número máximo de retransmissões (BOVA; KRIVORUCHKA, 1999). Esse protocolo deve obedecer aos seguintes critérios:

- o transporte deve fornecer uma entrega satisfatória com um número máximo de retransmissões, de modo a se evitar mensagens velhas;
- o transporte deve ocorrer de forma ordenada;
- o transporte deve ser baseado em mensagens;
- o transporte deve ter controle de fluxo;
- o transporte deve ter baixa sobrecarga e alto desempenho;
- as características de cada conexão virtual devem ser configuráveis;
- o transporte deve ter um método de “manter ativo”;
- o transporte deve ter um método de detecção de erro;
- o transporte deve prover uma transmissão segura.

O RUDP foi definido com o propósito de ser configurável individualmente. Desta forma, outros protocolos de transporte podem ser utilizados simultaneamente (BOVA; KRIVORUCHKA, 1999). Portanto, ao invés de ser um novo protocolo, ele é interpretado como uma nova camada dentro da camada de transporte, conforme Figura 16 (a).

A Figura 16 (b) demonstra como é realizada a transmissão da informação entre dois dispositivos, sendo PKT o pacote transmitido e ACK a confirmação de recebimento. Nota-se que o receptor retorna o pacote, para que o emissor confira se a informação foi transmitida corretamente. O emissor conferindo que o pacote está correto, alerta o receptor, com uma nova mensagem de confirmação, que o pacote pode ser encaminhado ao seu destino (aplicação ou roteamento) (MORA, 2005).

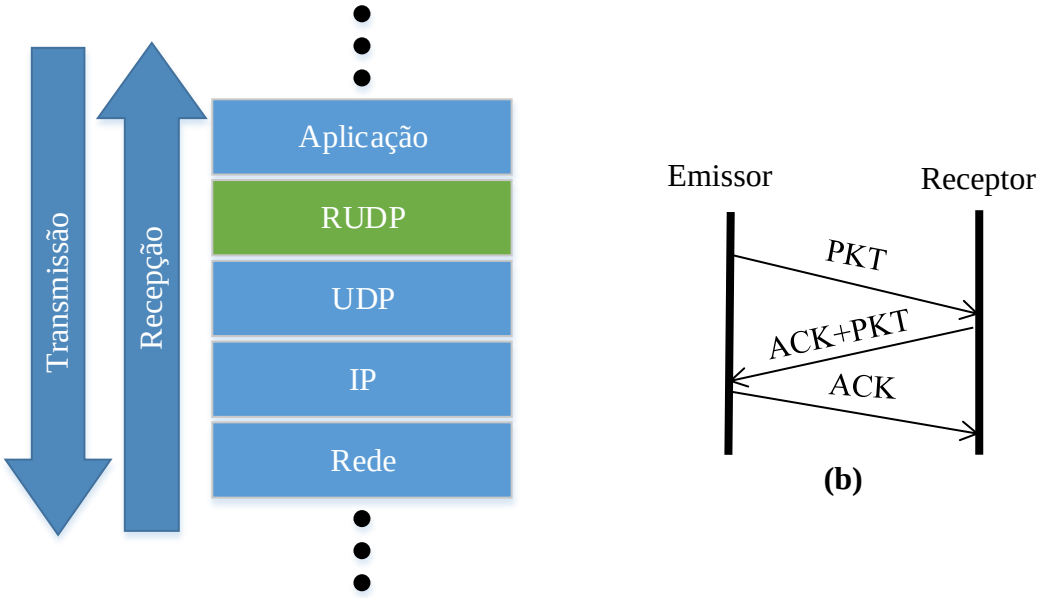


Figura 16 – Protocolo RUDP. (a) Modelo de rede com RUDP. (b) Modo de transmissão utilizando RUDP.
Fonte: Adaptado de Mora (2005).

O cabeçalho do protocolo RUDP é dividido conforme Figura 17 (BOVA; KRIVORUCHKA, 1999).

0	1	2	3	4	5	6	7	8 ... 15
SYN	ACK	EAK	RST	NUL	CHK	TCS	0	Tamanho do Cabeçalho (8 *bits*)
Número da Sequência (8 *bits*)								Número de Confirmação (8 *bits*)
Soma de Verificação (16 btis)								

Figura 17 – Cabeçalho RUDP.
Fonte: Adaptado de Bova e Krivoruchka (1999, p. 3).

Os dados de controle indicam quais propriedades estão compostas no pacote, conforme:

- SYN: indica se há o segmento de sincronização;
- ACK: representa se o número de verificação presente no cabeçalho é valido;
- EACK: indica que um pacote de confirmação estendido está presente;
- RST: indica se o pacote é um segmento de reinício;
- NUL: representa um segmento nulo;

- TCS: indica se o pacote faz parte do segmento de estabelecimento de conexão;
- CHK: indica se o campo de verificação de soma é referente somente ao cabeçalho, ou referente ao cabeçalho e dados;
- Tamanho do Cabeçalho: indica que posição os dados iniciam no pacote;
- Número da Sequência: informa o número da sequência do pacote, escolhido aleatoriamente no início da conexão, é incrementando a cada transmissão;
- Número de Confirmação: informa ao emissor o último pacote válido recebido;
- Soma de Verificação: campo contendo um código para verificação de integridade do cabeçalho.

3.4 CAMADA DE APLICAÇÃO

A camada de aplicação realiza função de interface entre o aplicativo que requisitou ou enviou alguma informação e o protocolo responsável pelo gerenciamento da comunicação, da camada de transporte. Essa camada é composta por diversos tipos de protocolos, com as mais diversas funções e objetivos, e os mais comuns utilizados em redes convencionais são (TORRES, 2001):

- HTTP (*Hyper Text Transfer Protocol*): Utilizado para realizar transferência de documentos hipermídia, acessados por meio de endereços IP ou por nomes fornecidos por DNS;
- DNS (*Domain Name Server*): Utilizado para identificar máquinas a partir de nomes ao invés de endereços IP;
- FTP (*File Transfer Protocol*): Utilizado para realizar transferências de arquivos;
- Telnet: Utilizado para realizar acesso remoto a maquina;
- SMTP (*Simple Mail Transfer Protocol*): Utilizado para enviar e receber e-mails.

Entretanto, diferente das redes convencionais, as RSSFs necessitam de métodos específicos para gerenciamento de dados, como gerenciamento e fusão de dados, sincronização de temporizadores e posicionamento. Muitas aplicações necessitam do posicionamento do sensor para que sua informação tenha algum significado, portanto cada dispositivo necessita saber sua exata posição, ou com taxas de desvio aceitáveis. Além do

posicionamento, as informações coletadas pelos sensores podem ser sensíveis em relação ao tempo que foi obtida, em horas, minutos, segundos, etc., portanto técnicas de sincronismo de temporizadores podem ser cruciais dependendo da aplicação desejada (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007; FAHMY, 2016).

Técnicas de fusão de dados são utilizadas para contornar falhas dos sensores, suas limitações de recursos e problemas de cobertura espacial e temporal. Elas, geralmente, são utilizadas para combinar informações originárias de diversas fontes, para que possam atingir maior eficácia na entrega de informações precisas, completas e confiáveis, do que se obtidas por uma única fonte (STOJMENOVIĆ, 2005; SOHRABY *et al.*, 2007; FAHMY, 2016). Além disso, técnicas de gerenciamento de dados podem envolver métodos para garantir a entrega de informação, como o caso do CoAP que fornece mecanismos de confirmação de entrega de pacote (SHELBY *et al.*, 2014).

3.4.1 *Constrained Application Protocol*

O *Constrained Application Protocol* (CoAP) é um protocolo que agrega serviços *Web* e métodos de controle de comunicação, para redes de dispositivos com recursos limitados (processamento, energia, redes, etc.). Foi desenvolvido para ser compatível com o HTTP (*Hypertext Transfer Protocol*), o que facilita sua integração com a rede *Web*, enquanto fornece métodos para atender as especificações de redes M2M (Máquina a Máquina) restritas de recursos, como: suporte a mensagens *multicast*, cabeçalhos de controle de tamanho reduzidos, troca de mensagens assíncronas e simplicidade. O CoAP contém métodos para emitir pedidos e respostas entre dispositivos fim-a-fim, formas para descobrir serviços e recursos, e incluí conceitos-chave de rede, como: URIs (*Uniform Resource Identifier*) e tipos de mídia de *Internet* (SHELBY *et al.*, 2014).

Apesar de ser definido como um protocolo de aplicação, o CoAP possui métodos de controle de comunicação, possibilitando habilitar métodos de confiabilidade para conexões UDP, suportando pedidos *unicast* e *multicast*. Para isso são utilizadas quatro tipos de mensagens: Confirmáveis (CON – *Confirmable*), Não Confirmáveis (NON – *Non-confirmable*), Confirmação (ACK – *Acknowledgement*) e Restabelecer (RST – *Reset*). As mensagens CON são aquelas que o emissor aguardará o retorno de uma mensagem ACK do receptor, confirmando fim-a-fim que a mensagem foi enviada devidamente. Por outro lado, as

mensagens NON não aguardam confirmação de recepção, de modo a desabilitar o método de confiabilidade. Quando um receptor não consegue processar, ou gerar uma mensagem de erro, de uma mensagem CON, o receptor envia uma mensagem RST ao invés do ACK. Cada mensagem CON e NON gerada possui um identificador de mensagem (*Message* ID) de dois *bytes*, e as mensagens ACK e RST devem possuir o mesmo ID da mensagem que recebeu. A Figura 18 representa as trocas de mensagens CON entre dois dispositivos, cliente e servidor, ao realizar um pedido de um dado de temperatura pelo cliente (SHELBY *et al.*, 2014).

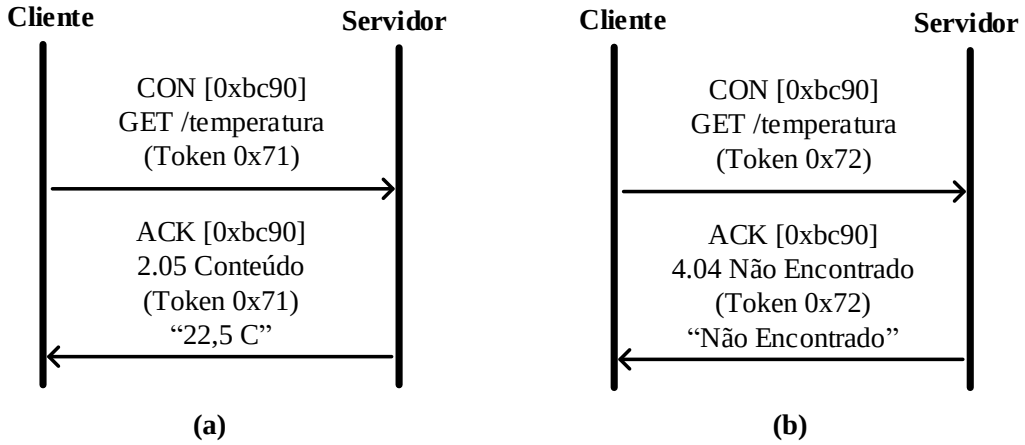


Figura 18 – Exemplo de troca de mensagens utilizando o CoAP. (a) O cliente envia a requisição da temperatura para o servidor e ele retorna uma mensagem de confirmação com a informação requisitada. (b) O servidor não encontra o valor requisitado, retornando a mensagem de confirmação com o erro específico.
Fonte: Adaptado de Shelby *et al.* (2014, p. 13).

O CoAP realiza trocas de mensagens compactas que, por padrão, são transportadas pelo protocolo UDP. Para isso, suas mensagens são codificadas em um formato binário, com um cabeçalho fixo de 4 *bytes*, seguido de campos opcionais de tamanhos variados, conforme demonstrado na Figura 19.

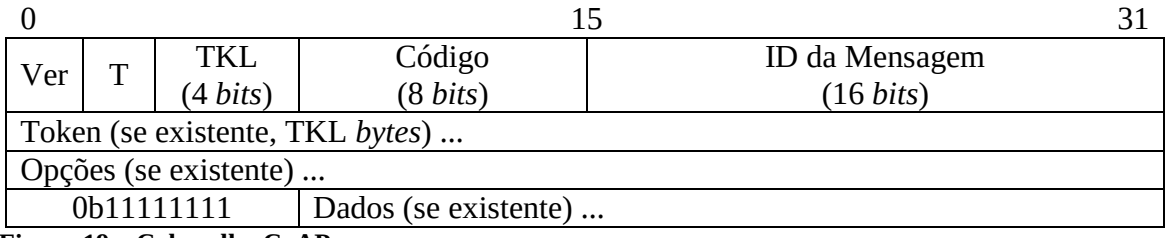


Figura 19 – Cabeçalho CoAP.
Fonte: Adaptado de Shelby *et al.* (2014, p. 16).

As definições de cada campo do cabeçalho CoAP são:

- Ver (versão): campo de valor inteiro sem sinal de 2 *bits* utilizado para indicar a versão do protocolo CoAP;
- T (Tipo): campo de valor inteiro sem sinal de 2 *bits* que indica o tipo da mensagem, 1 - CON, 2 - NON, 3 - ACK ou 4 – RST;
- TKL (*Token Length*): campo de valor inteiro sem sinal de 4 *bits* que demarca o tamanho do campo *Token*, de 0 a 8 *bytes*. Valores maior ou igual a 9 não devem ser utilizados e devem ser tratados como erro de formato se encontrados, pois são reservados;
- Código: campo de valor inteiro sem sinal de 8 *bits* utilizado para caracterizar o tipo da mensagem, podendo ser uma requisição ou resposta. Os valores suportados por esse padrão estão tabelados em Shelby *et al.* (2014, p. 87-88);
- ID da Mensagem: campo de valor inteiro sem sinal de 16 *bits* utilizado para detectar mensagens duplicadas e na validação do envio da mensagem pelo ACK ou RST;
- *Token*: campo de tamanho TKL *bytes* utilizado para relacionar requisições e respostas. O cliente deveria gerar *Tokens* de pares únicos entre origem e destino, entretanto, se não utilizado outro *Token* no destino, é possível omiti-lo;
- Opções: campo de tamanho variável que possui opções que devem conter em ordem seu tipo, tamanho e valor. As opções disponíveis podem ser verificadas em Shelby *et al.* (2014, p. 90);
- 0b11111111: campo de valor binário 11111111 designado para marcar o fim das opções e o início do campo de dados. Se existe dados no cabeçalho CoAP esse campo deve aparecer antes, caso contrário é omitido, indicando que o tamanho do campos Dados é zero;
- Dados: campo de tamanho variável que armazena a informação que está sendo transmitida pelo protocolo.

3.5 CONCLUSÃO

Para ser possível utilizar dispositivos com capacidade reduzida de processamento, armazenamento e comunicação, foram efetuadas análises dos modos de comunicação e gerenciamento de rede. Dessa forma, protocolos específicos para transmissão, controle de rotas e controle de rede, foram definidos para que seja possível realizar uma comunicação eficiente.

Conforme descrito no capítulo sobre camada de transporte, os protocolos TCP e UDP são amplamente utilizados em redes convencionais. Entretanto, estudos comprovam que os métodos do TCP são ineficientes para RSSF, pois sua latência é elevada para este propósito e seu cabeçalho ocupa grande parte do espaço de pacote disponível pelo protocolo IEEE 802.15.4. Por outro lado, apesar do protocolo UDP ter uma latência inferior, não há garantias de que a informação seja devidamente transmitida ao receptor (SOHRABY *et al.*, 2007). Tais estudos evidenciam a necessidade de novos métodos de controle de transmissão, que sejam mais robustos, de baixa latência e energeticamente eficientes.

Protocolos como RUDP e o CoAP foram desenvolvidos com esse propósito. Apesar deste trabalho citar o RUDP por ser o possível precursor de outros protocolos, observa-se que há outros métodos citados nas literaturas que demonstram ser mais eficientes em determinadas condições (YUE *et al.*, 2011; YUNUS *et al.*, 2011; MASIRAP *et al.*, 2016). Contudo, não há uma metodologia padrão ótima utilizada neste meio.

Apesar das técnicas citadas neste capítulo auxiliarem na estabilidade da rede, ainda é notável em sistemas reais suas falhas, pois são suscetíveis à instabilidade por fatores externos. Portanto, há uma necessidade de aprimoramento das técnicas adotadas, de forma a serem mais eficazes para RSSF. No capítulo 4 são evidenciados alguns parâmetros para o entendimento da estrutura de uma rede de comunicação, e como certos parâmetros podem ser cruciais ao seu desempenho.

4 ESTUDOS EM SISTEMAS DE FLUXO E FILAS

Como demonstrado no Capítulo 2, as RSSF são agregadoras de dados, que possuem aplicações que atendem aos mais diversos setores. Contudo, é necessário ressaltar suas imperfeições, as quais podem apresentar falhas críticas à aplicação. Como forma de aprimorar a QoS numa rede de comunicação, são estudados métodos matemáticos de controle de fluxo de dados. Entre os quais, teorias sobre filas apresentam ampla aplicabilidade em redes de computadores, pois estes são dependentes de filas para coordenar e controlar as transmissões de dados. Portanto, o propósito deste capítulo é apresentar os conceitos sobre teoria de sistemas de fluxo e filas e demonstrar os resultados de estudos que utilizam estas metodologias.

4.1 DEFINIÇÃO DE SISTEMAS DE FLUXO

Um sistema de fluxo é aquele que provê meios de transportar determinado objeto por canais de capacidade finita de um ponto ao outro. Capacidade finita refere-se ao fato de ser possível realizar o transporte de modo a satisfazer a demanda exigida numa taxa finita, por exemplo: o caixa de supermercado, o qual é capaz de contabilizar um único produto por vez (KLEINROCK, 1975).

Sistemas de fluxo podem ser caracterizados em duas classes principais: de fluxo estável ou instável. O primeiro caso refere-se aos sistemas de fluxo que operam numa maneira preditiva, ou seja, a quantidade do fluxo é bem definida e é constante durante o intervalo de interesse. Tal sistema é trivial de ser analisado em sistemas de canais únicos, sendo muito comum sua utilização em fábricas baseadas em linhas de produção. Entretanto, tal sistema se torna interessante ao utilizar múltiplos canais de transporte. Para saber qual o fluxo máximo nesse cenário, é necessário descobrir todas as possíveis rotas que podem ser adotadas pelo sistema, e dessa forma analisar uma a uma, verificando qual combinação resulta no maior fluxo permitido (KLEINROCK, 1975).

A classe de fluxo instável é categorizada por problemas de fluxos aleatórios ou estocásticos. Isto significa que, o tempo de utilização do canal, bem como a quantidade de fluxo que será utilizada, não pode ser prevista. Tais características tornam estes problemas de

natureza instável e complexos de serem estudados. Contudo, representam grande parcela dos sistemas do mundo real. Em comparação com a classe de fluxo estável para sistemas com canais únicos, esta classe de problemas não é trivial, tornando-se assunto de estudo sobre técnicas de como melhorar seu fluxo (KLEINROCK, 1975).

Por exemplo, considere um centro computacional o qual é responsável pelo processamento de cálculos matemáticos, desde os simples até os mais complexos, realizados a partir de pedidos aleatórios. Nesse cenário, os pedidos podem ocorrer enquanto a central está em modo de espera, como também podem ocorrer quando a central realiza algum processamento mais complexo. Ou seja, nem a central computacional, nem o responsável pelo pedido, podem definir o consumo computacional total, nem o fluxo exigido pelo sistema, o que o torna um sistema não preditivo e de difícil otimização (KLEINROCK, 1975).

4.2 TEORIA DE FILAS

O método para denotar sistemas de enfileiramento segue o padrão definido por Kendall (1953 *apud* FOGLIATTI; MATTOS, 2007, p. 10) *A/B/C/D/E*, de forma que: *A* – distribuição do tempo entre chegadas sucessivas; *B* – distribuição do tempo de serviço (atendimento); *C* – quantidade de servidores; *D* – capacidade de armazenamento do sistema; *E* – disciplina de atendimento. Caso os dois últimos parâmetros não estejam especificados, é interpretado que seu valor é infinito. Além disso, *A* e *B* são representadas a partir de funções do tipo M (exponencial), E_r (distribuição Erlang tipo *r*), D (determinístico) e G (geral), podendo haver outros símbolos especiais definidos ocasionalmente. Por exemplo, D/M/2/20 representa um sistema genérico composto por dois servidores com tempo entre chegada constante (determinístico), com tempo de serviço distribuído de modo exponencial e com capacidade de armazenamento de 20 (KLEINROCK, 1975; FOGLIATTI; MATTOS, 2007).

As disciplinas de atendimento, estabelecido pelo parâmetro *E*, representam a forma como os dados das filas são tratados e podem adotar as seguintes formas conhecidas:

- FIFO (*first in – first out*): o primeiro a ser inserido, será o primeiro a ser atendido;
- LIFO (*last in – first out*): o último a ser inserido, será o primeiro a ser atendido;
- PRI (*priority service*): atendimento com prioridades preestabelecidas;
- SIRO (*service in random order*): o atendimento ocorre de forma aleatória.

A Teoria das Filas provê métodos de avaliar a eficiência de um sistema a partir de suas características, as quais geralmente são mutáveis ao decorrer do tempo. Por serem mutáveis no tempo, sua modelagem é realizada a partir de variáveis aleatórias, às quais podem ser utilizadas como medidas de desempenho do sistema em regime estacionário (FOGLIATTI; MATTOS, 2007).

Um dos modelos de fila mais simples é o D/D/1/K/FIFO, o qual é possível determinar a distribuição dos tempos entre entradas de clientes e de serviço, com um único servidor com limite de *K* clientes, num sistema de atendimento FIFO. Analisando esse modelo, define-se que o tempo entre chegadas sucessivas é de $1/\lambda$, ou seja, λ é a taxa de chegadas sucessivas, e que o tempo de atendimento é $1/\mu$, ou seja, μ é a taxa de atendimentos. Se $\lambda \leq \mu$ não há formação de fila, sendo que todos os clientes serão devidamente atendidos antes da formação da fila. Entretanto, para $\lambda > \mu$ a fila poderá crescer até o limite estabelecido *K* (FOGLIATTI; MATTOS, 2007).

4.3 PROCESSOS MARKOVIANOS

Um processo markoviano é um tipo de processo estocástico em que dada a informação atual, qualquer informação passada é irrelevante para determinar o estado futuro. Esse processo não possui memória (*memoryless*), pois não há necessidade de armazenar dados. Processos markovianos de espaço de estado discreto são denominados cadeias de Markov, às quais podem ser representadas por um diagrama de fluxo conforme Figura 20, em que cada nodo representa um estado, e os arcos representam as transições entre estados no decorrer do tempo (FOGLIATTI; MATTOS, 2007).

Entretanto para representar melhor os processos de filas, são utilizados processos de "nascimento e morte" (*Birth-Death Process*), os quais podem ser descritos por uma cadeia de Markov homogênea, irredutível e de parâmetro contínuo, na qual é possível transitar somente por seus vizinhos imediatos. Uma cadeia de Markov é denominada homogênea se a probabilidade de transacionar não é alterada conforme o estado. Se há alguma forma de acessar um estado da cadeia a partir de qualquer ponto inicial, a cadeia é dita irredutível (FOGLIATTI; MATTOS, 2007).

A mudança de estados de n para $n + 1$, representa o processo de nascimento, a qual pode ocorrer numa taxa λ_n. O processo de morte é o oposto, quando a transição de estado

ocorre de n para $n-1$, contanto que $n > 0$, e pode ser estabelecido numa taxa μ_n. Na Figura 20 verifica-se como ocorre esse processo de nascimento/morte conforme suas taxas (FOGLIATTI; MATTOS, 2007).

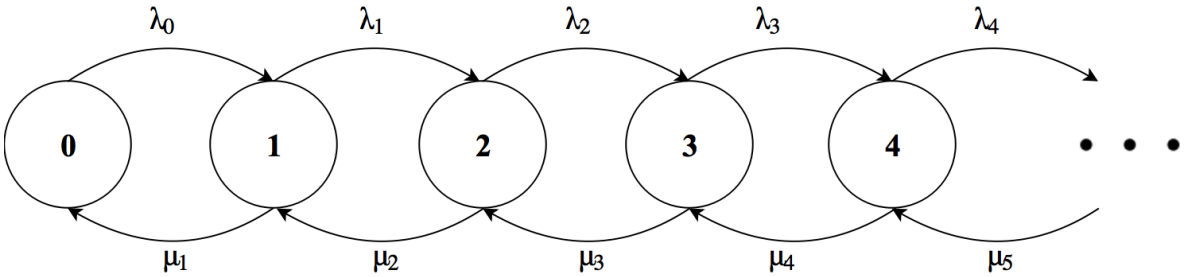


Figura 20 – Diagrama de fluxo de uma cadeia de Markov com processo de nascimento e morte.
Fonte: Fogliatti e Mattos (2007, p. 27).

4.4 ESTUDOS EM RSSF

Com base nas teorias de fluxo e filas, pesquisas têm sido realizadas com o enfoque em como melhorar o desempenho de rede de sensores sem fio ao analisar o *buffer* utilizado pela rede. A seguir é realizada uma breve análise de alguns trabalhos, ressaltando seus pontos relevantes.

4.4.1 O Paradoxo de Alocação

Baron *et al.* (2009) demonstram que uma rede estável pode se tornar instável ao incrementar o fluxo da rede, fenômeno denominado CAP (*Capacity Allocation Paradox*). De acordo com a teoria de filas, é possível estabilizar uma fila ao aumentar a taxa do tempo de serviço μ de uma fila com taxa de entrada λ, contanto que $\lambda < \mu$ (KLEINROCK, 1975). Da mesma forma que as teorias de informação demonstram que aumentar a capacidade de um canal C pode ajudar a transmitir a maioria dos códigos de informação numa taxa R, com uma taxa de erro pequena, contanto que $R < C$ (COVER; THOMAS, 1991 *apud* Baron *et al.*, 2009, p. 1359).

Baron *et al.* (2009) demonstra que ao adicionar mais capacidade no sistema seu desempenho pode não melhorar, com possibilidade de torná-lo instável. A Figura 21, por exemplo, representa uma rede simples de 2x1, na qual há duas fontes de informação A e B, um roteador R e um destino C. As fontes geram pacotes em taxas r_A e r_B, respectivamente, para seus respectivos *buffers* em R, usando a rota de capacidade de transmissão C_A e C_B. R, por sua vez, escalona os pacotes recebidos das fontes, e os envia para C, por meio de um canal de capacidade de transmissão C_R. Se os *buffers* finitos do roteador estiverem cheios, as fontes armazenam suas informações em seus próprios *buffers*. Na Figura 21, a fonte A não pode mais transmitir seus dados, pois o *buffer* correspondente no roteador encontra-se cheio, enquanto B pode enviar seus dados. Essa rede é considerada estável, se o tamanho esperado da fila estiver delimitado.

Incrementando a capacidade de transmissão de C_A ou C_B a rede pode se tornar instável. Por exemplo, ao aumentar a taxa de C_A, o *buffer* de R fica cheio, fazendo com que dessa forma o *buffer* de B também fique cheio, pois R não conseguirá manter o fluxo necessário. Esse fenômeno pode ocorrer com qualquer tamanho de rede, com qualquer tamanho de *buffer*, apesar de ser mais comum em redes com *buffers* pequenos.

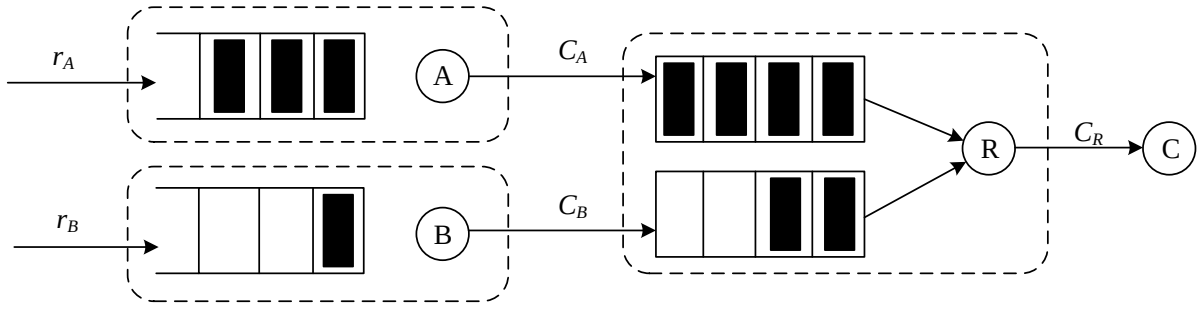


Figura 21 – Rede simples 2x1.
Fonte: Baron *et al.* (2009, p. 1359).

Para verificar esse fenômeno, são analisadas diversas técnicas de escalonamento no roteador, entre elas: prioridade fixa, *round-robin* e *round-robin* exaustivo. Além disso, são utilizadas três configurações diferentes de fluxo, sendo elas: tráfico fluido (sem *buffer*), *wormhole-switched* e, por fim, compartilhamento geral do processador (GPS – *General Processor Sharing*).

Baron *et al.* (2009) descreve as formas de trazer o sistema para estabilidade novamente, sendo uma das formas, não intuitiva, reduzir a velocidade de transmissão dos

fluxos estáveis. De outra forma, é possível adicionar mais capacidade, velocidade de transmissão e *buffer*, aos fluxos instáveis. Por último, trocar o método de escalonamento.

Mesmo o trabalho de Baron *et al.* (2009) não sendo destinado especificamente à RSSF, seu conceito pode ser ampliado e utilizado para analisar o comportamento de sistemas de comunicação. Em RSSF, as capacidades de transmissões C_A e C_B são diretamente relacionadas com a quantidade de dispositivos que utilizam as rotas fornecidas por A e por B. Quanto mais dispositivos utilizam a rede pela rota A, maior será a taxa r_A, portanto C_A será maior. Ressalta-se também que, por conta da capacidade limitada dos dispositivos utilizados em RSSF, R possui um único *buffer* par armazenar os dados provenientes de A e B. Tal fator torna o processo de escalonamento mais complexo para esse tipo de rede, implicando no escalonamento pelo método FIFO, na maioria dos casos. Portanto, em RSSF, cada dispositivo atua como um pequeno roteador, com capacidade finita, sem técnicas avançadas de escalonamento e com capacidades de transmissões variadas entre eles (BARON *et al.*, 2009).

