INSTITUTO FEDERAL DE SANTA CATARINA

VICTOR CESCONETTO DE PIERI

Simulação de uma rede LoRaWAN: subsídios a operação com alta densidade de nodos

São José - SC

Julho/2023

SIMULAÇÃO DE UMA REDE LORAWAN: SUBSÍDIOS A OPERAÇÃO COM ALTA DENSIDADE DE NODOS

Monografia apresentada ao Curso de Engenharia de Telecomunicações do campus São José do Instituto Federal de Santa Catarina para a obtenção do diploma de Engenheiro de Telecomunicações.

Orientador: Prof. Eraldo Silveira e Silva, Dr.

Coorientador: Prof. Saul Silva Caetano, Dr.

São José - SC

Julho/2023

Victor Cesconetto De Pieri

Simulação de uma rede LoRaWAN: subsídios a operação com alta densidade de nodos

Este trabalho foi julgado adequado para obtenção do título de Engenheiro de Telecomunicações, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

São José - SC, 10 de abril de 2023:

Prof. Eraldo Silveira e Silva, Dr.
Orientador
Instituto Federal de Santa Catarina

Prof. Saul Silva Caetano, Dr.
Coorientador
Instituto Federal de Santa Catarina

Prof. Arliones Stevert Hoeller Junior, Dr.
Instituto Federal de Santa Catarina

Prof. Mario de Noronha Neto, Dr.
Instituto Federal de Santa Catarina

Este trabalho é dedicado ao meu avô,
que tem uma curiosidade fora do comum.

AGRADECIMENTOS

Agradeço primeiramente a toda a minha familia, sem eles esse trabalho não seria possível, sempre me apoiaram nos estudos e isso tem sido gratificante tanto pra mim quanto para eles. Aos meus professores, principalmente o orientador, pela disponibilização do tempo e do conhecimento para a resolução deste trabalho. Aos meus amigos a todos que conheci na instituição, sempre me proporcionaram bons momentos e tornaram mais agradáveis os dias de estudo.

"Eu posso aceitar o fracasso.
Todo mundo falha em alguma coisa.
Mas eu não posso aceitar não tentar." M. Jordan

RESUMO

Os sistemas de IoT LoRa estão sendo amplamente utilizados para comunicação e transmissão de dados em uma variedade de aplicações, sendo a mais comum o monitoramento com sensores. Com o uso dessa tecnologia, é possível monitorar locais de difícil acesso devido ao seu longo alcance, sem a necessidade frequente de trocar as baterias dos dispositivos, graças ao seu baixo consumo de energia. Além disso, o custo reduzido dos sistemas LoRa aumenta a viabilidade de implantação. O objetivo deste trabalho é comparar um sistema real LoRa em funcionamento com um sistema simulado utilizando o framework FLoRa em conjunto com a ferramenta Omnet++. Serão consideradas métricas como o consumo de bateria e a Taxa de Extração de Dados (DER). Através da análise dessas métricas para a rede real e simulada, será possível avaliar a viabilidade de uma rede de alta densidade de nodos.

Palavras-chave: FLoRa. Omnet. IoT. LoRaWAN. LPWAN.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

DER Data Extraction Rate.

IDE Integrated Development Environment.

IFSC Instituto Federal de Santa Catarina.

IoT Internet of Things.

LNS Lora Network Server.

LPWAN Low Power Wide Area Network.

MQTT Message Queuing Telemetry Transport.

OTAA Over-the-Air Activation.

RSSI Received Signal Strength Indication.

Rx Recepção.

SNR Signal-Noise Ratio.

TTN The Things Network.

SUMÁRIO

1 INTRODUÇÃO

A implementação de projetos utilizando tecnologias *Low Power Wide Area Network* (LP-WAN) é uma das alternativas da Internet das Coisas (Internet of Things (IoT)). Como já existem inúmeras soluções de LP-WANs, fica fácil a escolha de uma tecnologia que melhor se adapte a um determinado cenário com soluções de baixo custo. As tecnologias são variadas, todas tem seus prós e contras, algumas demandam mais custo de bateria, outras menos, assim como taxa de transferência entre outras diferenças(VARGA; LTD, 2021).

Entre as tecnologias de Low Power Wide Area Network (LPWAN), existe o LoRa (Long Range), que é utilizado em muitos projetos que visam a baixa frequência de manutenção e a maior durabilidade de bateria, para conseguir manter uma comunicação em locais de difícil acesso. Existem alguns desafios para projetar uma rede LoRa atualmente, dentre eles: um número grande e altamente variável de nós; diversos cenários sem fio caracterizados por fatores ambientais exigentes (por exemplo, ambientes urbanos densos com muitos edifícios); interferência devido a outras redes co-localizadas operando nas mesmas bandas de frequência às vezes não-licenciadas (BOR et al., 2016a).

Como existem variadas tecnologias de redes e uma diversidade de cenários de uso, é comum o uso de ferramentas para apoiar o projeto de rede. O uso de simuladores de redes de computadores é frequente e permite avaliar o desempenho de redes, apoiar a configuração e o posicionamento de dispositivos, ajustar a operação, tomar decisões de expansão da infraestrutura e testar novos algoritmos e protocolos. Neste sentido, os acadêmicos da universidade de Aalto na Finlândia criaram um *framework* de simulação chamado FloRa. O FLoRa implementa as camadas física e de MAC do LoRa, suporta comunicações bidirecionais e permite simulações de ponta a ponta, incluindo a rede de backhaul. Segundo (SLABICKI GOPIKA PREMSANKAR, 2018), utiliza como base o Omnet++, que é uma biblioteca da linguagem C++ para simulações de projetos de rede.

Neste trabalho, pretende-se construir uma comparação entre uma rede real e uma rede simulada por meio de uma análise de métricas associadas à cobertura de rede, vazão, entre outras, com o objetivo inicial de verificar se a rede simulada se aproxima da real. A rede real escolhida é aquela que já está sendo implementada em outro trabalho acadêmico no Instituto Federal de Santa Catarina (IFSC) Campus São José, o que facilita as análises e a seleção das métricas.

Além disso também será feito uma simulação de um cenário expandido, alterando as configurações iniciais da rede, aumentando o número de nodos e usando taxa de dados adaptativo(ADR), para apoiar a construção de uma rede maior e mais robusta.

1.1 Objetivo Geral

Simular um cenário real de uma rede *LoRaWAN* implantada no IFSC São José, parametrizando modelos conhecidos de perda de caminho utilizando dados coletados na rede física, utilizando a métrica de *Taxa de Extração de Dados(Data Extraction Rate (DER))* com objetivo de mostrar o desempenho da rede. Os resultados da simulação poderão ser usados para dar subsídios à operação da rede real e em possíveis cenários expandidos.

1.2 Objetivos específicos

- Fazer medições na rede real para parametrizar modelos de perda de caminho como Sombreamento LogNormal e Hata Okumura para o cenario indoor e outdoor da regiao do IFSC.
- Simular um cenário real de rede *LoRaWAN*, comparando os resultados com as medições realizadas em nodos físicos.
- Incluir na simulação o aumento da densidade de nodos, e outros fatores tais como: carga adicional de dados e algoritmo de *ADR* de forma a avaliar situações que normalmente não seriam possíveis no cenário real.
- Analisar os resultados da simulação e fornecer subsídios à operação da rede *LoRaWAN* no cenário real investigado e em cenários expandidos

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados os conceitos básicos e as tecnologias envolvidas neste trabalho, na simulação, empregaremos conceitos de LPWAN e uma aplicação específica com camada física baseada em LoRa. A simulação será desenvolvida com o framework Omnet++ junto com o INET que é o framework para simulação de redes, e por fim juntando-se ao FLoRa que é um outro framework para simulações de redes LoRa especificamente.

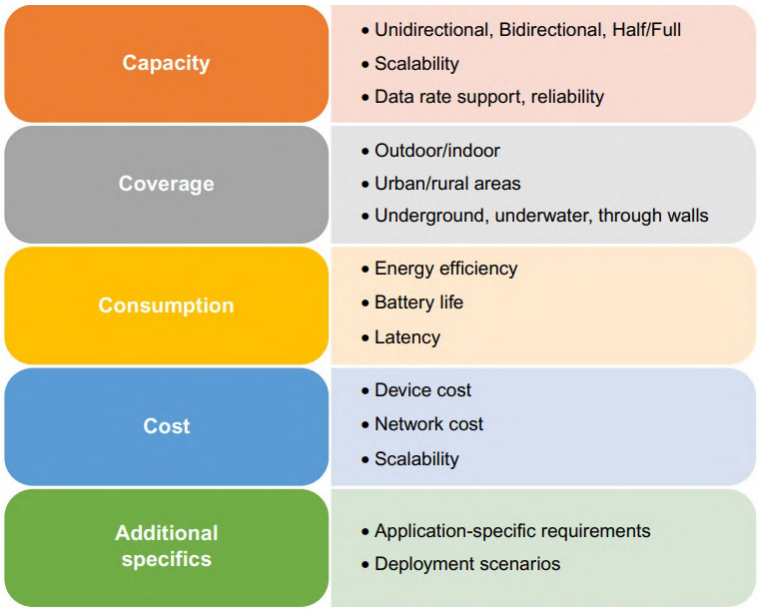
2.1 LPWAN

Muito consolidada atualmente, o uso da tecnologia de redes de baixa potência LPWAN é muito comum em projetos para IoT. Com o poder de conectar muitos dispositivos simultaneamente, junto de seu baixo custo energético e seu alto alcance, LPWANs garantem um acesso a informações em locais de difícil acesso, apenas utilizando um sensor, já que os gateways podem ficar bem distantes dos nodos. As tecnologias LPWAN podem ser utilizadas para uma ampla gama de aplicações, incluindo monitoramento ambiental, cidades inteligentes, serviços públicos inteligentes, agricultura, saúde, automação industrial, rastreamento de ativos, logística de transporte e muito mais (CHAUDHARI; ZENNARO; BORKAR, 2020).

As aplicações de LPWAN são muitas, cada solução LPWAN pode cobrir um determinado número de aplicações de acordo com o seu uso específico. Por exemplo, aplicações de cidades inteligentes requerem ampla cobertura de rádio para abranger todas as partes de uma cidade. Há muitos parâmetros que podem ser observados para uma solução LPWAN. O livro (CHAUDHARI; ZENNARO, 2020) cita 5 deles sendo os mais importantes: cobertura, capacidade, custo, consumo e caracteristicas adicionais de acordo com a Figura 1.

O longo alcance das redes LPWAN são possíveis por diversos fatores, um deles por exemplo é o uso de bandas abaixo dos *GHz* o que permite um comprimento de onda maior, possibilitando uma área maior de cobertura. O baixo custo de bateria é proporcionado por um *duty cycle* reduzido, os mecanismos *duty cycle* são adaptados baseados na aplicação. Se a aplicação precisa enviar ou receber mensagens os dispositivos finais acordam e enviam a mensagem, poupando bateria e ativando só quando necessário (RAZA; KULKARNI; SOORIYABANDARA, 2017). Existem diversas tecnologias LPWAN que tem *duty cycle* diferentes, por exemplo a LoRaWAN tem classes de dispositivos que se diferenciam no *duty cycle*.

Figura 1 – Aplicações LPWAN



Fonte: (CHAUDHARI; ZENNARO, 2020)

Outro fator que também contribui para as redes LPWAN serem muito utilizadas em *IoT* é a simplicidade dos protocolos *MAC*. Muitas outras tecnologias wireless (redes celulares e wifi), utilizam uma complexidade maior nesses protocolos. O *CSMA/CA* um protocolo *MAC* muito comum nas tecnologias acaba não servindo para as LPWANs por elas terem uma quantidade excessiva de dispositivos finais, por isso, as LPWANs mais comuns como LoRaWAN e *SigFox* utilizam o protocolo MAC chamado *ALOHA*. Como existem muitos dispositivos finais, a maioria das tecnologias LPWAN optam por reduzir a complexidade deles. Eles apenas servem como receptores e transmissores de dados. A complexidade da rede é voltada para as estações radio base, onde a recepção e transmissão é feita por multicanais ou sinais ortogonais simultaneamente (RAZA; KULKARNI; SOORIYABANDARA, 2017).

As redes LPWAN normalmente utilizam uma topologia de rede em estrela, onde os nodos se comunicam com uma estação base (*gateway*) sem a necessidade da comunicação entre si, utilizando apenas um salto. Na maioria das soluções o formato *nodo-gateway-servidor* de conexão é utilizado, com a comunicação do gateway até o servidor utilizando uma tecnologia diferente, como redes IP. No caso das LPWAN cada aplicação específica é utilizado uma tecnologia diferente, por exemplo para soluções de curto e médio alcance existem o ZigBee e o Sigfox que são tecnologias desenvolvidas para esse tipo de cenário. Quando se trata de longo alcance o LoRa é uma das LPWAN mais utilizadas, muito utilizada em locais de difícil acesso e permite uma longa distância entre os nodos e os gateways, além de sua longa vida de bateria e seu custo ser muito baixo. Com esses pontos podemos observar que dentre as características mais importantes da LPWAN o LoRa consegue abranger todas com suficiência, e sendo assim, como nosso projeto também

utiliza LoRa, a seguir discutiremos essa tecnologia.

2.2 LoRa

LoRa é uma técnica de *Spread Spectrum Modulation*, derivada da *Chirp Spread Spectrum (CSS)* integrada com um código de correção de erros desenvolvida pela Semtech. O LoRa oferece um equilíbrio entre sensibilidade da recepção e taxa de dados, opera em um canal de largura de banda fixa de 125 KHz nos canais para uplink e 500 KHz nos canais de *downlink* (SEMTECH, 2019).

A camada física permite longas distâncias, comunicação de baixa potência e opera em uma banda não-licenciada ISM sub-GHz. Como permite longas distâncias a taxa de bits tende a ser baixa. Segundo o artigo (BOR et al., 2016a) a comunicação LoRa é baseada em cinco fatores principais:

- **Fator de Espalhamento** (*Spreading Factor*): O fator de espalhamento é a razão entre a taxa de símbolos e a taxa de chip. O fator de espalhamento está diretamente relacionado à taxa de dados e a distância alcançada das conexões. Taxas de dados menores estão relacionadas com fatores de espalhamento mais altos, o que resulta em uma distância maior de comunicação. Entretanto, quanto maior o fator de espalhamento, maior a relação sinal-ruido(SNR) isso implica em um tempo maior de *airtime* . Escolhendo diferentes fatores de espalhamento, pode-se obter sinais ortogonais permitindo que o receptor possa com sucesso receber diferentes sinais de um mesmo canal ao mesmo tempo.

- **Potência de Transmissão** (*Transmission Power*): A potência de transmissão pode ser definida de acordo com a região. O rádio LoRa pode ser ajustado de -4 dBm para 20 dBm, e passos de 1 dB. Além disso as potências mais altas que 17dBm só podem ser usadas a 1 por cento de *duty cycle*.

- **Taxa de código** (*Coding Rate*) : Taxa de código é a taxa de código corretor de erros usado pelo modulador LoRa que oferece proteção contra rajadas de interferência, podem ser configurados entre 4/8 e 4/5 (BOR et al., 2016b). Quanto maior proteção de taxa de código maior o *airtime* da transmissão. Rádios com diferentes taxas de código (com o mesma frequência da portadora, fator de espalhamento e largura de banda), podem comunicar entre si usando um *header* explícito, onde a taxa de código dos dados do payload são armazenados no *header*.

- **Frequência da portadora** (*Carrier Frequency*) : A frequência da portadora depende das bandas licenciadas da região onde será utilizado o rádio. Pode ser programada em passos de 61 Hz, normalmente os rádios operam entre 137MHz até

1020MHz, porem as implementações são limitadas para as regiões, esse limite pode ser reduzido para 860MHz dependendo da região e do chip LoRa utilizado.

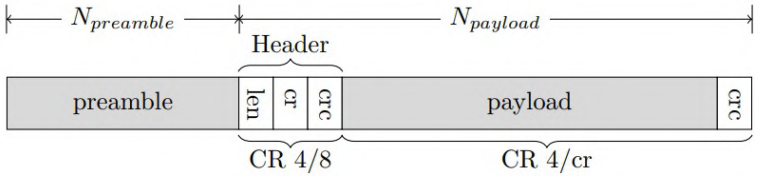
- **Largura de banda** (*Bandwidth*): A largura de banda influencia diretamente na taxa de dados das transmissões. Uma grande largura de banda proporciona uma alta taxa de dados, porém reduz a sensibilidade do receptor por ter um ruído adicional. Já uma largura de banda menor permite uma sensibilidade mais alta e uma menor taxa de dados. A largura de banda pode ser configurável de 7.8kHz até 500kHz, uma rede LoRa comumente opera em 500kHz, 250kHz ou 125kHz.