4.4.2 Impacto do *Buffer* Finito no Desempenho da RSSF

Al-Anbagi, Khanafer e Mauftah (2013) utilizam um modelo baseado no de Markov para verificar o impacto de um *buffer* finito no desempenho de uma RSSF. Nele são analisados, variando o tráfego da rede, os modelos de rede estrela e árvore de agrupamento, levando em consideração fatores, como: atraso fim-a-fim, confiabilidade e consumo energético. De forma resumida, o modelo proposto utiliza os seguintes critérios (AL-ANBAGI *et al.*, 2013):

- a RSSF opera em modo *beacon* no padrão IEEE 802.15.4;
- pacotes chegam à camada MAC numa taxa λ com distribuição de Poisson;
- são utilizados pacotes de confirmação de recebimento (ACK) na camada MAC;
- no modelo estrela o coordenador é o agregador de dados, enquanto no modelo de árvore de agrupamento o nodo raiz é o agregador de dados;
- os pacotes podem ser transmitidos em um único período de transmissão;
- todos os dispositivos operam com filas do tipo M/G/1/L/FIFO;
- o tempo de processo do pacote no *buffer* é desprezível.

Para avaliá-lo foi utilizado o simulador QualNet, com tráfego de rede gerado por distribuição de Poisson, numa rede variável distribuída em uma área de 900 m². Todo nodo da rede é capaz de comunicar com outro na mesma PAN, de forma a utilizar o método de CSMA/CA para ter acesso ao meio de comunicação. Na análise foram simulados três tamanhos de *buffers* da camada MAC: 512, 1024 e 2048 *byte*s.

A partir do trabalho de Al-Anbagi, Khanafer e Mauftah (2013) foi possível concluir que o tamanho do *buffer* na camada MAC tem impacto direto na qualidade da rede. Seus resultados demonstram que quanto maior o *buffer*, menor é o atraso fim-a-fim, com maior confiabilidade de entrega fim-a-fim e menor quantidade de energia consumida no processo. Isso ocorre pelo fato de que, com *buffers* maiores, os dispositivos armazenam primeiramente os pacotes, para depois transmiti-los, o que diminui a taxa de atraso. A diminuição do consumo energético ocorre por causa da redução de retransmissões, pois há um decaimento no número de colisões ao transmitir (AL-ANBAGI *et al.*, 2013).

4.4.3 Modelo Analítico de RSSF

Omondi (2015) e Omondi, Shah e Gemikonakli (2015) são trabalhos que descrevem um modelo analítico de RSSF, utilizando as teorias de filas, de forma a analisar o desempenho e disponibilidade de redes com falhas nos nodos com *buffers* finitos. Nesse trabalho, é utilizada a topologia de rede tipo árvore e um modelo de fila do tipo M/M/1/L/FIFO, o qual é bem representado por processos markovianos. De tal forma, o modelo proposto é uma representação de Markov de duas dimensões, conforme a Figura 22, nas quais o eixo vertical (j) representa os números de processos no sistema no decorrer do tempo, e o eixo horizontal (i) representa os estados de operação. Os estados de operação são definidos em: falha de canal (FC); falha no nodo (FM); ativo (R) e em espera S_{LP}. Neste modelo, o serviço fica disponível somente quando em estado ativo (R), enquanto em estado de falha (FM) sua única capacidade é a de recebimento de pacotes de dados. Apesar de ser possível receber dados em ambos os estados, a informação só é armazenada na fila quando há disponibilidade na mesma, sendo que o pacote é descartado caso contrário. Já no estado de falha de canal (FC) o dispositivo se torna inoperante, sem possibilidade de enviar ou receber informações. Desta maneira, os dispositivos neste estado são forçados a inicializarem a rotina de reparo para que seja possível retomar ao estado de operação (OMONDI, 2015).

Como descrito na seção 4.2, λ e μ são respectivamente as taxas de chegada e de atendimento, respectivamente, das filas que possuem uma capacidade de armazenamento L. Esse modelo adota, durante as operações, que os canais podem falhar e que seus tempos de falhas são distribuídos de modo exponencial no tempo em taxas ξ e ζ, respectivamente. Do mesmo modo que falhas podem ocorrer, foi definido que os métodos de reparo são executados imediatamente após sua detecção, retomando a total operacionalidade após seu reparo. Os tempos de recuperação foram definidos de forma a serem distribuídos, exponencialmente, em taxas η (reparo do nodo) e θ (reparo do canal), respectivamente. Como o modelo de rede utilizado é o de árvore por agrupamento, cada agrupamento é composto por um dispositivo de coordenação local, definido como cabeça do grupo (CH – *cluster head*). Cada CH possui uma taxa de recebimento de pacote de $\lambda_k = C\lambda$, no qual C é o número de dispositivos que compõem o grupo (OMONDI, 2015).

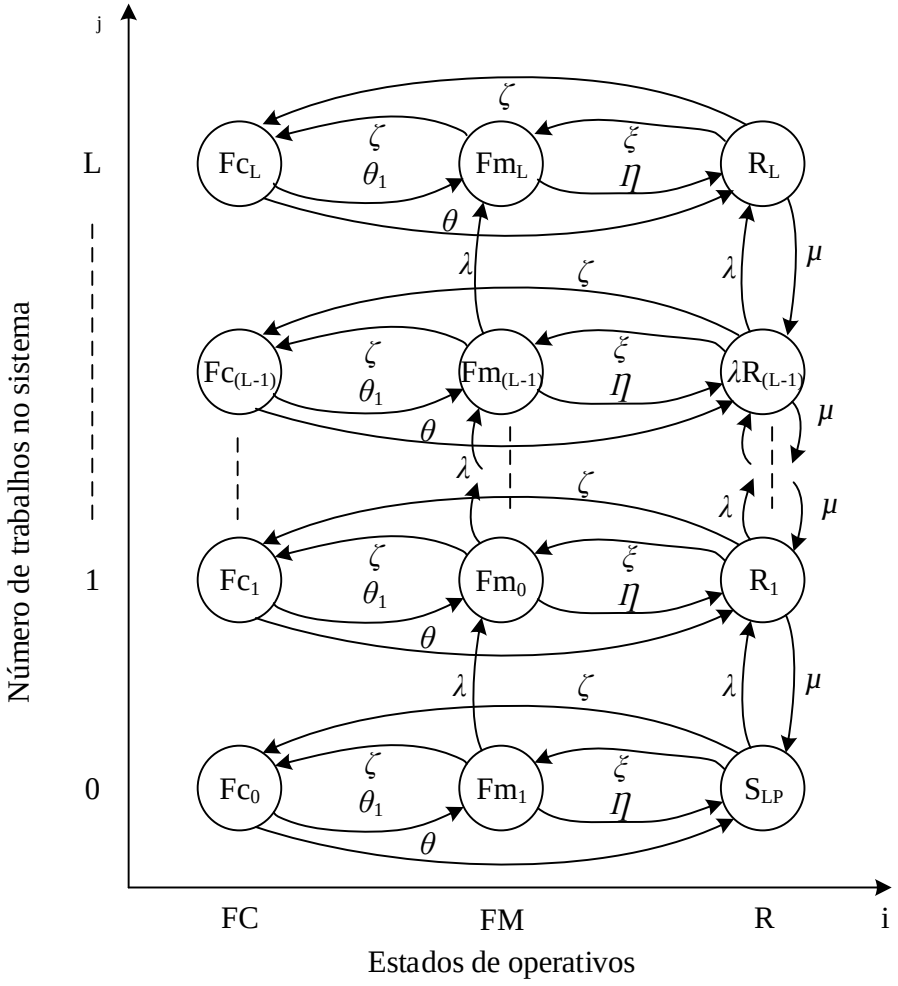


Figura 22 – Modelo de performabilidade.
Fonte: Adaptado de Omondi (2015, p. 133).

Para validar o modelo proposto, Omondi (2015) e Omondi, Shah e Gemikonakli (2015) desenvolveram um programa em linguagem de programação C++ para simulação, que utiliza uma abordagem de agendamento de eventos. Os parâmetros utilizados para validar o modelo estão descrito no Quadro 1.

Parâmetro	Valor
Nodos	25 a 35 por agrupamento
Taxa de chegada λ	0 a 14 pacotes por hora
Taxa de serviço μ	300 pacotes por hora
Taxa de falha ξ	0,001 a 0,01 por hora
Taxa de reparação η	0,5 por hora
Tamanho da fila L	10, 30, 50, 100, 500 e 1000

Quadro 1 – Parâmetros de simulação adotados.
Fonte: Adaptado de Omondi (2015, p. 94).

De acordo com Omondi (2015, p. 95), pôde-se observar que há um limite inferior e superior de capacidade de fila, a qual impacta no número médio de usuários na fila (MQL). Se a capacidade é baixa, o MQL permanece pequeno, entretanto mais dados são perdidos. Por outro lado, o MQL não se altera proporcionalmente caso a capacidade da fila seja incrementada indefinidamente, havendo um limite superior. Comparando as taxas de bloqueio em relação as taxas de chegada de dados, nota-se que há um bloqueio maior da rede quanto maior é a taxa de chegada de dados. De forma a reduzir essa taxa de bloqueio, pode-se incrementar a capacidade das filas. A partir desses dados, Omondi (2015) define sugestões de tamanhos de filas para cada cenário de aplicação, conforme o Quadro 2.

Categoria de Dados	Tamanho da Fila	Exemplos
Intensidade Baixa	10 a 30	Agricultura
Intensidade Média	30 a 50	*Body Area Networks*
Intensidade Normal	50 a 100	Erupção Vulcânica
Intensidade Elevada	100 a 500	Vídeo
Intensidade Muito Elevada	Mais de 500	Aplicações de tempo real

Quadro 2 – Propostas de tamanho de fila.
Fonte: Adaptado de Omondi (2015, p. 99).

4.5 CONCLUSÃO

As RSSF são sistemas de fluxo, em que cada sensor é responsável por transportar informações a partir de um canal de comunicação sem fio, composto de diversos pontos intermediários, até o receptor final. Em uma rede, cada ponto desse canal de comunicação contém uma fila de dados, denominada *buffer*, que realiza o armazenamento de cada informação a ser transmitida. Portanto, pode-se definir que uma RSSF é composta por dois tipos de fila, a fila local de cada dispositivo, e a fila global, composta por todos os dispositivos da rede. As informações são inseridas na fila local pelo dispositivo, ou pela informação recebida de outro que utilize este como rota. Portanto, a fila global pode ser entendida como o conjunto das filas locais necessárias para emitir uma informação de um dispositivo que esteja situado a alguns saltos de distância do receptor. A Figura 23 representa uma RSSF na qual cada dispositivo possua seu próprio *buffer* de capacidade quatro.

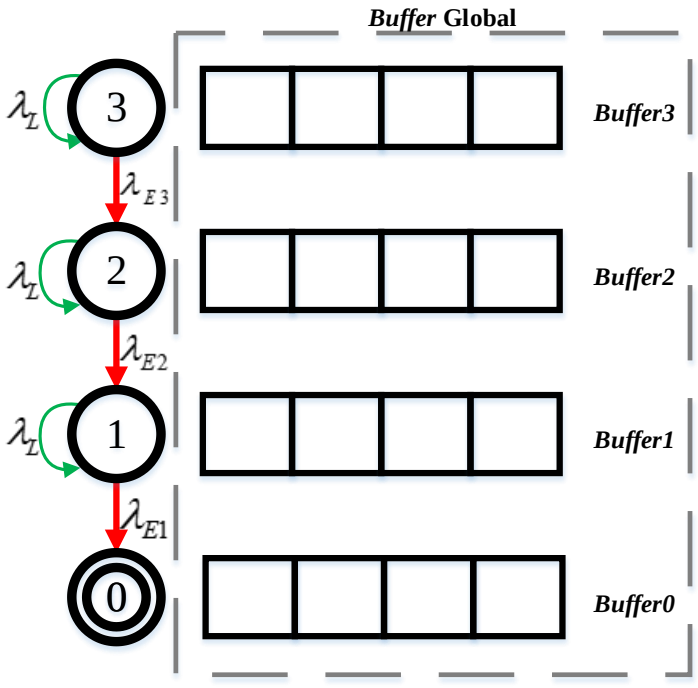


Figura 23 – Exemplo de fila numa RSSF. Os círculos simples representam dispositivos de uma RSSF, e o circulo duplo denotado por '0' representa o coordenador da rede. Cada dispositivo possui um *buffer* de capacidade quatro, representado ao lado de cada.
Fonte: Autoria própria.

No exemplo representado pela Figura 23, verifica-se que cada dispositivo cria uma nova informação em sua própria fila local numa taxa λ_L, denotando o processo de início de

transmissão de uma amostra de um dispositivo de uma RSSF. Além disso, cada dispositivo é responsável pelo roteamento e armazenamento das informações provenientes dos dispositivos anteriores, sendo que cada um gera numa taxa λ_E diferente. Isso ocorre por causa da disposição da rede, na qual cada dispositivo mais próximo ao agregador emitirá mais informação do que os mais afastados, devido ao processo de roteamento. Ademais, pode-se verificar que se trata de uma fila global com prioridade, pois os dispositivos mais próximos ao agregador de dados, no exemplo o dispositivo 1, irá inserir suas informações no início da fila, sendo processados mais rapidamente do que os dispositivos mais longes, no exemplo o dispositivo 3. Portanto, se a taxa de geração de informação λ_L for muito maior do que as taxas de transmissão de dados entre os dispositivos λ_E, a rede irá ficar presa somente com as informações provenientes do dispositivo 1.

Os estudos realizados por Baron *et al.* (2009), Omondi (2015) e Omondi *et al.* (2015) demonstram que aumentar a capacidade de armazenamento das filas em redes de comunicação pode aumentar a QoS da rede. Entretanto, esses mesmos estudos demonstram que o oposto também ocorre, de forma que aumentar o *buffer* torna a rede instável, perdendo mais informações. Além disso, mesmo que aumentar a quantidade de memória em um *buffer* estabilize a rede, é provável que um dispositivo de RSSF não consiga fornecer a quantidade de memória necessária, pois são dispositivos de recursos limitados. Portanto, torna-se evidente a necessidade um método no qual seja possível garantir a QoS com um custo mínimo de memória, pois somente gerenciar o tamanho do *buffer* numa rede não é o suficiente.

5 PROPOSTA DE AUMENTO DE CONFIABILIDADE

Conforme demonstrado nos capítulos anteriores, RSSF é uma tecnologia de ampla utilidade. Contudo, para que seja possível implantá-las, é necessário utilizar componentes de baixo custo, fator limitante de sua funcionalidade. Métodos foram desenvolvidos especificamente para serem utilizados nesse tipo de rede, pois os convencionais, como protocolos TCP e UDP, não atendem completamente as necessidades das RSSFs. Além disso, ao analisar o comportamento de sistemas de fluxo de dados, nota-se que aumentar a capacidade de armazenamento de informações transientes não resolve problemas de instabilidade do sistema de comunicação. Pois, se a taxa de geração de informação (λ) for maior que a taxa de tratamento (μ), seria necessário uma capacidade de armazenamento infinita para ser possível processar todos os dados.

Portanto, com a finalidade de desenvolver um método que atenda as necessidades de confiabilidade, de modo a utilizar o mínimo possível de memória dos dispositivos, foram agregados métodos de transporte, roteamento e transmissão em um novo protocolo de comunicação denominado µNet. Dessa forma, o protocolo engloba as camadas de transporte, rede e enlace de forma mais efetiva para as restrições das RSSFs. O µNet utiliza de métodos de confirmação de transmissão de mensagem ponto-a-ponto, para aumentar a confiabilidade na entrega de informações em redes multissaltos e, por padrão, utiliza *buffer* de rede unitário, de modo que cada dispositivo de uma RSSF possa enviar ou rotear somente um pacote por vez, possibilitando utilizar a memória de modo mais efetivo em outros recursos do sistema. Por utilizar métodos de confirmação ponto-a-ponto, torna-se possível controlar o *buffer* de rede, de modo que a informação não seja descartada enquanto não tiver sido transmitida adequadamente para o próximo dispositivo. Desse modo, conforme descrito na seção 4.5, a rede possui somente uma fila global, em que cada salto faz parte de uma parcela do *buffer* total de armazenamento de pacotes. Contudo, apesar de ser um valor padrão, o protocolo pode ser configurado para operar com tamanhos variados de *buffers*.

O método de confirmação ponto-a-ponto utilizado pelo µNet é composto por três tarefas, sendo elas:

1- enviar o pacote ao próximo nodo;
2- adicionar o pacote recebido no *buffer* da rede, se o mesmo possuir espaço livre;
3- retornar um pacote de confirmação de recebimento, somente se o pacote recebido for armazenado no *buffer* sem erros.

Quando um remetente envia uma informação para um dispositivo (receptor), o segundo só emite uma mensagem de confirmação quando for possível armazenar o pacote transmitido. Quando o remetente recebe a mensagem de confirmação emitida pelo receptor, o remetente libera o espaço de memória ocupada pela informação para ser utilizada novamente. Com esse método, é possível aumentar a confiabilidade na entrega de informações fim-a-fim em redes multissaltos.

Nas seções posteriores deste Capítulo, são explicados os procedimentos adotados para realizar os métodos de confirmação de transmissão, roteamento e transporte das informações fim-a-fim para redes multissaltos. Códigos, projetos e outras especificações do µNet estão disponíveis no repositório GitHub (BARRIQUELLO et al., 2018).

5.1 MÉTODO DE CONFIRMAÇÃO DE TRANSMISSÃO

Para validar a transmissão ponto-a-ponto de um pacote (PKT), o receptor confirma o recebimento pela camada de enlace por uma mensagem ACKR (*Acknowledgement of the Radio*) que é fornecida pelo rádio transmissor. Caso o pacote seja inserido no *buffer* do receptor, então o receptor envia a confirmação da camada de transporte, pela mensagem ACKN (*Acknowledgement of the Network*). Por fim, o emissor envia uma ACKR para o receptor para confirmar o recebimento do ACKN e libera o *buffer* do emissor para armazenar um novo pacote. A Figura 24 (a) demonstra como é realizada a troca de mensagens ponto-a-ponto com o protocolo µNet e Figura 24 (b) demonstra o processo de transmissão em caso de falhas.

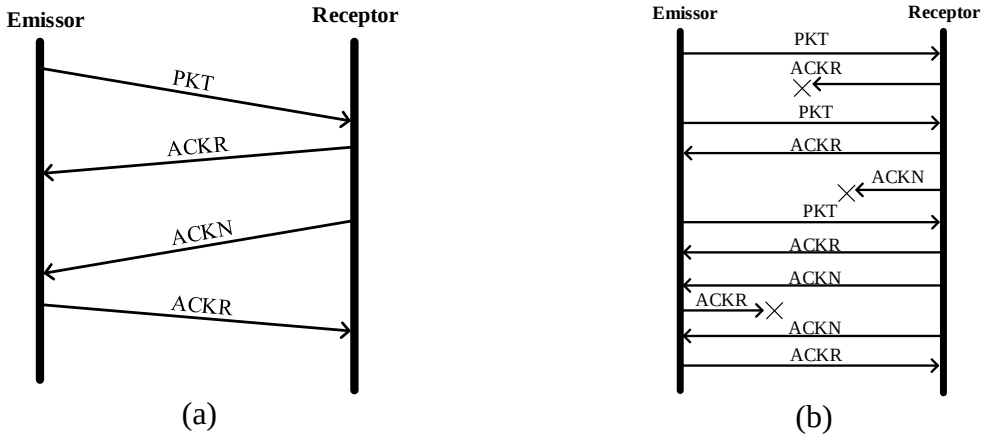


Figura 24 – Transmissão ponto-a-ponto com protocolo µNet. (a) Exemplo de transmissão bem sucedida. (b) Exemplo de retransmissões em caso de falhas.
Fonte: Autoria própria.

Cada transmissão é composta por tentativas de transmissão com intervalos distintos entre o tipo de retransmissão. Caso o pacote ACKR não retorne, define-se que houve problema na comunicação entre o emissor e o receptor, por interferência no meio ou mal funcionamento dos dispositivos. Nesse caso, deve-se aguardar um período pequeno entre retransmissão, por padrão definido em 10 ms para o padrão IEEE 802.15.4. Entretanto, se retornar o pacote ACKR e o ACKN não, denota-se que o *buffer* da rede estava ocupado ou houve erro na transmissão de retorno do ACKN por interferência no meio de comunicação, necessitando que o emissor aguarde novamente para uma nova tentativa de envio. Nesse caso, deve-se aguardar um tempo mais elevado do que o caso anterior, pois se o receptor estiver com o *buffer* ocupado, deve ser disponibilizado tempo suficiente para esvaziá-lo. Por padrão, foi utilizado um valor inicial de 64 ms de espera, que é incrementado a cada retransmissão, até um valor máximo de 256 ms pelo algoritmo *Trickle*. Se o pacote ACKN não retornar dentro desse intervalo, o PKT é enviado novamente ao receptor, realizando as mesmas retransmissões efetuadas caso o pacote ACKR não retorne. A quantidade de tentativas para transmissões é configurada no protocolo e, por padrão, definida em 3 tentativas se o ACKR não retornar e em 30 se o ACKN não retornar. Esses valores foram definidos de modo empírico, analisando o comportamento de redes reais e simuladas.

Portanto, com esse método, garante-se a entrega de pacotes a um salto de distância e que o pacote será encaminhado pelo nodo vizinho, pois está armazenado em seu buffer para processamento. Caso o pacote recebido não possa ser reservado na memória, o pacote de confirmação não é emitido. Ao finalizar o tempo de espera do emissor, sem um retorno da confirmação de recebimento, o emissor poderá tentar enviar novamente o pacote, até que se esgote o número de tentativas. Com esse método, se não houver retorno da confirmação ACKN, indica-se que o fluxo de dados nesse momento é superior à capacidade da rede, ou que uma fonte de interferência iniciou no meio de comunicação entre as mensagens ACKR e ACKN.

A Figura 25 demonstra o processo de comunicação do µNet em cenários com múltiplos saltos, na qual dois sensores enviam dois pacotes distintos, “X” e “Y”, ao “Receptor”. Para isso, o “Sensor 1” deve enviar seu pacote pelos sensores 2, 3 e ‘n’ para que chegue ao “Receptor”, enquanto o “Sensor 3” necessita passar somente pelo sensor ‘n’. Nesse caso, o “Sensor ‘n’” denota que poderia haver mais sensores no meio do caminho. Conforme o exemplo da Figura 25, o “Sensor 1” envia com sucesso seu pacote ao “Sensor 2”, enquanto, em tempos próximos, o “Sensor 3” envia sua informação ao “Sensor ‘n’”. Contudo, quando o “Sensor 2” tenta enviar o pacote, o “Sensor ‘n’” encontra-se ocupado com o pacote “Y” do

“Sensor 3”, necessitando aguardar um tempo de “Espera”. Enquanto o “Sensor 2” aguarda liberar o *buffer*, o “Sensor ‘n’” finaliza seu processo de roteamento, finalizando a entrega do pacote “Y” para o receptor. Após o tempo de espera, o “Sensor 2” envia seu pacote ao “Sensor ‘n’”, que após finalizar o processo de roteamento, entrega o pacote “X” ao receptor. Nesse caso, ambos emissores conseguem enviar com sucesso seus pacotes. Entretanto, se a quantidade de tentativas se esgote, o pacote é descartado e perdido.

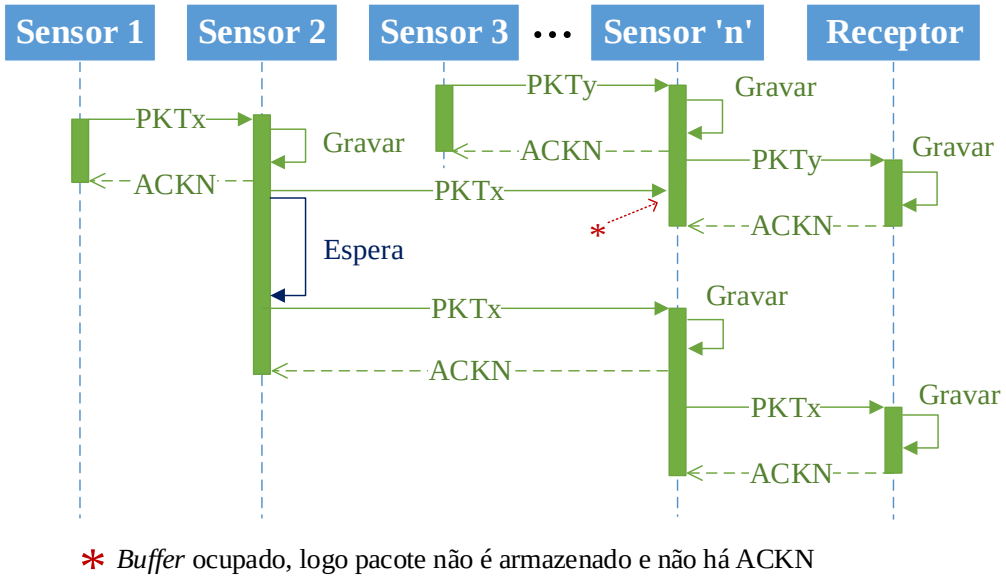


Figura 25 – Fluxograma de transmissão multissaltos com µNet.
Fonte: Autoria própria.

Para realizar os controles de transmissão de pacotes, um novo cabeçalho de informações de 5 *bytes* é utilizado, conforme a Figura 26.

LLC (8 *bits*)	Tipo do Pacote (8 *bits*)	Tamanho do Cabeçalho (8 *bits*)	Número Máximo de Saltos (8 *bits*)	Próximo Cabeçalho (8 *bits*)

Figura 26 – Cabeçalho de rede µNet.
Fonte: Autoria própria.

A descrição de função de cada campo da Figura 26 é:

- LLC: *Logical Link Control*, informação do tipo de protocolo de comunicação utilizado para realizar controle de transmissão, podendo ser µNet, UDP, TCP, etc.;
- Tipo do Pacote: informação referente ao tipo que esse pacote pertence, podendo ser:

    - *Broadcast*: envia mensagem a todos os nodos vizinhos;
    - *Unicast Down*: envia uma mensagem em direção ao coordenador da rede;
    - *Unicast Up*: enviar uma mensagem em direção a um nodo mais distante do coordenador da rede;
    - *Unicast Acknowledge Down*: mensagem de retorno para confirmação de recebimento de uma mensagem *Unicast Down* prévia;
    - *Unicast Acknowledge Up*: mensagem de retorno para confirmação de recebimento de uma mensagem *Unicast Up* prévia;
    - *Multicast Down*: enviar uma mensagem a diversos nodos em direção ao coordenador da rede;
    - *Multicast Up*: enviar uma mensagem a diversos nodos em direção oposta ao coordenador da rede;
- Tamanho do Cabeçalho: quantidade, em *byte*s, do cabeçalho de informação do pacote;
- Número Máximo de Saltos: quantidade de saltos que determinada informação pode percorrer antes de ser descartado;
- Próximo Cabeçalho: indica se o próximo cabeçalho é do tipo Controle (CTL) ou Aplicação (APL);

5.2 MÉTODO DE ROTEAMENTO

Para realizar os processos de roteamento com o protocolo µNet, utilizou-se um endereçamento de 64 *bit*s para os dispositivos do µNet, assim como o endereço estendido do protocolo IEEE 802.15.4 (IEEE..., 2015). Dos quais são utilizados 16 *bit*s para informar o prefixo e a qual área de rede (PANID) o dispositivo pertence, conforme o padrão IEEE 802.15.4. Além disso, são utilizados 48 *bit*s para endereçamento global de rede, sendo 32 *bit*s (ADDR32) para endereçamento de redes numa mesma PAN e outros 16 *bit*s (ADDR16) são utilizados como endereço local da rede. A Figura 27 representa os cabeçalhos de endereços de origem e destino do µNet utilizados pelo processo de roteamento.

PANID Origem (16 *bit*s)	ADDR32 Origem (32 *bit*s)	ADDR16 Origem (16 *bit*s)
PANID Destino (16 *bit*s)	ADDR32 Destino (32 *bit*s)	ADDR16 Destino (16 *bit*s)

Figura 27 – Endereçamento µNet.
Fonte: Autoria própria.

Para realizar a formação de rota no µNet, o coordenador da rede deve iniciar a transmissão por uma mensagem de *broadcast* na qual ele indica aos dispositivos vizinhos a sua existência. Quando algum dispositivo vizinho recebe a notificação da existência do coordenador, ele indica aos seus dispositivos vizinhos a existência de uma rota. Isso se perpetua até que todos os dispositivos na rede obtenham ao menos uma rota. Se um novo dispositivo entra na rede, ele pode enviar uma mensagem de requisição de rota aos seus vizinhos. Dessa forma, se alguém possuir uma rota até o coordenador, esse novo dispositivo é notificado com a informação de sua rota.

Por convenção, o µNet utiliza as seguintes nomenclaturas para rotas, de forma que a distância é medida pela quantidade de saltos necessários entre o emissor e coordenador da rede:

- rotas descendentes (*downward*): representa a direção das mensagens que partem dos nodos mais distantes em direção ao coordenador da rede;
- rotas ascendentes (*upward*): representa a direção das mensagens originadas no coordenador da rede em direção aos nodos mais distantes;
- nodo: um dispositivo que pertença à rede;
- nodo pai: dispositivo que fornece ao menos um caminho para outros dispositivos poderem enviar informações ao coordenador da rede;
- nodo filho: dispositivo que utiliza um nodo pai para enviar sua informação ao coordenador da rede.

A criação de rotas descendentes se inicia com o coordenador indicando sua atividade aos nodos vizinhos. Os dispositivos próximos, sabendo a rota para o coordenador, propagam essa informação a sua vizinhança, até que todos os dispositivos que pertençam a rede tenham sua rota estabelecida, ou até que uma requisição de um novo dispositivo ocorra. Para definir qual das rotas descendentes será utilizada, cada dispositivo avalia a qualidade do sinal e a quantidade de saltos necessários para enviar uma informação até o coordenador da rede. A rota com melhor sinal de rádio entre o nodo atual e o nodo pai e menor quantidade de saltos ao coordenador é a escolhida como padrão.

A definição das rotas ascendentes ocorre somente após as rotas descendentes serem estabelecidas. Ao formar as rotas descendentes, cada dispositivo armazena os nodos filhos que fazem parte de sua sub-rede. Portanto, quando é necessário emitir uma mensagem a um nodo filho, verifica-se na tabela de roteamento ascendente qual o dispositivo vizinho que possui rota ao destinatário desejado. A Figura 28 representa uma rede µNet e sua tabela de roteamento ascendente, sendo que cada nodo possui uma lista com os próximos saltos necessários para enviar a informação aos nodos filhos.