Alterando esses fatores, podemos criar vários cenários de rede onde podemos avaliar as taxas de dados, consumo de energia e resiliência aos ruídos.

A estrutura de pacotes do LoRa demonstrado na Figura 2, inicia com o *preâmbulo*, programável de 6 a 65535 *símbolos*, somados com o rádio, que adiciona mais 4.25 *símbolos*. Ainda possui um *header* opcional, que descreve o tamanho e a taxa de código de correção de erros do payload. O *payload* que contém de 1 até 255 *bytes* vem logo após o *header* opcional, o header que é sempre transmitido, utiliza uma taxa de correção de erros de 4/8 e tem seu próprio *CRC* (BOR et al., 2016a).

O *airtime* da transmissão depende da combinação do tamanho do *payload*, fator de espalhamento, largura de banda e da taxa de correção de erros. Dependendo de cada configuração de transmissão o *airtime* pode variar, por exemplo um pacote com 20 bytes pode variar de 9 milisegundos a 2,2 segundos.

Figura 2 – Estrutura de quadros LoRa



Fonte: (BOR et al., 2016a)

É importante ressaltar que, no contexto da comunicação LoRa, quando duas transmissões ocorrem simultaneamente e seus sinais se sobrepõem no receptor, é importante entender como o receptor lida com essa situação. Existem vários fatores que influenciam se o receptor será capaz de decodificar um ou ambos os pacotes, ou se não conseguirá decodificar nenhum deles.

Esses fatores incluem a frequência da portadora (CF), o fator de espalhamento (SF), a potência de transmissão e o sincronismo das transmissões. Para determinar se ocorre uma colisão entre duas transmissões, é considerado o intervalo de recepção de cada pacote, que define quando a recepção do pacote começa e termina. O ponto médio desse intervalo é calculado, assim como o seu comprimento.(BOR et al., 2016a)

A colisão ocorre quando o valor absoluto da diferença entre os pontos médios das transmissões é menor do que a soma dos comprimentos médios. Em outras palavras, se a diferença entre os pontos médios das transmissões for menor do que a soma dos comprimentos dos intervalos, isso indica que os pacotes estão se sobrepondo e há uma colisão. Essa colisão pode resultar na perda de pacotes ou na incapacidade de decodificar corretamente os dados transmitidos.(BOR et al., 2016a)

Portanto, compreender e analisar o comportamento de colisão além dos outros fatores é fundamental para otimizar o desempenho das transmissões LoRa, garantindo a melhor utilização do espectro e minimizando a interferência entre os pacotes. Essas informações são essenciais para projetar sistemas LoRa mais eficientes e confiáveis, especialmente em ambientes com múltiplos dispositivos transmitindo simultaneamente.

2.2.1 LoRaWAN

Para explicar melhor o funcionamento do LoRa, precisamos falar do LoRaWAN, que se constitui nas camadas de enlace e rede das redes LoRa, ou seja, é o protocolo da camada de rede e enlace(de acordo com as 7 camadas padrão da internet) que garante o funcionamento dos dispositivos que usam a camada física LoRa. Os nodos LoraWAN podem ser sensores ou atuadores que são conectados aos gateways por meio de *RF* utilizando a modulação LoRa.

Na maioria das aplicações LoRaWAN os nodos são autônomos, operados por bateria, eles digitalizam as leituras do ambiente e as enviam ao gateway. Quando são produzidos, os dispositivos LoRa possuem muitos identificadores únicos, esses identificadores são usados para administrar o dispositivo e ativá-lo com segurança. Esses identificadores únicos também são usados para o envio de dados criptografados a nuvem. (SEMTECH, 2019)

A especificação do LoRaWAN define um padrão aberto aos protocolos de rede da arquitetura de redes LoRa, que são baseados no protocolo ALOHA de enlace. A arquitetura de rede consiste em uma topologia hierárquica onde os nodos LoRa comunicam com os *gateways* e os gateways se comunicam com os Servidores de rede LoRa (*Lora Network Server (LNS)*) conforme a Figura 3.

Cada nodo não está necessariamente associado a um *gateway*, qualquer *gateway* pertencente a rede que é alcançado pelo nodo pode receber dados. Os gateways trabalham como encaminhadores de mensagem, recebem as mensagens dos nodos, e as enviam para o servidor LoRa. Os servidores LoRa por sua vez enviam as mensagens com os dados coletados pelos nodos aos servidores de aplicação (SLABICKI GOPIKA PREMSANKAR, 2018).

Os pacotes de *uplink* enviados por um nodo são recebidos por todos os gateways

que ele alcançar. Esse esquema reduz muito o erro de pacotes e o consumo de bateria. Já os *gateways* enviam os dados ao *LNS* via tráfego IP, que pode ser por cabo *Ethernet*, *Wi-Fi* ou conexões celulares. Os gateways na rede LoRa funcionam apenas como encaminhadores de mensagens, eles somente checam a integridade dos dados recebidos via RF.

Se os dados não estão íntegros, ou seja, o *CRC* está incorreto, a mensagem é descartada. Se está correta, o *gateway* encaminha ao *LNS*, junto com os metadados que incluem o nivel *RSSI* da mensagem, bem como um *timestamp* adicional. Para pacotes de *downlink*, o gateway executa as requisições de transmissão vindas do *LNS*. Enquanto múltiplos *gateways* recebem a mesma mensagem LoRa de RF, o *LNS* faz o trabalho de remover as mensagens duplicadas. Baseado no nivel *RSSI* das mensagens idênticas, o *LNS* seleciona o *gateway* que recebeu a mensagem com o melhor *RSSI* do nodo. (SEMTECH, 2019)

O gerenciamento da rede LoRaWAN é feito pelo *LNS*, ele ajusta a rede para , possui as configurações de funcionamento da rede e estabelece conexões seguras usando o algoritmo de 128-*bits AES* para o transporte dos dados fim-a-fim (dos nodos as aplicações em nuvem). Também controla o tráfego que flui até o nodo e vice-versa, garantindo que a recepção e a transmissão dos dados seja transportada pelo melhor sinal de transmissão, além de verificar a autenticidade de todos os sensores na rede, a integridade de todas as mensagens, e por fim, o *LNS* não tem acesso a nenhum dos dados da aplicação.

Segundo (SEMTECH, 2019) os *LNSs* em sua maioria contém:

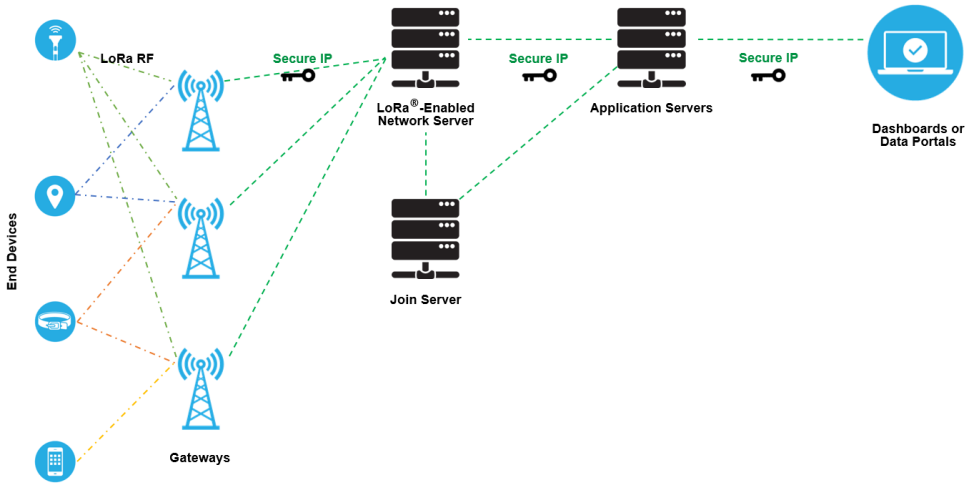
- Checagem de endereço de dispositivo.
- Autenticação dos quadros e gerenciamento de contador de quadros.
- Confirmações de mensagens recebidas.
- Adaptação de taxa de dados usando o protocolo *ADR*.
- Responde a todas as requisições dos dispositivos vindas da camada de enlace.
- Encaminha os *payloads* de uplink para os *servidores da aplicação*.
- Enfileira os *payloads* de donwlink vindos de qualquer *servidor de aplicação* para qualquer dispositivo conectado na rede.
- Encaminha mensagens de requisição e permissão entre os dispositivos e o *servidor de ingresso* (Para a versão 1.1 do LoRaWAN).

Os *servidores de aplicação (AS)* são responsáveis pelo gerenciamento do sistema. Eles possuem o papel de receber as informações dos dados dos sensores e para isso precisam de uma implementação livre de erros, utilizando ferramentas para o manuseio dos

dados vindos dos sensores. Sendo assim, como os gateways garantem o encaminhamento dos dados ao servidor, sem modificá-los, o papel do *AS* é garantir que esses dados sejam consistentes, provendo além de uma atomicidade de dados, um nível de segurança adequado usando dados do dispositivo final como:

- *DevEUI* (Número serial de identificador único do nodo)
- *AppKey* (Chave criptográfica de aplicação)
- *NETID* (Identificador de rede)
- *AppSKey* (Chave criptográfica de sessão)
- *NwkSkey* (Chave criptográfica do *LNS*)

Figura 3 – Rede LoRa



Fonte: (SEMTECH, 2019)

Para adicionar um novo dispositivo final (nodo) o *AS* checa se a entrada desse dispositivo específico está na lista de *dispositivos suportados*, verificando se o *DevEUI* do nodo e sua *AppKey* combinam. Após uma verificação de sucesso, o *servidor de aplicação* responde para o *LNS* um *AppNonce* (valor randômico ou identificador único fornecido por um *LNS*). Assim o *LNS* concatena um *Network ID (NETID)* e também alguns parâmetros de configuração dentro do código de integridade de mensagem *MIC* para enviar de volta ao nodo. O nodo valida o *MIC* e descriptografa a mensagem para obter o *AppNonce* e o *NETID* que são usados para criar as chaves de sessão *AppSKey* (ELDEFRAWY et al., 2019). Os novos nodos podem ser sincronizados com o sistema de duas maneiras, uma é *OTAA (Over-the-air Activation)*, esse modo de ativação corresponde a ativação

remota do nodo; a outra é *(ABP) Activation By Personalization*, esse modo corresponde a ativação manual onde as chaves são pré-configuradas no nodo.

Os nodos LoRaWAN operam em três modos diferentes: A,B e C. Todos os dispositivos devem permitir o suporte para a classe A e os classe B devem ter suporte para classes A e B, já os de classe C devem suportar todas as classes. Esses modos de operação definem como os nodos irão se comunicar na rede.

- **Classe A**: Nos dispositivos classe A a maior parte do tempo eles se encontram no estado ocioso e permitem transmissões bidirecionais. Quando observam uma mudança no ambiente seja relacionado a qualquer coisa que o dispositivo monitora, ele acorda e inicia o *uplink*, retornando os dados sobre as informações das mudanças pela rede.

  Para a resposta ele possui duas janelas, Rx1 e Rx2, entre as janelas há um intervalo de *sleep* configurável geralmente em termos do delay da transmissão de *uplink* e cada janela opera com uma duração de um segundo (tipicamente configurado). Se o dispositivo não receber mais nada nessas janelas de Recepção (Rx), ele entra em *sleep* novamente até ser acordado recebendo dados para enviar novamente (SEMTECH, 2019). Se o *downlink* da transmissão acontece durante a primeira janela (Rx1) o mesmo canal que está sendo usado por *uplink* é usado para *downlink* também, enquanto o *Fator de espalhamento* é determinado baseado no parâmetro *RX1DROffset*. No caso do segunda janela ser usada para *downlink*, um canal com um *Fator de espalhamento* fixo é usado.

  É responsabilidade do Lora Network Server (LNS) escalonar o tráfego do *downlink* no tempo exato para o sucesso do controle de tempo. Os dispositivos Classe A por frequentemente estarem em *sleep*, são os que menos consomem bateria (HAXHIBEQIRI et al., 2018).

Figura 4 – Classe A - Nenhuma recepção de dados

Fonte: (SEMTECH, 2019)

- **Classe B**: Também operando em classe A os dispositivos classe B, também chamados de *beacons* classe B, tem em seu funcionamento um processo chamado *beaconing*. Esse processo o *beacon* sincronizado com o tempo precisa fazer um *broadcast* pela rede pelos gateways, onde os nodos alinham seu tempo de acordo com a referência imposta pela rede(SEMTECH, 2019). Em relação a bateria os dispositivos de classe

Figura 5 – Classe A - Rx1 recebendo dados

Fonte: (SEMTECH, 2019)

Figura 6 – Classe A - Rx2 recebendo dados

Fonte: (SEMTECH, 2019)

B, têm um consumo maior em relação aos dispositivos de classe A, porque precisam abrir mais janelas de recepção que podem ser apenas de *beacons*, não necessariamente de um *downlink*.

- **Classe C**: Os dispositivos classe C também implementam a classe A, além de ter sua comunicação sempre ativa, sem *sleep*. O único momento que os dispositivos classe C estão impossibilitados de receber dados, são quando estão em modo de transmissão. O funcionamento da recepção é semelhante a classe A, porém na segunda janela (Rx2) a recepção nunca é desligada até alguma informação ser transmitida(HAXHIBEQIRI et al., 2018). O consumo de bateria dos dispositivos classe C é o maior dentre os três modos de operação, justamente por seu funcionamento nunca ter um *sleep* e sempre estar pronto para receber novos *downlinks*, porém, comumente os dipositivos classe C são ligados diretamente à energia sem o uso de baterias.

O protocolo LoRaWAN facilita muitas configurações de nodos. Além de ter uma certa facilidade com a adição de um novo dispositivo na rede, o LoRaWAN possui um instrumento chamado *Adaptative Data Rate (ADR)*, que gerencia dinamicamente os parâmetros do nodo no objetivo de aumentar a taxa de entrega de pacotes. O *ADR* gerencia a taxa de dados e a potência de transmissão dos nodos, para caso um dispositivo entrar na rede e o *uplink* for configurado pelo *LNS* o ADR adicionará um*uplink ADR bit*. O ADR se encarrega de configurar automaticamente as taxas de *uplink* do nodo caso o dispositivo não for configurado pelo *LNS*. O ADR do nodo e do *LNS* rodam assincronamente (HAXHIBEQIRI et al., 2018).

2.3 Simulação de redes LoRa e o simulador *OMNet++*

Para o propósito deste estudo foi escolhido o simulador *OMNet++*, que mais a frente será explicado, permitindo com seu framework *INET* desenvolver uma simulação de rede robusta, onde pode-se selecionar protocolos de rede entre outros parâmetros. Para as simulações de redes LPWAN e mais especificamente redes LoRa, existe uma ferramenta dentro do Omnet++, o FLoRa, que será explicado adiante, é um outro framework que trabalha junto com o INET possibilitando uma implementação fiel ao sistema LoRa que será utilizado no projeto.

2.3.1 Avaliação de desempenho por simulação

Para avaliar o desempenho de um sistema pode-se utilizar quatro técnicas: modelo analítico, análises por simulação e um sistema real ou uma combinação dessas três. Em particular, a simulação é interessante pois permite estudar cenários que normalmente não poderiam ser avaliados em experimentos reais de forma simples. Com métricas pré-estabelecidas pode-se construir uma simulação com muitas variáveis em um ambiente desejado e assim fazer testes muito próximos do mundo real, permitindo ajustar pontos de funcionamento, prevendo comportamento de expansões da rede e otimizando o funcionamento da mesma.

Nesta seção será exposto quais os conceitos para construir uma simulação de um sistema de eventos discretos, para a construção de uma simulação com o *OMNet++* junto com o *INET* e o *FLoRa*.

A partir do *OMNet++* precisamos adicionar uma biblioteca robusta para simulação de redes, o *INET*, que permite construção de simulações de redes utilizando protocolos já conhecidos, como por exemplo: TCP, UDP, IP e IEEE 802.11 (INET-DOCUMENTATION, 2022). E além disso temos o *FLoRa* que é um framework de código abertro para simulações de redes LoRa baseado no INET. Criado pela universidade Aalto na Finlândia, esse framework possibilita a criação de simulações LoRa, que permite a análise de desempenho de redes LoRa em ambientes variados.