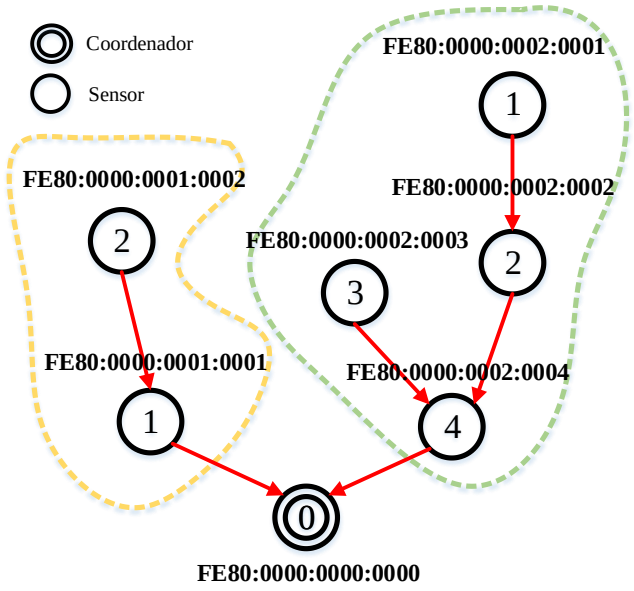


Tabela de roteamento ascendente

Nº	Salto ADDR16	Destino ADDR64
0	0001	FE80:0000:0001:0001
0	0001	FE80:0000:0001:0002
1	0002	FE80:0000:0001:0002
0	0004	FE80:0000:0002:0004
0	0004	FE80:0000:0002:0003
0	0004	FE80:0000:0002:0002
0	0004	FE80:0000:0002:0001
4	0003	FE80:0000:0002:0003
4	0002	FE80:0000:0002:0002
4	0002	FE80:0000:0002:0001
2	0001	FE80:0000:0002:0001

Figura 28 – Exemplo de rede µNet com tabela de roteamento ascendente. A tabela da direita apresenta as rotas ascendentes que cada dispositivo da figura esquerda. O campo "Nº" é o identificador de cada nodo, "Salto ADDR16" o próximo salto que a informação deve ser enviada e "Destino ADD64" denota o destino final que o pacote deve chegar. Portanto, para o dispositivo "0" enviar um pacote para o destino "FE80:000:0002:0001", deve-se transmitir o pacote para o salto de ADDR16 "0002".

Fonte: Autoria própria.

Conforme a Figura 28, há duas redes com o mesmo PANID FE80, a amarela (esquerda) com endereçamento ADDR32 de 0000:0001 e a verde (direita) com 0000:0002. Nota-se a existência de dispositivos com o mesmo endereçamento de ADDR16, e contanto que os mesmos não estejam no mesmo endereçamento de rede ADDR32 e no mesmo raio de comunicação, a rede funciona sem problemas. Por exemplo, o coordenador não poderia ter dois dispositivos com ADDR16 de 0001 como vizinho, pois isso causaria que a mesma informação fosse recebida por ambos vizinhos. Nesse caso, a mensagem de confirmação seria gerada pelos dois dispositivos e posteriormente descartada pelo dispositivo que não possuí a rota correta do pacote.

De forma a manter uma rota ativa e a tabela de roteamento atualizada, é enviada uma mensagem de controle descendente ao coordenador. Essa mensagem de controle deve ser transmitida com mais frequência quando a rede está sendo criada, no processo de estabilização de rotas de cada dispositivo. A partir do momento que é verificado a estabilidade dessa rota, a frequência de transmissão da mensagem de controle é reduzida. Com isso, os dispositivos conseguem verificar as rotas ascendentes e quais dispositivos estão ativos e operantes. Essa mensagem de controle é omitida caso alguma mensagem descendente tenha sido enviada dentro do intervalo de tempo do controle. O controle da transmissão da mensagem de controle ocorre pelo algoritmo *Trickle*, com intervalos entre 4 segundos, quando a rota está sendo criada ou modificada, e 9 minutos quando as rotas estão estabilizadas, valores definidos empiricamente a partir de análises de redes com protocolo µNet.

Além disso, o processo de roteamento do µNet utiliza dois *buffers* separados, um para as rotas ascendentes e outro para as rotas descendentes, definidos para não bloquear os fluxos das rotas. Como o fluxo natural de um RSSF é dos sensores para o coordenador, sua fila global se torna maior a medida que há congestionamento próximo ao coordenador. Entretanto, o fluxo ascendente (do coordenador aos sensores), em geral, é menor. Portanto, se utilizado um único *buffer* para as duas rotas, as pacotes com destino aos sensores podem ser bloqueados pelo *buffer* ocupado dos pacotes que tem como destino o coordenador.

5.3 MÉTODO DE TRANSPORTE

O método de transporte utilizado pelo µNet é similar ao UDP, sem conexão e sem confirmação de envio fim-a-fim, utilizando uma porta de origem e uma de destino para identificar à qual aplicação o pacote pertence. Entretanto, utiliza-se de apenas um *byte* para as portas, pois leva-se em consideração que uma RSSF não deve conseguir utilizar mais do que 255 aplicações em um dispositivo, por causa da capacidade de memória restrita dos dispositivos. Além disso, não são realizados cálculos por soma de verificação nessa camada, pois essa confirmação já é realizada na camada de enlace pelo padrão IEEE 802.15.4, e se o pacote não estiver correto, deve ser descartado nas camadas inferiores. A Figura 29 apresenta o cabeçalho de transporte utilizado no protocolo µNet, em que a Porta de Origem e Porta de

Destino são valores para informar de qual aplicação o pacote foi originado e para qual deve ser entregue; e Tamanho dos Dados, informa o tamanho em *bytes* dos Dados que estão sendo transportados pelo µNet.

Porta de Origem (8 *bit*s)	Porta de Destino (8 *bit*s)	Tamanho dos Dados (8 *bit*s)	Dados

Figura 29 – Cabeçalho de transporte µNet.
Fonte: Autoria própria.

5.4 INTEGRAÇÃO COM IEEE 802.15.4 E MENSAGENS DE CONFIRMAÇÃO

Para a solução proposta para RSSF, o protocolo µNet utiliza o padrão IEEE 802.15.4 para gerenciar as transmissões entre rádios. Entretanto, nota-se que esse protocolo pode ser utilizado com qualquer protocolo de transmissão, contanto que respeitada as condições do protocolo µNet. Portanto, o cabeçalho IEEE 802.15.4 utilizado para RSSF é composto pelos campos conforme descrito no Capítulo 3 e pela Figura 30.

Tamanho do Pacote (8 *bits*)	Quadro de Controle (16 *bits*)	Número de Sequência (8 *bits*)	PAN ID (16 *bits*)	Endereço de Destino (16 *bits*)	Endereço de Origem (16 *bits*)	Dados	FCS (16 *bits*)

Figura 30 – Cabeçalho IEEE 802.15.4 para o protocolo µNet.
Fonte: Autoria própria.

A partir do cabeçalho da Figura 30, é definida a mensagem ACKR utilizada para confirmar que o pacote foi transmitido, de tamanho total de 6 *bytes*, conforme representação da Figura 31. Nota-se que o "Quadro de Controle" possui valor hexadecimal 2, o que representa uma mensagem do tipo de confirmação, conforme as especificações do protocolo IEEE 802.15.4 (IEEE..., 2015).

Tamanho do Pacote	Quadro de Controle (0x02)	Número de Sequência	FCS

Figura 31 – Mensagem de confirmação ACKR do protocolo µNet.
Fonte: Autoria própria.

A partir dos cabeçalhos descritos anteriormente, o µNet é um protocolo com cabeçalho de tamanho de 24 *bytes* e, ao utilizar o padrão de comunicação IEEE 802.15.4, seu

cabeçalho eleva-se para 36 *bytes*. A partir do cabeçalho de rede e de transporte, é definido o pacote de confirmação de rede (ACKN), conforme representado na Figura 32, de tamanho total (TAM) de 21 *byte*s, pois não utiliza as informações do cabeçalho de transporte para realizar as confirmações.

IEEE 802.15.4	LLC	*ACK DOWN* ou *ACK UP*	TAM	SALTOS	CTL	ORIG. 64 *bit*s	DEST. 64 *bit*s

Figura 32 – Mensagem de confirmação ACKN do µNet.
Fonte: Autoria própria.

6 MATERIAIS E MÉTODOS

Para validar se a solução proposta no Capítulo 5 atende aos requisitos de confiabilidade para RSSF, adotou-se métodos de comparação com soluções existentes e utilizadas em RSSFs. O Contiki OS é um sistema operacional de código aberto que contém protocolos e métodos específicos para ambientes de RSSF. Por ser de código aberto, eles são testados por uma comunidade que auxilia na correção de problemas e rápida evolução do sistema. Portanto, a partir das informações coletadas e apresentadas nos capítulos anteriores, definiu-se configurar o Contiki OS com os protocolos de rede mais usuais em RSSF, como: CoAP, UDP, IPv6, 6LoWPAN, RPL e IEEE 802.15.4. O método µNet foi, originalmente, desenvolvido para o sistema operacional de tempo real BRTOS (*Brazilian Real-Time Operation System*). Portanto, são analisados dois sistemas operacionais com conjuntos de protocolos distintos para RSSF.

Realizam-se diversas comparações das taxas de confiabilidade de entrega de pacotes entre os protocolos do Contiki OS e do BRTOS, analisando a porcentagem de pacotes recebidos corretamente nas camadas de aplicações entre transmissores e receptores. O simulador Cooja (Contiki OS Java) é utilizado para obter dados de comparação, o qual permite controlar e gerenciar sistemas de comunicação sem fio utilizando protocolos de comunicação IEEE 802.15.4, com alguns dispositivos emulados fornecidos pela ferramenta. Além disso, esse simulador permite utilizar códigos que são gravados em dispositivos físicos, o que torna fácil a migração do simulador para ambientes de testes reais.

6.1 MATERIAIS

Todas as simulações foram realizadas em um computador com processor Intel Core i5-4210U de 1,7 GHz, de 2 núcleos físicos e 4 lógicos, e com 8 GB de memória RAM.

Os dados foram adquiridos por uma ferramenta computacional capaz de simular ambientes de rede de sensores sem fio, o Cooja. Esse *software* possui um conjunto de funcionalidades que permite interagir e extrair dados dos dispositivos emulados e controlar o meio de comunicação. Esta ferramenta vem sendo utilizada em trabalhos relacionados com RSSF, o qual permite um meio de comparação da solução proposta com outros trabalhos publicados pela comunidade científica, como: Ancillotti *et al.*, (2014), Gardasevic *et al.* (2015), Kim *et al.* (2015), Lodhi *et al.* (2016), entre outros.

A ferramenta Cooja é disponibilizada pelo Contiki OS numa máquina virtual destinada à ferramenta VMWare, a qual possui uma licença gratuita para máquinas não comerciais. Essa máquina virtual utiliza o Ubuntu 14.04 LTS 32-*bit*, um sistema operacional baseado em GNU/Linux, o qual é gratuito para uso. Foi configurado para a máquina virtual a disponibilidade de quatro processadores, 3 GB de memória de processamento (RAM) e 18 GB de memória de disco rígido (HD).

O simulador Cooja disponibiliza alguns dispositivos configurados com rádios para serem utilizados na rede. Para realizar os testes foi escolhido o dispositivo virtualizado Wismote, Figura 33, o qual é composto por um processador MSP430F5437 operando a 16 MHz, com memória de programa até 256 kB e memória RAM de 16 kB (ARAGO SYSTEMS, 2011). Juntamente com ele é utilizado o rádio CC2520 que opera a 2.4 GHz com o padrão IEEE 802.15.4, numa taxa de transmissão de 250 kbps com capacidade de 127 *byte*s por pacote (TEXAS..., 2007).

Figura 33 – Placa do dispositivo Wismote.
Fonte: Arago Systems (2011, p. 1).

6.1.1 BRTOS

O *Brazilian Real-Time Operating System* (BRTOS), ou Sistema Operacional de Tempo Real Brasileiro, foi desenvolvido para ser um sistema operacional preemptivo que utilize o mínimo necessário de memória de dados e de programa (ao menos 100 *byte*s de RAM e 2 kB de memória de programa), de forma que seja possível utilizá-lo com sistemas computacionais simples, como microcontroladores com unidade de processamento entre 8 e 32 *bits*. O sistema provê um conjunto de ferramentas para auxiliar seu funcionamento, como: semáforos, *mutex*, caixas de mensagens, filas e temporizadores, além de possuir um simulador para ser utilizado no Windows (DENARDIN; BARRIQUELLO, 2017).

O sistema permite a instalação de até 32 tarefas com prioridades distintas em seu sistema, de forma que se garanta que a tarefa de maior prioridade será executada antes das demais. Seu escalonador de tarefas utiliza o conceito de prioridade de execução, portanto devem-se considerar os tempos de execução necessários de cada tarefa, para o sistema não perder sua característica de tempo real. A organização temporal deve ser realizada ao projetar o sistema, no qual o desenvolvedor deve testar e validar que toda tarefa seja executada na forma e tempo corretos (DENARDIN; BARRIQUELLO, 2017).

Até o momento, há porte oficial para os seguintes processadores:

- Freescale: Kinetis (ARM Cortex-M4), Coldfire V1 e HCS08;
- ST: STM32F4xx (ARM Cortex-M4F);
- NXP: LPC11xx (ARM Cortex-M0), LPC176x (ARM Cortex-M3);
- Renesas: RX600 (RX62N);
- Texas Instruments: MSP430, Stellaris LM3S8968 (ARM Cortex-M3), Stellaris LM4F120H5QR (ARM Cortex-M4F);
- Atmel: ATMEGA328/128;
- Microchip: PIC18.

Todo código do BRTOS está sob a licença do MIT (permissiva), a qual permite a reutilização do código fornecido, contanto que sua licença seja distribuída com o programa (DENARDIN, 2010).

6.1.2 Contiki OS

Contiki OS (Contiki *Operating System*) é um sistema operacional cooperativo, escrito em linguagem de programação C de código aberto, desenvolvido para aplicações em Internet das Coisas (*Internet of Things* – IoT). Ele tem suporte para os padrões de internet IPv6 e IPv4 ao mesmo tempo que fornece padrões de comunicação 6LoWPAN, RPL e CoAP, de forma a maximizar a vida da bateria dos dispositivos (CONTIKI, 2018a).

O Contiki OS utiliza o mecanismo denominado *protothreads* para controlar as tarefas, que é uma mistura dos métodos de gerenciamento de eventos e de multitarefas. Com *protothreads*, os gerenciadores de eventos podem ser configurados para acionar ou bloquear determinadas tarefas. Com esses métodos, o sistema é capaz de funcionar em dispositivos com quantidade restrita de memória, ao necessitar pelo menos de 10 kB de memória RAM e 30 kB de memória de programa para comportar um sistema com rede IPv6 e RPL (CONTIKI, 2018a).

Até o momento, o Contiki OS possui portabilidade de código para os dispositivos citados no Quadro 3, e está definido sob a licença "3-clause BSD-style", tornando-o gratuito para sistemas comerciais e não comerciais, contanto que os direitos estejam devidamente situados nos cabeçalhos dos arquivos (CONTIKI, 2018a).

MCU/SoC	Rádio	Plataformas	Suporte ao Cooja
TI CC2538	Integrado / CC1200	RE-Mote	Não
nRF52832	Integrado	nRF52 DK	Não
RL78	ADF7023	EVAL-ADF7023DB1	Não
TI CC2538	Integrado	cc2538dk	Não
TI MSP430x	TI CC2420	exp5438, z1	Sim
TI MSP430x	TI CC2520	wismote	Sim
Atmel AVR	Atmel RF230	avr-raven, avr-rcb, avr-zig*bit*, iris	Não
Atmel AVR	TI CC2420	micaz	Sim
Freescale MC1322x	Integrado	redbee-dev, redbee-econotag	Não
TI MSP430	TI CC2420	sky	Sim
TI MSP430	TI CC1020	msb430	Não
TI MSP430	RFM TR1001	esb	Sim
Atmel Atmega128 RFA1	Integrado	avr-atmega128rfa	Não
Microchip pic32mx795f512l	Microchip mrf24j40	seed-eye	Não
TI CC2530	Integrado	cc2530dk	Não
6502	-	apple2enh, atari, c128, c64	Não
Nativo	-	native, minimal-net, cooja	Sim

Quadro 3 – Plataformas suportadas pelo Contiki.
Fonte: Adaptado de Conitki (2018b).

6.1.3 Cooja

Cooja (Contiki OS Java) é uma ferramenta desenvolvida originalmente para realizar simulações e testes de programas criados com o sistema operacional Contiki OS. É uma fermentada desenvolvida em Java, cujo principal objetivo é fornecer um ambiente de simulação que permite aos desenvolvedores, tanto ver suas aplicações executando em redes de larga escala, como em detalhes extremos nos dispositivos de *hardware* emulados. A maior vantagem desse simulador é a facilidade no manuseio, tornando-o uma ótima ferramenta para desenvolvimento rápido. Contudo, seu método para emular dispositivos é realizado em nível de *hardware*, o que proporciona uma inspeção precisa do comportamento do sistema, ao custo de uma simulação mais lenta (ÖSTERLIND, 2006; CONTIKI, 2018a).

Além disso, essa ferramenta possui a característica de ser extensível, de modo a possibilitar que os usuários insiram interfaces dos dispositivos emulados e aditivos (*plug-ins*) à ferramenta simuladora. Algumas das interfaces provenientes de cada dispositivo são: posição espacial do dispositivo, botões, LEDs, transmissores e sensores. Os *plug-ins* são utilizados para realizar a interação com o simulador, como: controlar velocidade de simulação, verificar tráfego de rede, acessar memória de cada dispositivo, etc. (ÖSTERLIND, 2006; CONTIKI, 2018a). A Figura 34 apresenta a tela de interação do Cooja com o usuário da ferramenta, em que os retângulos coloridos representam:

- vermelho: interface visual da rede, entre as funções cita-se: distância entre nodos, rota estabelecida por cada nodo, sinais de tráfego da rede, etc.;
- azul: controle da simulação, iniciar, pausar, passo único e reiniciar;
- verde: saída serial de todos os dispositivos, possui filtro para mostrar somente as informações pertinentes;
- amarelo: *Simulation script*, ferramenta de automação de simulação;
- roxo: linha do tempo de cada dispositivo quanto à usabilidade do rádio;
- marrom: espaço para anotações.

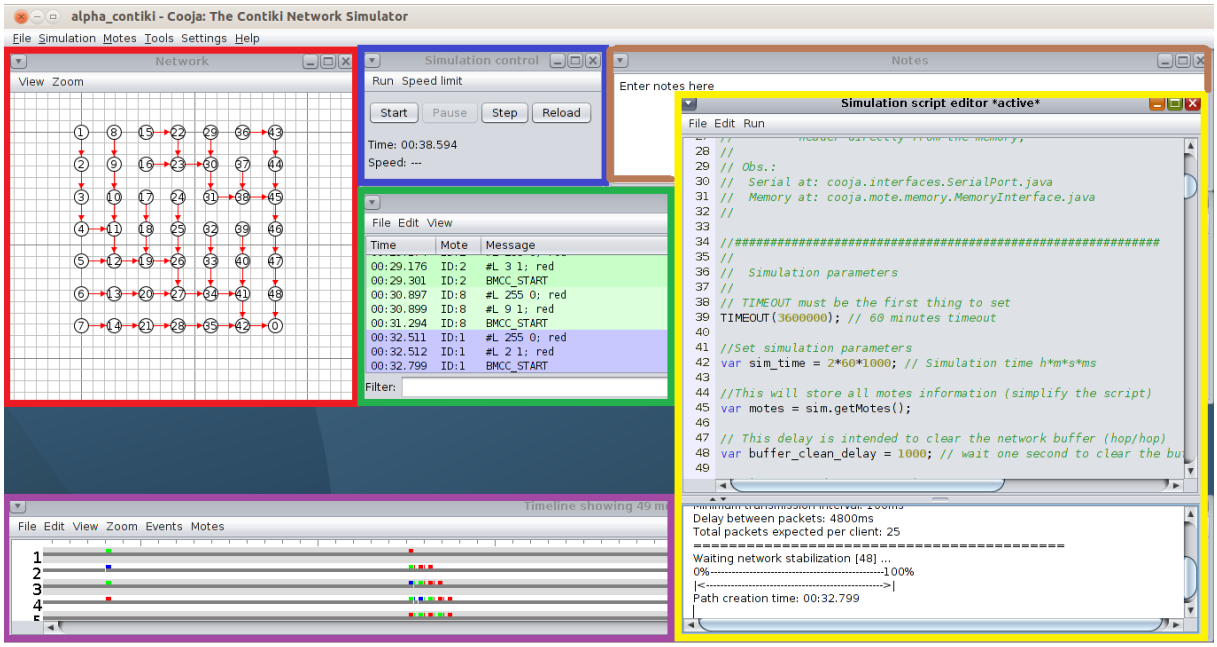


Figura 34 – Interface gráfica do Cooja.
Fonte: Adaptado de Contiki (2018a) e Dunkels *et al.* (2015).

O Cooja é responsável por fornecer uma ferramenta visual, simular o meio de comunicação dos rádios e integrar ferramentas externas com suas aplicações. Para controlar os mecanismos de emulação dos dispositivos, são utilizadas duas ferramentas, o *MSPSim* para emular microprocessadores da família MSP430 da Texas Instruments; e *Avrora* para emular os microprocessadores da família AVR da Atmel. Com isso o Cooja se torna uma ferramenta útil para simular redes heterogêneas, com diferentes dispositivos numa mesma rede (ÖSTERLIND, 2006; ROUSSEL et al., 2016; CONTIKI, 2018a).

O *MSPSim* utiliza códigos binários compilados em formato “ELF” (*Executable and Linkable Format*) ou em “IHEX” (Intel HEX) (FINNE *et al.*, 2018). Entretanto, para o Cooja compreender que se trata de um código compilado para uma plataforma específica, deve-se renomear o arquivo binário no formato: “código.plataforma”, no qual “código” é um nome genérico do arquivo e “plataforma” se refere àquelas que são suportadas pelo emulador, como: *SkyMote*, *Zolertia Z1* e *Wismote*. Por exemplo, ao compilar um código para a plataforma *Wismote*, deve-se renomear o arquivo “código.elf” para “código.wismote”. Portanto, ao renomear um arquivo binário para a plataforma específica, é possível simulá-lo independente do código utilizado ao compilar (ROUSSEL *et al.*, 2016).

A melhor forma de automatizar e controlar as simulações do Cooja é pela ferramenta de *script*, que possibilita recuperar informações quando o Cooja é utilizado sem a interface gráfica (Figura 34). Após criar uma nova simulação, deve-se acessar o editor de *script* dentro do Cooja para adicionar códigos escritos em *JavaScript* à simulação. Entre as aplicabilidades

dessa ferramenta, citam-se os exemplos comuns: controle temporal da simulação, repetição de simulações, enviar e receber mensagens dos dispositivos, ler e escrever na memória, armazenar resultados e análises em arquivos, entre outras funcionalidades. Além do mais, essa ferramenta permite acessar qualquer método público codificado em *Java* no programa Cooja (CONTIKI; ÖSTERLIND, 2014).

6.2 MÉTODOS

Para analisar o protocolo µNet pela comparação com outros protocolos, é necessário definir parâmetros, normas e procedimentos adotados nas simulações. As subseções seguintes descrevem os procedimentos utilizados para integrar o BRTOS com o simulador Cooja, as configurações dos cenários de RSSF, as métricas de análise de desempenho, a aplicação responsável por gerenciar e coletar as informações da rede e as configurações utilizadas para realizar uma comparação efetiva entre protocolos. Além disso, todos os códigos utilizados no simulados Cooja, para realizar as análises, estão disponíveis em repositório *online* GitHub (GODOI *et al.*, 2017).

6.2.1 Integração BRTOS e Cooja

Como descrito nas seções 6.1.2 e 6.1.3, a ferramenta de simulação Cooja foi desenvolvida para o Contiki OS, que possui os mais recentes protocolos para RSSF, como IPv6, RPL, 6LoWPAN e IEEE 802.15.4. Contudo, é possível integrá-la com qualquer tipo de código compilado, conforme explicado na seção descritiva do Cooja. Portanto, para utilizar o BRTOS com esta ferramenta, foram necessários alguns procedimentos:

- definir o dispositivo a ser utilizado na rede, conforme disponibilidade do simulador;
- realizar a portabilidade do BRTOS para o dispositivo selecionado;
- implementar os métodos de acesso ao rádio do simulador no BRTOS com µNet.

Para evitar contratempos na implementação e portabilidade do BRTOS com µNet para o simulador, optou-se por utilizar o dispositivo virtualizado Wismote, o qual possui maior quantidade de memória disponível. O MSP430 possui dois modos de endereçamento, 16 *bits*, e 20 *bits*, também classificado como MSP430X (TEXAS..., 2016b). Para ser possível utilizar toda memória disponível pelo Wismote (total de 256 kB), foi necessário realizar a portabilidade do BRTOS para o modo MSP430X, o qual possibilita endereçar mais memória do que a arquitetura de 16 *bits*, cujo total de endereçamento é de 65.535 (TEXAS..., 2016b).

Ao realizar a portabilidade de sistema para um determinado processador, torna-se necessário utilizar linguagem *Assembly* para ter acesso aos registradores exclusivos do processador, para efetuar o processo de troca de contexto do sistema. Esta troca é responsável por armazenar toda informação dos registradores do processador na memória do dispositivo e de carregá-los com novas informações provenientes de outro processo. Dessa forma, é possível parar e salvar as operações que estavam sendo realizadas, para que sejam retomadas quando necessário. O Apêndice A apresenta os códigos *Assembly* utilizados para realizar a troca de contexto no BRTOS.

Por fim, foi necessário implementar o código para controlar o rádio CC2520 (TEXAS..., 2007), de modo compatível com os métodos fornecidos pelo simulador Cooja. Para tal, foi utilizada parte do código fornecido pelo Contiki, de forma a facilitar o processo de integração.

6.2.2 Cenário de Comunicação

As simulações efetuadas a partir da ferramenta Cooja, tiveram seus meios de comunicação controlados pelo modelo UDGM (*Unit Disk Graph Medium*), que permite configurar as distâncias de transmissão (D_{TX}) e de interferência (D_{TI}) dos módulos de comunicação. A distância de transmissão é o raio de alcance que um dispositivo consegue enviar uma mensagem a outro de forma correta. Também, definido como alcance de propagação de sinal emitido pelo rádio. A distância de interferência se refere ao raio de alcance que o sinal emitido pelo rádio, ao enviar uma mensagem, corromperá outros sinais de comunicação. Não é possível detectar qual a informação/dado que esse sinal está transportando, mas sua energia ainda é suficiente para causar interferências em outras comunicações.

Além disso, o UDGM possibilita a configuração de taxa de sucesso (P_S) para enviar e receber mensagens entre dispositivos. A taxa de sucesso é uma propriedade probabilística que define, se uma mensagem transmitida será recebida, de forma a simular perda de informações por interferências provenientes do meio. O cálculo de P_S utilizado pelo UDGM é definido pela equação (3), retirada do arquivo "UDGM.java" do código fonte do Cooja, conforme demonstrado no ANEXO A (DUNKELS *et al.*, 2015; ROSENDAL; ELSTS, 2015).

$$P_S = P_{TX} \times \left(1 - \frac{D^2}{(D_{TX} \times F_{Rádio})^2} \times (1 - P_{RX})\right) \quad (3)$$

De modo que, P_{TX} e P_{RX} são fatores configuráveis que controlam as probabilidades de transmissão e recepção, respectivamente e $F_{Rádio}$ é a razão entre a potência de sinal configurada e potência máxima do rádio. A $F_{Rádio}$ controla a quantidade de energia consumida pelo rádio, assim como a distância de propagação do sinal criado. Por padrão, o Cooja utiliza os raios de alcance D_{TX} e D_{TI} com 50 m e 100 m, respectivamente, e a potência do radio CC2520 é configurado com 66,6% de sua capacidade máxima, configuração comum em dispositivos comerciais. A partir desses valores, é possível verificar que a distância máxima de transmissão é de 33,3 m (66,6% de D_{TX}), e a distância de propagação do sinal de interferência é de 66,6 m (66,6% de D_{TI}). Os valores da Tabela 1 apresentam as configurações utilizadas no controlador UDGM.

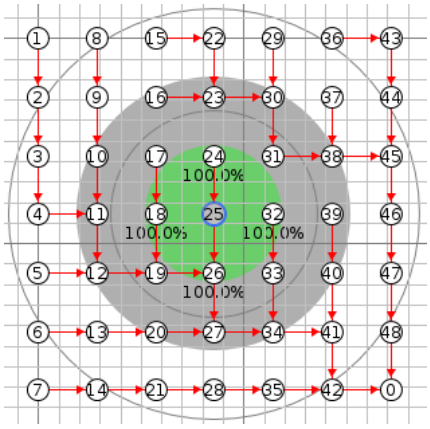
Tabela 1 – Valores de configuração do modelo UDGM.

Parâmetro	Valor
D_{TX}	50 metros
D_{TI}	100 metros
$F_{Rádio}$	66,6%
P_{TX}	100%, 95%, 90% e 85%
P_{RX}	100%, 95%, 90% e 85%

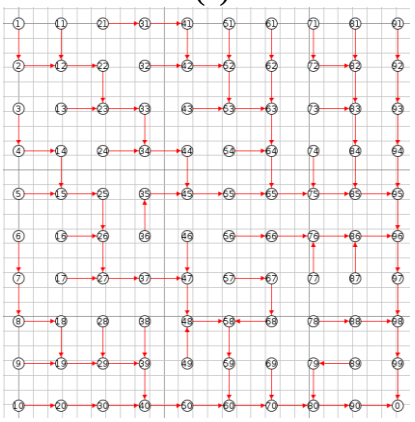
Fonte: Autoria própria.