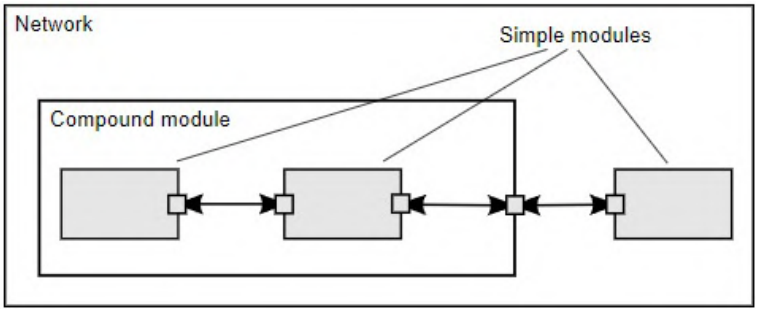
2.3.2 *OMNet++* e INET

Uma das ferramentas a ser utilizadas é o *OMNet++* desenvolvido pela OpenSim, uma robusta biblioteca *opensource* para a linguagem C++ que permite construir simulações de todo o tipo e é especializado em simulações de tempo discreto. Um sistema de eventos discretos é um sistema onde mudanças de estado (eventos) acontecem em instâncias discretas no tempo, e os eventos levam tempo zero para acontecer. Não é um simulador de uma aplicação ou sistema em particular, mas ele proporciona uma arquitetura flexível e genérica para escrever ferramentas de simulação (KIM et al., 2010).

O modelo de sistema do *OMNet++* possui módulos onde eles se comunicam com a passagem de mensagens. As mensagens podem ser enviadas por meio de conexões que abrangem módulos(módulo composto) ou diretamente para outros módulos. O modelo em si é chamado de rede no *OMNet++*, e os módulos ficam agrupados em algo chamado *módulo composto* de acordo com a Figura 7 (VARGA; LTD, 2021).

Os modelos *OMNet++* são organizados hierarquicamente e dentro desses modelos existem módulos conectados que se comunicam entre si. O módulo *top-level* é o *módulo do sistema*, é o módulo principal, onde contém os sub módulos que podem conter outros submódulos, esses por sua vez são chamados de *módulos compostos*.

Figura 7 – Módulos OMNet++



Fonte: (VARGA; LTD, 2021)

Cada módulo contém uma implementação em C++ usando a biblioteca *OMNet++*(VARGA; LTD, 2021). Quando montados em maiores componentes usando uma linguagem de alto nível (*NED - Network Description*). Os módulos não podem chamar funções de outros módulos, apesar de serem conectados. Entretanto, eles interagem passando *mensagens* pelas *portas* de conexão. Essas *mensagens* podem carregar estrutura de dados, incluindo pacotes de comunicação de rede. A arquitetura do *OMNet++* que desenvolvedores encapsulam um leque de funções em um componente que se assemelha a interação entre entidades em sistemas distribuidos (KIM et al., 2010).

As *mensagens* representam quadros ou pacotes de uma rede, trabalhos ou clientes numa rede de filas ou em outros tipos de entidades móveis. As *mensagens* podem conter uma complexa estrutura de dados arbitrários (VARGA; LTD, 2021). O *gates* ou *portas* são as interfaces de entrada e saída dos módulos, as *mensagens* passam por eles para irem de um módulo para o outro. Cada conexão ou *link* é criada com um único nível de hierarquia de módulos, dentro de um módulo composto por exemplo, pode-se conectar as *portas* correspondentes de dois submódulos, ou a *porta* de um submódulo e a *porta* de um módulo composto, de acordo com a figura 7 (VARGA; LTD, 2021).

Os módulos possuem parâmetros, que podem ser configurados por *NED* ou por arquivos de configuração. Os parâmetros podem ser usados para customizar um simples módulo ou para parametrizar a topologia do modelo. Usam de variáveis ou métodos

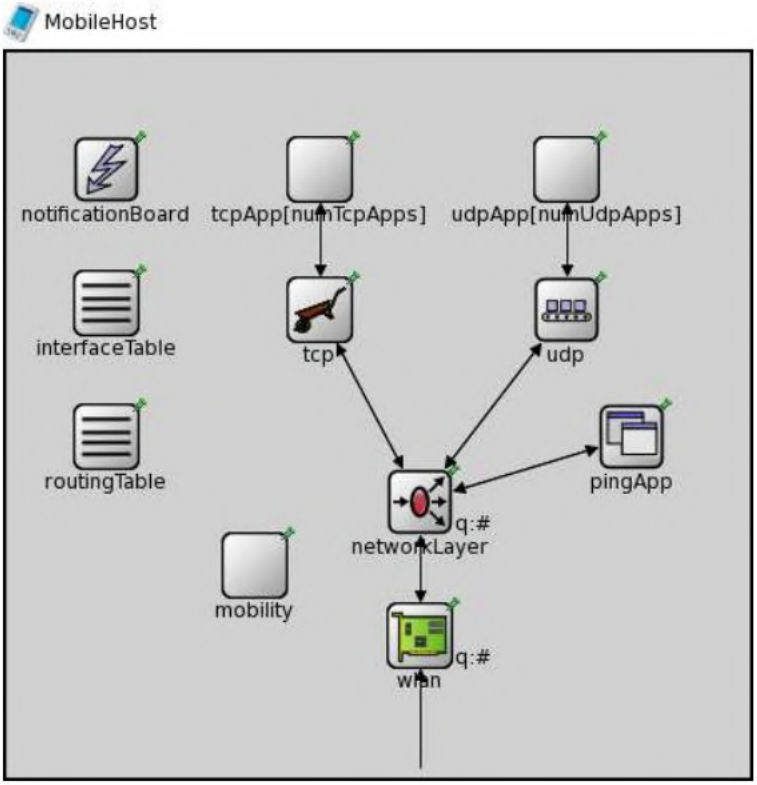
desenvolvidos em *C++* para iniciar junto com um módulo, além disso, pode-se definir o número de submódulos de um módulo composto e as rotinas que cada um irá executar em cada um deles.

Uma das ferramentas do *OMNet++* mais utilizadas para simulação de redes de computadores é o *INET*, como já dito anteriormente, ele implementa os protocolos da camada de rede, enlace e roteamento, o que facilita a construção de módulos de simulação para aplicações de rede. Os protocolos são representados como módulos, e podem ser combinados para implementar *hosts* e dispositivos de rede como *roteadores*, *switche s*, *access points* entre outros.

Na Figura 8, temos um exemplo de uma aplicação sem fio construída com o *OMNet++* usando *INET*. Dentro do *host* os módulos para os protocolos são conectados de acordo com suas camadas no modelo de rede. Os módulos das camadas superiores representam as aplicações, os módulos nas camadas do meio implementam os protocolos de transporte e por fim os módulos na camada inferior é um módulo composto que implementa os protocolos de enlace e rede (KIM et al., 2010).

Além do INET, as simulações para redes especificas utilizam outros *framework*, no caso deste projeto será utilizado o *framework* para redes LoRa que será apresentado logo adiante.

Figura 8 – Estrutura do Módulo INET



Fonte: (KIM et al., 2010)

O OMNeT++ fornece suporte integrado para registrar resultados de simulação, por meio de vetores de saída e escalares de saída. Os vetores de saída são dados de séries temporais registrados a partir de módulos ou canais e podem ser utilizados para registrar informações como atrasos de pacotes, comprimentos de filas, estado de módulos, entre outros. Já os escalares de saída são resultados resumidos calculados durante a simulação e podem ser números individuais ou resumos estatísticos, como média, desvio padrão, mínimo, máximo, etc(VARGA; LTD, 2021).

Existem duas maneiras de coletar e registrar os resultados no OMNeT++: através do mecanismo de sinais, utilizando estatísticas declaradas, ou diretamente no código C++ com o uso da biblioteca de simulação. O método baseado em sinais e estatísticas declaradas foi introduzido na versão 4.1 do OMNeT++ e é recomendado por permitir registrar os resultados de forma personalizada sem a necessidade de modificar extensivamente o modelo de simulação(VARGA; LTD, 2021).

Durante um estudo de simulação, são realizados múltiplos experimentos com o objetivo de investigar questões como o impacto do número de nodos nos tempos de resposta na rede. Cada experimento envolve a execução do modelo de simulação com diferentes conjuntos de parâmetros, e várias medições são conduzidas para coletar os resultados. A fim de mitigar o viés introduzido pelo fluxo de números aleatórios utilizado na simulação, cada medição é repetida várias vezes com sementes de números aleatórios distintas, e os resultados são posteriormente calculados.

As ferramentas de análise de resultados do *OMNeT++* podem aproveitar as etiquetas de experimento, medição e repetição registradas nos arquivos de resultados e exibir as execuções de simulação e os resultados registrados na interface do usuário.

Essas etiquetas podem ser especificadas explicitamente no arquivo ini usando as opções de configuração experiment-label, measurement-label e replication-label. Se elas estiverem ausentes, o padrão é o seguinte:

- experiment-label = "$configname"
- measurement-label = "$iterationvars"
- replication-label = "$repetition,seed-set=<seedset>"

Ou seja, a etiqueta de experimento padrão é o nome da configuração; a etiqueta de medição é concatenada a partir das variáveis de iteração; e a etiqueta de repetição contém a variável do loop de repetição e seed-set.

Essa capacidade de registrar resultados no OMNeT++ é fundamental para obter uma visão completa do comportamento do modelo durante a simulação. Ela permite

analisar métricas importantes, como atrasos, tempos de espera em filas e utilização de recursos, auxiliando na compreensão do desempenho e no refinamento do modelo.

2.3.3 FLoRa

Existe um *framework* específico para simulações LoRa dentro do *OMNet*. Esse *framework* inclui uma biblioteca com código aberto chamada *FLoRa*, publicada em 2017, desenvolvida por pesquisadores na Universidade de Aalto na Finlândia onde existem vários projetos de pesquisa no tema LoRa e LoRaWAN. Este *framework* permite a implantação de vários nodos e vários *gateways* na rede, além de permitir configurar diversos fatores como a posição dos dispositivos em coordenadas x e y, coeficientes de perda de propagação, ativar ou desativar a tecnologia *ADR* nos dois nodos do *LNS* (YASCARIBAY et al., 2022).

Como dito anteriormente, a simulação feita com o *FLoRa* permite configurar todos os parâmetros de transmissão da camada física LoRa: Fator de espalhamento, frequência central, largura de banda, taxa de código e potência de transmissão (SLABICKI GOPIKA PREMSANKAR, 2018). O *framework* também permite coletar métricas para analisar os problemas da rede sendo simulada, com isso é possível resgatar os resultados mais facilmente.

O foco deste projeto é verificar a escalabilidade da rede com maior número de nodos além dos existentes na rede física, mobilidade dos nodos para análise da cobertura e erros de rede, além de testar o funcionamento do *ADR* que é um mecanismo automático de adaptação de dados dos dispositivos, que também é possível de ser implementado na simulação. Todos esses dados o *framework FLoRa* permite configurar de acordo com o desejo do usuário. Erros de propagação, colisão de pacotes, consumo de energia e taxa de entrega de pacotes também são outras métricas possíveis de análise da rede LoRa no simulador (YASCARIBAY et al., 2022). No *FLoRa* há métricas disponíveis para a simulação, dentre elas temos os parâmetros de entrada e as estatisticas de saída.

Dos parametros de entrada:

- **Número de nodos**: Numero de dispositivos finais nodos é determinado pelo projeto, há possibilidade de aumentar o numero de nodos em grandes quantidades, porém isso acarreta em um tempo de simulação maior, podendo durar horas ou até dias.
- **ADR (Adaptative Data Rate)**: É possível ativar ou desativar o ADR tanto nos dispositivos quanto nos *LNSs* para testes.
- **Numero de *gateways***: A configuração de vários *gateways* é possível, pois o *LNS* é configurado para filtrar os pacotes duplicados e tratá-los para que seja enviado

somente um ao nodo com melhor qualidade de sinal (SLABICKI GOPIKA PREMSANKAR, 2018).

- **Mobilidade dos nodos**: Simular um sistema onde os nodos não são fixos e podem se movimentar livremente no ambiente.

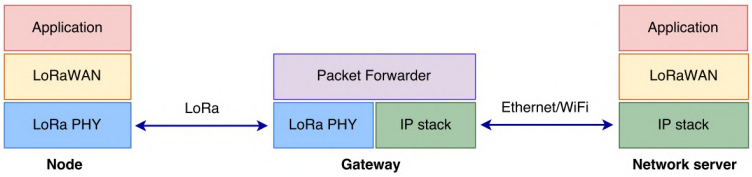
Das estatísticas de saída:

- **Consumo de energia**: Consumo total de energia dos nodos, normalmente informada em *Joules(J)*.
- **Perdas de propagação**: As perdas de propagação são mostradas através dos valores de RSSI(Received Signal Strenght Indicator ou Indicador de Potência do Sinal Recebido) abaixos do limiar de sensibilidade do receptor.
- **DER(Taxa de Extração de Dados)**: A Taxa de Extração de Dados mostra a porcentagem de entrega de dados do sistema, muito útil para verificar o desempenho do sistema. Quanto mais próxima a DER estiver de 1, mais eficiente é a rede.(BOR et al., 2016a)

O *FLoRa* suporta ambientes urbanos e suburbanos, assim como os dados de perda de percurso da transmissão de cada ambiente simulado. Seguindo experimentos feitos em (BOR et al., 2016a) os casos de simulações feitas com ambientes urbanos têm maior perda de percurso do que em suburbanos, portanto tem menor área de cobertura. A transmissão eficiente do LoRa depende se as transmissões se interferem entre si. Assume-se que duas transmissões com fator de espalhamento diferentes não se colidem devido à seus canais ortogonais.

De acordo com a Figura 9, o *FLoRa* implementa módulos *OMNet++* que replicam a camada física LoRa e a camada de enlace (LoRaWAN). As transmissões vindas dos nodos de múltiplos canais podem ser recebidas pelos *gateways* simultaneamente de acordo com as especificações do LoRaWAN na simulação (SEMTECH, 2019). A camada física pode ser implementada utilizando módulos *INET* como os enlaces de *Ethernet* e *Wifi*.

Figura 9 – Estrutura de módulos FLoRa

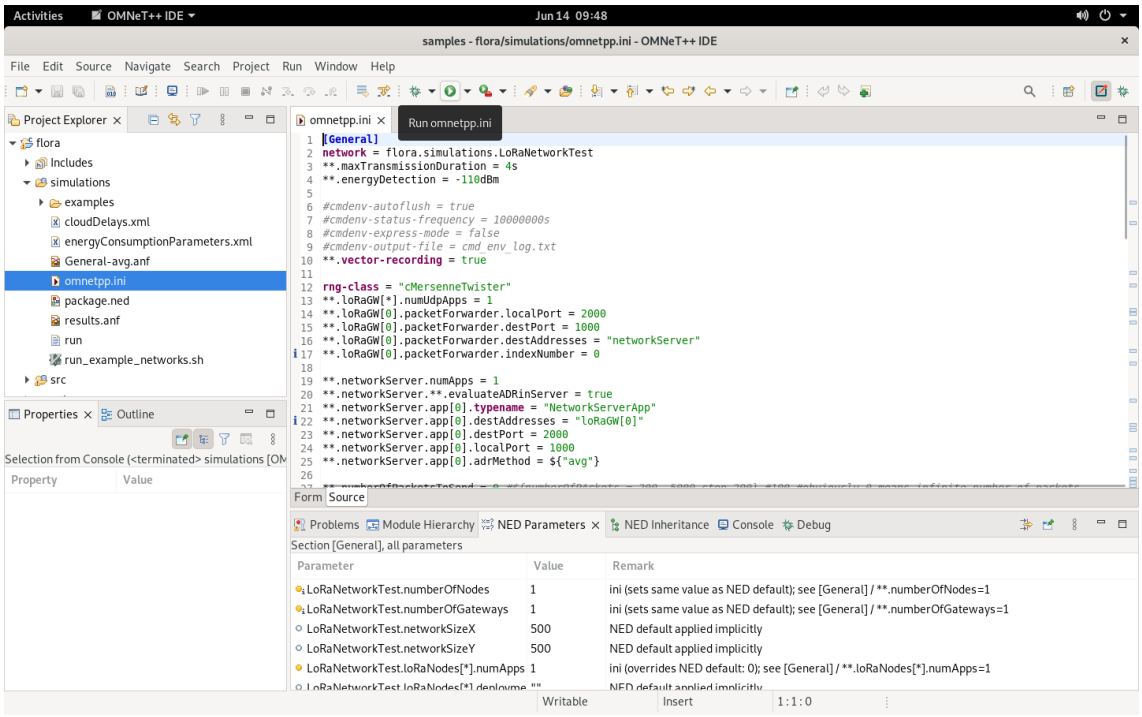


Fonte: (SLABICKI GOPIKA PREMSANKAR, 2018)

A adoção do *FLoRa* nos fornece modelos de *perda de percurso* que incluem dois tipos já implementados: os modelos *Hata Okumura* e o *Log-normal shadowing*(Sombreamento

Log-Normal). Na próxima seção, serão apresentados mais detalhes sobre cada um deles. Todo o código-fonte do *FLoRa* está disponível no *GitHub*, e a página de tutoriais contém informações úteis sobre a implementação. Na Figura 10, é possível observar várias configurações similares às do *OMNet*, que permitem definir os parâmetros de simulação para LoRa e LoRaWAN(FLORA-DOCUMENTATION, 2022).

Figura 10 – Fatores de simulação no OMNet++



Fonte: (FLORA-DOCUMENTATION, 2022)

Por fim o *FLoRa* nos permite construir uma simulação fiel à uma rede LoRa, configurando os principais parâmetros do sistema, para uma melhor análise de dados de desempenho de rede sobre o projeto definido. Isso implica que podemos construir uma simulação com muitos parâmetros a fim de reproduzir um cenário mais próximo do real.

2.4 Modelos de perda de percurso

A perda de percurso é um fenômeno comum em sistemas de comunicação sem fio, que ocorre devido à atenuação do sinal à medida que ele se propaga pelo meio de transmissão. A atenuação pode ser causada por diversos fatores, como a absorção pelo meio de transmissão, a difração ao redor de obstáculos, a reflexão em superfícies e a dispersão pelo meio.

No contexto específico do LoRa, a perda de percurso é um fator crucial a ser considerado na implantação de uma rede eficiente. Isso porque o LoRa opera em frequências de rádio de baixa potência, o que significa que o sinal transmitido é relativamente fraco e,

portanto, mais suscetível a atenuações ao longo do percurso percorrido (YASCARIBAY et al., 2022).

A modelagem das perdas de percurso é essencial para determinar a cobertura da rede LoRaWAN e para otimizar a colocação dos dispositivos na rede. A escolha adequada da potência de transmissão e da sensibilidade do receptor pode ajudar a minimizar as perdas de percurso e melhorar o desempenho geral da rede. Portanto, é importante que sejam feitas simulações realistas das perdas de percurso para garantir que a rede LoRa atinja sua máxima eficiência e alcance (BORGHETTI; PAOLINI; POLESE, 2019). Existem modelos matemáticos que podem ser usados para simular as perdas de percurso, como o modelo de *Log-normal* e o modelo de *Hata-Okumura* (BARBIROLI et al., 2019), nesse sentido, esses foram os dois modelos escolhidos para utilizar na simulação.

2.4.1 Modelo Sombreamento Log-Normal

O modelo Log-normal é um dos modelos mais comuns usados na modelagem de perdas de percurso em comunicações sem fio. Ele é baseado na suposição de que a perda de percurso segue uma distribuição log-normal. Em outras palavras, o modelo considera que o logaritmo da perda de percurso é uma variável aleatória normalmente distribuída (RAPPAPORT, 2009).

A fórmula para o cálculo da perda de percurso usando o modelo log-normal é dada pela Equação 2.1 onde $PL(d)$ é a perda de percurso na distância d, $PL(d_0)$ é a perda de percurso na referência de distância d_0, n é o expoente de perda, X é uma variável aleatória normal com média zero e desvio padrão σ, que corresponde ao sombreamento.(RAPPAPORT, 2009).

$$PL(d) = PL(d_0) + 10 \cdot n \cdot \log_{10}(\frac{d}{d_0}) + X(0, \sigma) \tag{2.1}$$

2.4.2 Modelo Hata-Okumura

O modelo de Hata-Okumura, por sua vez, é um modelo empírico amplamente utilizado para calcular a perda de percurso em sistemas de comunicação móvel. Ele foi originalmente desenvolvido para sistemas de telefonia celular, mas também pode ser usado em outros sistemas de comunicação sem fio, como LPWANs e consequentemente o LoRa (HATA, 1980). O modelo leva em consideração fatores como a frequência da portadora, a altura da antena e a distância entre o transmissor e o receptor. A fórmula para o cálculo da perda de percurso usando o modelo de Hata-Okumura é dada pela Equação 2.2 onde $PL(d)$ é a perda de percurso na distância d, f é a frequência da portadora em MHz, h_b é a altura da antena base em metros, A, B, C e D são constantes que dependem do ambiente de propagação e do tipo de terreno(RAPPAPORT, 2009).

$$PL(d) = A + B \cdot \log_{10}(d) + C \cdot (\log_{10}(f) - 3)^2 - D \cdot \log_{10}(h_b) \tag{2.2}$$

No Capítulo 4, daremos um passo adiante ao aplicar os modelos de Sombreamento Log-Normal e Hata-Okumura, discutidos anteriormente, como base para a parametrização das simulações em *OMNet++*. Ao utilizar essas teorias consolidadas, poderemos explorar e analisar o desempenho de redes LoRa de em diferentes cenários e condições de propagação. Essa abordagem nos permitirá construir simulações, contribuindo para uma compreensão mais aprofundada do comportamento dos sistemas LoRa explicados anteriormente.

3 CONFIGURAÇÃO DO SISTEMA DE COLETA DE DADOS

Este capítulo aborda os preparativos essenciais para a configuração do sistema de coleta de dados reais e para a realização de simulações. Na primeira parte, serão apresentados os procedimentos necessários para a coleta de dados reais, fornecendo informações sobre o processo e os recursos utilizados, como o dispositivo utilizado na coleta, o gateway posicionado na caixa d'água do IFSC Campus São José e os Servidores de Rede e de Aplicação, responsáveis por receber e processar os dados para uma melhor visualização e tratamento dos dados.

Na seção 3.2, serão abordados os preparativos para as simulações, nos quais será configurado o ambiente de simulação, alterando valores padrão do simulador para parametrizar os modelos de perda de percurso e configurar os parâmetros de rádio. Por fim, será tratada a parte dos resultados da simulação, na qual podemos visualizar as estatísticas fornecidas pelo *OMNet++* e realizar análises do desempenho da rede por meio de dados como *DER* (Taxa de Extração de Dados ou *Data Extraction Rate*), colisões de pacotes e a média de *RSSI* (*Received Signal Strength Indicator* ou Indicador de Potencia do Sinal Recebido) recebido por cada nodo.

3.1 Preparativos para a coleta de dados

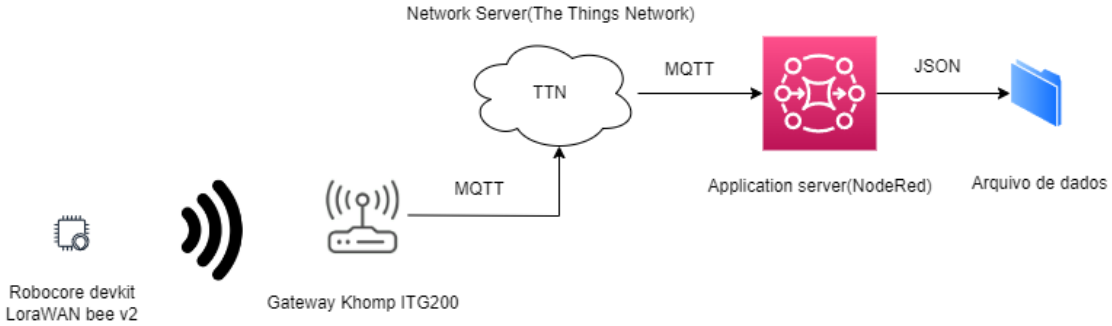
Nesta seção, serão apresentados os detalhes de configuração do sistema de coleta de dados na rede *LoRaWAN*. Será abordado o processo de programação do firmware no Arduino Integrated Development Environment (IDE), a configuração do servidor de rede *TTN* (The Things Network) (THETHINGSNETWORK, 2023) e do servidor de aplicação *Node-RED* (NODERED, 2023), utilizando o protocolo *MQTT* (*Message Queuing Telemetry Transport*). Também serão mostrados os esquemáticos da configuração e alguns trechos de código que demonstram as alterações realizadas.

Os equipamentos utilizados para as medições foram uma placa *ESP32*, juntamente com um módulo *LoRa SX1276M0* (DATASHEET. . . , 2019), conectados a uma placa base desenvolvida pela *RoboCore* (ROBOCORE-DEVKIT, 2021). Foi integrado um *gateway Khomp IGT200* (DATASHEET-IGT200, 2019), fornecido pelo campus do IFSC São José, ao sistema.

Também será abordada a configuração do servidor de rede *TTN* para receber dados dos dispositivos *LoRa*, incluindo os passos de criação de aplicativo e dispositivo,

bem como a configuração dos parâmetros de comunicação. Será detalhada também a configuração do servidor de aplicação *Node-RED*, utilizando *MQTT* para receber os dados do *TTN* e processá-los. Além disso, serão apresentados os esquemáticos da configuração, ilustrando as conexões entre dispositivos *LoRa*, sensores e os servidores *TTN* e *Node-RED*, fornecendo uma visão geral do sistema (Figura 11).

Figura 11 – Sistema de coleta de dados físicos



Fonte: Elaborado pelo Autor

3.1.1 Programação do firmware usando o Arduino IDE

Primeiramente, foi necessário configurar o *firmware* do nodo Robocore utilizando a plataforma de desenvolvimento *Arduino IDE*. Para isso, foi preciso instalar bibliotecas que são compatíveis com o processador *ESP32* (ESPRESSIF..., 2019) presente na placa. Em seguida, foi realizado uma pesquisa nos manuais *LoRa* para obter informações sobre os comandos *AT* (LORA..., 2020). Esses comandos *AT* permitem interagir com os módulos *LoRa* por meio de uma conexão serial, proporcionando aos usuários a capacidade de configurar parâmetros, definir modos de operação e controlar a transmissão e recepção de dados, entre outras funcionalidades.

Os comandos AT utilizados foram :

- AT+CH <canal> <ativar/desativar(1/0)>: Esse comando AT ativa ou desativa um canal nos canais de frequência disponíveis do dispositivo. Foram ativados os canais da *TTN* do 8 ao 16 de 125KHz cada, iniciando por 916MHz até 918MHz.
- AT+REGION <valor>: Ativa todos os canais referentes a região configurada, a regiao configurada foi a AU915(Austrália 915MHz).
- AT+TP <valor>: Configura a potencia de transmissão, começando por zero que corresponde a máxima potencia.
- AT+DEVEUI: (*DEVEUI* é uma abreviação para *Device EUI*), que deve retornar o *Device EUI* do módulo, um código de 16 dígitos hexadecimais (8 bytes). *Device*

EUI representa o endereço do dispositivo para a rede, basicamente como o *MAC Address* da conexão *Wi-Fi* do *ESP32*, portanto ele é extremamente importante e único para cada módulo.

- AT+DR <valor>: Corresponde ao Data Rate do dispositivo, foi selecionado o valor que corresponde a 125KHz e fator de espalhamento 12.
- AT+ADR <valor>: Foi mantido desativado o *ADR* no experimento.
- AT+AppEUI <valor>: O *AppEUI* é o identificador único da aplicação na qual o dispositivo vai se conectar, então este valor precisa estar configurado corretamente para que haja comunicação entre a aplicação e o dispositivo. Isso ajuda a evitar comunicações indesejadas entre dispositivos e aplicativos que não estão autorizados a interagir entre si.
- AT+APPKEY <valor>: Este valor precisa estar igual ao da aplicação, para o dispositivo poder ingressar na rede. Essa chave é compartilhada entre o dispositivo e a rede *LoRaWAN* para garantir a confidencialidade e a integridade das comunicações.
- AT+NJM <valor>: Network Join mode, corresponde ao modo de ingresso do dispositivo a rede, foi selecionado 1, que ativa o modo *Over-the-Air Activation (OTAA)(Over-the-Air Activation)*. Ao usar o *OTAA* para o ingresso, a rede, por sua vez, autentica o dispositivo e atribui a ele um conjunto de chaves de segurança, incluindo o *AppEUI* (Endereço EUI da aplicação) e o *AppKey* (Chave de aplicação).

No *Arduino IDE*, existe uma biblioteca que já implementa todos esses comandos por meio de métodos, os quais foram utilizados na etapa de configuração do *firmware*, como exemplificado no trecho de código presente na Código 3.1. É importante mencionar que a rotina de ativação dos canais Código 3.2 foi desenvolvida separadamente, fora do escopo da biblioteca padrão do *Arduino IDE*.

Código 3.1 – Trecho da rotina setup do firmware no Arduino IDE

```
1  //Configura o data rate SF12 125kHz
2  resposta = lorawan.set_DR(0);
3  if(resposta == CommandResponse::OK){
4    Serial.println(F("SF = 12 BW = 125kHz"));
5  } else {
6    Serial.println(F("Erro ao configurar o DR"));
7  }
8  //Potencia maxima - 14dBm
9  resposta = lorawan.set_TXPower(0);
10 if(resposta == CommandResponse::OK){
11   Serial.println(F("Potencia de transmissao configurada para o maximo"));
12 } else {
```

```
13      Serial.println(F("Erro ao configurar a Potencia"));
14    }
15    //set ADR
16    resposta = lorawan.set_ADR(0);
17    if(resposta == CommandResponse::OK){
18      Serial.println(F("ADR setado"));
19    } else {
20      Serial.println(F("Erro ao configurar o ADR"));
21    }
22    //set Region
23    uint8_t region;
24    resposta = lorawan.set_Region(1);
```

Código 3.2 – Codigo de desativação de canais no Arduino IDE

```
1   Serial.println("Configurando canais...");
2   int i=0;
3   while(i < 72){
4     if((i < 9 || i > 16) && i != 60){
5       resposta = lorawan.deactivate_Channel(i);
6       if(resposta == CommandResponse::OK){
7         Serial.print(F("Desativou canal :"));
8         Serial.println(i);
9       } else {
10        Serial.println(F("Erro ao desativar o canal"));
11      }
12      delay(1000);
13    }
14    i++;
15  }
```

O método *deactivate_Channel* (Código 3.3) foi desenvolvido na biblioteca para executar essa rotina usando o comando AT+CH para inicializar somente os canais necessários na rotina de inicialização do firmware da placa RoboCore. Assim foi possível ter garantias que após o uso do comando AT+REGION, que ativa todos os canais de 912MHz a 926MHz, o firmware desative os canais que não serão utilizados.

Código 3.3 – Código de desativação do canal desenvolvido

```
1 CommandResponse SMW_SX1276M0::deactivate_Channel(uint8_t channel){
2   char data[12] = { CHAR_EOS };
3   snprintf(data, sizeof(data), "%d status=0%c", channel, CHAR_EOS);
4   _send_command(CMD_CH, 1, data);
5   CommandResponse res = _read_response(SMW_SX1276M0_TIMEOUT_WRITE); // this command
      takes almost 1 s to reply
6   return res;
7 }
```

Após configurar a inicialização do equipamento no setup, foi implementado um código para enviar mensagens a cada 15 segundos, a fim de coletar dados de *RSSI* e *SNR*. Esses dados são automaticamente adicionados ao *payload* dos pacotes *Message Queuing Telemetry Transport (MQTT)* quando recebidos pelo *gateway* e enviados para a aplicação. Além disso, o *payload* de dados também inclui um valor simbólico de 4 bytes, que representa a potência configurada.

3.1.2 Configuração do Servidor de Rede *TTN*

Na *TTN* o *gateway* já estava pré-configurado, então apenas precisamos associá-lo à aplicação. No entanto, para o registro do dispositivo nodo, foi necessário realizar algumas configurações específicas, conforme ilustrado na Figura 12.

Foram realizadas configurações específicas para adaptar o plano de frequência à região da Austrália, seguindo as diretrizes e especificações regionais brasileiras para o *LoRaWAN*. A frequência selecionada foi de 915MHz, que está em conformidade com os requisitos locais. A versão do *LoRaWAN* utilizada é a 1.0.3, garantindo a compatibilidade com o sistema. O modo de ativação escolhido para o dispositivo foi o *OTAA*. Além disso, o dispositivo foi configurado para operar na classe A, possibilitando a comunicação bidirecional entre o dispositivo e a rede *LoRa*. Após isso, bastou configurar corretamente o *AppEUI* e o *AppKey* no firmware do dispositivo para ingressar na rede.