Para avaliar o desempenho do protocolo µNet e dos protocolos de comunicação disponibilizados no Contiki OS, foram propostos três cenários de RSSF com distribuições espaciais distintas. Todos os cenários utilizam o modelo de distribuição em um plano cartesiano de duas dimensões, definindo-se nomes para cada um deles: *Alpha* para o primeiro, *Beta* para o segundo e *Gama* para o terceiro. No primeiro e segundo cenários foram utilizados 49 e 100 dispositivos, respectivamente, distribuídos em um formato matricial quadrado, com

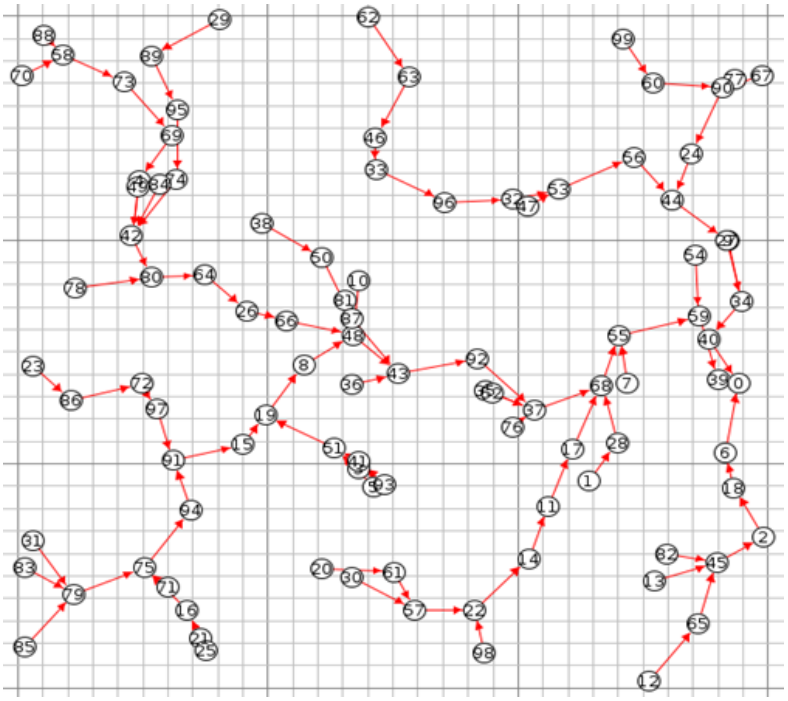
distâncias equivalentes entre eles de 29 m, conforme representado na Figura 35 (a) e (b). O terceiro cenário foi construído com 100 dispositivos distribuídos de forma aleatória, garantindo-se que cada um tenha, ao menos, um vizinho com quem se comunicar, conforme Figura 35 (c). O primeiro e segundo cenários foram escolhidos como forma de representar conjuntos residenciais distribuídos de maneira simplista, de forma que há uma residência ao lado da outra, com ao menos um dispositivo inserido exatamente a mesma distancia do outro, e o terceiro um caso no qual não há uma organização linear dessas residências, o que deve ocorrer na maioria dos casos. Por ser um cenário aleatório, calculou-se a distância média entre os dispositivos, pois esse parâmetro é relacionado com P_S. O Quadro 4 apresenta a relação dos cenários utilizados, e suas configurações.



(a)



(b)



(c)

Figura 35 – Cenários de simulação *Alpha*, *Beta* e *Gama*. (a) Cenário *Alpha* da esquerda superior com 49 dispositivos distribuídos em forma matricial. (b) Cenário *Beta* da esquerda inferior com 100 dispositivos distribuídos em forma matricial. (c) Cenário *Gama* da direita com 100 dispositivos distribuídos em forma aleatória.

Fonte: Adaptado de Dunkels *et al.* (2015).

Nome do Cenário	Quantidade de Dispositivos	Distância Média Entre Dispositivos	Distribuição
Alpha	49	29 metros	Matricial
Beta	100	29 metros	Matricial
Gama	100	23,5 metros	Aleatório

Quadro 4 – Configuração espacial dos cenários.
Fonte: Autoria própria

Como os dois primeiros cenários são distribuídos de forma simétrica, as probabilidades P_S entre cada dispositivo é a mesma, dependendo somente dos parâmetros configuráveis P_{TX} e P_{RX}. Entretanto, o terceiro cenário contém probabilidades distintas entre dispositivos, pois a distância entre eles não é equivalente. Contudo, calculou-se a probabilidade P_S média entre todos os dispositivos da rede, para condição do cenário Gama, verificando uma discrepância de até 3% entre os cenários *Alpha* e *Beta*. Todos os valores utilizados no cenário de comunicação estão descritos na Tabela 2.

Em cada cenário, foram utilizados os valores de probabilidades P_{TX} e P_{RX}: 100%, 95% e 85%. O cenário com 100% representa redes que não possuem interferência externa à própria rede, denominado cenário ideal, ou sem falhas. Com probabilidades inferiores a 85%, as taxas de confiabilidade de entrega de pacotes do Contiki OS são muito pequenas ao comparar com o método proposto, conforme apresentados no capítulo de resultados. Portanto, limitaram-se as análises gráficas entre probabilidades de 85% a 100% para uma comparação entre os métodos utilizados. Além disso, não são realizadas análises de 90% de probabilidade nos cenários *Beta* e *Gama*, pois, conforme simulado no cenário *Alpha*, os resultados de 95% e 85% são suficientes para realizar estipular a eficiência do método, de modo a diminuir o tempo de coleta de resultados.

Tabela 2 – Taxa de sucesso dos cenários a partir das configurações UDGM.

Configurações UDGM		Taxas de Sucesso de Cada Cenário		
P_{TX}	P_{RX}	P_S – *Alpha*	P_S – *Beta*	P_S – *Gama*
100%	100%	100%	100%	100%
95%	95%	91,5%	91,5%	92,6%
90%	90%	83,3%	-	-
85%	85%	75,6%	75,6%	78,7%

Fonte: Autoria própria.

A partir da Figura 35 (a), representação do cenário *Alpha*, nota-se que a área em verde representa a área de comunicação de raio D_{TX}, e a área cinza de raio D_{TI} representa a interferência causada pelas transmissões. Portanto, como no exemplo, o nodo 25 consegue se comunicar com os nodos 18, 24, 26 e 32, ao mesmo tempo em que gera interferência nos

nodos 10, 11, 12, 16, 17, 19, 20, 23, 27, 30, 31, 33, 34, 38, 39 e 40. As setas em vermelho representam os caminhos adotados dos nodos até a central de armazenamento de dados, nesse caso representado pelo nodo “0”. Ressaltando que as rotas demonstradas na Figura 35 não são necessariamente as utilizadas por cada protocolo de comunicação, sendo somente ilustrativa, pois a métrica de cada protocolo de roteamento define suas rotas e, geralmente, não são estáticas.

6.2.3 Análise de Desempenho

Para analisar o desempenho da confiabilidade do protocolo µNet, utilizou-se de métodos comparativos com protocolos padronizados e utilizados em RSSF. Por conveniência, tais protocolos são disponíveis junto com o sistema operacional Contiki OS, que disponibiliza os códigos fonte à comunidade. Portanto, é utilizado o Contiki OS configurado com os protocolos e padrões de comunicação: CoAP, UDP, IPv6, RPL (OF0), 6LoWPAN e IEEE 802.15.4. Para padronizar a comparação, é desenvolvida para ambos os sistemas operacionais uma aplicação denominada *Benchmark*, a qual é responsável por controlar e sincronizar as mensagens geradas e recebidas por cada dispositivo da RSSF. A Figura 36 apresenta o conjunto de protocolos utilizados em cada sistema operacional.

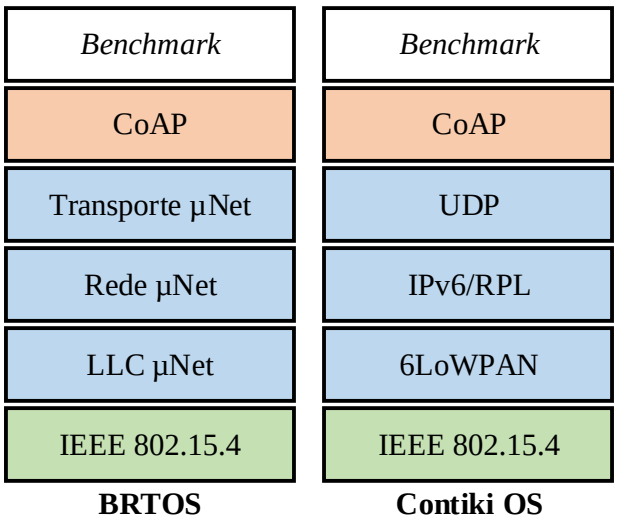


Figura 36 – Conjunto de protocolos de comunicação para RSSF. Lado direito os protocolos de comunicação para o BRTOS, e a direita os protocolos utilizados no Contiki OS.
Fonte: Autoria própria.

Para avaliar o protocolo µNet, comparou-o com dois conjuntos de protocolos, o primeiro com UDP, IPv6, RPL (OF0 por padrão), 6LoWPAN e IEEE 802.15.4, sem utilizar o CoAP, para facilitar a interpretação esse conjunto de protocolo será descrito como C-UDP, Contiki OS com protocolo de transporte UDP. O segundo conjunto de protocolos utiliza o CoAP para gerenciar as transmissões, adicionando seu método de confirmação de entrega fim-a-fim, denominado conjunto C-CoAP, Contiki OS com CoAP. Do mesmo modo, o protocolo µNet foi avaliado sem o CoAP, denominado conjunto B-µNet (BRTOS µNet), e com o protocolo, denominado conjunto B-CoAP (BRTOS µNet com CoAP). A implementação do protocolo CoAP para o BRTOS, foi realizada pela portabilidade do módulo utilizado no Contiki OS, de modo a reduzir os problemas por módulos distintos.

Inicialmente foi estipulada uma base de comparação, ao analisar as confiabilidades do conjunto C-UDP, que não contém confirmações ponto-a-ponto ou fim-a-fim, a não ser pela confirmação de entrega de pacote fornecido pelo padrão IEEE 802.15.4. Dessa forma, é possível verificar o impacto dos cenários simulados com interferência do meio, e o que esperar da rede com conjunto C-UDP, que não contém processos de garantia de entrega de informação de rede. Após estabelecer a base de comparação, foi analisado o conjunto B-µNet, e os impactos de seus métodos na RSSF. Por seguinte, utiliza-se o conjunto C-CoAP para validar se o processo de confirmação ponto-a-ponto apresenta vantagem em relação ao processo de confirmação fim-a-fim, utilizado nas RSSF. Por fim, é analisado o conjunto B-CoAP, de modo a verificar o que ocorre com a RSSF ao utilizar dois protocolos com confirmações distintas, ponto-a-ponto e fim-a-fim.

Foram utilizados sistemas operacionais distintos, pois os protocolos já haviam sido integrados e testados para suas respectivas plataformas. Ao migrar de plataformas pode-se esperar maior complexidade em suas portabilidades, e não detectar problemas sutis que poderiam ter maior impacto na comparação dos protocolos. Portanto, optou-se realizar a análise dos protocolos em seus sistemas operacionais originais, evitando problemas relacionados a erros de portabilidade. Além disso, nas análises de transmissão ponto-a-ponto, descrita na seção 7.2, observa-se que os tempos de processamentos de pacotes recebidos por ambos os sistemas operacionais são próximos.

As análises de desempenho dos conjuntos de protocolos são realizadas ao variar a taxa λ_L de geração de pacotes, de modo a demonstrar a estabilidade de cada conjunto de protocolos à medida que λ_L se torna maior do que a taxa de λ_E de roteamento entre dispositivos, conforme descrito na seção 4.5. A partir disso, é possível validar o sistema quando a quantidade de fluxo é estável, $\lambda_L < \lambda_E$, e quando o sistema se torna instável

$\lambda_L > \lambda_E$, pois não é possível estimar a taxa λ_E. Entretanto, para validar o limite de estabilidade de fluxo, é necessário realizar um processo de sincronismo entre os dispositivos da RSSF, na qual são disponibilizados intervalos de tempos únicos para que cada um tenha capacidade de emitir ao seu destino sem interferência da própria rede. Apesar desse processo não representar a maioria dos sistemas reais, que possuem taxas λ_L aleatórias, esse modelo permite analisar os limites de desempenho de uma RSSF.

A aplicação *Benchmark* foi desenvolvida para os sistemas BRTOS e Contiki OS, para controlar a taxa λ_L de geração de pacotes, intervalos de espera, a quantidade de mensagens emitidas por dispositivos, contabilizar a taxa efetiva de pacotes que chegaram ao destino e sincronizar os processos dos dispositivos. Nota-se que o sincronismo ocorre apenas na camada de aplicação (*Benchmark*), pois realizar sincronismo dos protocolos das camadas inferiores não representaria o processo aleatório de roteamento envolvido em uma RSSF real. Para controlar a taxa λ_L, utiliza-se um intervalo periódico de tempo entre transmissões P, em que cada pacote de dados é criado e enviado. Como o simulador Cooja atribui um identificador único (ID) a cada dispositivo da rede, é possível distribuir P entre os dispositivos de forma que cada um inicie em um instante distinto do outro. Desse modo, é possível controlar o tempo que cada pacote gerado por um dispositivo terá para ser roteado até seu destino, definido por T_R, antes que outro dispositivo gere um novo pacote. Para tal, utilizou-se a equação (4) para contabilizar o tempo de espera de cada dispositivo, T_{ID}, para iniciar sua transmissão dentro de um período P.

$$T_{ID} = (qc - ID)\,\frac{P}{qc}, \qquad \forall\ 0 < ID \leq qc \tag{4}$$

Em que:

P: período entre transmissões de pacote, $1/\lambda_L$;

T_{ID}: tempo de espera único para cada dispositivo dentro de um P;

ID: identificador único de cada dispositivo;

qc: quantidade total de clientes na rede, sem considerar o coordenador.

Se $ID = qc$, então será o primeiro a enviar a informação, com $T_{ID} = 0$. Se $ID = 1$, então o dispositivo 1 será o último a enviar sua informação. Com esse método, é possível controlar os intervalos de tempo T_R para que um pacote seja roteado e entregue ao seu destino, conforme equação (5).

$$T_R = T_{ID-1} - T_{ID}, \qquad \forall \;\; 0 < ID \leq qc \tag{5}$$

A Figura 37 demonstra a divisão de dois períodos entre transmissões, P, entre todos os dispositivos da RSSF, e as setas indicando para cima denotam quando o dispositivo inicia sua transmissão no intervalo P. Nota-se que cada mensagem transmitida, possui um tempo T_R antes que outro dispositivo adicione uma mensagem na rede.

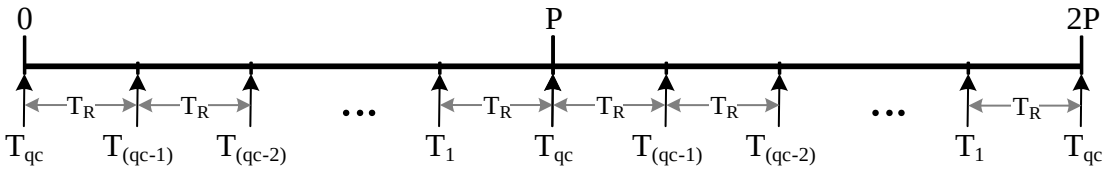


Figura 37 – Distribuição do período entre transmissões entre dispositivos. Representação de dois intervalos entre transmissões P e sua divisão entre todos os dispositivos da rede, em que cada um possui um período único entre início de transmissões.
Fonte: Autoria própria.

Esse processo de controle de transmissão é compatível com os conjuntos de protocolos C-UDP e B-µNet, pois são protocolos que não possuem confiabilidade fim-a-fim. Entretanto, com a adição do protocolo CoAP, não é possível garantir esses intervalos de transmissão, pois seu processo não permite transmissões sucessivas de dados, até que a mensagem seja confirmada, ou que estoure o tempo limite de espera de confirmação. Por causa disso, o intervalo P adiciona somente um atraso extra, sem fornecer sua característica de controle de filas. Portanto, utiliza-se um tempo fixo de 1 segundo após finalizar a transmissão do pacote com o CoAP, como medida preventiva de sobrecarga dos nodos mais próximos ao coordenador.

A proposta inicial do protocolo µNet é utilizar um *buffer* unitário em seu processo de armazenamento de pacotes em roteamento. Consequentemente, é realizada uma comparação de desempenho do protocolo µNet ao analisar dois tamanhos de *buffer* em sua rede, unitária pelo conjunto B-µNet, e de 16 posições, pelo conjunto denominado B-µNet16, o mesmo valor utilizado pelo Contiki OS. Essa análise permite demonstrar o impacto de utilizar um *buffer* unitário em uma RSSF.

O desempenho de cada conjunto de protocolos é contabilizado pela taxa de confiabilidade (C), que é a relação entre quantidade de pacotes recebidos (Pkt_{Rec}) e a soma da quantidade de pacotes emitidos por cada dispositivo (Pkt_{Totais}), conforme equação (6).

$$C = \frac{Pkt_{Rec}}{Pkt_{Totais}} \quad (6)$$

Cada dispositivo da rede foi configurado para transmitir 100 pacotes (*qp*), portanto no cenário *Alpha* são gerados 4.800 pacotes, e nos cenários *Beta* e *Gama* são gerados 9.900 pacotes, conforme calculado pela equação (7). Além disso, é verificada a taxa de confiabilidade de cada dispositivo, de modo a avaliar quais são os dispositivos mais afetados, conforme os números de saltos entre eles e coordenador aumentam.

$$Pkt_{Totais} = qp \times qc \quad (7)$$

Os intervalos entre transmissões, *P*, foram definidos após realizar análise dos tempos de transmissão ponto-a-ponto e a partir dos resultados obtidos nas simulações, alterando-o à medida que fosse necessário. Portanto, foram utilizados diferentes intervalos para cada um dos cenários, variando de um intervalo mais elevado até 1 segundo, conforme denotados na Tabela 3.

Tabela 3 – Valores de *P* para os conjuntos C-UDP e B-µNet. Faixa de períodos entre transmissões utilizadas nos cenários simulados e a taxa de variação para as faixas de períodos.

Nome do Cenário	Cenário Ideal (segundos)	Cenário com Interferência (segundos)	Variação (segundos)
Alpha	Entre 4,5 s e 1 s	Entre 10 s e 1 s	0,5 s
Beta	Entre 11 s e 1s	Entre 16 s e 1 s	1 s
Gama	Entre 11 s e 1s	Entre 16 s e 1 s	1 s

Fonte: Autoria própria

6.2.4 Aplicação *Benchmark*

A aplicação *Benchmark* realiza o controle e sincronismo dos dispositivos, e, para tal, foi particionada em dois códigos, o primeiro adicionado junto aos dispositivos, denominado *Benchmark* Embarcado, e o outro para a ferramenta de automatização de simulação (*script*) do Cooja, denominado código *Benchmark Script*. O código embarcado realiza as operações de enviar as mensagens, temporizar as transmissões e quantificar a confiabilidade da RSSF. O *Benchmark JavaScript* utiliza as funcionalidades internas do simulador para obter exatidão nos processos de sincronismo e coleta de dados dos dispositivos. Para isso, o *script* utiliza a interface de porta serial para sincronizar a rede, e os métodos de acesso direto a memória para coletar as informações da rede.

Para realizar o sincronismo entre todos os dispositivos, é necessário que o código embarcado seja coordenado pelo *script*. Assim, o código embarcado tem por função controlar o intervalo e quantidade de pacotes emitidos, e o *script* é responsável por avaliar se todos os dispositivos estão prontos para enviar as mensagens, e coordená-los para que iniciem a transmissão síncrona entre eles. O início da transmissão síncrona, somente ocorre quando todos os dispositivos possuem uma rota padrão ao coordenador da rede, e estão com suas rotas estabilizadas por alguns instantes.

A Figura 38 apresenta o fluxograma de operação dos códigos *Benchmark*, separados na esquerda pelo processo do código embarcado, e na direita pelo processo utilizado no *script*. Ao iniciar o processo, primeiramente, o *Benchmark Script* coleta informações sobre os dispositivos que estão sendo simulados e aguarda que cada dispositivo da rede estabeleça uma rota até o coordenador. As informações coletadas de quantidade de clientes, quantidade de pacotes a ser enviado e período entre transmissões de pacote são sincronizadas entre todos os dispositivos pelo *Benchmark Script*. Depois que a rede estabelece suas rotas, é utilizado um "Tempo de Estabilização" de quinze segundos para as redes composta de 49 dispositivos e de vinte segundos para uma rede de 100 dispositivos. Este parâmetro é utilizado para que as mensagens de controle para formação de rota se tornem menos frequentes e não impactem significantemente nas análises realizadas. Se não utilizado, poderia haver congestionamento no meio de comunicação, o que impactaria negativamente no desempenho da simulação e podendo causar perturbações nas simulações dos protocolos analisados. Embora seja natural que ocorram transições das rotas durante a simulação, esse método ajuda a minimizar possíveis falhas e sobrecarga da rede na inicialização do processo. Os valores utilizados foram

baseados em análise visual no simulador Cooja do comportamento da rede e, apesar de não serem os valores ideais, foi possível verificar que foram suficientes para seu propósito. Mesmo com a distribuição do período de transmissão para cada dispositivo, é necessário que eles iniciem ao mesmo tempo, para que não ocorram sobreposições dos tempos. Para tal, o *Benchmark Script* envia uma mensagem a todos dispositivos para que iniciem ao mesmo tempo, garantindo o sincronismo entre todos. Isso somente é possível devido à ferramenta de simulação, pois possibilita controlar todos os dispositivos ao mesmo tempo.

Quando um dispositivo finaliza a quantidade necessária de pacotes a transmitir, ele emite uma mensagem de controle ao *Benchmark Script*, indicando seu estado. Quando o último dispositivo envia sua mensagem de fim de transmissão, é aguardado um "Tempo de transmissão do *buffer*" cuja função é de aguardar que o último pacote possa chegar ao seu destino, e ao mesmo tempo, receber alguns pacotes que possam estar presos nos *buffers*. Esse método é necessário, pois não há como garantir que o pacote chegue ao seu destino. Assim como o "Tempo de Estabilização", foram utilizados valores obtidos por análise da RSSF no simulador, sendo de dez segundos para redes de 49 dispositivos e sessenta segundos para 100 dispositivos.

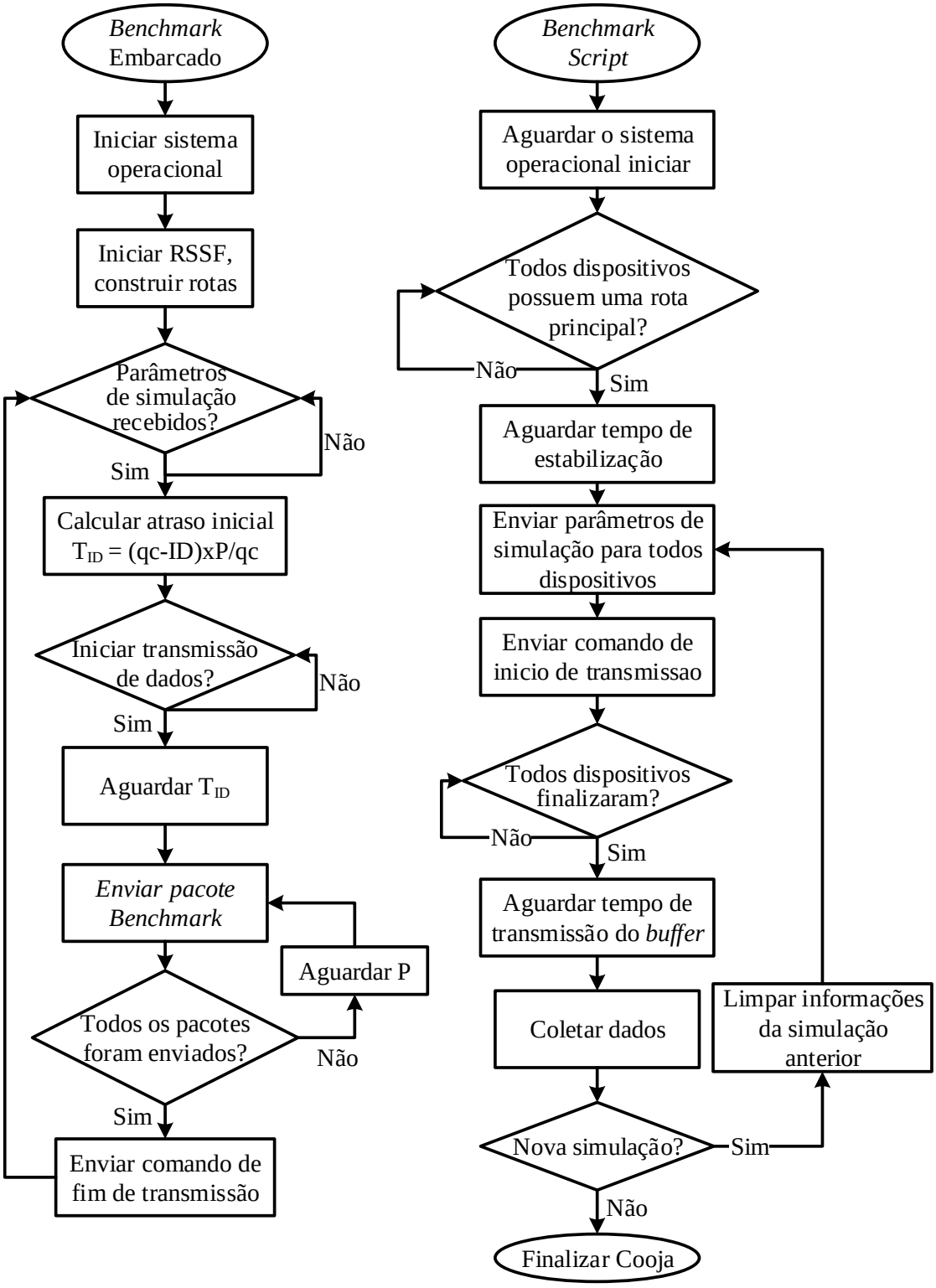


Figura 38 – Fluxograma dos algoritmos *Benchmark*. A esquerda o fluxograma do código embarcado, inserido em cada dispositivo da RSSF. A direita o *script* utilizado no automatizador de simulação do Cooja, escrito em *JavaScript*.

Fonte: Autoria própria.

6.2.5 Configurações e Padrões Adotados

As características configuráveis dos protocolos de RSSF devem ser padronizadas entre as comparações para que a proposta de análise seja válida. Os valores adotados são utilizados em todos os cenários simulados e por cada conjunto de protocolo analisado.

Existem métodos para diminuir a energia consumida pelos dispositivos ao controlar a taxa de trabalho, ligando e desligando-os em momentos bem definidos. Contudo, o objetivo das análises propostas não é minimizar a quantidade de energia consumida, e sim analisar qual a taxa de confiabilidade mais efetiva dos conjuntos dos protocolos propostos. Portanto, foi definido que os rádios devem ficar sempre ligados, ao utilizar taxa nula de controle de trabalho, denominado *null duty cycle*.

Apesar do protocolo UDP não fornecer nenhum meio de verificar se o pacote foi devidamente entregue ao seu destino, o protocolo IEEE 802.15.4 utiliza uma mensagem de confirmação entre os transmissores para verificar, se a mensagem foi entregue ao dispositivo mais próximo. Essa confirmação é utilizada no conjunto de protocolos do Contiki OS, e é essencial para o protocolo proposto µNet.

A camada de enlace dos dois sistemas operacionais foi configurada com o protocolo CSMA/CA, de modo a verificar se o meio está ocupado, antes de enviar alguma informação. Ambos sistemas operacionais realizam 3 retransmissões, caso não seja possível enviar uma mensagem por causa do meio estar ocupado. Além disso, o µNet pode realizar até 30 retransmissões, enquanto não receber a confirmação ACKN.

Como configuração padrão do Contiki OS, o *buffer* de rede possui espaço para armazenar até 16 pacotes. O protocolo µNet utiliza dois tamanhos de *buffer*, unitário conforme proposta inicial, e de 16 pacotes, para comparar como o mesmo tamanho do *buffer* do Contiki OS pode influenciar na confiabilidade da rede.

No Contiki OS foi utilizado IPv6 para endereçar os dispositivos, ao utilizar o módulo otimizado para RSSF denominado uIP. Além disso, foi utilizado o protocolo 6LoWPAN para compressão de informações, utilizando o método HC06 do Contiki OS, configurado por padrão. Como a função objetivo MRHOF, do protocolo RPL, é relevante para dispositivos com restrições de energia, ele é comparado com a função OF0, com intuito de verificar qual o método adequado para ser utilizado nos cenários de simulação propostos.

O CoAP foi configurado para enviar mensagens do tipo CON, que requer o retorno de uma mensagem de confirmação pelo receptor. Foram utilizadas as configurações padrões

de tempo e tentativas de transmissão, 4 tentativas espaçadas em intervalos de 4 s, 8 s, 16 s e 32 s, totalizando no máximo 60 s para aguardar a confirmação de que a mensagem tenha chegado ao destino. Para não ocorrer sobrecargas na rede ao utilizar configurações desnecessárias desse protocolo, não foram utilizados URIs ou quaisquer outros parâmetros que acarretem no aumento do cabeçalho CoAP, totalizando em 6 *bytes* para enviar uma mensagem com informações. Entre os seis *bytes* estão inclusos os 4 *bytes* mínimos do pacote CoAP, 1 *byte* para destinado ao *Token* utilizado na comunicação e 1 *byte* utilizado pelo CoAP para marcar o inicio da informação que será transmitida.

O Quadro 5 contém as configurações gerais que foram utilizadas nos cenários de simulações e nos protocolos de comunicação.

Parâmetro	Configuração
Protocolos Utilizados	µNet, UDP, IPv6, 6LoWPAN, RPL e IEEE 802.15.4
Controle de ciclo de funcionamento do rádio	*null duty cycle*
Controle de Acesso ao Meio	CSMA/CA
Número de retransmissões da camada de enlace do Contiki OS	3 retransmissões
Número de retransmissões até receber o ACKR do µNet	3 retransmissões
Número de retransmissões até receber o ACKN do µNet	30 retransmissões
Tempo de espera do ACKR do µNet	10 milissegundos
Tempo de espera do ACKN do µNet	Entre 64 e 256 milissegundos
Método de compressão 6LoWPAN	HC06
Métricas de roteamento RPL	OF0 e MRHOF com ETX
Configuração do CoAP	Mensagens CON
Tentativas de retransmissão CoAP	4
Intervalos entre tentativas CoAP	4, 8,16 e 32 segundos
Quantidade de dispositivos	49 e 100
Quantidade de coordenadores de rede	1
Tempo de Estabilização da Rede 49 Nodos	15 segundos
Tempo de Estabilização da Rede 100 Nodos	20 segundos
Tempo de Liberação do *Buffer* da Rede 49 Nodos	10 segundos
Tempo de Liberação do *Buffer* da Rede 100 Nodos	60 segundos
Quantidade de pacotes emitidos por dispositivo (*qp*)	100 pacotes
Período *P* entre transmissões na aplicação *Benchmark*	Entre 16 e 1 segundo(s)
Modelo de controle do meio de comunicação	UDGM
Probabilidades P_{TX} e P_{RX} do modelo UDGM	100%, 95%, 90% e 85%

Quadro 5 – Quadro geral de configurações adotadas nas simulações.
Fonte: Autoria própria.

7 RESULTADOS

Os resultados obtidos nesse trabalho foram coletados por simulações realizadas na ferramenta Cooja. Os conjuntos de protocolos C-UDP e C-CoAP tem como objetivo constituir uma base de comparação para a solução proposta e os conjuntos B-µNet, B-µNet16 e B-CoAP tem como objetivo validar a solução proposta com diversas perspectivas de operação. Como o objetivo desse trabalho é aprimorar a confiabilidade de transmissão de dados fim-a-fim em RSSF, os conjuntos são validados quanto a sua confiabilidade de entrega de pacotes fim-a-fim, em cenários ideais e com fontes de interferência externa à rede. A interferência é simulada com as ferramentas do Cooja, ao definir as probabilidades de que um pacote seja emitido e recebido, adequadamente, numa transmissão ponto-a-ponto, definidas pelas variáveis P_{TX} e P_{RX}, às quais foram variadas em conjunto, entre 100% (cenário ideal), 95%, 90% e 85%.