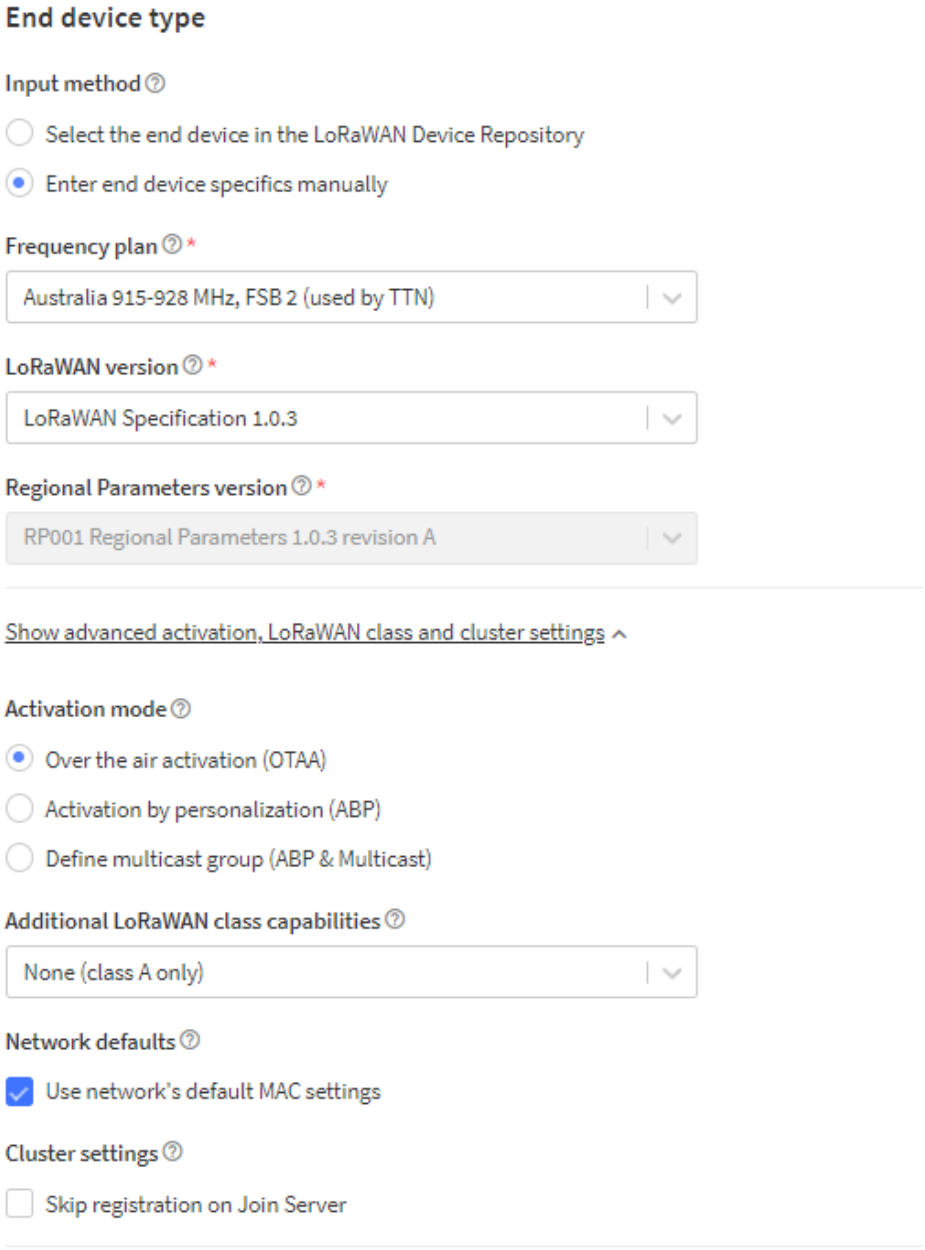
3.1.3 Configuração do Servidor de Aplicação *Node-RED*

O *Node-RED* é uma plataforma de programação visual de código aberto, desenvolvida pela IBM. Ele facilita a visualização de mensagens vindas do protocolo MQTT, através do Servidor de Rede de redes LoRaWAN. Foi escolhido o Node-RED para o servidor de Aplicação por ser uma ferramenta anteriormente utilizada em outros trabalhos do campus IFSC São José, como por exemplo o (BEDAQUE, 2016).

Para a configuração do *Node-RED*, foi preciso iniciar um servidor *Node-RED* em uma maquina virtual, e se conectar ao servidor broker *MQTT* da *TTN*. Após essa conexão estar estabelecida, foi necessário criar um bloco utilizando os componentes disponíveis na interface gráfica do servidor conforme Figura 13.

O primeiro bloco *IFSC* corresponde à conexão com o servidor *TTN*, que irá receber os pacotes. Ela está conectada a um bloco de *json* que transforma o *payload* da mensagem em um *JSON (Java Script Object Notation)*, e o envia para um bloco *debug*, onde podemos ver as mensagens em modo *debug* para entender o formato delas. Após entender como estavam chegando as mensagens, foi possível criar um bloco de 'Function' mostrado Figura 14 onde usamos uma função em *JavaScript* para filtrar os dados do *payload* e enviá-los ao bloco 'Dados' onde um arquivo em formato *JSON* é escrito linha por

Figura 12 – Registro de dispositivo na *TTN*

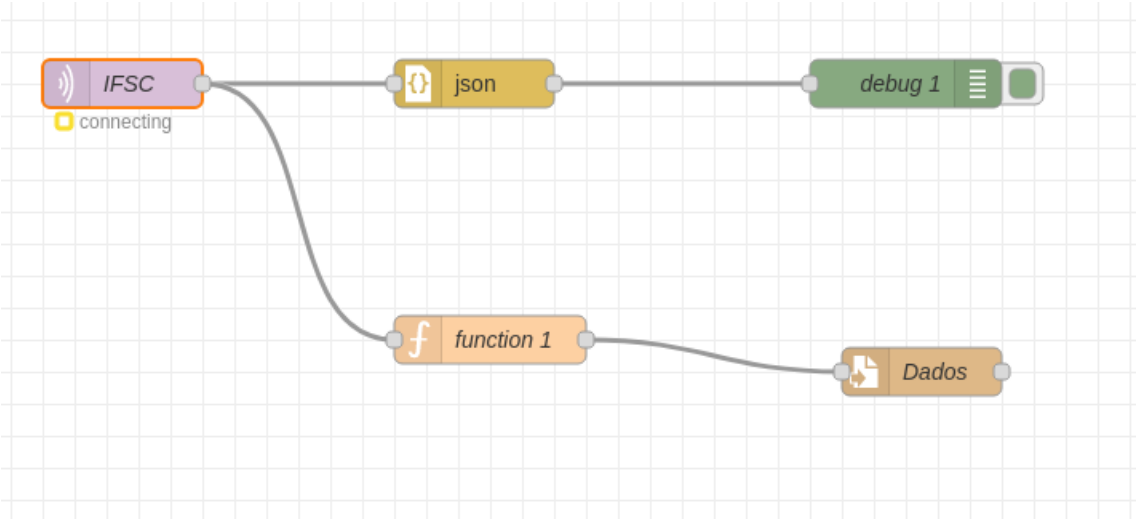


Fonte: Elaborado pelo Autor

linha com todos os pacotes recebidos em tempo real.

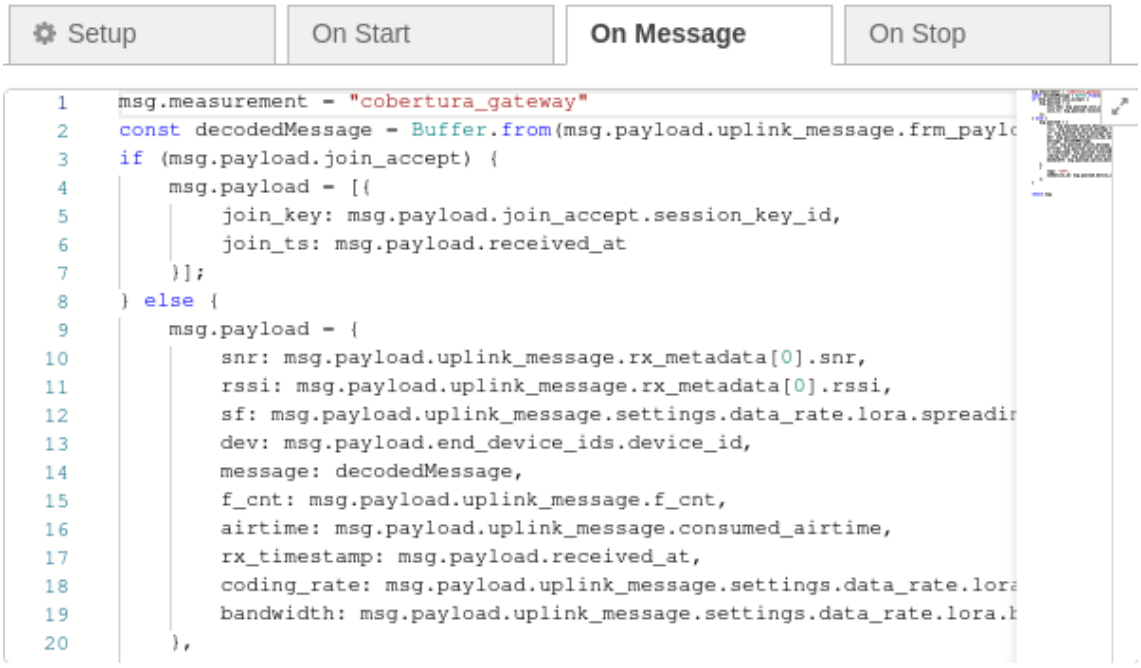
A Figura 14 mostra alguns dos valores recebidos no *payload* do servidor da *TTN*. Tais como *SNR*, *RSSI*, fator de espalhamento (*SF*), ID do dispositivo (devEUI), mensagem do *payload* (message), contador de frames (*f_cnt*), *airtime*, a estampa de tempo dos pacotes recebidos (*rx_timestamp*), a taxa de código (*coding_rate*) e por fim a largura de banda (*bandwidth*). Com esses dados foi possível analisar os valores e desenvolver os cálculos demonstrados no próximo capitulo.

Figura 13 – Esquemático de blocos do *Node-RED*



Fonte: Elaborado pelo Autor

Figura 14 – *Function* do *Node-RED*



Fonte: Elaborado pelo Autor

3.2 Preparação para coleta de estatísticas na simulação *FLoRa*

Após as configurações do sistema para as medições reais, precisamos verificar quais os pontos onde iremos alterar as informações do mecanismo de simulação após parametrizar os modelos no Capítulo 4. Iniciaremos configurando o ambiente de simulação onde alteraremos arquivos para estabelecer as rotinas e os valores dos parametros de diversos fatores, como as configurações de radio e os parametros do modelo de perda de percurso.

Em seguida vamos discutir sobre a coleta de estatisticas de saida do simulador,

explicando quais as estatisticas que foram utilizadas para a visualização dos resultados.

3.2.1 Configuração do ambiente de simulação

As primeiras alterações seriam no arquivo *.ini* da simulação (**??**), onde configuraremos os parâmetros de rádio, tais como potência de transmissão, frequencia da portadora, sensibilidade, largura de banda, fator de espalhamento, ganhos das antenas e taxa de código. Além disso, outras configurações do dispositivo estão vinculadas ao arquivo *.ini*, como posicionamento e quantidade de nodos e *gateways*, tamanho do *payload* da mensagem, tamanho máximo da área de simulação, modelo de propagação, modelos de antenas e consumo de energia. Todos esses fatores influenciam nos resultados da simulação. Infelizmente, a simulação não abrange o eixo Z de um possível plano 3D. Atualmente, usando o *FLoRa*, só é possível executar simulações em 2D.

Tabela 1 – Parâmetros de configuração do *.ini*

Parâmetro	Valor
Fator de Espalhamento(SF)	12
Potência de Transmissão	14dBm
Faixa de frequência	915MHz
Largura de banda	125kHz
Taxa de código	4
ADR	desativado

Os parâmetros dos modelos de perda podem ser ajustados no arquivo .ned. No modelo Hata-Okumura, existem dois parâmetros, K1 e K2 (Código 3.4), que serão explicados no Capítulo 4. Já no modelo de Sombreamento Log-normal, é possível configurar a distância de referência (d0), o expoente de perda (n) e o desvio padrão (σ) da variável aleatória gaussiana (Código 3.5). Alguns parâmetros de configuração do Sombreamento Log-normal, por exemplo, precisaram ser definidos diretamente no código (Figura **??**), pois os arquivos .ned dos modelos não possuem esses parâmetros, como é o caso do PL(d0).

Código 3.4 – Parâmetros Hata-Okumura

```
package flora.LoRaPhy;

import inet.physicallayer.wireless.common.pathloss.FreeSpacePathLoss;

module LoRaHataOkumura extends FreeSpacePathLoss
{
    parameters:
        double K1 = default(120.60);
        double K2 = default(32.43);
        @class(LoRaHataOkumura);
}
```

Código 3.5 – Parâmetros Sombreamento Log-Normal

```
package flora.LoRaPhy;

import inet.physicallayer.wireless.common.pathloss.FreeSpacePathLoss;

module LoRaLogNormalShadowing extends FreeSpacePathLoss
{
    parameters:
        double d0 = 1260m @unit(m);
        double gamma = 3.6;
        double sigma = 4.78;
        @class(LoRaLogNormalShadowing);
}
```

Código 3.6 – Parâmetros de perda de referencia PL(d0)

```
double LoRaLogNormalShadowing::computePathLoss(mps propagationSpeed, Hz frequency,
    m distance) const
{
    // parameters taken from paper "Do LoRa Low-Power Wide-Area Networks Scale?"
    double PL_d0_db = 103;
    double PL_db = PL_d0_db + 10 * gamma * log10(unit(distance / d0).get()) +
    normal(0.0, sigma);
    return math::dB2fraction(-PL_db);
}

m LoRaLogNormalShadowing::computeRange(W transmissionPower) const
{
    // parameters taken from paper "Do LoRa Low-Power Wide-Area Networks Scale?"
    double PL_d0_db = 103;
    double max_sensitivity = -117;
    double trans_power_db = round(10 * log10(transmissionPower.get()*1000));
    EV << "LoRaLogNormalShadowing transmissionPower in W = " << transmissionPower
    << " in dBm = " << trans_power_db << endl;
    double rhs = (trans_power_db - PL_d0_db - max_sensitivity)/(10 * gamma);
    double distance = d0.get() * pow(10, rhs);
    return m(distance);
}
```

Para se adequarem ao escopo do estudo deste trabalho, foi necessário realizar alterações nessas configurações, como por exemplo na Código 3.6, o valor padrão de PL(d0) foi alterado para 103.

Outro fator que precisou ser colocado diretamente no código Código 3.7 foi a sensibilidade da rede, foi necessário alterar os valores de sensibilidade que se adequassem ao menor valor de *RSSI* visualizado nas medições reais, de -117dBm.

Código 3.7 – Parâmetros de sensibilidade do receptor

```
1 if(reception->getLoRaSF() == 12)
2     {
3         if(reception->getLoRaBW() == Hz(125000)) sensitivity = W(math::dBmW2mW
    (-117) / 1000);
4         if(reception->getLoRaBW() == Hz(250000)) sensitivity = W(math::dBmW2mW
    (-116) / 1000);
5         if(reception->getLoRaBW() == Hz(500000)) sensitivity = W(math::dBmW2mW
    (-109) / 1000);
6     }
```

3.2.2 Coleta de estatísticas no Omnet/ *FLoRa*

O *FLoRa* implementa alguns códigos que permitem coletar estatísticas referentes à rede simulada dentro do simulador Omnet. Essas estatísticas abrangem principalmente a parte física da rede, juntamente com algumas estatísticas da aplicação, como a quantidade de pacotes enviados e recebidos. Uma das estatísticas observadas é a média de *RSSI* mostrada na Figura 15. Essa média é calculada a partir de um vetor contendo todos os valores de *RSSI* dos pacotes recebidos durante a simulação (*Count*). Além disso, são determinados o desvio padrão (*StdDev*) e a variância amostral (*Variance*) para permitir uma análise comparativa com as medições reais.

Figura 15 – Estatisticas *FLoRa* - Média *RSSI*

omnetpp.ini | LoRaLogNormalShadowing.cc | results.anf | LoRaReceiver.cc

Browse Data

Here you can see all data that come from the files specified in the Inputs page.

All (119,657 / 119,657) Parameters (77,520 / 77,520) Scalars (32,220 / 32,220) Histograms (2,700 / 2,700) Vectors (60 / 7,217)

experim... | measurement filter | module filter | or of RSSI per node

Module	Name	Count	Mean	StdDev	Variance
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-91.215527	6.602849	43.597611
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-91.642546	6.025033	36.301026
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-91.629685	6.953007	48.344311
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	20	-92.783091	5.900954	34.821257
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	20	-92.296647	6.244581	38.994794
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-93.762885	6.708740	45.007195
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-89.830284	6.636822	44.047411
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-90.518507	5.526089	30.537662
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	20	-93.388177	5.732864	32.865734
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-90.584115	6.471034	41.874281
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-94.378007	5.632793	31.728354
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-96.491313	5.413732	29.308492
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-94.124689	5.392132	29.075085
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	20	-95.890230	4.970998	24.710820
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-99.199658	4.948090	24.483592
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-99.340608	5.402362	29.185513
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	21	-97.831819	5.535111	30.637451
LoRaNetworkTest.networkServer.app[0]	Vector of RSSI per node	20	-96.715514	4.424979	19.580443

Fonte: Elaborado pelo Autor

Outra estatística que a simulação fornece para compreender o desempenho da rede é a *DER*, que se refere à taxa de pacotes enviados e recebidos pela rede. Com a *DER*

podemos ter uma estimativa da perda de pacotes que estamos tendo na rede simulada, assim podendo provar se a rede é viável ou não pela proposta apresentada. No *FLoRa*, podemos extrair esses dados para cada parâmetro de fator de espalhamento configurado nos nodos, conforme mostrado na Figura 16. Pode-se observar que, na simulação em questão, foram configurados apenas nodos com fator de espalhamento 12, portanto os outros valores estão em zero.

Figura 16 – Estatisticas *FLoRa* - DER

omnetpp.ini | LoRaLogNormalShadowing.cc | results.anf | LoRaReceiver.cc

Browse Data

Here you can see all data that come from the files specified in the Inputs page.

All (119,657 / 119,657) Parameters (77,520 / 77,520) Scalars (420 / 32,220) Histograms (2,700 / 2,700) Vectors (60 / 7,217)

experim... | measurement filter | module filter | DER *

ent	Measurement	Replicati	Module	Name	Value
	$0="avg", $1=2500	#0	LoRaNetworkTest.networkServer.app[0]	DER SF11	0
	$0="avg", $1=2500	#0	LoRaNetworkTest.networkServer.app[0]	DER SF12	0.947368
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF7	0
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF8	0
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF9	0
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF10	0
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF11	0
	$0="avg", $1=2500	#1	LoRaNetworkTest.networkServer.app[0]	DER SF12	0.894737
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF7	0
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF8	0
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF9	0
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF10	0
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF11	0
	$0="avg", $1=2500	#2	LoRaNetworkTest.networkServer.app[0]	DER SF12	0.950000
	$0="avg", $1=2500	#3	LoRaNetworkTest.networkServer.app[0]	DER SF7	0
	$0="avg", $1=2500	#3	LoRaNetworkTest.networkServer.app[0]	DER SF8	0
	$0="avg", $1=2500	#3	LoRaNetworkTest.networkServer.app[0]	DER SF9	0
	$0="avg", $1=2500	#3	LoRaNetworkTest.networkServer.app[0]	DER SF10	0

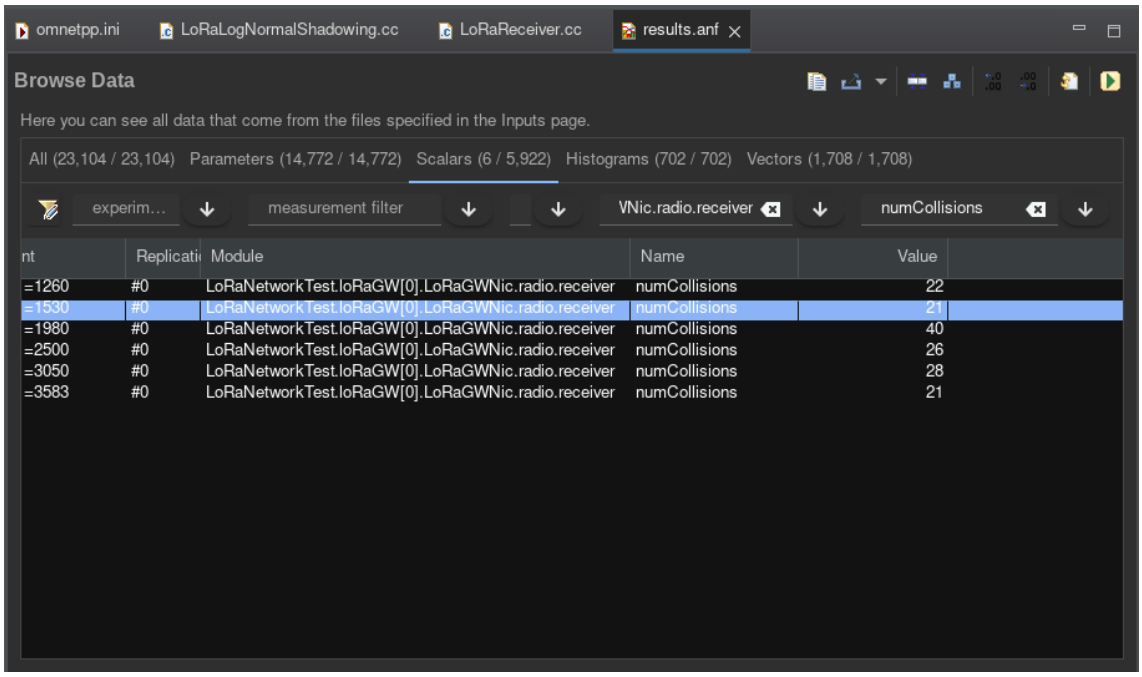
Fonte: Elaborado pelo Autor

As colisões de pacote é algo importante a se ressaltar e o simulador também nos permite visualizar, o cálculo de como as colisões se comportam no simulador é algo derivado do módulo *INET* e segue a mesma linha da explicação do fim da seção 2.2. O *FLoRa* nos mostra a quantidade de colisões (*numCollision* na Figura 17) que ocorreram durante os pacotes enviados na simulação, com isso podemos ter uma noção de mais um fator que deixaria a DER com valores mais baixos.

A Figura 17 apresenta estatísticas de colisão obtidas em uma simulação com dez nodos nos pontos de coleta. Essas estatísticas indicam o número de colisões observadas para diferentes distâncias entre os nodos e o *gateway*.

Os valores exibidos para cada execução nas simulações, como o valor 2500 na Figura 16, correspondem a uma etiqueta específica de distância configurada no arquivo *.ini*, conforme explicado na subseção 2.3.2. Observa-se que essas execuções foram repetidas várias vezes com a mesma distância para permitir a variação das sementes na variável aleatória mencionada na Equação 2.1. Dessa forma, é possível visualizar um valor de saída mais confiável e consistente.

Figura 17 – Estatisticas *FLoRa* - Colisões



Fonte: Elaborado pelo Autor

Após todas as configurações anteriormente citadas, é possível construir um cenário de simulação utilizando o cálculo dos parâmetros dos modelos de perda e configurando os parâmetros da simulação. Além da construção do cenário, é possível visualizar as estatísticas do resultado com mais precisão por meio das flags do *OMNeT++* para as múltiplas execuções de um experimento.

4 AVALIAÇÃO DE DESEMPENHO POR SIMULAÇÃO DE CENÁRIOS BASEADOS EM UMA REDE LORA OUTDOOR

Neste capitulo serão apresentado inicialmente os experimentos para a coleta de dados do cenário físico, para preparar os dados para a parametrização dos modelos de perda de percurso Sombreamento de Log-Normal e Hata-Okumura, demonstrando os calculos para achar o coeficiente de perda e o desvio padrao da variavel aleatoria gaussiana, assim como os parametros K1 e K2 do modelo Hata Okumura.

Com o objetivo de validar e comparar os resultados obtidos com base nos modelos estudados, neste capítulo, iremos descrever detalhadamente as etapas e procedimentos adotados para coletar os dados de campo no ambiente de interesse. Além disso, iremos apresentar a implementação das simulações utilizando os coeficientes calibrados dos modelos de Sombreamento de Log-Normal e Hata-Okumura. Isso nos permitirá avaliar o desempenho da rede em condições reais e simuladas, permitindo uma análise comparativa entre os resultados.

4.1 Coleta de dados para parametrização dos modelos de propagação

Antes de prosseguirmos com a análise comparativa, foi necessário realizar uma coleta de dados para obter informações relevantes sobre o comportamento da rede LoRa. Inicialmente, foi coletado dados de *Received Signal Strength Indication (RSSI)* e *Signal-Noise Ratio (SNR)* em diversos pontos da rede. Essas métricas são essenciais para compreender a qualidade do sinal recebido pelos dispositivos na rede.

4.1.1 Coleta de dados para parametrização outdoor

Para o desenvolvimento de um modelo de perda de caminho em uma rede LoRa na região outdoor, foi realizado um experimento de coleta de dados na beira-mar de São José. Esse ambiente, próximo ao campus do IFSC São José, foi selecionado devido à sua relevância para a aplicação prática da tecnologia LoRa em cenários ao ar livre. A escolha da beira-mar de São José como local para coleta de dados se baseou em diversos fatores. Primeiramente, a proximidade com o campus do IFSC São José facilitou o acesso à área de estudo, permitindo a realização de medições precisas e repetíveis. Além disso,

a beira-mar é onde foi possível conseguir pontos com visada para o *gateway*, posicionado na caixa d'água do campus a 30 metros de altura.

Tabela 2 – Parâmetros de configuração do nodo

Parâmetro	Valor
Fator de Espalhamento(SF)	12
Potência de Transmissão	14 dBm
Faixa de frequência	915 a 928MHz
Largura de banda	125kHz
Classe	A
Modo de autenticação	OTAA
Taxa de código	4/5

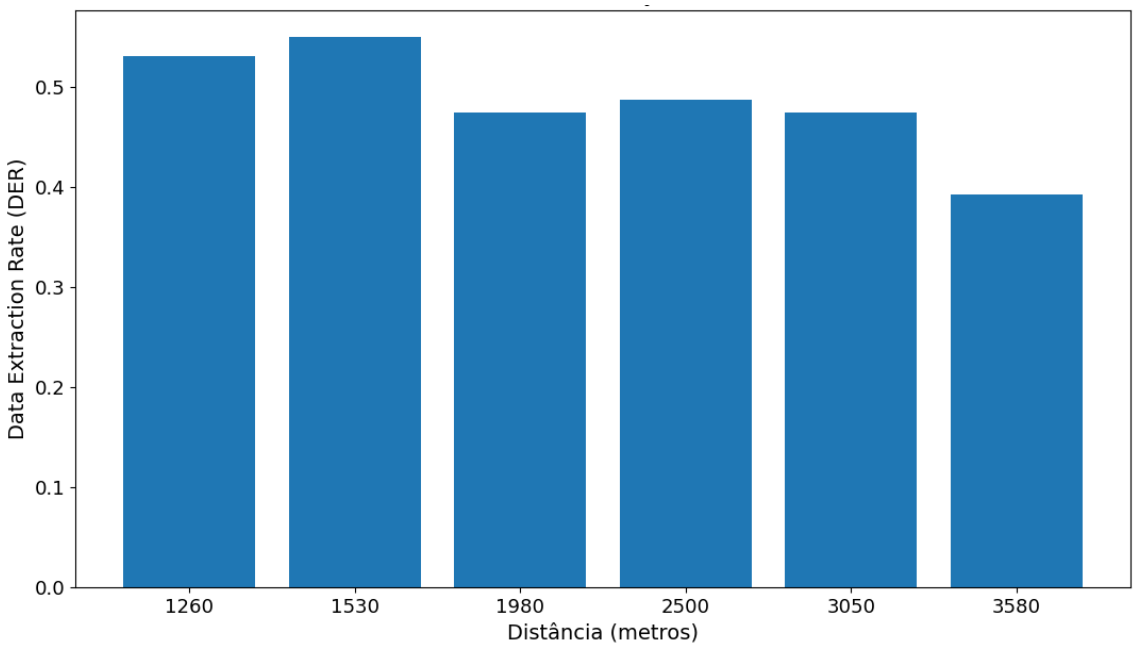
A Tabela 2 mostra as configurações nodo utilizadas, cuidadosamente selecionadas para o experimento. O fator de espalhamento de 12 permite detectar o menor valor possível de RSSI, garantindo maior precisão na medição da intensidade do sinal recebido. A largura de banda de 125kHz otimiza o uso do espectro de frequência disponível, proporcionando comunicação eficiente e confiável. A potência de transmissão de 14 dBm foi definida para alcançar a maior distância possível. As antenas possuem ganhos de 2,1 dBi (transmissora) e 5 dBi (receptora), aumentando a intensidade do sinal transmitido e recebido. A taxa de código de 4/5 garante boa eficiência na transmissão de dados, equilibrando taxa de transferência e robustez contra erros. A função de ADR foi desativada para permitir controle manual da taxa de dados, adequado para o cenário de aplicação específico. Essas configurações proporcionam um ambiente adequado para medições precisas e coleta eficiente de dados para a parametrização dos modelos de propagação.

Durante o experimento, foram selecionados pontos estratégicos na beira-mar de São José, garantindo uma cobertura abrangente e representativa do ambiente de interesse. Em cada ponto de medição, foram instalados dispositivos LoRaWAN configurados para coletar dados relevantes de recepção de potencia do sinal, como o *RSSI* e o *SNR*.

A coleta de dados foi realizada em diferentes horários ao longo do dia, com o objetivo de capturar variações nas condições de propagação do sinal LoRa. Em cada ponto de coleta, foram registrados, em média, os dados de 20 transmissões do nodo para o gateway, permitindo calcular a média do RSSI recebido. Além disso, o firmware do nodo foi configurado para enviar um determinado número de mensagens ao longo do tempo, o que possibilitou calcular a *DER* das transmissões, conforme ilustrado na Figura 18.

A semelhança nas taxas de DER entre os diferentes pontos ocorre devido à ausência teórica de perdas nessas localidades, conforme ilustrado na Figura 3 do estudo realizado por (LINKA et al., 2018). A mencionada figura evidencia que perdas significativas somente seriam esperadas a partir de uma distância de 3000 metros em relação ao receptor. A explicação para essa similaridade nas taxas de DER reside no fato de que, devido à simultaneidade de utilização do mesmo gateway por outros nodos, podem ter

Figura 18 – Taxa de Extração de Dados (DER) dos dados coletados



Fonte: Elaborado pelo Autor

ocorrido colisões de pacotes, impactando negativamente as taxas de DER e resultando em valores na faixa de 0,5 a 0,6, em vez de apresentarem valores próximos a 1. Além disso em outras ocasiões as recepções de pacotes na mesma região foram medidas por outros trabalhos como (BEDAQUE, 2016), demonstrando valores de recepção próxima ou igual a 100%.

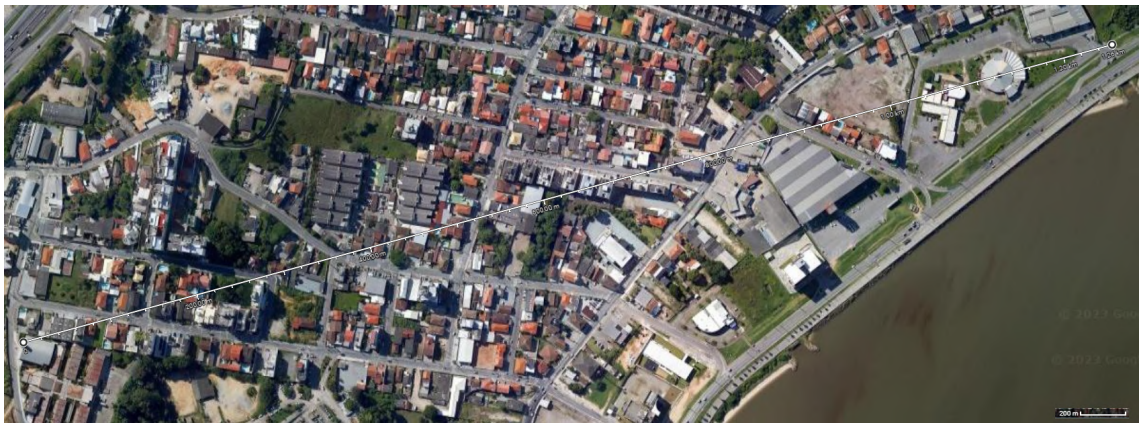
Foram consideradas diferentes distâncias entre os dispositivos de transmissão e recepção para abranger uma variedade de cenários de comunicação. Nesse experimento, as distancias coletadas com visada foram : 1260 metros (Figura 19a), 1530 metros (Figura 19b), 1980 metros (Figura 20a), 2500 metros (Figura 20b), 3000 metros (Figura 21a), 3580 metros (Figura 21b).