Além de verificar a confiabilidade das redes, analisa-se o tempo necessário para que a confiabilidade seja atingida, ao variar os intervalos entre transmissões dos protocolos que não possuem confirmação fim-a-fim e ao verificar o tempo necessário para que o CoAP transmita a mesma quantidade de mensagens com suas configurações padrões de confirmação fim-a-fim. Por fim, são analisadas as confiabilidades de cada dispositivo da rede, a medida que se incrementa a quantidade de saltos necessários para emitir um pacote até o receptor. Para tal, calcula-se a média das confiabilidades dos dispositivos que possuem a mesma quantidade de saltos, em cada intervalo entre transmissões e em cada cenário. Esses resultados são apresentados por imagens em que suas cores representam a média da confiabilidade, conforme o intervalo entre transmissões de cada cenário simulado.

7.1 TAMANHO DE CABEÇALHOS

A configuração padrão do protocolo IEEE 802.15.4 para RSSF utiliza taxa de comunicação de 250 kbps (32 *byte*s por milissegundo) e 127 *byte*s de armazenamento por pacote, de forma que é necessário utilizar os recursos de forma eficaz. O conjunto C-UDP, com os protocolos IEEE 802.15.4, IPv6 e UDP, utiliza 60 *byte*s de informações para cabeçalho de controle de transmissões, enquanto o B-µNet, com IEEE 802.15.4 e µNet,

utiliza apenas 36 *byte*s. Ao utilizar 6LoWPAN, foi possível verificar que o C-UDP reduz em, no máximo, 21 *byte*s a quantidade de informações utilizadas nos cabeçalhos, totalizando um cabeçalho de 39 *byte*s. Apesar do 6LoWPAN reduzir o tamanho, não pode-se esperar que a taxa de compactação seja uniforme para todos os dispositivos da rede, pois sua técnica de otimização depende de diversos fatores que tornam cada dispositivo com um tamanho de cabeçalho diferente do outro. Por causa disso, realizou-se uma simulação de tempo menor do cenário *Beta*, que resultou em 11 mil pacotes compactados, cujo valor médio deles foi utilizado para contabilizar o tamanho do cabeçalho do C-UDP com a ação do 6LoWPAN. Portanto, a Figura 39 e Tabela 4 apresentam os tamanhos dos cabeçalhos dos protocolos utilizados no B-µNet e C-UDP, sem e com 6LoWPAN.

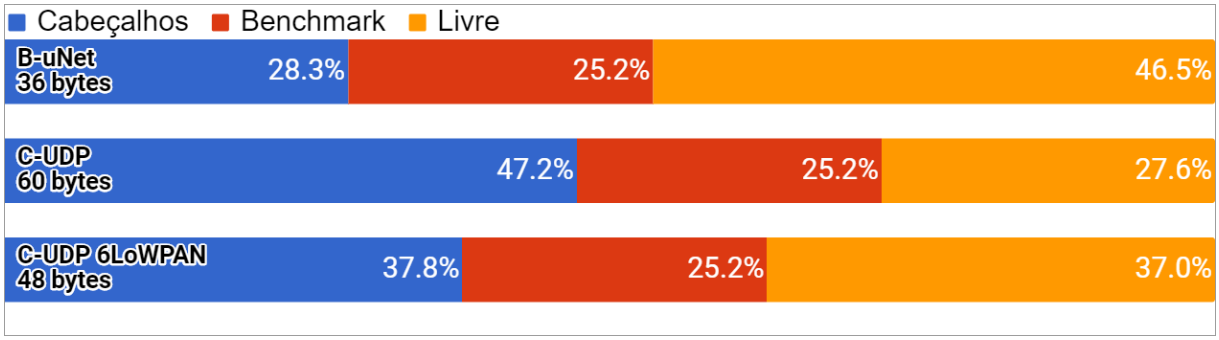


Figura 39 – Proporção do tamanho de cabeçalhos dos conjuntos de protocolos. Representação da porcentagem utilizada do padrão IEEE 802.15.4, que disponibiliza até 127 *bytes*. Em azul os cabeçalhos dos conjuntos B-µNet e C-UDP, em vermelho o tamanho ocupado pela aplicação *Benchmark* e em amarelo o que restou de espaço livre.
Fonte: Autoria própria.

Como o protocolo CoAP não é influenciado pelas métricas do 6LoWPAN, necessita-se adicionar 6 *byte*s, conforme configuração da seção 6.2.5, à soma total de cada método de comunicação. A Tabela 4 representa a quantidade de *byte*s utilizada por cada conjunto de protocolos.

Tabela 4 – Tamanho de cabeçalhos de cada protocolo.

Protocolo	B-CoAP	C- CoAP	C- CoAP 6LoWPAN
Benchmark	32 *bytes*	32 *bytes*	32 *bytes*
CoAP	6 *bytes*	6 *bytes*	6 *bytes*
µNet	24 *bytes*	-	-
UDP + IPv6	-	48 *bytes*	30 *bytes* (valor médio)
IEEE 802.15.4	12 *bytes*	12 *bytes*	12 *bytes*
Total	74 *bytes*	98 *bytes*	80 *bytes*

Fonte: Autoria própria.

A partir desses valores, pode-se concluir que o modo de operação B-µNet consome uma quantidade menor do tamanho total disponível para um pacote de dados. Nota-se que não são utilizados endereços IPv6 para o µNet, não sendo possível acessá-los diretamente. Entretanto, pode-se utilizar um dispositivo específico para roteamento entre redes, com maiores capacidades de processamento e memória, no qual traduziria a informação para o formato necessário. Por fim, essas informações explicam algumas diferenças entre os tempos de transmissões de dados ponto-a-ponto apresentados no capítulo 7.2.

7.2 TRANSMISSÃO PONTO-A-PONTO

De modo a compreender como a confiabilidade de entrega de pacotes em uma rede multissaltos é afetada, primeiro é necessário analisar como ocorrem as transmissões de informações ponto-a-ponto, e os tempos de comunicação exigidos para transmitir um pacote sem interferência da própria rede. Para tal, foram utilizados os conjuntos de protocolos C-UDP e B-µNet para medir os tempos de transmissão ideal, sem interferência de outros no momento em que realiza a transmissão, de um pacote entre dois dispositivos vizinhos, medidos por uma funcionalidade da própria ferramenta de simulação, à qual possibilita extrair informações à respeito das atividades do meio de comunicação sem que seja necessário adicionar códigos aos sistemas embarcados. Por realizar essa análise em momentos que outros dispositivos não atuem na rede, é possível mitigar parcialmente a ação do CSMA/CA, o que elevaria o tempo final dessa análise se o meio de comunicação estivesse ocupado.

O tempo total para enviar uma informação entre dois dispositivos vizinhos é calculado pela soma dos intervalos de tempos necessários para realizar as tarefas de transmissão, confirmação e processamento da informação. Com a abordagem adotada pelo conjunto B-µNet, são classificados oito intervalos necessários para realizar a comunicação entre dois dispositivos X (emissor) e Y (receptor), sendo eles:

1- tempo utilizado pelo dispositivo de comunicação sem fio (rádio) para transmitir um pacote de dados (PKT) de X para Y;
2- tempo de inversão dos modos operacionais do rádio, de recepção para transmissão (INV1);
3- tempo de transmissão da mensagem de confirmação de rádio (RAT1) de Y para X;

4- tempo de processamento do pacote de dados (PROC) pelo dispositivo Y. Durante este intervalo, o pacote (PKT) é analisado e verifica-se se é necessário gerar um pacote ACKN com o µNet;
5- tempo de transmissão do pacote de confirmação de rede (NAT) de Y para X;
6- tempo de inversão dos modos operacionais do rádio, de recepção para transmissão (INV2);
7- tempo de transmissão da segunda confirmação de rádio (RAT2) de X para Y;
8- tempo transcorrido entre o final da recepção da confirmação RAT2 e transmissão do pacote de dados (PKT) de Y para um terceiro dispositivo da rota, denominado tempo de roteamento (ROT). Durante esse intervalo, são analisados os endereços de origem e destino do PKT para executar as rotinas de roteamento e de acesso ao meio.

Por outro lado, o modo de comunicação adotado pelo C-UDP utiliza menos processos, pois não possui modo de confirmação ponto-a-ponto. Portanto o C-UDP contém somente quatro intervalos, sendo eles:

1- tempo utilizado pelo dispositivo de comunicação sem fio (rádio) para transmitir um pacote de dados (PKT) de X para Y;
2- tempo de inversão dos modos operacionais do rádio, de recepção para transmissão (INV1);
3- tempo de transmissão da mensagem de confirmação de rádio (RAT1) de Y para X;
4- tempo transcorrido entre o final da recepção da confirmação RAT1 e transmissão do pacote de dados (PKT) de Y para um terceiro dispositivo da rota, denominado tempo de roteamento (ROT). Durante esse intervalo, são analisados os endereços de origem e destino do PKT para executar as rotinas de roteamento e de acesso ao meio.

A Figura 40 representa as mensagens trocadas entre os dispositivos, conforme texto descritivo. Setas entre emissor e receptor são as trocas de mensagens pelo meio de comunicação sem fio, enquanto os colchetes, ao lado das barras, representam os tempos necessários para cada operação. Nota-se que ACKN é a mensagem de confirmação adotado pelo protocolo µNet e ACKR é a mensagem de confirmação utilizada pelo protocolo IEEE 802.15.4.

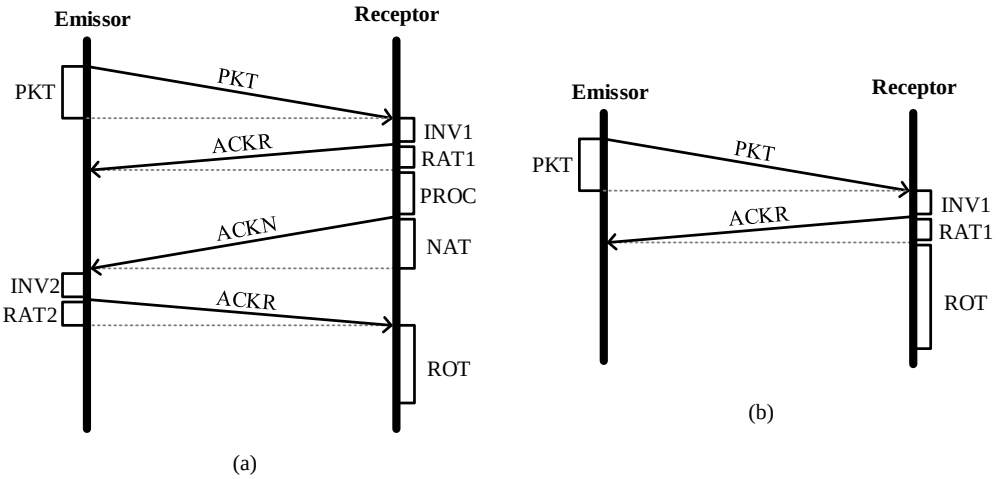


Figura 40 – Representação da transmissão ponto-a-ponto. (a) processo B-µNet e (b) processo C-UDP.
Fonte: Autoria própria.

A Tabela 5 apresenta os dados coletados pelo simulador de acordo com a representação textual anterior. Nota-se que para o C-UDP, os tempos representados por PROC, NAT, INV2 e RAT2 não são aplicáveis, pois a metodologia não utiliza uma confirmação na camada de transporte. Para facilitar a visualização, a Figura 41 apresenta uma representação gráfica dos dados.

Tabela 5 – Tempo para enviar um pacote ponto-a-ponto.

#	Definição	Representação	B-µNet (ms)	C-UDP (ms)
1	Transmissão do pacote de dados	PKT	2,32	2,54
2	Inversão do rádio entre RX e TX	INV1	0,27	0,25
3	Transmissão da confirmação do rádio	RAT1	0,36	0,36
4	Processamento do pacote de dados	PROC	1	NA
5	Transmissão de confirmação da rede	NAT	1,12	NA
6	Inversão do rádio entre RX e TX	INV2	0,27	NA
7	Transmissão de confirmação do rádio	RAT2	0,36	NA
8	Roteamento	ROT	1,10	2,44
9	Total (t_{PP})	-	6,8	5,59

Fonte: Autoria própria.
Nota: Intervalos que não são aplicáveis ao C-UDP estão denotados com "NA".

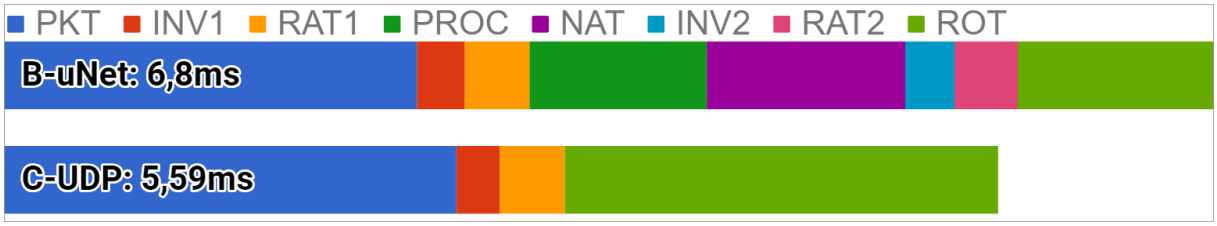


Figura 41 – Representação gráfica da Tabela 5.
Fonte: Autoria própria.

A diferença de transmissões do pacote de dados é devida ao tamanho do cabeçalho de cada metodologia, pois o tamanho da informação utilizada na camada de aplicação *Benchmark* é exatamente a mesma nos dois casos. O B-µNet é composto, no total, por um cabeçalho de 36 *byte*s enquanto o C-UDP é composto por um cabeçalho em média de 42 *byte*s. Com isso, numa rede na qual a taxa de transmissão é dada por 250 kbps, apresenta uma diferença de aproximadamente 0,2 ms entre o B-µNet e o C-UDP. Além disso, os resultados apresentam valores ideais de transmissão ponto-a-ponto, podendo variar dependendo do tipo de ação tomada pelo método CSMA/CA ao ocorrer falhas na comunicação.

A partir dos valores apresentados na Tabela 5, pode-se estimar um intervalo de tempo mínimo necessário para que cada dispositivo envie uma informação até o agregador de dados através de um caminho multissalto. Um dispositivo que está situado a h saltos de seu destino, cujo tempo de transmissão ponto-a-ponto é de t_{PP}, necessita de tempo t_h para enviar sua informação até o destino, conforme equação (8).

$$t_h = t_{PP} \times h \tag{8}$$

Portanto, para um dispositivo situado a 12 saltos de distância de seu destino, é necessário 81,6 ms para que o conjunto B-µNet transmita a informação e de 67,08 ms para o conjunto C-UDP. A partir disso, o intervalo de tempo ideal, P_I, necessário para que cada dispositivo envie ao menos uma mensagem até o coordenador, é dado pela somatória de t_h de cada dispositivo, conforme equação (9), no qual h_i representa a quantidade de saltos do dispositivo i.

$$P_I = \sum_{i=1}^{qc} t_{PP} \times h_i \tag{9}$$

Contudo, as rotas de cada dispositivo são mutáveis no tempo, de forma que esse intervalo possa aumentar e diminuir, de acordo com o protocolo de roteamento. Além disso, em um sistema real não é possível prever todas as condições e estados que o sistema possa adotar, o que pode modificar o intervalo P_I. Entretanto, é possível realizar uma simplificação do tempo necessário ao utilizar o maior tempo dos dispositivos em cada cenário. Por exemplo, no cenário *Alpha* são necessários 12 saltos para que a informação do dispositivo mais afastado seja entregue ao coordenador, enquanto no cenário *Beta* e *Gama* são necessários 18 saltos.

Então, pode-se estipular que no pior caso, todos os dispositivos da rede necessitarão de, pelo menos, a mesma quantidade de tempo ideal do que o dispositivo mais distante. Portanto, o intervalo de tempo mínimo entre transmissões, P, para o pior caso de cada cenário e conjunto de protocolos, é contabilizado pela equação (10), o que resulta na Tabela 6.

$$P = t_{pp} \times h \times qc \tag{10}$$

Tabela 6 – Intervalo entre transmissões *P* para o pior caso.

Cenário	qc	h	$P_{B-\mu Net}$	P_{C-UDP}
Alpha	48	12	3,92 s	3,22 s
Beta e *Gama*	99	18	12,12 s	9,96 s

Fonte: Autoria própria.

Ao utilizar esses valores de P nos conjuntos de protocolos C-UDP e B-µNet, espera-se que as comunicações dos dispositivos da rede operem sem falhas, pois $\lambda < \mu$ e não haveria duas transmissões simultâneas.

Por fim, comparam-se os tempos de processamento dos sistemas operacionais Contiki OS e BRTOS dos pacotes recebidos. O tempo de roteamento do Contiki OS, de 2,42 ms, contém os processos de descompactar o pacote, processo reverso do 6LoWPAN, e identificar a próxima rota do pacote. Portanto, espera-se que tenha um tempo superior para seu processamento. Por outro lado, o BRTOS divide o tempo de processamento em duas instâncias, PROC e ROT, os quais somam 2,2 ms. Nos processos utilizados pelo BRTOS, contabiliza-se somente o tempo de roteamento, pois o pacote não é compactado por outros métodos. A partir desses valores, percebe-se que os tempos de processamentos são próximos, o que indica que a diferença de sistemas operacionais não é crucial ao tipo de análise realizada.

7.3 ANÁLISE DE CONFIABILIDADE DO CONJUNTO C-UDP

Conforme descrito no Capítulo 5.3, Materiais e Métodos, o conjunto C-UDP é utilizado como padrão de comparação para validar a confiabilidade do conjunto B-µNet. Contudo, para utilizá-lo como método de comparação, é necessário realizar a análise de desempenho do conjunto de protocolos C-UDP, para obter seus valores de confiabilidade e assim compará-lo. Além disso, comparam-se as funções objetivo do RPL, OF0 e MRHOF com métrica ETX, de modo a verificar qual a melhor função objetivo para os cenários de simulação proposto, e assim validar se o processo de roteamento adotado pelo µNet é adequado, pois suas métricas são semelhantes à da OF0. Nos cenários ideais, espera-se que ambas as métricas possuam taxas de confiabilidade elevadas para intervalos elevados, pois não deve haver perda de dados por colisões internas da rede ou interferências externas. Os gráficos da Figura 42 apresentam as análises de confiabilidade das funções objetivo OF0 e MRHOF no conjunto de protocolos C-UDP em cenários ideais.

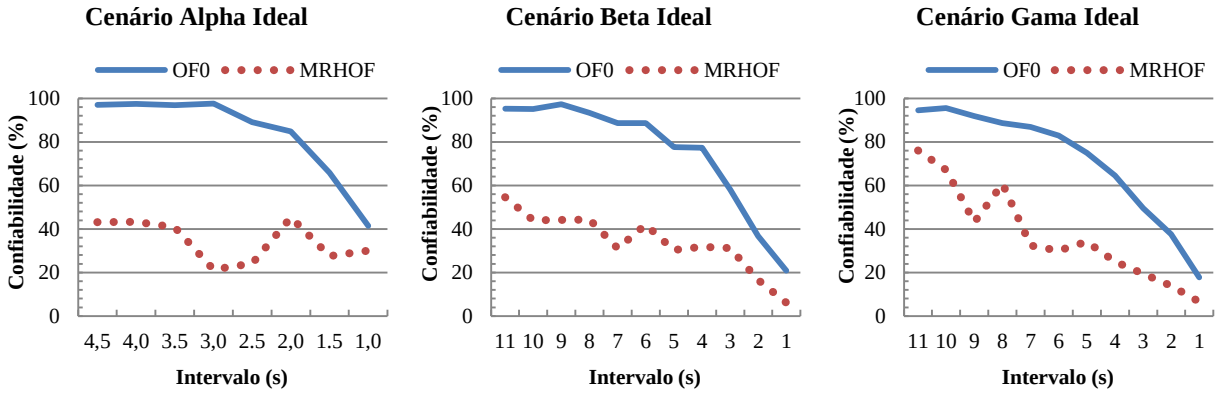


Figura 42 – Comparação entre OF0 e MRHOF em cenários ideais. Comparação das Funções Objetivo do conjunto C-UDP. Eixo vertical apresenta a confiabilidade *C* em % e o eixo horizontal contém a variação dos períodos entre transmissão, *P*, em segundos.
Fonte: Autoria própria.

Como no cenário ideal não há perda de informação por interferência externa à rede, os intervalos em que $\lambda < \mu$ acima de 3 segundos para o cenário *Alpha* e 9 segundos para os cenários *Beta* e *Gama*, conforme Tabela 6, é possível verificar que o conjunto C-UDP, com função objetivo OF0, é capaz de entregar próximo de 98% dos pacotes gerados em sua rede. Estipula-se que a OF0 não tem capacidade de atingir 100% de confiabilidade, devido às mensagens de controle de rede e a possíveis mudanças nas rotas de dispositivos vizinhos interferirem na comunicação dos pacotes de dados. Contudo, a MRHOF obteve os piores

resultados em todos os cenários simulados, atingindo no melhor caso 43%, 54% e 76% nos respectivos cenários. Durante a simulação, foi possível observar que a MRHOF realiza maior variação de rotas do que a OF0, o que indica uma maior troca de mensagens de controle de rede. Por causa da troca elevada de rotas, alguns nodos perdem suas rotas por curtos intervalos de tempo no decorrer da simulação, dessa forma a informação não consegue ser entregue ao seu destino, aumentando drasticamente a perda de informação. É possível verificar que o cenário *Gama* foi o menos afetado, pois, por ser aleatório, acabou possuindo menor quantidade de rotas possíveis para formar, tendo menos variação durante os períodos de simulação.

A seguir, foram realizadas simulações em cenários com simulação de interferência externa à rede. Os gráficos da Figura 43 apresentam as análises de confiabilidade das funções objetivo OF0 e MRHOF no conjunto de protocolos C-UDP em cenários com interferência simulada. Apesar de ambas as funções terem sido afetadas drasticamente pela aplicação de interferência na rede, verifica-se que a função MRHOF obteve os piores resultados de confiabilidade. Ao utilizar taxa de sucesso de transmissão de 95%, verifica-se que a confiabilidade da OF0 fica próxima de 60% no cenário *Alpha* e 40% para os cenários *Beta* e *Gama*, enquanto a MRHOF fica entre 20% e 30% na maioria dos intervalos dos cenário *Alpha*, *Beta* e *Gama*. Quanto maior a taxa de interferência na rede, é notável que a taxa de confiabilidade diminui para ambas funções, com valores abaixo de 10% de entrega nos piores casos, o que demonstra que ambas funções não auxiliam na confiabilidade de entrega de dados. Nota-se que os valores de confiabilidade da função MRHOF foram obtidos com as configurações padrões providos no código do Contiki OS. Portanto, deve ser possível melhorar sua confiabilidade ao modificar suas configurações. Entretanto, verifica-se que essa métrica é sensível aos cenários simulados, o que a torna inviável para as simulações propostas.

Além disso, verifica-se que os cenários de simulação são próprios para a função OF0, pois nos cenários ideais foi possível atingir os valores próximos aos ideais, conforme calculados e demonstrados na seção 7.2. Por fim, verifica-se que as métricas de roteamento utilizadas no µNet também são válidas para os cenários propostos, visto que utiliza métricas similares a da função objetivo OF0.

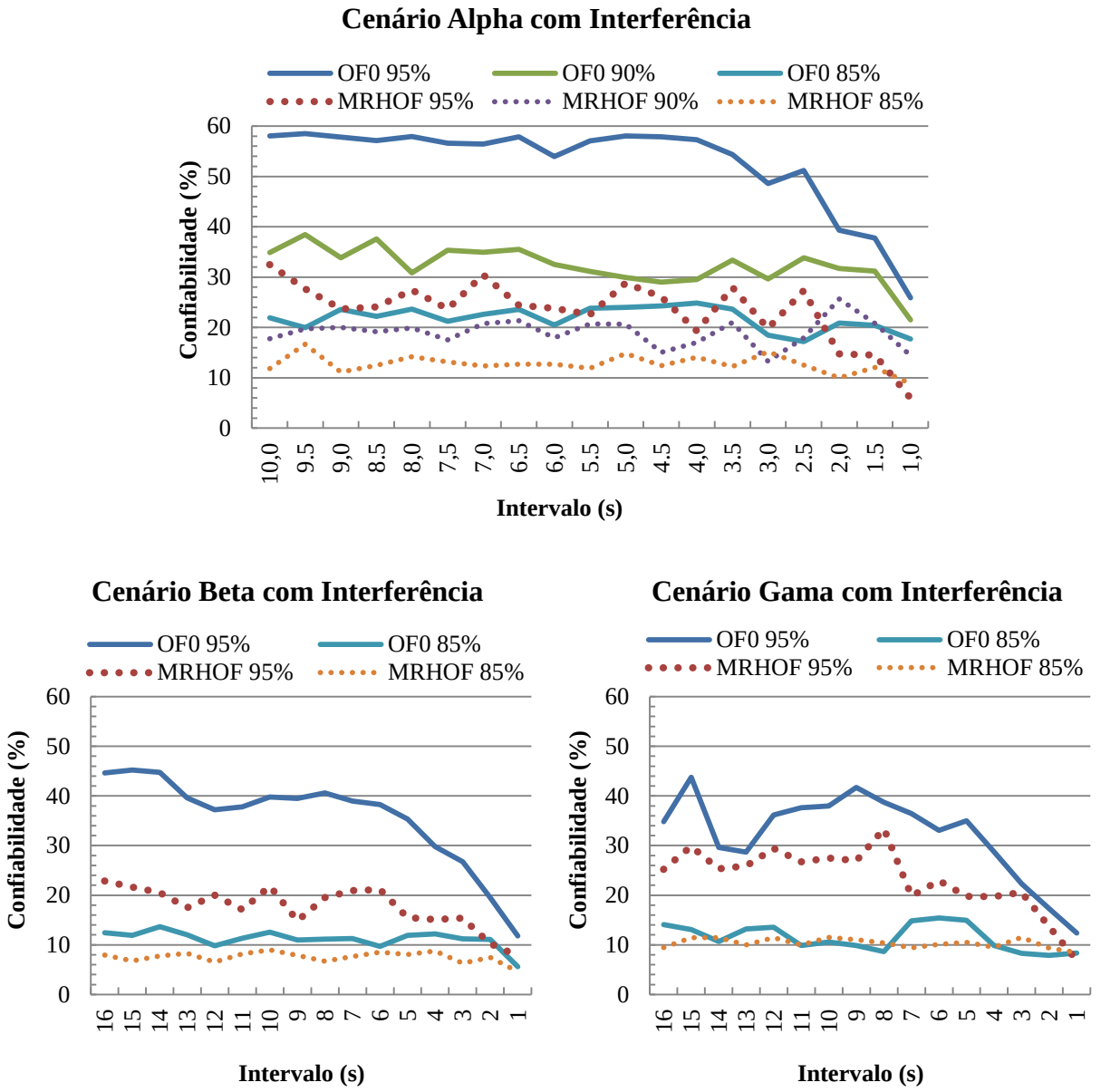


Figura 43 – Comparação entre OF0 e MRHOF em cenários com interferência. Comparação das Funções Objetivo do conjunto C-UDP. Eixo vertical apresenta a confiabilidade C em % e o eixo horizontal contém a variação dos períodos entre transmissão, P, em segundos. Cada OF possui valores iguais de P_{TX} e P_{RX}: 95%, 90% e 85%.

Fonte: Autoria própria.

7.4 ANÁLISE DE CONFIABILIDADE DO CONJUNTO B-µNET

Após realizar as análises do conjunto de protocolo C-UDP, foi verificado que a função de roteamento com melhor desempenho de confiabilidade é a OF0, a qual é utilizada para validar o conjunto B-µNet. Por padrão, o protocolo µNet utiliza *buffer* de rede de tamanho unitário, de modo que armazena somente um pacote por vez. Portanto, primeiro é realizada a análise do B-µNet de *buffer* unitário, comparando-o com o C-UDP, e após realiza-se a análise do B-µNet com *buffer* de tamanho 16, o mesmo utilizado pelo C-UDP. A Figura 44 realiza a comparação de confiabilidade do conjunto C-UDP e B-µNet nos cenários ideais.

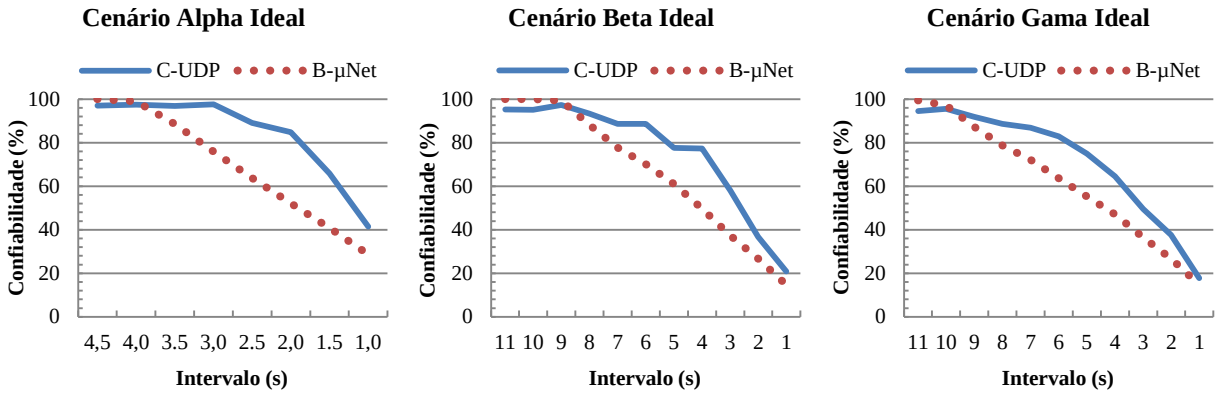


Figura 44 – Comparação entre C-UDP e B-µNet em cenários ideais. Eixo vertical apresenta a confiabilidade C em % e o eixo horizontal contém a variação dos períodos entre transmissão, P, em segundos.
Fonte: Autoria própria.