Tabela 3 – Distancias coletadas, media RSSI e coordenadas

Distancia(m)	Coordenadas (X,Y)	Media RSSI(dBm)
1260	(-27.605029, -48.621421)	-89
1530	(-27.604181, -48.618796)	-88.95
1980	(-27.603130, -48.613895)	-92.76
2500	(-27.603714, -48.616478)	-94
3000	(-27.601985, -48.604001)	-99.31
3580	(-27.611150, -48.597553)	-113.81

Dessa forma conseguimos construir um gráfico de decaimento do RSSI em relação a distância na Figura 22. Facilitando assim a visualização da perda de caminho do sinal em relação a distância. Foram coletados dados de outras distâncias (200 metros, 400 metros, 750 metros, 1760 metros, 1900 metros, 4000 metros), porém as outras são posições com muitos obstáculos e geraram dados inconsistentes para uma curva para a parametrização

Figura 19 – Fotos de satélite dos pontos coletados - parte 1



(a) 1260m

(b) 1530m

Fonte: Google Maps

de um modelo, com valores demonstrando a variação entre pontos que tem pouca visada e muita visada para o *gateway*.

Mesmo configurando o fator de espalhamento para 12 e a potência do dispositivo para o máximo (14 dBm), o menor valor de RSSI encontrado foi de -117 dBm. Isso tende a acontecer porque o ambiente urbano possui muitas interferências, ruídos, sombreamento e multipercurso. Segundo (SANTOS, 2019, p. 71) é natural que isso aconteça, uma vez que o fabricante geralmente realiza testes em condições de laboratório controladas, o que pode influenciar nas medições de sensibilidade.

Para visualizar a variação dos valores de RSSI em relação à distância entre os dispositivos, foi construído um *boxplot* (gráfico de caixa) (Figura 23). O *boxplot* permite analisar a distribuição dos valores de RSSI, destacando medidas estatísticas importantes e possíveis anomalias.

Além disso, também foi plotado um boxplot (Figura 24) separado para os valores

Figura 20 – Fotos de satélite dos pontos coletados - parte 2

(a) 1980m

(b) 2500m

Fonte: Google Maps

de *SNR* em cada ponto de medição. Isso ajudou a entender a relação entre a relação sinal-ruído e a distância entre os dispositivos, fornecendo uma visão geral do comportamento do sinal na rede LoRa física.

4.1.2 Parametrização do Sombreamento Log-Normal

Com base nos dados coletados e utilizando técnicas matemáticas como demonstrado no exemplo 4.9 do (RAPPAPORT, 2009, p. 93), é possível calcular o expoente de perda do sombreamento de log normal(subseção 2.4.1).

Para achar as estimativas $\hat{p}_i$ em função de n temos:

$$\hat{p}_i = -89 - 10 \cdot n \cdot \log\left(\frac{d_n}{1260}\right) \tag{4.1}$$

Figura 21 – Fotos de satélite dos pontos coletados - parte 2

(a) 3000m

(b) 3580m

Fonte: Google Maps

As somas dos erros ao quadrado:

$$\begin{aligned} J(i) = (-89 - (\hat{p}_1))^2 + (-88,95 - (\hat{p}_2))^2 + (-92,76 - (\hat{p}_3))^2 + \\ (-94 - (\hat{p}_4))^2 + (-99,31 - (\hat{p}_5))^2 + (-113,81 - (\hat{p}_6))^2 \end{aligned} \tag{4.2}$$

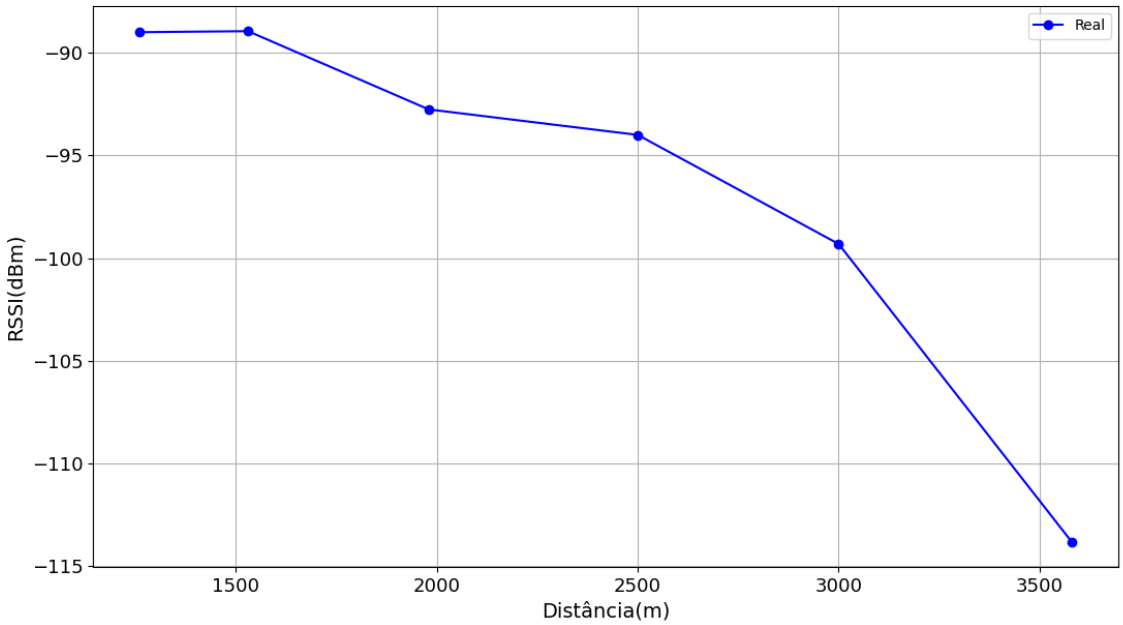
Substituindo todos os valores e simplificando a equação e aplicando a derivada teremos:

$$\frac{dJ(n)}{dn} = 96,36n - 347,15 \tag{4.3}$$

Igualando a 0 e isolando o n teremos:

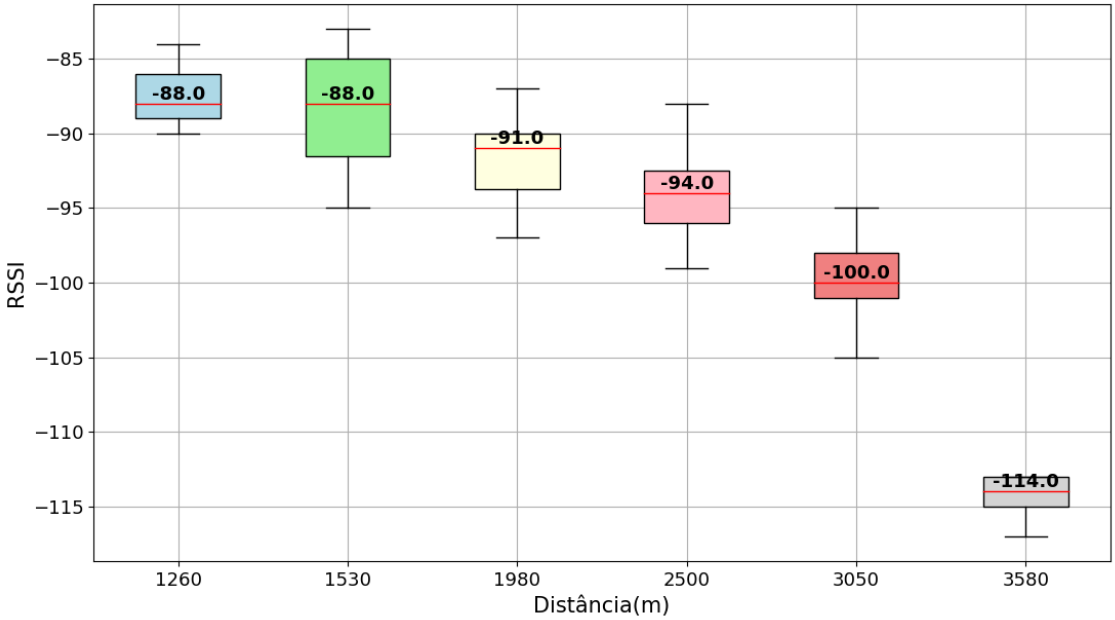
$$n = 3,6 \tag{4.4}$$

Figura 22 – Gráfico RSSI x Distância



Fonte: Elaborado pelo Autor

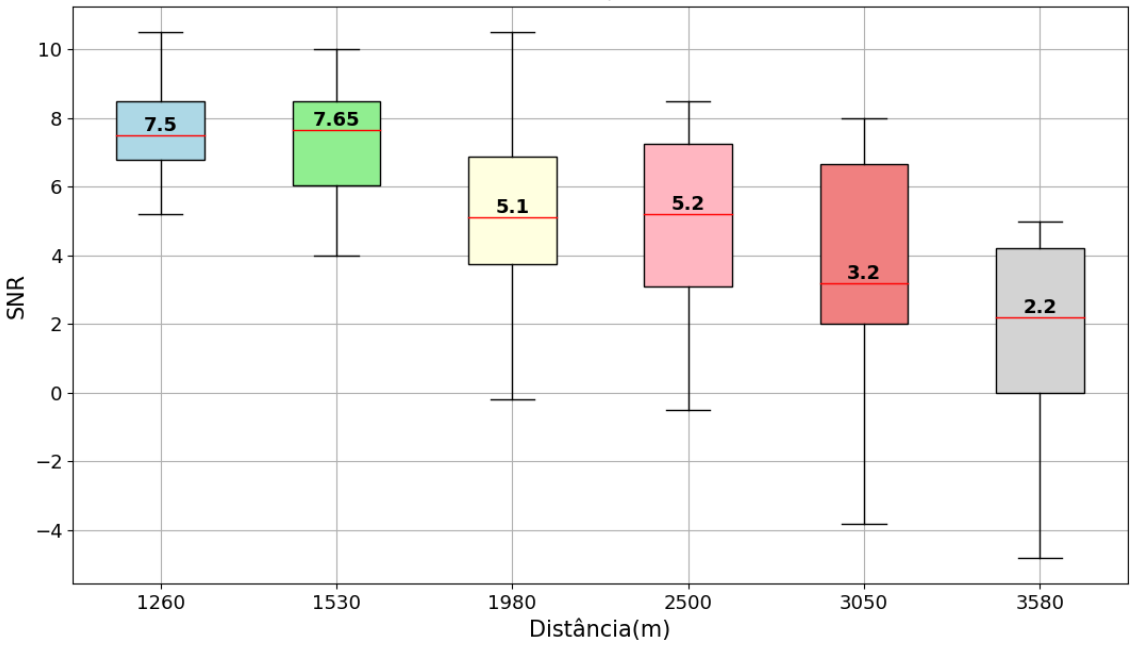
Figura 23 – Gráfico Box RSSI x Distância



Fonte: Elaborado pelo Autor

Esse é o valor de n calculado para o ambiente que foi medido na área da beira-mar próximo ao campus do IFSC São José. Outra medição foi feita por outro trabalho (BEDAQUE, 2016), porém as medições foram feitas com um gateway posicionado em outra localização, o que possibilitou leituras com visada direta. Isso não foi possível neste estudo, resultando em um expoente maior, que se adequa ao expoente de rádio-celular urbano sombreado da tabela 4.2 do livro de Rappaport (2009) (RAPPAPORT, 2009). Esse expoente calculado (3,6) será utilizado nas simulações posteriormente.

Figura 24 – Gráfico Box de SNR x Distância



Fonte: Elaborado pelo Autor

Continuando o cálculo, precisamos achar a variável aleatória Xσ, para isso precisamos utilizar a variância amostral.

$$\sigma^2 = J(n)/6 \tag{4.5}$$

Onde 6 é o numero de amostras e *J(3,57)* é :

$$J(3.6) = 137,38 \tag{4.6}$$

Agora calculando a variância amostral e em seguida achando o desvio padrão para a variável aleatória gaussiana de media 0:

$$\sigma^2 = 137,38/6 = 22,89 \tag{4.7}$$

$$\sigma = \sqrt{37,97} = 4,78 \tag{4.8}$$

Então com o valor do desvio padrão conseguimos adicionar uma variável aleatória gaussiana com média zero e desvio padrão de 4,78 para a equação de perda de caminho de sombreamento de log-normal. Substituindo na equação:

$$PL(d) = -89 + 10 \cdot 3,6 \cdot \log_{10}(\frac{d}{1260}) + X\sigma \sim N(0,4,78) \tag{4.9}$$

Ao achar esses parâmetros, conseguimos adicionar aos cálculos da simulação uma perda de percurso parecida com a achada nas medições reais, com isso será possível simular um ambiente que se aproxima do real e será possível analisar a rede com parâmetros diversos como por exemplo uma alta densidade de nodos.

4.1.3 Parametrização do Hata-Okumura

Para a parametrização da fórmula do modelo Hata-Okumura, utilizamos a variante para ambientes urbanos descrita na equação 5 do capítulo 3, seção 3.2 do artigo de referência (GRIVA et al., 2023). Essa equação representa a mediana do sinal em perdas, foi escolhida devido ao fato de o simulador já estar pré-configurado para recebê-la e também por se adequar melhor ao ambiente em que estamos realizando os cálculos. A variante para ambiente urbano é expressa pela equação:

$$L_{urbano} = 69.5+26.16\cdot\log(fc)-13.82\cdot\log(hb)-a(hm)+(44.9-6.55\cdot\log(hb))\cdot\log(R) \quad (4.10)$$

Na equação 4.10 especificamente para o LoRa, *fc* representa a frequência de transmissão, *hb* é a altura da antena do *gateway* , *hm* é a altura da antena do nodo e R é a distância entre os dois dispositivos em Km. A constante *a(hm)* é o fator de correção para a altura da antena móvel, essa constante é determinada pela equação 4.11 em especifico que também é fornecida pelo artigo (GRIVA et al., 2023), equação 6 do capitulo 3, seção 3.2.

$$a(hm) = (1.11 \cdot \log(fc) - 0.7) \cdot hm - (1.56 \cdot \log(fc) - 0.8) \quad (4.11)$$

A utilização da variante L_{urbano} nos permite uma melhor modelagem do ambiente real do experimento, considerando os fatores relevantes para a perda de propagação do sinal. Dessa forma, podemos obter resultados mais precisos e representativos da performance do sistema de comunicação sem fio no contexto urbano. Além disso, também utilizamos uma variação para cidades de pequeno e médio porte, demonstrada na Equação 4.11, para tentar chegar a algo mais parecido com o cenário real.

Para adequar essa formula a simulação, precisamos calcular a equação em partes e dividi-la em dois para que possamos adicionar aos dois coeficientes K1 e K2 que são esperados:

$$K_1 = 69.5 + 26.16 \cdot \log(fc) - 13.82 \cdot \log(hb) - a(hm) \quad (4.12)$$

$$K_2 = 44.9 - 6.55 \cdot \log(hb) \quad (4.13)$$

Utilizando os parametros para o calculo fc = 915MHz, hm = 80m(por estar em cima de uma torre de 30m no topo de uma colina onde se encontra o campus e as medições foram feitas no nivel do mar), para calcular cada coeficiente temos:

$$K_1 = 120.60; K_2 = 32.43 \tag{4.14}$$

Assim a equação na simulação do Hata-Okumura ficará expressa por:

$$PL(dB) = 120.60 + 32.43 \cdot \log(R) \tag{4.15}$$

Após calcularmos os parâmetros do modelo de propagação Hata-Okumura e do Sombreamento Log-Normal, podemos avançar para a etapa de simulação na seção 4.2. Esses cálculos e considerações são fundamentais para obtermos uma compreensão mais precisa da propagação de sinal em no cenário específico. Com base nesses resultados, poderemos realizar simulações que refletirão de maneira mais fiel as condições reais de transmissão e recepção. Essa próxima etapa nos permitirá avaliar e analisar o desempenho do sistema em termos de taxa de erro de pacote, cobertura e escalabilidade.

4.2 Simulações outdoor de alta densidade de nodos utilizando os modelos calculados

Agora que já possuímos os coeficientes de perda anteriormente calculados na seção 4.1, podemos iniciar a construção de uma simulação com uma alta densidade de nodos, a fim de entender como funcionaria a escalabilidade de uma rede LoRaWAN implementadas no ambiente dos arredores do IFSC Campus São José, simulando uma cidade inteligente, ou outra aplicação de IoT desejada.

4.2.1 Experimento para seleção do meio de propagação a ser usado

Nessa seção faremos um comparativo entre os valores de perda reais e os simulados, demonstraremos quais as diferenças entre ter apenas um gateway e diversos, verificaremos a DER e a distancia máxima sem perdas utilizando os modelos calculados.