Nos cenários ideais de comunicação, é possível verificar que o conjunto C-UDP possui taxa de confiabilidade próxima a 98% e B-µNet apresenta valores de 100% nos intervalos em que $\lambda < \mu$. Entretanto, ao diminuir os intervalos, é notável que o conjunto C-UDP possui taxas de confiabilidade maiores do que o B-µNet. Portanto, com intervalos entre transmissões de 3 segundos no cenário *Alpha* é possível entregar 98% de dados com o conjunto C-UDP, enquanto são necessários 4 segundos para o conjunto B-µNet. Para os cenários *Beta* e *Gama* nota-se que são necessários ao menos 9 e 10 segundos, respectivamente, para ambos conjuntos de protocolos ficarem próximos de 100% de confiabilidade. Apesar do valor ser abaixo do estimado para o B-µNet, esse aspecto é esperado, pois é estimado que seja necessário 12,12 segundos para o pior caso, no qual cada dispositivo atribui interferência a todos os outros, enquanto realiza sua comunicação, impossibilitando comunicações paralelas. Como nos cenários maiores o raio do sinal de

interferência afeta uma proporção menor de dispositivos do que em cenários menores, é possível realizar mais transmissões simultâneas sem que um interfira com o outro, de modo a ser possível diminuir o intervalo entre transmissões.

Contudo, nota-se que o B-µNet é mais lento que o C-UDP, algo já esperado, devido a necessidade de enviar mais informações para controle da rede. Entretanto, pelo fato do B-µNet utilizar um *buffer* unitário (127 *byte*s), enquanto o C-UDP utiliza um *buffer* de capacidade 16 pacotes (2.032 *byte*s), demonstra que, apesar da diferença, nesse cenário, não há perdas de informação significantes. Isto torna o método acessível aos dispositivos com pouca capacidade de memória. Além disso, durante as simulações foi observado que as rotas utilizadas por ambos os métodos foram similares, com menor taxa de troca do que a função MRHOF, pois ambos dependem somente da quantidade de saltos, o que valida o método de comparação entre C-UDP e B-µNet.

Apesar da confiabilidade do B-µNet ser menor do que C-UDP nos cenários ideais com intervalores menores, a partir Figura 45 é possível verificar a eficácia da confiabilidade de ambos os conjuntos de protocolos ao simular fontes de interferência externa à rede.

Ao comparar com os dados retirados do cenário ideal, pôde-se observar que o C-UDP não é robusto em cenários com interferência, pois mesmo com intervalos de transmissão elevados, há perdas significativas de dados. Por outro lado, ao utilizar o B-µNet, nota-se que é possível garantir a entrega de informações, se fornecido o tempo necessário para realizar as tentativas de transmissões. Ao analisar o cenário *Alpha*, verifica-se que ao utilizar 7 segundos com probabilidade de entrega de informação de 95%, é possível ter uma taxa de confiabilidade próxima a 100%, o que não ocorre com nenhum intervalo do conjunto C-UDP. Além disso, com probabilidade de entrega de 90% e 85%, é possível verificar que com 10 segundos de intervalo, o B-µNet obtém 90% e 80% de confiabilidade, enquanto o C-UDP fica próximo de 35% e 20%, respectivamente.

Nos cenários, *Beta* e *Gama*, nota-se que o B-µNet fica entre de 80% e 90% de confiabilidade com 95% de probabilidade de entrega, e entre 50% e 60% com probabilidade de entrega de 85%. Por outro lado, o C-UDP entrega, no melhor, caso 45% de pacotes com probabilidade de transmissão de 95% e entre 10% e 15% com probabilidades de 85%. Em todos os cenários, o B-µNet consegue fornecer maior taxa de confiabilidade do que C-UDP, o que demonstra que a solução proposta tem potencial para ser utilizada em ambientes que possuem maior taxa de interferência externa. Além disso, a confiabilidade do conjunto B-µNet pode ser ajustada de acordo com o intervalo entre as transmissões, fator que não pode

ser observado para o C-UDP em cenários com ruído e em intervalos entre transmissões maiores.

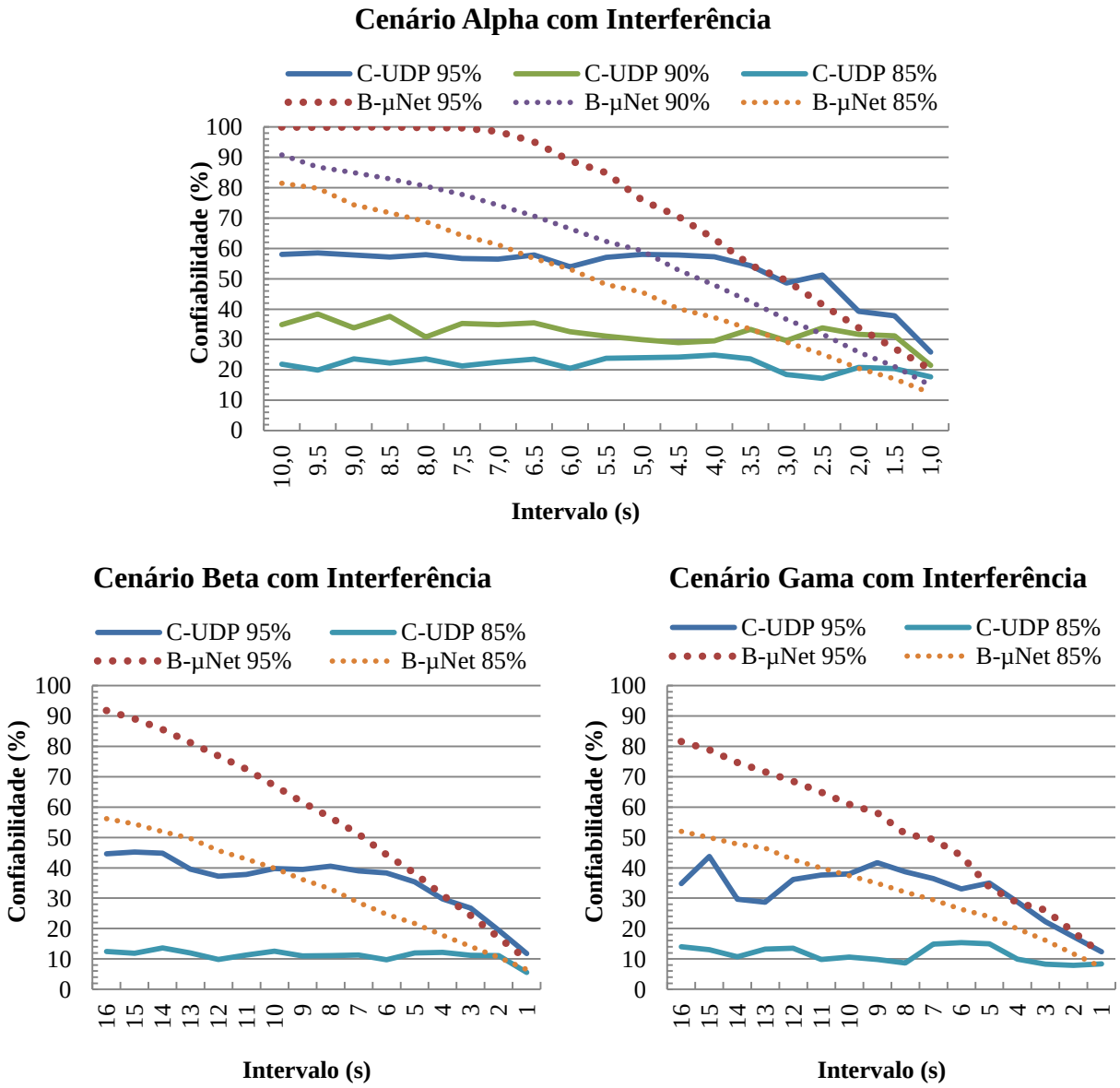


Figura 45 – Comparação entre C-UDP e B-µNet em cenários com interferência. Eixo vertical apresenta a confiabilidade C em % e o eixo horizontal contém a variação dos períodos entre transmissão, P, em segundos. Cada OF possui valores iguais de P_{TX} e P_{RX}: 95%, 90% e 85%.

Fonte: Autoria própria.

7.4.1 Impacto na Confiabilidade do B-µNet pelo Tamanho do *Buffer*

Apesar do B-µNet ter obtido melhor desempenho do que o conjunto C-UDP, é necessário avaliar o impacto na confiabilidade da rede ao utilizar o *buffer* unitário. Para tal, é realizada a comparação entre o B-µNet de *buffer* unitário e de *buffer* de 16 posições, valor utilizado no C-UDP, conforme configuração padrão do Contiki OS. Para facilitar a escrita e descrição das soluções, a rede com *buffer* de 16 posições foi denominado B-µNet16. Portanto, a Figura 46 apresenta os resultados das comparações em cenários ideais e a Figura 47 os resultados dos cenários com simulação de interferência.

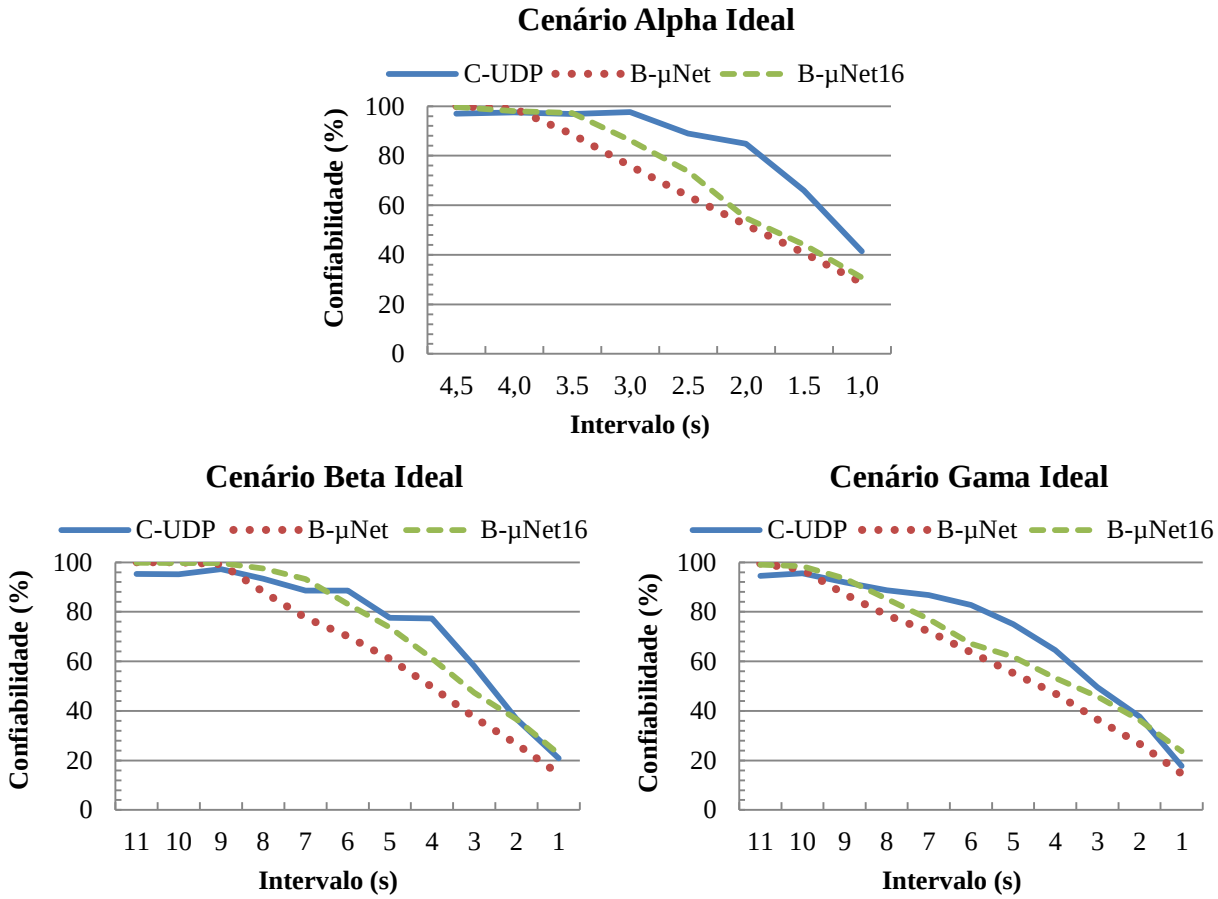


Figura 46 – Comparação entre C-UDP, B-µNet de *buffer* unitário e de capacidade 16 em cenários ideais. As curvas denotadas por B-µNet representam a solução com *buffer* unitário, enquanto B-µNet16 representa a solução de *buffer* com capacidade 16.

Fonte: Autoria própria.

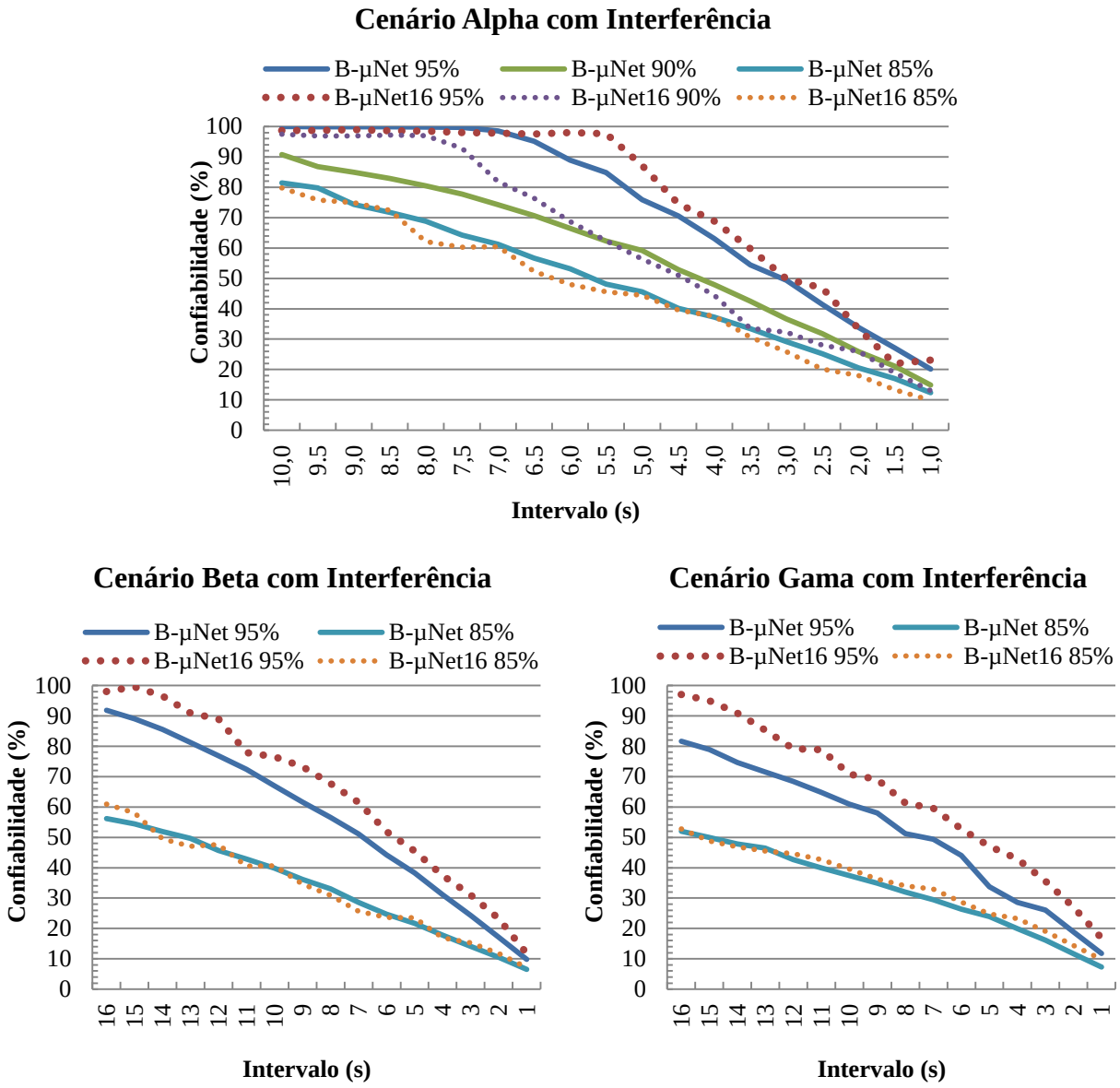


Figura 47 – Comparação entre B-µNet com *buffer* unitário e de capacidade 16 em cenários com interferência. Eixo vertical apresenta a confiabilidade C em % e o eixo horizontal contém a variação dos períodos entre transmissão, P, em segundos. Cada OF possui valores iguais de P_{TX} e P_{RX}: 95%, 90% e 85%.

Fonte: Autoria própria.

Nos cenários ideais, verifica-se que o B-µNet16 obteve maior confiabilidade do que o B-µNet, entretanto, menor do que o C-UDP. A partir disso, verifica-se que o *buffer* não é a causa principal do B-µNet ter menor desempenho do que o C-UDP. Ainda assim, seu impacto é significativo na solução, de modo a aumentar a confiabilidade em aproximadamente 10% a medida que os intervalos diminuem. Por outro lado, nos cenários com simulação de interferência, é possível verificar que há um ganho nos casos em que a probabilidade de entrega de pacotes é de 95%. No cenário *Alpha*, a confiabilidade se mantém elevada por mais intervalos, possibilitando aumentar a velocidade de transmissão de dados. Nos cenários *Beta* e

Gama, verifica-se que o ganho é constante para todos os intervalos, o que indica que quanto maior a rede, mais benéfico é o uso de *buffers* maiores. Entretanto, nota-se quanto maior a interferência na rede, menor a contribuição do *buffer* na rede, verificável pelas probabilidades de entrega de 85%, na qual ambas as soluções tiveram desempenhos próximos em todos os cenários simulados.

Portanto, apesar do *buffer* auxiliar na confiabilidade de entrega de pacotes, seu impacto não é tão significativo quanto o método de confirmação de entrega de pacotes, tendo seu melhor desempenho em redes maiores e com interferência moderada. Caso a fonte de ruído seja intensa, não há necessidade de utilizar um *buffer*, pois ele não auxilia com precisão na correção desse problema.

7.5 ANÁLISE DE CONFIABILIDADE COM PROTOCOLO CoAP

O CoAP é um dos protocolos que podem ser utilizados em RSSF para enviar mensagens com métodos de confirmação de recebimento. Portanto, são realizadas comparações utilizando-o com os conjuntos de protocolos C-UDP e B-µNet, denominando-os C-CoAP e B-CoAP, respectivamente. Como o CoAP utiliza mensagens de confirmação fim-a-fim para validar se a mensagem foi devidamente enviada, seu processo é composto por tentativas de transmissão de envio, enquanto não receber a confirmação do receptor. A aplicação *Benchmark* envia uma nova mensagem ao receber a confirmação de que a mensagem foi enviada devidamente, ou se atingir o limite de retransmissões e tempos de espera. Com a adição do CoAP, não é possível realizar um cálculo exato do tempo de simulação pelo intervalo entre transmissões. Por causa disso, são apresentados os tempos de simulação total, no qual são enviados todos os pacotes de cada cenário, 4.800 pacotes no cenário *Alpha* e 9.900 pacotes nos cenários *Beta* e *Gama*. Entretanto, é possível definir o tempo máximo que o CoAP deve atingir, com a configuração padrão, para enviar 100 pacotes por dispositivo, sendo de 100 minutos (1 hora e 40 minutos), pois cada pacote pode aguardar até 1 minuto para receber a confirmação do receptor.

As Tabelas 7 e 8 apresentam, respectivamente, os resultados de confiabilidade e tempo de simulação dos conjuntos C-CoAP e B-CoAP no cenário *Alpha*, variando os intervalos entre transmissões de 9 segundos a 1 segundo, de forma a verificar o que ocorre com o método *Benchmark* ao adicionar o CoAP como gerenciador de transmissões.

Tabela 7 – Confiabilidade e tempo de simulação do conjunto C-CoAP no cenário *Alpha*. As colunas *C* apresentam os valores de confiabilidade de cada conjunto, a coluna Tempo refere-se ao tempo necessário para enviar todas as mensagens de todos os dispositivos, descrito em horas, minutos e segundos, e a coluna *P* é o intervalo entre transmissões.

P	*Alpha* 100%		*Alpha* 95%		*Alpha* 90%		*Alpha* 85%	
	C	Tempo	*C*	Tempo	*C*	Tempo	*C*	Tempo
9 s	99,87%	1h 48m 32s	91,73%	1h 50m 46s	74,73%	1h 50m 25s	57,54%	1h 51m 21s
7 s	99,75%	1h 46m 56s	94,79%	1h 47m 01s	73,58%	1h 46m 45s	54,73%	1h 48m 21s
5 s	99,96%	1h 44m 00s	93,27%	1h 44m 22s	76,06%	1h 44m 38s	53,06%	1h 44m 50s
3 s	98,44%	1h 41m 32s	88,60%	1h 40m 39s	71,67%	1h 41m 38s	53,10%	1h 41m 36s
1 s	98,50%	1h 38m 21s	86,81%	1h37m 13s	73,27%	1h 37m 00s	53,52%	1h 37m 13s

Fonte: Autoria própria.

A partir dos resultados dos cenários simulados apresentados na Tabela 7, verifica-se que a adição do intervalo *P* entre as transmissões, não contribui com o aumento da confiabilidade do C-CoAP. Contudo, incrementa-se o tempo máximo necessário para realizar as tentativas de envio de mensagens da rede, conforme se aumenta o *P*. Além disso, verifica-se que o tempo necessário para enviar todos os pacotes é próximo do limite máximo disponível, calculado por 100 minutos do CoAP mais o intervalo entre retransmissões do *Benchmark* multiplicado pela quantidade de pacotes emitidos. Tal comportamento indica que, apesar do método de confirmação do CoAP melhorar a confiabilidade da rede, há diversos problemas de desempenho na rede. Assim como o conjunto C-UDP, pacotes podem ser perdidos por causa de interferências ao realizar trocas de mensagens de controle na rede, ou quando um dispositivo modifica sua rota, de modo que a mensagem de confirmação não consiga retornar de modo apropriado ao emissor da informação.

Tabela 8 – Confiabilidade e tempo de simulação do conjunto B-CoAP no cenário *Alpha*. As colunas *C* apresentam os valores de confiabilidade de cada conjunto, a coluna Tempo refere-se ao tempo necessário para enviar todas as mensagens de todos os dispositivos, descrito em horas, minutos e segundos, e a coluna *P* é o intervalo entre transmissões.

P	*Alpha* 100%		*Alpha* 95%		*Alpha* 90%		*Alpha* 85%	
	C	Tempo	*C*	Tempo	*C*	Tempo	*C*	Tempo
9 s	100%	15m 39s	87,67%	49m 05s	85,33%	57m 56s	82,98%	1h 05m 41s
7 s	89,00%	31m 33s	90,35%	48m 51s	85,77%	58m 34s	85,69%	1h 11m 03s
5 s	89,00%	30m 37s	86,87%	49m 24s	84,40%	58m 58s	83,90%	1h 10m 49s
3 s	87,33%	32m 09s	89,08%	48m 32s	85,67%	59m 44s	82,71%	1h 08m 37s
1 s	90,75%	31m 22s	90,21%	49m 03s	85,94%	58m 49s	84,77%	1h 10m 01s

Fonte: Autoria própria.

Do mesmo modo, de acordo com a Tabela 8, a confiabilidade não se altera significativamente ao modificar o intervalo *P*, com exceção do cenário *Alpha* ideal com

intervalo de 9 segundos. Contudo, verifica-se que em determinados cenários a confiabilidade incrementa enquanto o intervalo *P* reduz, pois ao ocorrer perda de dados o fator de correção pelo CSMA/CA, ou mesmo o modo de realizar as tentativas de transmissão, podem alterar o resultado final. Além disso, a método de simulação de interferência utiliza métodos probabilísticos, o que ocasiona em resultados distintos em cada simulação. Conforme verificado nos tempos de transmissões ponto-a-ponto, e nas simulações anteriores do conjunto B-µNet, são necessários 4 segundos de intervalo entre transmissão para que confiabilidade da rede fique próxima à 100% no cenário *Alpha* ideal. Com a adição do CoAP, há uma mensagem de confirmação que transita do destinatário ao emissor, portanto, o tempo necessário nesse caso é o dobro do tempo anterior, algo em torno de 9 segundos. Nesse intervalo, λ é inferior a μ, de modo que a rede possui capacidade de enviar todas as suas informações, sem concorrência do meio de comunicação e sem a necessidade das retransmissões fornecidas pelo protocolo CoAP.

Além disso, nota-se que os tempos de simulação em cada cenário não modificam em relação ao intervalo entre transmissões, situando-se entre 30, 49, 58 e 70 minutos para cada cenário. Esses valores são abaixo do limite máximo permitido pelo CoAP, de 100 minutos, o que demonstra que a perda de pacotes na rede ocorre devido a concorrência do meio de comunicação, pelos dispositivos. Ao enviar a mensagem, cada dispositivo deve fazer suas confirmações ponto-a-ponto, e para retornar a confirmação fim-a-fim, as confirmações ponto-a-ponto ocorrem mais uma vez. Por esse motivo, mesmo com as retransmissões em intervalos distintos pelo CoAP, algumas mensagens são descartadas, diminuindo a confiabilidade final do conjunto B-CoAP. O mesmo não ocorre no conjunto C-CoAP, pois a rede possui menor concorrência do meio de comunicação por utilizar menos mensagens de confirmação, de modo a obter confiabilidades significativas, mesmo com intervalos entre transmissões reduzidos. Contudo, devido às confirmações ponto-a-ponto, o B-CoAP finaliza antes do tempo máximo de simulação em todos os casos, pois as mensagens que transitam do receptor ao emissor (fim-a-fim) possuem menores chances de serem perdidas, o que ocasiona em menor quantidade de retransmissões pelo protocolo CoAP. Por outro lado, as mensagens de confirmação com o conjunto C-CoAP são perdidas nas rotas do coordenador ao sensor, resultando em maior quantidade de retransmissão de pacotes, prolongando o tempo de comunicação do protocolo CoAP e, consequentemente, em tempos de simulações maiores do que o B-CoAP.

Devido à invariância da confiabilidade e do tempo total de simulação pela adição de intervalos entre transmissões, foi utilizado o intervalo entre transmissões de 1 segundo nas análises dos cenários *Beta* e *Gama*, como medida preventiva para não sobrecarregar os sistemas operacionais. Além disso, para validar o conjunto B-CoAP é necessário que λ seja maior do que μ, para que o CoAP realize suas funções. Portanto, a Tabela 9 apresenta as análises de desempenho dos conjuntos C-CoAP e B-CoAP, assim como os melhores resultados obtidos pelo conjunto B-µNet em cada cenário simulado. Nota-se que a coluna *P* denota somente os intervalos entre transmissões do conjunto B-µNet, pois os conjuntos com o CoAP utilizam intervalo fixo de 1 segundo.

Tabela 9 – Comparação de confiabilidade e tempo de simulação entre conjuntos C-CoAP, B-CoAP e B-µNet. As colunas *C* apresentam os valores de confiabilidade de cada conjunto, a coluna Tempo refere-se ao tempo necessário para enviar todas as mensagens de todos os dispositivos, descrito em horas, minutos e segundos, e a coluna *P* é o intervalo entre transmissões adotado no conjunto B-µNet.

Cenário	P_{TX} e P_{RX}	C-CoAP		B-CoAP		B-µNet		
		C	Tempo	*C*	Tempo	*C*	Tempo	*P*
Alpha	100%	98,50%	1h 38m 21s	90.75%	31m 22s	98,96%	06m 41s	4 s
	95%	86,81%	1h 37m 13s	90.21%	49m 03s	99,71%	16m 26s	10 s
	90%	73,27%	1h 37m 00s	85.94%	58m 49s	90,79%	16m 26s	10 s
	85%	53,52%	1h 37m 13s	84.77%	1h 10m 01s	81,44%	16m 26s	10 s
Beta	100%	99,67%	1h 38m 49s	75.17%	1h 02m 22s	99,51%	18m 54s	11 s
	95%	78,12%	1h 38m 16s	69.36%	1h 23m 25s	81,60%	27m 02s	16 s
	85%	34,61%	1h 38m 49s	58.14%	1h 36m 27s	52,03%	27m 02s	16 s
Gama	100%	98,51%	1h 38m 54s	75.54%	1h 04m 26s	99,23%	15m 38s	9 s
	95%	78,28%	1h 38m 57s	68.66%	1h 26m 37s	91,82%	27m 02s	16 s
	85%	34,42%	1h 39m 23s	57.40%	1h 43m 32s	56,15%	27m 02s	16 s

Fonte: Autoria própria.

A partir dos resultados da Tabela 9 e dos resultados abordados anteriormente, é possível verificar que o C-CoAP possui maior confiabilidade do que com o C-UDP, aproximadamente, entre 20% e 30%, ao custo de um tempo muito maior de comunicação. Por outro lado, ao comparar o B-CoAP com o B-µNet, nota-se que a confiabilidade diminui, à medida que o tempo aumenta para enviar a mesma quantidade de mensagens, causado pelo congestionamento do meio de comunicação. Ao comparar o C-CoAP com o B-CoAP, observa-se que nas situações em que a interferência simulada do meio é menor, o C-CoAP obteve maior confiabilidade, e quanto maior a interferência do meio, a confiabilidade do

B-CoAP se sobressai. Entretanto, na maioria dos casos, o tempo necessário para transmitir a mesma quantidade de mensagens pela rede é menor com o B-CoAP.

A partir desses resultados, conclui-se que não é viável utilizar dois protocolos para confirmação, ponto-a-ponto e fim-a-fim, pois o excesso de mensagens de controle no meio influencia para que o desempenho da rede diminua, com menor confiabilidade e maior tempo para emitir todos os pacotes. Portanto, o B-µNet possui melhor capacidade de confiabilidade com menor tempo total para entregar as informações da rede do que o C-CoAP e o B-CoAP, se o intervalo entre transmissões P for adequado para o tamanho da rede, além de possibilitar correções de erro em pontos específicos da rede com as mensagens de controle de transmissão ponto-a-ponto, evitando a propagação de erros e sobrecarga na rede.

7.6 ANÁLISE DE CONFIABILIDADE POR QUANTIDADE DE SALTOS

As análises realizadas, nas seções anteriores, têm como objetivo verificar a confiabilidade da rede como um todo, ao avaliar a quantidade de pacotes recebidos no coordenador pela quantidade total de pacotes emitidos por todos os dispositivos. Entretanto, é necessário verificar o comportamento da rede à medida que aumenta a quantidade de saltos entre cada dispositivo e o coordenador. Para realizar essa análise, estipula-se a quantidade mínima de saltos necessários para cada dispositivo enviar uma mensagem ao coordenador, pois no cenário *Gama* é possível que um dispositivo possua quantidades distintas de saltos por causa dos protocolos de roteamento. Nos cenários *Alpha* e *Beta*, por causa da sua distribuição matricial, cada dispositivo possui uma quantidade fixa de saltos necessários. Contudo, os valores de mínimos de saltos necessários foram obtidos por meio de simulações, analisando o comportamento individual das rotas de cada dispositivo. A Figura 48 apresenta a soma da quantidade de dispositivos que possuem a mesma quantidade de saltos.

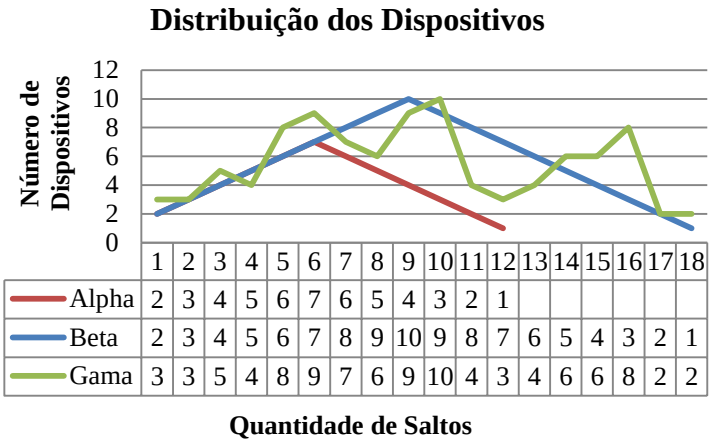


Figura 48 – Soma da quantidade de dispositivos com mesma quantidade de saltos.

No cenário *Alpha* a maior concentração de dispositivos se encontram a 6 saltos de distância do coordenador, enquanto no *Beta* a maioria está a 9 saltos. Por outro lado, o cenário *Gama* possui a maior concentração de dispositivos a 10 saltos de distância. As distribuições dos cenários *Alpha* e *Beta* adotam um padrão linear, por causa da distribuição matricial, o que não ocorre na distribuição aleatória do cenário *Gama*, com picos nas quantidades 6 e 16 saltos. Nota-se também que os cenários *Beta* e *Gama*, coincidentemente, possuem a mesma quantidade de saltos para os dispositivos mais longes, de 18 saltos. Ao realizar a média de saltos necessários pelos dispositivos de cada cenário, conclui-se que o cenário *Alpha* possui em média 6,125 saltos, o *Beta* possui 9,09 saltos e o *Gama* possui 9,16 saltos. Devido aos valores de distancia máxima e média de saltos serem equivalentes nos cenários *Beta* e *Gama*, os resultados obtidos nas análises anteriores, de confiabilidade ao variar os intervalos entre transmissão, tiveram comportamentos similares. Apesar de serem similares, o cenário *Beta* obteve melhores valores de confiabilidade, pois a distribuição dos dispositivos no cenário *Gama* contém maior quantidade de dispositivos com 15, ou mais, saltos.

De modo a analisar o comportamento da rede, conforme aumenta o número de saltos de cada dispositivo ao coordenador, foi realizada a média da confiabilidade entre os dispositivos que contém a mesma quantidade de saltos. Os resultados das Figuras 50, 51, 52 e 53 apresentam, em escala de cor, a média de confiabilidade por quantidade de salto de cada dispositivo, nos cenários *Alpha*, *Beta* e *Gama*. São avaliados ao variar o intervalo entre transmissões dos conjuntos C-UDP, B-µNet e B-µNet16, e com o intervalo entre transmissões de 1 segundo para os conjuntos C-CoAP e B-CoAP, conforme resultados demonstrados anteriormente. Cada confiabilidade é denotada na escala de cor demonstrada na Figura 49,

sendo que cores em tons avermelhados representam confiabilidades próximas de 0%, e cores em tons azuis escuros representam valores próximos de 100%.

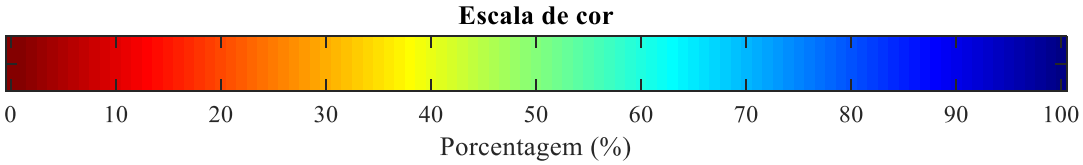


Figura 49 – Escala de cor conforme taxa de confiabilidade.
Fonte: Autoria própria.

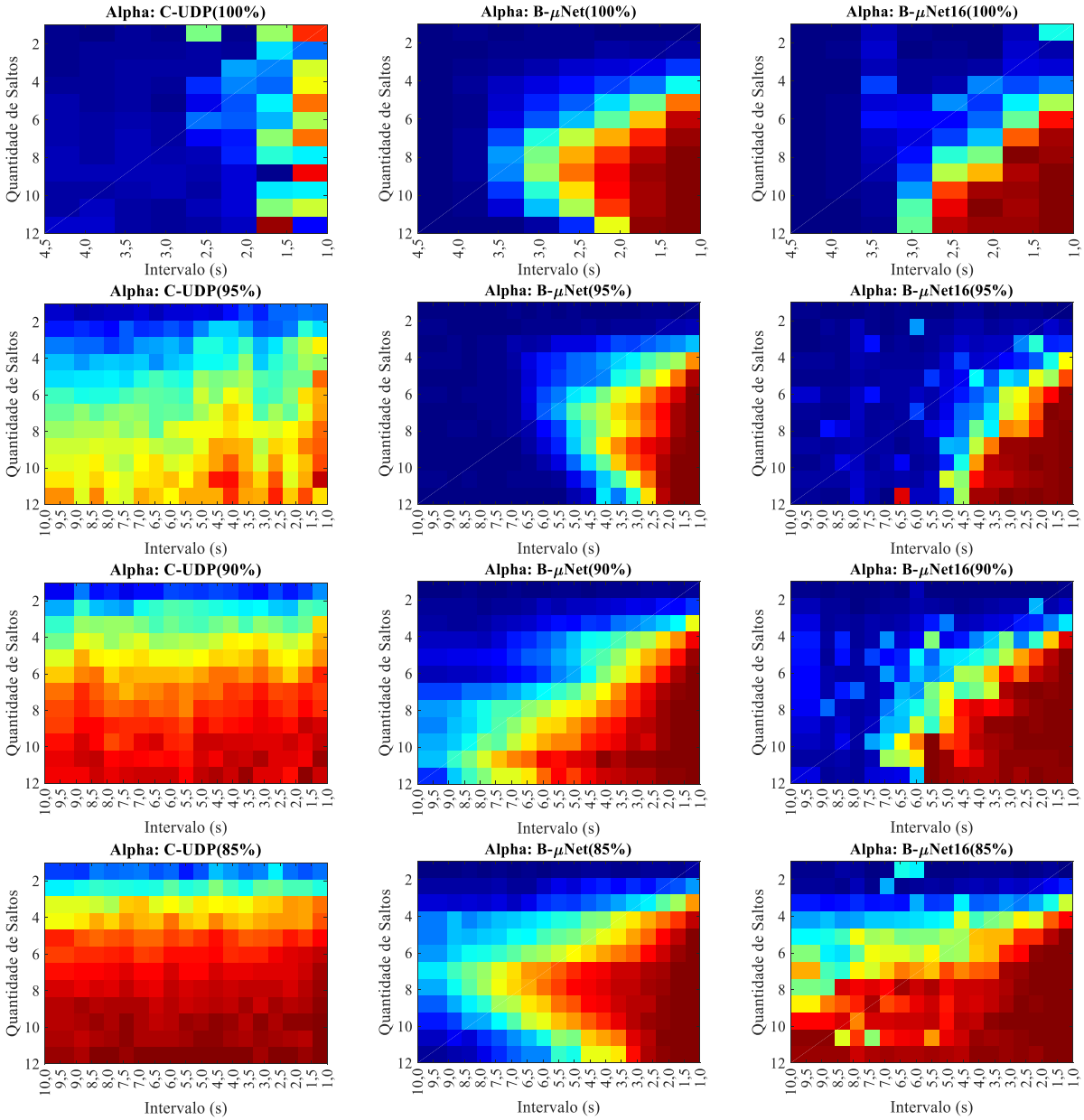


Figura 50 – Média de confiabilidade por quantidade de saltos do cenário *Alpha*. Eixo vertical representa a quantidade de saltos necessários para enviar uma mensagem ao coordenador, eixo horizontal contém a variação dos períodos entre transmissão, *P*, em segundos.
Fonte: Autoria própria.

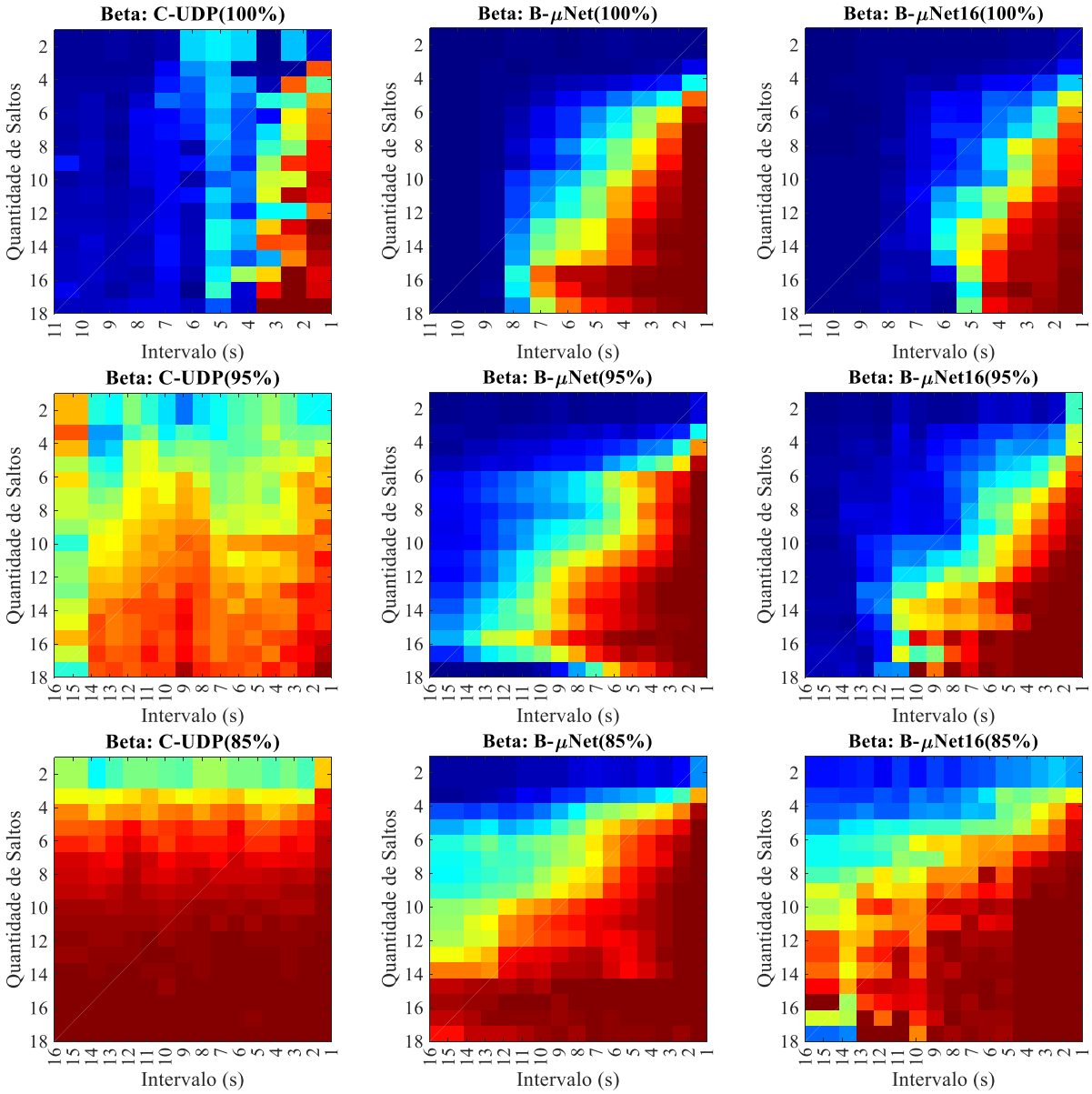


Figura 51 – Média de confiabilidade por quantidade de saltos do cenário *Beta*. Eixo vertical representa a quantidade de saltos necessários para enviar uma mensagem ao coordenador, eixo horizontal contém a variação dos períodos entre transmissão, *P*, em segundos.

Fonte: Autoria própria.

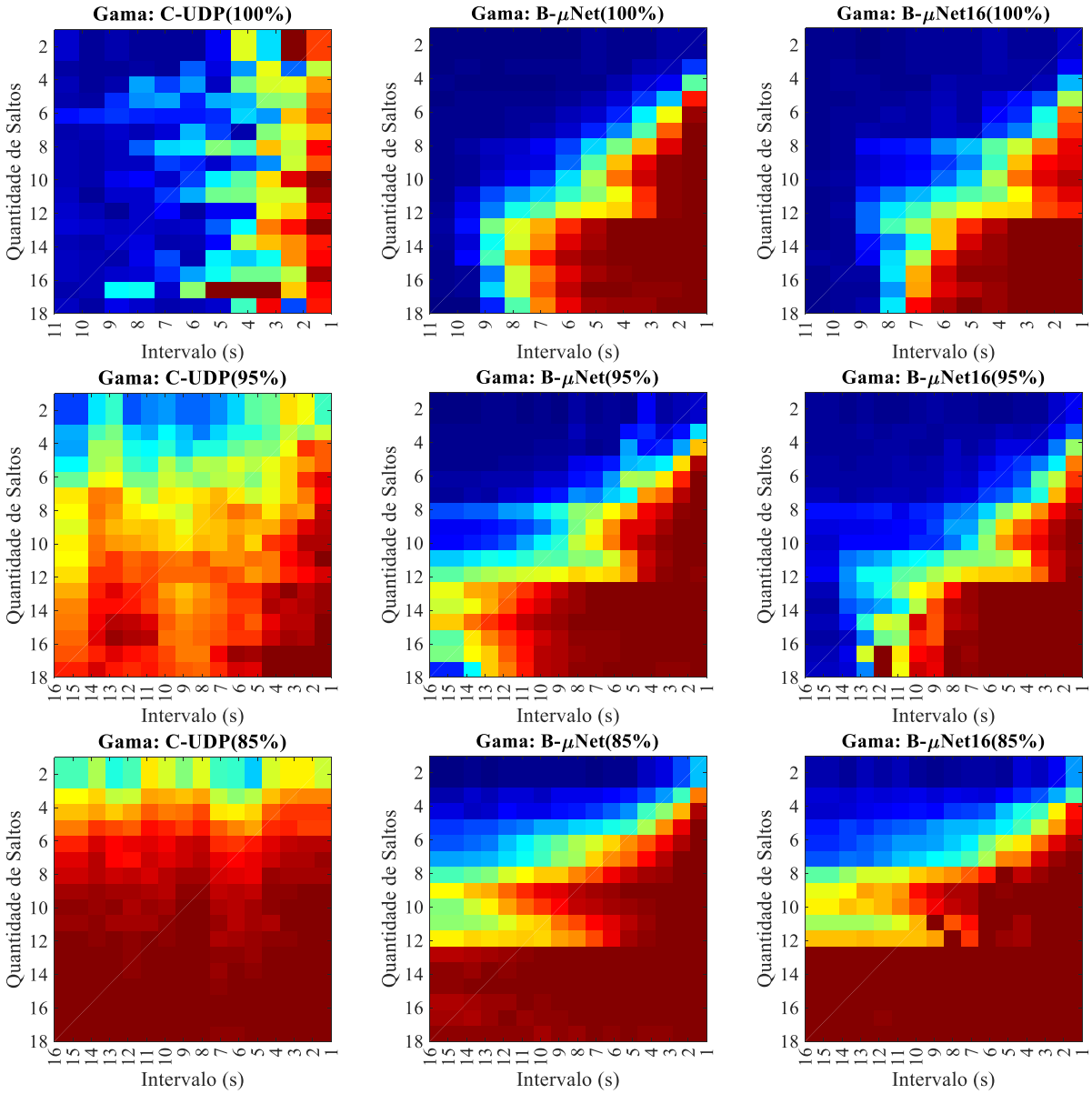


Figura 52 – Média de confiabilidade por quantidade de saltos do cenário *Gama*. Eixo vertical representa a quantidade de saltos necessários para enviar uma mensagem ao coordenador, eixo horizontal contém a variação dos períodos entre transmissão, *P*, em segundos.

Fonte: Autoria própria.

Em todos os cenários, o C-UDP apresenta elevada confiabilidade em intervalos maiores nos cenários ideais, diminuindo a confiabilidade de modo esparso, independente da quantidade de saltos. Entretanto, ao adicionar as interferências nas simulações, observa-se que todos os dispositivos têm sua confiabilidade afetada, principalmente os mais longes do coordenador, independente do intervalo utilizado entre as transmissões.

O B-µNet, por outro lado, apresenta o mesmo comportamento para todos os cenários e em qualquer nível de interferência utilizada. Quanto maior o intervalo de tempo entre as transmissões, maior a confiabilidade da rede. Entretanto, conforme se reduz o intervalo, os dispositivos que estão mais distantes começam a falhar primeiro, tornando-os inativos nos piores casos. Por outro lado, verifica-se que é possível melhorar a confiabilidade do B-µNet nos cenários com interferência, ao aumentar o intervalo de tempo entre as transmissões, comportamento que não é visto no C-UDP.

O B-µNet16 apresenta o mesmo comportamento do B-µNet, com a diferença de que a confiabilidade é distribuída entre todos os dispositivos da rede. Contudo, assim como o B-µNet, ao diminuir os intervalos, os dispositivos que estão mais longes se tornam inativos.

Nota-se que no cenário *Gama* há uma demarcação nos dispositivos com mais de 12 saltos, pois possuem menor confiabilidade do que o cenário *Beta*. Esse comportamento ocorre pela distribuição dos dispositivos na rede, pois há uma concentração maior de dispositivos com 12 saltos ou mais no cenário *Gama* do que no *Beta*.

Além do mais, verifica-se que as taxas de interferência têm maior impacto nos dispositivos mais distantes, pois o erro é cumulativo a cada salto, pelo método de simulação de interferência adotado. No cenário *Alpha*, por exemplo, o nodo de número 1 é o mais afetado, pois as informações precisam passar por 12 saltos para chegar ao seu destino. Com isso, num ambiente em que a taxa de sucesso de transmissão ponto-a-ponto seja de 91%, a interferência total (fim-a-fim) para esse dispositivo é de aproximadamente 32,25%, calculado pelo produtório das taxas de sucesso de transmissão ponto-a-ponto. O mesmo pode ser esperado nos cenários *Beta* e *Gama*, que com 18 saltos a probabilidade de uma informação ser entregue ao seu destino é de aproximadamente 18,31%.

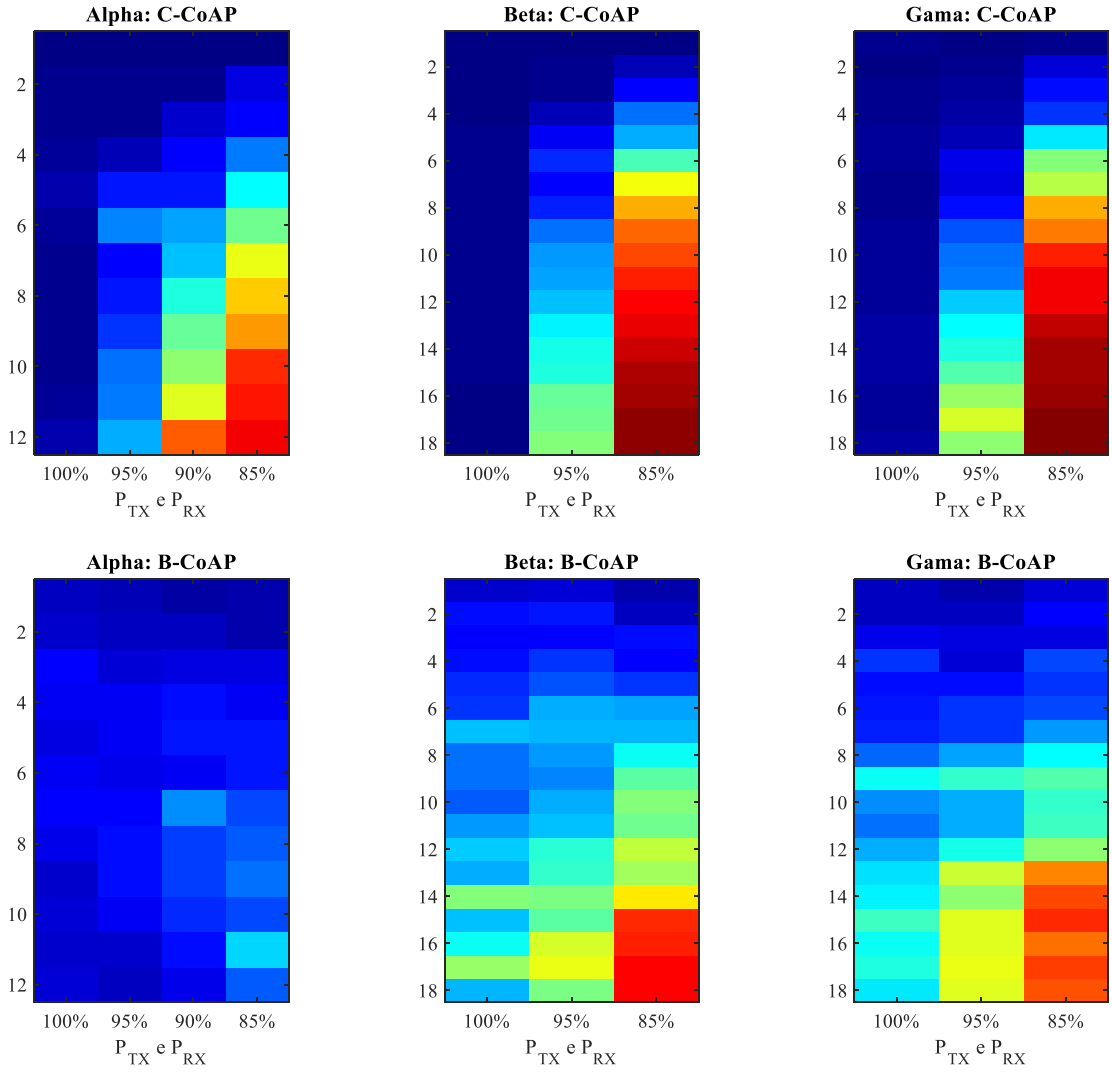


Figura 53 – Média de confiabilidade por quantidade de saltos dos conjuntos C-CoAP e B-CoAP. Eixo vertical representa a quantidade de saltos necessários para enviar uma mensagem ao coordenador, eixo horizontal contém as probabilidades de sucesso para enviar uma mensagem ponto-a-ponto, P_{TX} e P_{RX}.
Fonte: Autoria própria.

A partir de observações do conjunto C-CoAP, quanto maior a interferência no meio, menor a confiabilidade dos dispositivos mais distantes, com similaridades com o comportamento do B-µNet. Por outro lado, o B-CoAP consegue distribuir melhor o erro da rede, de modo que os dispositivos mais afastados não fiquem inoperantes por conta da interferência. Apesar do B-CoAP apresentar menor confiabilidade geral da rede, nos cenários ideais, ele fornece maior confiabilidade geral e distribuída da rede ao inserir fontes de interferência.

8 CONCLUSÃO

Rede de Sensores Sem Fio é uma tecnologia capaz de auxiliar nos campos civil, hospitalar, militar, industrial e ambiental. Aplicações de sensoriamento e atuação no meio são estudadas e aplicadas de forma que seja possível melhorar a qualidade de vida, reduzir custos, maximizar lucros, entre outras formas de beneficiar e facilitar a vida das pessoas. Entretanto, tal tecnologia ainda enfrenta sérios problemas como: recursos limitados nos dispositivos utilizados (memória e processamento), eficiência energética, segurança, interferência do meio de comunicação e falhas de transmissão de dados. Apesar das técnicas utilizadas na literatura para melhorar a comunicação entre dispositivos na rede, análises prévias em ambientes controlados demonstram que, mesmo com o avanço nos métodos empregados, há falhas que devem ser corrigidas.

Neste trabalho, foi proposto um novo protocolo, denominado µNet, para realizar o controle do fluxo de transmissões de dados em RSSFs. Tal protocolo tem como objetivo melhorar a confiabilidade fim-a-fim em transmissão de pacotes na rede, ao utilizar um método adicional de confirmação. Pela falta de ferramentas específicas e padronizadas para análise de confiabilidade e desempenho de RSSF, realizou-se comparações com protocolos e métodos de transmissão destinados a RSSFs utilizados no meio cientifico, como o CoAP, UDP, IPv6, RPL, 6LoWPAN e IEEE 802.15.4, implementados no sistema operacional Contiki OS. Para simplificar a escrita, foram definidos nomes para os conjuntos de protocolos analisados, conforme descrito na seção 6.2.3. Por utilizar o Contiki OS, foi utilizada a ferramenta de simulação Cooja para RSSFs, a qual fornece ferramentas essenciais para controle e monitoramento da rede de comunicação.

Ao utilizar a ferramenta de simulação Cooja, foi possível verificar o desempenho de ambas as soluções, assim como obter os tempos mínimos para realizar a comunicação entre dois dispositivos vizinhos. Verificou-se que o tamanho de cabeçalho final é menor com o µNet do que com os outros protocolos, devido a estrutura ser destinada especificamente à RSSF. Por ser aproximadamente 10% menor, torna-se melhor para ser utilizado com o protocolo IEEE 802.15.4, que possibilita a transmissão de no máximo 127 *bytes* por pacote. Além disso, conforme esperado, o protocolo µNet necessita de mais tempo para enviar, receber e rotear um pacote do que os protocolos utilizados no Contiki OS, pois é necessário utilizar o meio de comunicação para enviar mais dois pacotes de controle de transmissão. De

modo a totalizar num valor, aproximado, de envio ponto-a-ponto mínimo de 6,8 ms para o µNet e 5,59 ms para o Contiki.

Para analisar a confiabilidade de cada conjunto de protocolo utilizado, contabilizou-se a quantidade de pacotes transmitidos com sucesso ao coordenador, em cenários simulados sem e com interferência no meio de comunicação. Foram utilizados cenários diversificados em quantidade de dispositivos, distribuição espacial e quantidade de interferência simulada. Além disso, os conjuntos de protocolos analisados foram testados sem e com adição de processos de confiabilidade fim-a-fim, provido pelo protocolo CoAP.

Primeiramente, verificou-se a confiabilidade do conjunto C-UDP ao utilizar as funções OF0 e MRHOF do RPL. Observou-se que, apesar de utilizar o mesmo conjunto de protocolos, ambas as métricas de roteamento obtiveram resultados significativamente diferentes. O C-UDP com OF0 obteve taxas de confiabilidade elevadas em cenários ideais, reduzindo à medida que o intervalo de transmissão reduzia. Por outro lado, não foi possível obter taxas de confiabilidade elevadas com o MRHOF, ao utilizar os mesmos parâmetros. Notou-se também que, ao simular a interferência no meio, ambos os conjuntos de protocolos tiveram suas taxas de confiabilidade afetadas significativamente, sendo que a confiabilidade da função OF0 permanecer melhor do que a da MRHOF. Observou-se que a baixa confiabilidade do MRHOF foi causada pela quantidade de troca de rotas durante a simulação. Entre essas trocas de rotas, observou-se que, em certos instantes, as rotas eram perdidas, ou seja, não havia rota que chegasse ao coordenador.

Ao analisar o conjunto B-µNet, notou-se um comportamento semelhante ao do C-UDP em cenários ideais, sendo que o C-UDP manteve sua confiabilidade maior à medida que o intervalo de transmissão diminuía, pois o B-µNet necessita de mais tempo para realizar suas transmissões ponto-a-ponto. Por outro lado, nos cenários com interferência de simulação, é possível notar a discrepância de confiabilidade entre os métodos, sendo que o em alguns casos B-µNet conseguiu entregar todas suas informações, enquanto o C-UDP teve sua confiabilidade afetada significativamente. Portanto, o verifica-se que o B-µNet é um conjunto de protocolos que consegue obter maior confiabilidade do que o C-UDP em cenários com interferência.

Por ser destinado à RSSFs com dispositivos de baixa capacidade, o protocolo µNet utiliza por padrão um *buffer* de rede de tamanho unitário. Portanto, com intuito de averiguar o impacto de utilizar o *buffer* unitário, comparou-o com *buffer* de capacidade de armazenamento 16, mesmo valor adotado pelo Contiki OS. Conforme os resultados demonstrados na seção 7.4.1, observou-se que o incremento do *buffer* ocasionou maior

confiabilidade nos cenários com menor interferência. Em cenários em que a interferência é elevada, o tamanho do *buffer* não foi fator para melhorar a confiabilidade. Isso demonstra que o processo de confirmação de entrega apresenta maior impacto na confiabilidade do que somente ajustar o tamanho do *buffer* da rede, conforme demonstrado no Capítulo 4. Portanto, apesar do tamanho do *buffer* poder incrementar a confiabilidade em certos cenários, o processo de confirmação ponto-a-ponto consegue manter a confiabilidade elevada mesmo com um *buffer* unitário, o que é significante em redes de dispositivos com recursos limitados.

Em sequência, foi realizada análises dos conjuntos de protocolos com a adição do CoAP, de modo a adicionar métodos de confirmação de entrega fim-a-fim na rede. O conjunto C-CoAP apresentou uma melhora significativa em sua confiabilidade em cenários com interferência, ao custo de um tempo de transmissão muito maior se comparado ao C-UDP. Por outro lado, o B-CoAP teve sua confiabilidade reduzida e seu tempo de transmissão aumentado significativamente. Nota-se que, ao utilizar dois modos de confirmação, a rede fica sobrecarregada com métodos de confirmação, reduzindo a taxa efetiva de pacotes enviados e recebidos com sucesso. Nos cenários simulados, um erro que ocorre em um dos dispositivos é propagado pela rede inteira, pois os intervalos entre transmissões utilizados não possibilitam que os dispositivos tenham tempo suficiente para recuperação de erros. Tal fator é agravado ainda mais quando as mensagens necessitam ser transmitidas e confirmadas por métodos fim-a-fim, reduzindo a efetividade de comunicação da rede. Além disso, se um dispositivo tem sua rota alterada, todos os dispositivos que dependem dela tem suas comunicações afetadas enquanto a rede não se adequa à nova rota. Portanto, ao comparar o C-CoAP, B-CoAP e B-µNet, nota-se que o B-µNet possui maior confiabilidade, se for disponibilizado o intervalo entre transmissões adequado para o cenário que está inserido, pois realiza tratativa de erros de forma localizada, evitando-se a propagação do erro.

Por fim, analisa-se a confiabilidade média por quantidade de saltos para cada conjunto de protocolos proposto e seus respectivos cenários. Nota-se que nos cenários ideais, o C-UDP distribui a confiabilidade em todos os saltos da rede, enquanto o B-µNet, com *buffer* unitário ou maior, tem seus dispositivos mais distantes afetados primeiro. Desse modo, os dispositivos que estão mais distantes, não conseguem enviar suas mensagens ao coordenador da rede, conforme diminui o intervalo entre transmissões. Essa característica não aparenta ter relação com o *buffer*, visto que o comportamento do B-µNet não se altera em relação ao tamanho do *buffer*. Por outro lado, à medida que se insere interferência na rede, o C-UDP e o C-CoAP apresentam o mesmo comportamento do B-µNet, independente do intervalo entre

transmissões. Contudo, nota-se que o conjunto B-CoAP distribui a taxa de confiabilidade na rede, de modo a evitar que os dispositivos mais distantes fiquem sem comunicação. Portanto, apesar de reduzir a taxa de confiabilidade da rede geral e aumentar o tempo necessário de transmissão, nota-se que pode ser uma medida eficiente para distribuir o erro da rede, de forma que os dispositivos mais distantes não percam sua comunicação.

A partir dos resultados obtidos, e das análises realizadas, conclui-se que o protocolo µNet obteve êxito em sua função de aumentar a confiabilidade na entrega de pacotes em RSSFs afetadas por fontes de interferências. Quanto maior a interferência no meio, mais discrepante a diferença de confiabilidade dos conjuntos de protocolos abordados. Se disponibilizado intervalos entre transmissões maiores, é possível obter com o µNet uma confiabilidade maior de entrega fim-a-fim, numa rede com meio de comunicação com interferência. Tal comportamento demonstra a viabilidade da solução proposta para sistemas que necessitam de garantia de entrega de dados. Por seu cabeçalho ser menor, em comparação ao utilizado pelo C-UDP e por ser possível utilizar *buffer* de rede unitário, o protocolo µNet é adequado para dispositivos de baixa capacidade de memória, característica essencial para RSSFs. Portanto, esse protocolo é eficaz para gerenciar as comunicações em aplicações implantadas em *smart cities*, que contém elevados níveis de interferência no meio de comunicação.

Como sugestão de trabalhos futuros, que poderão aprimorar e/ou complementar o trabalho realizado, cita-se:

- Realizar análises em ambientes reais, de modo a verificar o desempenho real das técnicas adotadas em ambientes com interferências e suas aplicabilidades para *smart cities*.
- Realizar análises de consumo energético com o protocolo µNet, comparando-o com soluções adotadas nesse tipo rede e viabilizá-lo para cenários em que a restrição energética seja o problema principal.
- Definir métodos para configurar os intervalos entre comunicações de modo dinâmico e adaptável a qualquer cenário, com níveis distintos de interferência e de distribuição espacial de dispositivos. Com as informações disponibilizadas no coordenador é possível definir o tamanho da rede e a distância de cada dispositivo, com elas é possível notificar a rede de sensoriamento qual seria um intervalo adequado entre transmissões de mensagens. Além disso, ao verificar a quantidade de retransmissões ponto-a-ponto necessárias para cada dispositivo, é

possível detectar possíveis focos de interferência ou sobrecarga na rede, e ajustá-la para se adaptar a tais problemas.

- Análises de tamanho mínimo de *buffer* necessário para redes que utilizem métodos de confirmação ponto-a-ponto, para maximizar o desempenho de comunicação com o menor custo de memória.
- Modificar a ordem das mensagens de controle do protocolo µNet, de modo a requisitar o espaço do *buffer* do dispositivo vizinhos antes de realizar a transmissão do pacote de dados. Assim, é possível evitar o modelo de tentativa e erro utilizado pelo µNet, possibilitando reservar antes o espaço para armazenar a informação e, somente quando o dispositivo estiver disponível, informar aos seus vizinhos que pode receber uma informação. Com esse método, deve-se verificar se há vantagens em confiabilidade e desempenho de comunicação em relação ao µNet utilizado nesse trabalho.
- Adaptar os parâmetros de tempo e retransmissão do protocolo CoAP para redes que utilizem métodos de confirmação ponto-a-ponto. Como o CoAP foi desenvolvido com objetivo de ser utilizado com o protocolo UDP, seus intervalos entre retransmissões podem ser baixos para protocolos que necessitem de mais tempo para realizar as transmissões ponto-a-ponto, ou mesmo redes que possuem dispositivos com muitos saltos. Portanto, adequar o protocolo CoAP para protocolos que utilizem confirmação ponto-a-ponto pode contribuir na distribuição do erro de confiabilidade da rede de forma a evitar desconexão total dos dispositivos mais distantes quando a rede fica instável.

REFERÊNCIAS

AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES. **Resolução nº 680, de 27 de junho de 2017.** 2017. Disponível em: <http://www.anatel.gov.br/legislacao/resolucoes/2017/936-resolucao-680>. Acesso em: 5 jun. 2018.

AL-ANBAGI, I.; EROL-KANTARCI, M.; MOUFTAH, H. T. Tuning Guaranteed Time Slots of IEEE 802.15.4 for Transformer Health Monitoring in the Smart Grid. **IEEE Wireless Communications and Networking Conference (WCNC)**, v. 4, p. 3420–3425, 2014.

AL-ANBAGI, I.; KHANAFER, M.; MOUFTAH, H. T. MAC finite buffer impact on the performance of cluster-tree based WSNs. **IEEE International Conference on Communications**, p. 1485–1490, 2013.

ANCILLOTTI, E.; BRUNO, R.; CONTI, M. Reliable Data Delivery With the IETF Routing Protocol for Low-Power and Lossy Networks. **IEEE Transactions on Industrial Informatics**, v. 10, n. 3, p. 1864–1877, 2014.

ARAGO SYSTEMS. **WiS Mote: Module capteurs/actionneurs sans-fil basse consommation pour applications WSN.** 2011. Disponível em: <http://www.arago-systems.com/images/stories/WiSMote/Doc/wismote.pdf>. Acesso em: 5 jun. 2018.

BARON, A.; GINOSAR, R.; KESLASSY, I. The Capacity Allocation Paradox. **Proceedings - IEEE INFOCOM**, p. 1359–1367, 2009.

BARRIQUELLO, C. H.; DENARDIN, G. W.; GODOI, F. N. DE. **BRTOS uNet.** Disponível em: <https://github.com/brtos/brtos-unet>. Acesso em: 5 jun. 2018.

BOVA, T.; KRIVORUCHKA, T. **Reliable UDP protocol.** 1999. Disponível em: <https://tools.ietf.org/html/draft-ietf-sigtran-reliable-udp-00>. Acesso em: 5 jun. 2018.

BUI, L.; ERYILMAZ, A.; SRIKANT, R.; et al. Asynchronous congestion control in multi-hop wireless networks with maximal matching-based scheduling. **IEEE/ACM Transactions on Networking**, v. 16, n. 4, p. 826–839, 2008.

CHENG, B.; ZHAO, S.; WANG, S.; et al. Lightweight Mashup Middleware for Coal Mine

Safety Monitoring and Control Automation. **IEEE Transactions on Automation Science and Engineering**, v. 14, n. 2, p. 1245–1255, 2016.

CONTIKI. **Contiki.** 2018a. Disponível em: <http://www.contiki-os.org/>. Acesso em: 5 jun. 2018.

_____. **Contiki Hardware.** 2018b. Disponível em: <http://www.contiki-os.org/hardware.html>. Acesso em: 5 jun. 2018.

CONTIKI; ÖSTERLIND, F. **Using Cooja Test Scripts to Automate Simulations.** 2014. Disponível em: <https://github.com/contiki-os/contiki/wiki/Using-Cooja-Test-Scripts-to-Automate-Simulations>. Acesso em: 5 jun. 2018.

COVER, T. M.; THOMAS, J. A. **Elements of Information Theory**. New York: Wiley, 1991.

DAELY, P. T.; REDA, H. T.; SATRYA, G. B.; et al. Design of Smart LED Streetlight System for Smart City With Web-Based Management System. **IEEE Sensors Journal**, v. 17, n. 18, p. 6100–6110, 2017.

DAGHER, R.; MITTON, N.; AMADOU, I. Towards WSN-Aided navigation for vehicles in smart cities: An application case study. **2014 IEEE International Conference on Pervasive Computing and Communication Workshops, PERCOM WORKSHOPS 2014**, p. 129–134, 2014.

DEERING, S.; HINDEN, R. Internet Protocol, Version 6 (IPv6) Specification Status. **Internet Engineering Task Force**, p. 1–39, 1998.

DENARDIN, G. W. **O que é o BRTOS?** 2010. Disponível em: <https://brtosblog.wordpress.com/2010/10/06/o-que-e-o-brtos/>. Acesso em: 5 jun. 2018.

DENARDIN, G. W.; BARRIQUELLO, C. H. **BRTOS: Brazilian Real-Time Operating System.** 2017. Disponível em: <https://github.com/brtos>. Acesso em: 5 jun. 2018.

DUNKELS, A.; ÖSTERLIND, F.; SCHMIDT, O. **Instant Contiki 3.0.** 2015. Disponível em: <https://sourceforge.net/projects/contiki/files/Instant Contiki/>. Acesso em: 5 jun. 2018.

FAHMY, H. M. A. **Wireless Sensor Networks: Concepts, Applications, Experimentation and Analysis**. 1 ed. Springer Singapore, 2016.

FINNE, N.; ERIKSSON, J.; DUNKELS, A.; et al. **Official MSPSim git repository.** 2018. Disponível em: <https://github.com/contiki-ng/mspsim>. Acesso em: 5 jun. 2018.

FOGLIATTI, M. C.; MATTOS, N. M. C. **Teoria de Filas**. Rio de Janeiro: Interciência Ltda., 2007.

GADDOUR, O.; KOUBÂA, A. RPL in a nutshell: A survey. **Computer Networks (Comput. Netw.), Elsevier**, v. 56, n. 14, p. 3163–3178, 2012.

GANERIWAL, S.; KUMAR, R.; SRIVASTAVA, M. B. Timing-Sync Protocol for Sensor Networks. **Proc. 1st International Conference on Embedded Networked Sensor System (SenSys)**, v. 47, n. 6, p. 34–40, 2003.

GARDASEVIC, G.; MIJOVIC, S.; STAJKIC, A.; et al. On the performance of 6LoWPAN through experimentation. **IWCMC 2015 - 11th International Wireless Communications and Mobile Computing Conference**, p. 696–701, 2015.

GHARAIBEH, A.; SALAHUDDIN, M. A.; HUSSINI, S. J.; et al. Smart Cities: A Survey on Data Management, Security and Enabling Technologies. **IEEE Communications Surveys & Tutorials**, v. X, n. X, p. 1–55, 2017.

GHAYVAT, H.; MUKHOPADHYAY, S.; GUI, X.; et al. WSN- and IOT-based smart homes and their extension to smart buildings. **Sensors**, v. 15, n. 5, p. 10350–10379, 2015.

GIACCONE, P.; LEONARDI, E.; SHAH, D. Throughput region of finite-buffered networks. **IEEE Transactions on Parallel and Distributed Systems**, v. 18, n. 2, p. 251–263, 2007.

GNAWALI, O.; LEVIS, P. **The Minimum Rank with Hysteresis Objective Function The.** 2012. Disponível em: <https://tools.ietf.org/pdf/rfc6719.pdf>. Acesso em: 5 jun. 2018.

GODOI, F. N. DE; DENARDIN, G. W.; BARRIQUELLO, C. H. **uNet for Cooja.** 2017. Disponível em: <https://www.github.com/fabricio-godoi/unet4cooja>. Acesso em: 5 jun. 2018.

GRUMAZESCU, C.; VL, V.; SUBA, G. WSN Solutions for Communication Challenges in Military Live Simulation Environments. **International Conference on Communications (COMM)**, p. 319–322, 2016.

GUNGOR, V. C.; HANCKE, G. P. Industrial Wireless Sensor Networks : Challenges , Design Principles , and Technical Approaches. **IEEE Transactions on Industrial Electronics**, v. 56, n. 10, p. 4258–4265, 2009.

HOSNI, I.; HAMDI, N. Cross layer optimization of end to end delay in WSN for smart grid communications. **2016 International Symposium on Signal, Image, Video and Communications, ISIVC 2016**, p. 217–223, 2016.

HUI, J.; THUBERT, P. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. , p. 1–24, 2011.

HUSSAIN, A. M.; KHAN, P.; SUP, K. K. WSN Research Activities for Military Application. **Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on**, p. 271–274, 2009.

IEEE COMPUTER SOCIETY. **8802-2-1994 - ISO/IEC/IEEE International Standard - Information Technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 2: Logical link control**. The Institute of Electrical and Electronics Engineers, Inc., 1994.

_____. IEEE Standard for Low-Rate Wireless Networks. New York: The Institute of Electrical and Electronics Engineers, Inc. 2015.

JASMIN, U.; VELAYUTHAM, R. Enhancing the Security in Signature Verification for WSN with Cryptographic Algorithm. **International Conference on Circuits, Power and Computing Technologies [ICCPCT-2014]**, p. 1584–1588, 2014.

JAVVIN TECHNOLOGIES. **Network Protocols Handbook**. 2ª ed. Saratoga CA, 2005.

JOO, C.; SHROFF, N. B. Performance of random access scheduling schemes in multi-hop wireless networks. **IEEE/ACM Transactions on Networking**, v. 17, n. 5, p. 1481–1493, 2009.

KAR, K.; LUO, X.; SARKAR, S. Throughput-optimal scheduling in multichannel access point networks under infrequent channel measurements. **IEEE Transactions on Wireless Communications**, v. 7, n. 7, p. 2619–2629, 2008.

KEEN, H. **IEEE 802.2 Logical Link Control (LLC).** 2011. Disponível em: <http://www.iee

e802.org/2/>. Acesso em: 5 jun. 2018.

KENDALL, D. G. Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chains. **Ann. Math. Statist.**, v. 24, p. 338–354, 1953.

KIM, H.-S.; IM, H.; LEE, M.-S.; et al. A Measurement Study of TCP over RPL in Low-power and Lossy Networks. **Journal of Communications and Networks**, v. 17, n. 6, p. 1, 2015.

KLEINROCK, L. **Queueing Systems**. New York: John Wiley & Sons, Inc., 1975.

KLEINROCK, L.; TOBAGI, F. Packet Switching in Radio Channels: Part I--Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics. **IEEE Transactions on Communications**, v. 23, n. 12, p. 1400–1416, 1975.

KNOBLOCH, F.; BRAUNSCHWEIG, N. A Traffic-Aware Moving Light System Featuring Optimal Energy Efficiency. **IEEE Sensors Journal**, v. 17, n. 23, p. 7731–7740, 2017.

LAVRIC, A.; POPA, V.; SFICHI, S. Street Lighting Control System Based On Large- Scale WSN : A Step Towards A Smart City. **International Conference and Exposition on Electrical and Power Engineering**, p. 16–18, 2014.

LE, L. B.; MODIANO, E.; SHROFF, N. B. Optimal Control of Wireless Networkds With Finite Buffers. **IEEE/ACMTRANSACTIONS ON NETWORKING**, v. 20, n. 4, p. 1316–1329, 2012.

LIN, X.; SHROFF, N. B. The impact of imperfect scheduling on cross-layer congestion control in wireless networks. **IEEE/ACM Transactions on Networking**, v. 14, n. 2, p. 302–315, 2006.

LODHI, M. A.; REHMAN, A.; KHAN, M. M.; et al. Multiple path RPL for low power lossy networks. **APWiMob 2015 - IEEE Asia Pacific Conference on Wireless and Mobile**, p. 279–284, 2016.

LU, X.; WANG, S.; LI, W.; et al. Development of a WSN based real time energy monitoring platform for industrial applications. **IEEE 19th International Conference on Computer Supported Cooperative Work in Design (CSCWD)**, p. 337–342, 2015.

MAHOOR, M.; SALMASI, F. R.; NAJAFABADI, T. A. A hierarchical smart street lighting system with brute-force energy optimization. **IEEE Sensors Journal**, v. 17, n. 9, p. 2871–2879, 2017.

MASIRAP, M.; AMARAN, M. H.; YUSSOFF, Y. M.; et al. Evaluation of Reliable UDP-Based Transport Protocols for Internet of Things (IoT). **IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)**, p. 200–205, 2016.

MICHAELIS. **Dicionário Brasileiro da Língua Portuguesa: buffer.** 2017. Disponível em: <http://michaelis.uol.com.br/busca?r=0&f=&t=&palavra=buffer>. Acesso em: 5 jun. 2018.

MORA, O. Reliable Transport Layer Protocol in Low Performance 8-bit Microcontrollers. US 2005/0129064 A1 2005.

MUDUMBE, M. J.; ABU-MAHFOUZ, A. M. Smart water meter system for user-centric consumption measurement. **Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN 2015**, p. 993–998, 2015.

MUNIR, A.; ANTOON, J.; GORDON-ROSS, A. Modeling and Analysis of Fault Detection and Fault Tolerance in Wireless Sensor Networks. **ACM Transactions on Embedded Computing Systems**, v. 14, n. 1, p. 1–43, 2015.

NANDURY, S. V; BEGUM, B. A. Strategies to Handle Big Data for Traffic Management in Smart Cities. **International Conference on Advances in Computing, Communications and Informatics (ICACCI)**, p. 356–364, 2016.

NHAT-QUANG NHAN; MINH-THANH VO; TUAN-DUC NGUYEN; et al. Improving the performance of mobile data collecting systems for electricity meter reading using Wireless Sensor Network. **The 2012 International Conference on Advanced Technologies for Communications**, , n. Atc, p. 241–246, 2012.

OLSSON, J. 6LoWPAN demystified. **Texas Instruments**, p. 13, 2014.

OMONDI, F. A. **Modelling and Performability Evaluation of Wireless Sensor Networks**. Middlesex University, 2015.

OMONDI, F. A.; SHAH, P.; GEMIKONAKLI, O.; et al. An Analytical Model for Bounded WSNs with Unreliable Cluster Heads and Links. **40th Annual IEEE Conference on Local**

Computer Networks, p. 201–204, 2015.

OREKU, G. S. Reliability in WSN for security: Mathematical approach. **International Conference on Computer Applications Technology (ICCAT)**, p. 1–6, 2013.

ÖSTERLIND, F. A sensor network simulator for the Contiki OS. **SICS Research Report** 2006.

PENG, L.; WANG, F.; WANG, C. An Improved Algorithm of Node Credibility Management for Lightweight WSN. **International Conference on Networking and Network Applications (NaNA)**, p. 99–102, 2016.

POSTEL, J. **User Datagram Protocol**. Sterling, VA: IETF, 1980.

_____. **Transmission Control Protocol**. Sterling, VA: IETF, 1981.

PUVANESHWARI S; VIJAYASHAARATHI S. Efficient Monitoring system for cardiac patients using Wireless Sensor Networks (WSN). **2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)**, p. 1558–1561, 2016.

RAMAKRISHNAN, R.; GAUR, L. Smart electricity distribution in residential areas: Internet of Things (IoT) based advanced metering infrastructure and cloud analytics. **2016 International Conference on Internet of Things and Applications (IOTA)**, p. 46–51, 2016.

ROSENDAL, F.; ELSTS, A. **UDGM.** 2015. Disponível em: <https://github.com/contiki-os/contiki/blob/master/tools/cooja/java/org/contikios/cooja/radiomediums/UDGM.java>. Acesso em: 5 jun. 2018.

ROUSSEL, K.; SONG, Y.-Q.; ZENDRA, O. Using Cooja for WSN Simulations: Some New Uses and Limits. **EWSN 2016 — NextMote workshop**, p. 319–324, 2016.

SHAH, J.; MISHRA, B. IoT enabled Environmental Monitoring System for Smart Cities. **International Conference on Internet of Things and Applications (IOTA)**, p. 383–388, 2016.

SHELBY, Z.; HARTKE, K.; BORMANN, C. The Constrained Application Protocol (CoAP).

IETF 2014.

SICHITIU, M. L.; VEERARITTIPHAN, C. Simple, accurate time synchronization for wireless sensor networks. **2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.**, v. 2, n. C, p. 1266–1273, 2003.

SOARES, L. F. G.; LEMOS, G.; COLCHER, S. **Redes de Computadores: Das LANs, MANs e WANs às Redes ATM**. 2ª ed. Rio de Janeiro: Elsevier Editora Ltda., 1995.

SOHRABY, K.; MINOLI, D.; ZNATI, T. **Wireless Sensor Networks: Technology, Protocols, and Applications**. New Jersey: John Wiley & Sons, Inc., 2007.

STOJMENOVIĆ, I. **Handbook of Sensor Networks: Algorithms and Architectures**. New Jersey: John Wiley & Sons, Inc., 2005.

SUDARSONO, A.; KRISTALINA, P.; HARUN, M. U.; et al. An Implementation of Secure Data Sensor Transmission in Wireless Sensor Network for Monitoring Environmental Health. **2015 International Conference on Computer, Control, Informatics and Its Applications An**, p. 93–98, 2015.

TASSIULAS, L.; EPHREMIDES, A. Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks. **IEEE Transactions on Automatic Control**, v. 37, n. 12, p. 1936–1948, 1992.

TEXAS INSTRUMENTS. **CC2520 Datasheet: 2.4 GHZ IEEE 802.15.4/ZIGBEE® RF TRANSCEIVER.** 2007. Disponível em: <http://www.ti.com/lit/ds/symlink/cc2520.pdf>. Acesso em: 5 jun. 2018.

_____. **MSP430F543x and MSP430F541x Mixed-Signal Microcontrollers.** 2014. Disponível em: <http://www.ti.com/lit/ds/slas612e/slas612e.pdf>. Acesso em: 5 jun. 2018.

_____. **CC2650 CC2650 SimpleLink™ Multistandard Wireless MCU.** 2016a. Disponível em: <http://www.ti.com/lit/ds/symlink/cc2650.pdf>. Acesso em: 5 jun. 2018.

_____. **MSP430x5xx and MSP430x6xx Family: User's Guide.** 2016b. Disponível em: <http://www.ti.com/lit/ug/slau208q/slau208q.pdf>. Acesso em: 5 jun. 2018.

THUBERT, P. **Objective Function Zero for the Routing Protocol for Low-Power and**

Lossy Networks (RPL). 2012. Disponível em: <https://tools.ietf.org/html/rfc6552>. Acesso em: 5 jun. 2018.

TOH, S.; LEE, S.; CHUNG, W. WSN Based Personal Mobile Physiological Monitoring and Management System for Chronic Disease. **Third 2008 International Conference on Convergence and Hybrid Information Technology**, p. 467–472, 2008.

TORII, YOSHITAKA; OTSUKA, T.; ITO, T. A diversity sensor connection capability WSN for disaster information gathering system. **IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)**, p. 1–6, 2016.

TORRES, G. **Redes de Computadores Curso Completo**. Rio de Janeiro: Axcel Books do Brasil Editora Ltda, 2001.

ULLAH, R.; FAHEEM, Y.; KIM, B. S. Energy and Congestion-Aware Routing Metric for Smart Grid AMI Networks in Smart City. **IEEE Access**, p. 13799–13810, 2017.

UNAWONG, S.; MIYAMOTO, S.; MORINAGA, N. Techniques to Improve the Performance of Wireless LAN under ISM Interference Environments. **Fifth Asia-Pacific Conference on ... and Fourth Optoelectronics and Communications Conference on Communications**, v. 1, p. 802–805, 1999.

VASSEUR, J.; KIM, M.; PISTER, K.; et al. RFC 6551 - Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks. **Internet Engineering Task Force**, p. 1–30, 2012.

VILGELM, M.; GURSU, M.; ZOPPI, S.; et al. Time Slotted Channel Hopping for smart metering: Measurements and analysis of medium access. **2016 IEEE International Conference on Smart Grid Communications, SmartGridComm 2016**, , n. 1, p. 109–115, 2016.

VISCONTI, P.; ORLANDO, C.; PRIMICERI, P. Solar Powered WSN for monitoring environment and soil parameters by specific app for mobile devices usable for early flood prediction or water savings. **IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)** 2016.

WAGH, S. S.; MORE, A.; KHAROTE, P. R. Performance Evaluation of IEEE 802.15.4

Protocol under Coexistence of WiFi 802.11b. **Procedia Computer Science**, v. 57, p. 745–751, Elsevier Masson SAS 2015.

WINTER, T.; THUBERT, P.; BRANDT, A.; et al. **RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks.** 2012. Disponível em: <https://tools.ietf.org/html/rfc6550>. Acesso em: 5 jun. 2018.

YI, L.; ZHONGYONG, F. The Research of Security Threat and Corresponding Defense Strategy for WSN. **Seventh International Conference on Measuring Technology and Mechatronics Automation**, p. 1274–1277, 2015.

YICK, J.; MUKHERJEE, B.; GHOSAL, D. Wireless sensor network survey. **Computer Networks**, v. 52, n. 12, p. 2292–2330, 2008.

YOON, Y.; KIM, J.; JUNG, M.; et al. Radio Propagation Characteristics in the Large City and LTE protection from STL interference. **ICACT Transactions on Advanced Communications Technology (TACT)**, v. 3, n. 6, p. 542–549, 2014.

YUE, Z.; REN, Y.; LI, J. **Performance evaluation of UDP-based high-speed transport protocols**. 2011.

YUNUS, F.; ISMAIL, N.-S. N.; ARIFFIN, S. H. S.; et al. Proposed Transport Protocol for Reliable Data Transfer in Wireless Sensor Network (WSN). **4th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO)**, p. 1–7, 2011.

APÊNDICES

APÊNDICE A – Código *Assembly* para troca de contexto do processador MSP430X

MSP430				MSP430X			
Salvar		**Carregar**		**Salvar**		**Carregar**	
PUSH.W	R2	POP.W	R4	MOV.W	6(SP),2(SP)	POPX.A	R4
PUSH.W	R15	POP.W	R5	MOV.W	4(SP),6(SP)	POPX.A	R5
PUSH.W	R14	POP.W	R6	MOV.W	2(SP),4(SP)	POPX.A	R6
PUSH.W	R13	POP.W	R7	ADDX.A	#4,SP	POPX.A	R7
PUSH.W	R12	POP.W	R8	PUSHX.A	R15	POPX.A	R8
PUSH.W	R11	POP.W	R9	ADDX.A	#4,SP	POPX.A	R9
PUSH.W	R10	POP.W	R10	POPX.W	R15	POPX.A	R10
PUSH.W	R9	POP.W	R11	SWPB	R15	POPX.A	R11
PUSH.W	R8	POP.W	R12	RLAM.W	#4,R15	POPX.A	R12
PUSH.W	R7	POP.W	R13	ADD.W	SR,R15	POPX.A	R13
PUSH.W	R6	POP.W	R14	PUSHX.W	R15	POPX.A	R14
PUSH.W	R5	POP.W	R15	ADDX.A	#(-4),SP	POPX.A	R15
PUSH.W	R4	BIC	#240, 0(R1)	PUSHX.A	R14	reti	
		POP.W	R2	PUSHX.A	R13		
				PUSHX.A	R12		
				PUSHX.A	R11		
				PUSHX.A	R10		
				PUSHX.A	R9		
				PUSHX.A	R8		
				PUSHX.A	R7		
				PUSHX.A	R6		
				PUSHX.A	R5		
				PUSHX.A	R4		

ANEXOS

ANEXO A – Código UDGM do Cooja para cálculo das probabilidades de sucesso de transmissão ponto-a-ponto

```
/*
 * Copyright (c) 2009, Swedish Institute of Computer Science.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the Institute nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 */

package org.contikios.cooja.radiomediums;

<...>

/**
 * The Unit Disk Graph Radio Medium abstracts radio transmission range as circles.
 *
 * It uses two different range parameters: one for transmissions, and one for
 * interfering with other radios and transmissions.
 *
 * Both radio ranges grow with the radio output power indicator.
 * The range parameters are multiplied with [output power]/[maximum output power].
 * For example, if the transmission range is 100m, the current power indicator
 * is 50, and the maximum output power indicator is 100, then the resulting transmission
 * range becomes 50m.
 *
 * For radio transmissions within range, two different success ratios are used [0.0-1.0]:
 * one for successful transmissions, and one for successful receptions.
 * If the transmission fails, no radio will hear the transmission.
 * If one of receptions fail, only that receiving radio will not receive the transmission,
 * but will be interfered throughout the entire radio connection.
 *
 * The received radio packet signal strength grows inversely with the distance to the
 * transmitter.
 *
 * @see #SS_STRONG
 * @see #SS_WEAK
 * @see #SS_NOTHING
 *
 * @see DirectedGraphMedium
 * @see UDGMVisualizerSkin
 * @author Fredrik Osterlind
 */
```

```
@ClassDescription("Unit Disk Graph Medium (UDGM): Distance Loss")
public class UDGM extends AbstractRadioMedium {
  private static Logger logger = Logger.getLogger(UDGM.class);

  public double SUCCESS_RATIO_TX = 1.0; /* Success ratio of TX. If this fails, no radios
receive the packet */
  public double SUCCESS_RATIO_RX = 1.0; /* Success ratio of RX. If this fails, the single
affected receiver does not receive the packet */
  public double TRANSMITTING_RANGE = 50; /* Transmission range. */
  public double INTERFERENCE_RANGE = 100; /* Interference range. Ignored if below
transmission range. */

  <...>

  public double getSuccessProbability(Radio source, Radio dest) {
        return getTxSuccessProbability(source) * getRxSuccessProbability(source, dest);
  }
  public double getTxSuccessProbability(Radio source) {
    return SUCCESS_RATIO_TX;
  }
  public double getRxSuccessProbability(Radio source, Radio dest) {
        double distance = source.getPosition().getDistanceTo(dest.getPosition());
    double distanceSquared = Math.pow(distance,2.0);
    double distanceMax = TRANSMITTING_RANGE *
    ((double) source.getCurrentOutputPowerIndicator() / (double)
source.getOutputPowerIndicatorMax());
    if (distanceMax == 0.0) {
      return 0.0;
    }
    double distanceMaxSquared = Math.pow(distanceMax,2.0);
    double ratio = distanceSquared / distanceMaxSquared;
    if (ratio > 1.0) {
        return 0.0;
    }
    return 1.0 - ratio*(1.0-SUCCESS_RATIO_RX);
  }
  <...>
}
```

Fonte: Rosendal e Elsts (2015).