Primeiramente configuramos a simulação com apenas um nodo para verificar as curvas de perda dos modelos em relação a curva da média *RSSI* medida no real, para isso foi necessário alterar o *.ini* da simulação *FloRa* para colocar o nodo nas mesmas posições das medições reais.

Após configurar a simulação no Omnet a partir dos dados calculados, podemos executar simulações com somente um nodo e varias as distancias para ver os valores médios

de RSSI coletados em cada distancia. A Tabela 4 mostra os valores do sombreamento log normal e a Tabela 5 mostra os valores do Hata-Okumura.

Tabela 4 – Valores simulados do modelo Log-Normal

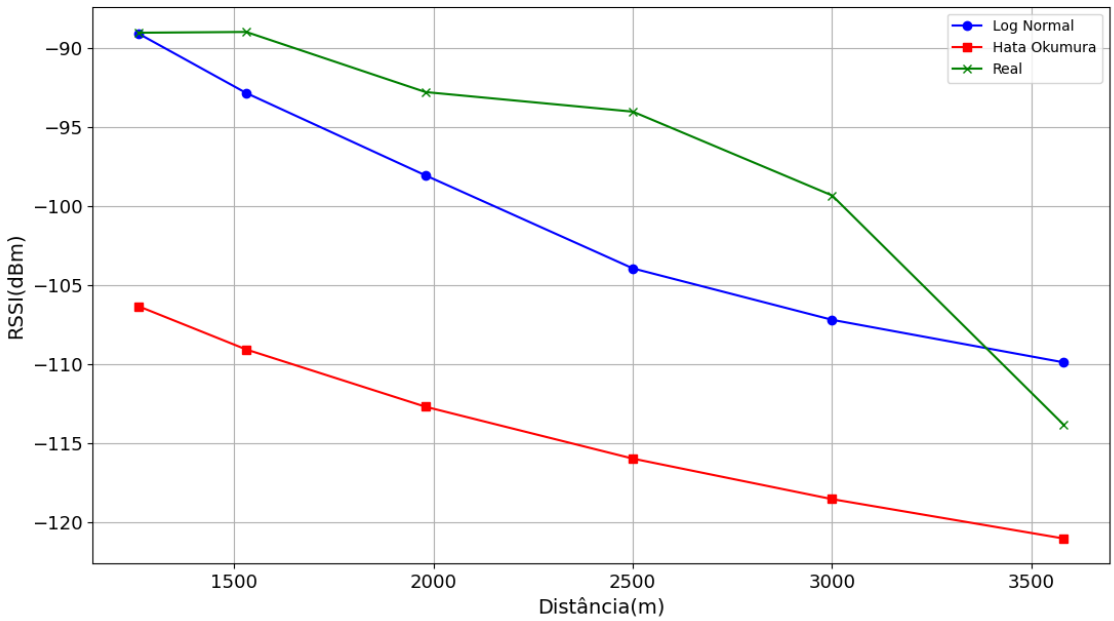
Distância (m)	RSSI (dBm)
1260	-89.06
1530	-92.82
1980	-98.04
2500	-103.93
3000	-107.19
3580	-109.88

Tabela 5 – Valores simulados do modelo Hata-Okumura

Distância (m)	PL (dBm)
1260	-106.33
1530	-109.06
1980	-112.69
2500	-115.98
3000	-118.55
3580	-121.04

Podemos verificar a curva de decaimento de RSSI dos modelos HataOkumura e Sombreamento Log-Normal pela Figura 25. Para melhorar a precisão dos valores calculados utilizamos a sensibilidade do receptor de 133.25dBm para SF 12, dados capturados do artigo (BOR et al., 2016a, p. 62).

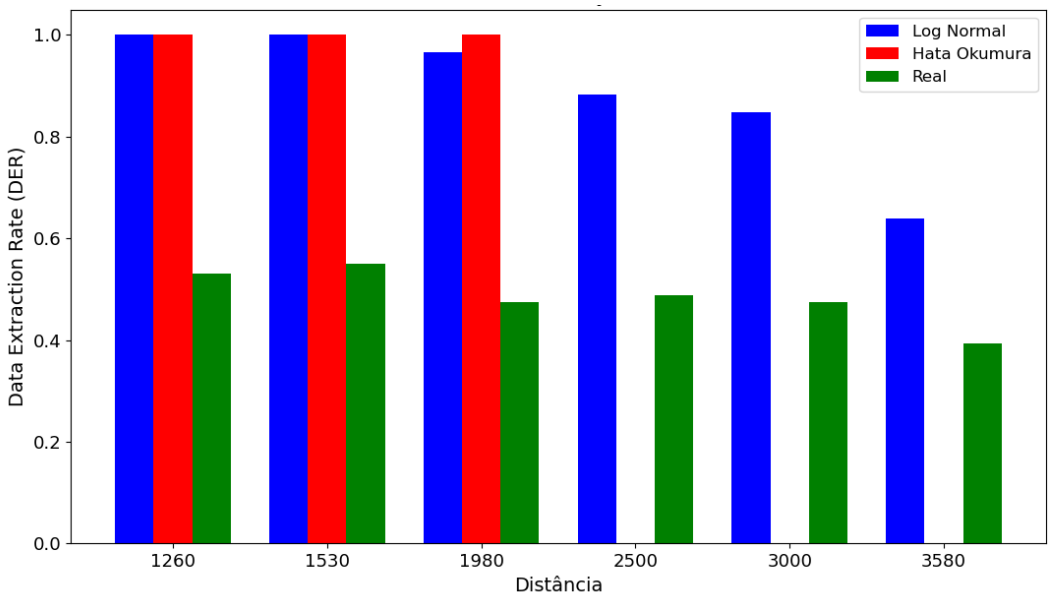
Figura 25 – Gráfico RSSI dos modelos simulados X real



Fonte: Elaborado pelo Autor

Além disso, também fizemos uma estimativa de taxa de extração de dados (*DER*) para cada distancia utilizando Log-Normal, representada pela Figura 26.

Figura 26 – DER- Modelos x Real



Fonte: Elaborado pelo Autor

De acordo com a Figura 26 a distância máxima para o modelo Hata-Okumura ficaria entre 1980 e 2500 metros. No entanto, para o modelo de Sombreamento Log-Normal, verificamos que a DER começa a cair significamente depois dos 3000 metros de distancia. Dessa forma, nas próximas simulações, optaremos pelo modelo de Sombreamento Log-Normal, pois ele se aproxima mais da curva real de perdas, tornando os experimentos mais fidedignos ao ambiente real.

Foi observado também que a curva do modelo Hata-Okumura ficou um pouco abaixo nos valores de RSSI, talvez porque a região não se configuraria como cidade de medio porte e sim um ambiente suburbano, porém, para adequá-lo à simulação teriamos que modificar o calculo do modelo no código fonte, ja que precisariamos de mais um coeficiente, e a simulação só permite o cálculo utilizando K1 e K2.

4.2.2 Alta densidade de nodos em uma região próxima ao Campus simulando uma Cidade Inteligente

Nesta subseção, conduziremos simulações que envolvem uma alta densidade de nós distribuídos em uma região próxima ao campus, com o objetivo de simular um ambiente semelhante ao de uma cidade inteligente. Os nodos serão espalhados em distâncias de 100 metros a 3000 metros ao redor do gateway, a fim de verificar algumas das métricas anteriormente destacadas, como a DER do sistema e as colisões de pacotes, com o intuito de analisar a viabilidade da região para as aplicações IoT.

Após a análise dos estudos apresentados no artigo (LINKA et al., 2018), constatou-se que à medida que um nodo se aproxima da distância de 3000 metros do gateway,

ocorrem perdas significativas de pacotes, evidenciadas por leituras de valores muito baixos de RSSI. É importante destacar que, com base nas medições físicas realizadas, o menor valor de RSSI observado foi de -117 dBm, o que indica que a distância para a ocorrência de perdas de pacotes é menor do que os 3000 metros mencionados anteriormente.

No contexto citado acima foram feitos experimentos variando o numero de nodos em numeros de 50, 200, 500, 1000. Em um primeiro experimento foi optado por utilizar a variação o numero de gateways de 1 a 8 para observar as colisões e a *DER* quando temos diferentes valores de intervalo de transmissão de pacotes.

Os nodos foram posicionados de forma randomica, utilizando a função de distribuição uniforme(*uniform do OMNet++*), com o máximo de 2500 metros para x e para y, e os gateways foram posicionados de forma simétrica, conforme Tabela 6. Na simulação não há nenhum obstaculo que possa fazer com que o sinal tenha perdas além dos parametros do modelo. Assim o posicionamento dos gateways só irá interferir em qual deles a informação chegará, sem afetar nos dados de DER.

Tabela 6 – Posicionamento dos gateways

Gateway	X	Y
1	500m	250m
2	1000m	250m
3	1500m	250m
4	2000m	250m
5	500m	1250m
6	1000m	1250m
7	1500m	1250m
8	2000m	1250m

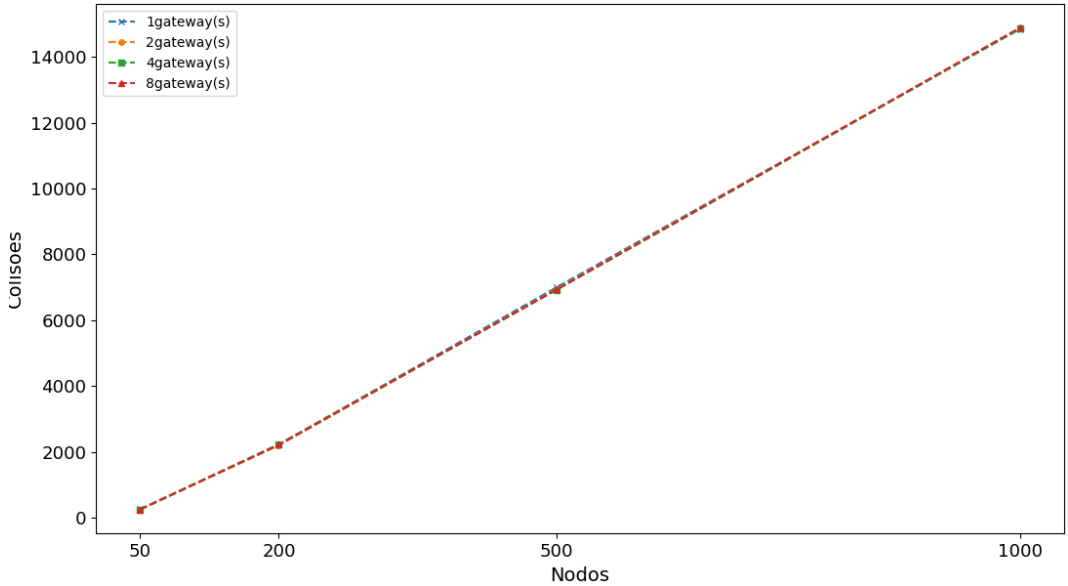
Os experimentos foram separados em intervalo de envio de mensagens pelos nodos: 60, 300 e 500 segundos. A partir disso foi observado que a quantidade de gateways não influencia as colisões de pacotes mesmo quando muda-se a quantidade de mensagens enviadas pelos nodos, pelos intervalos de 60(Figura 27), 300(Figura 28) e 500(Figura 29) segundos.

Com a DER o caso muda um pouco, mostrando que nos três casos a entrega de pacotes entre 2 gateways e 4 gateways se aproxima, mesmo com o aumento do numero de nodos e o maior numero de pacotes enviados. Podemos observar também que quando o intervalo de mensagem é de 60s as curvas de DER(Figura 30) tendem a ser mais abruptas pela maior quantidade de pacotes enviados.

A configuração para 2 gateways, além de se aproximar da curva de 4 gateways e ficar muito próxima da DER de 4 gateways para 50 nodos nos 3 casos, mostra que, no caso em que temos 300s (Figura 31) de intervalo de mensagens, a DER mostrou-se superior, indicando que mesmo com o dobro de gateways, a entrega de pacotes pode ser inferior.

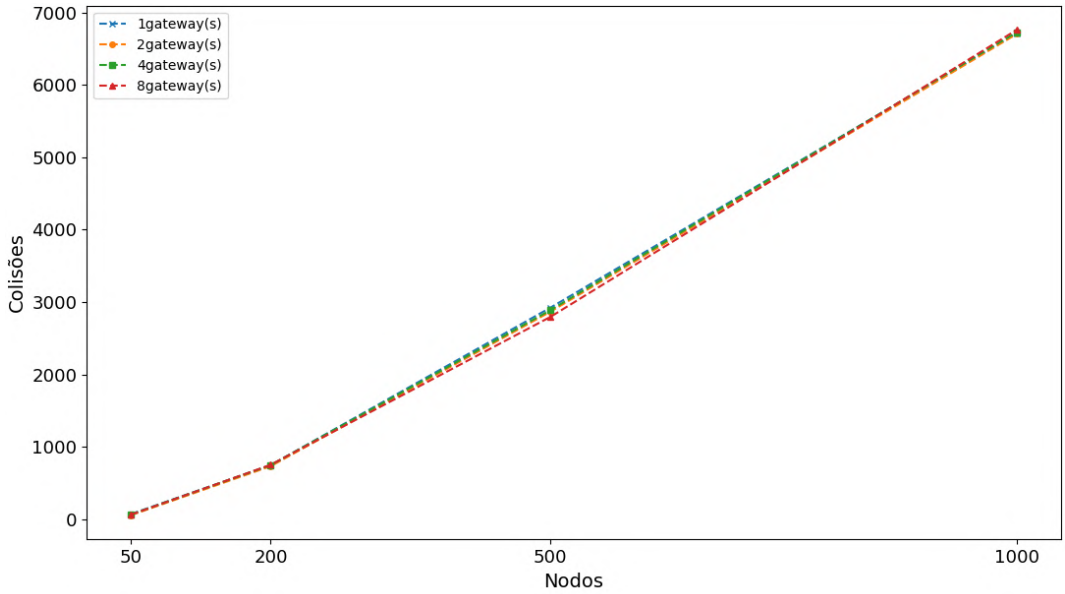
Mesmo para 8 gateways, as perdas são semelhantes, independentemente do in-

Figura 27 – Colisões para um intervalo de mensagem de 60s



Fonte: Elaborado pelo Autor

Figura 28 – Colisões para um intervalo de mensagem de 300s

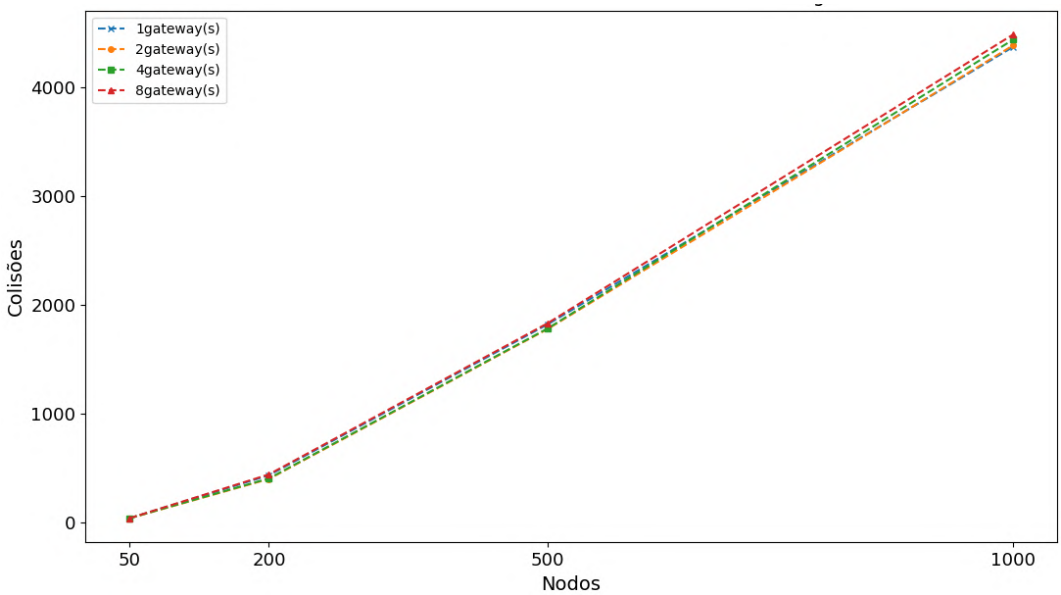


Fonte: Elaborado pelo Autor

tervalo de mensagens. Quando adicionamos mais nodos à rede, elas tendem a ser mais suavizadas à medida que reduzimos a carga de pacotes enviados. Isso é observado na DER de todas as outras configurações (Figura 30, Figura 31, Figura 32), mostrando que, à medida que a carga de pacotes enviados é reduzida, as curvas tendem a ser mais suaves.

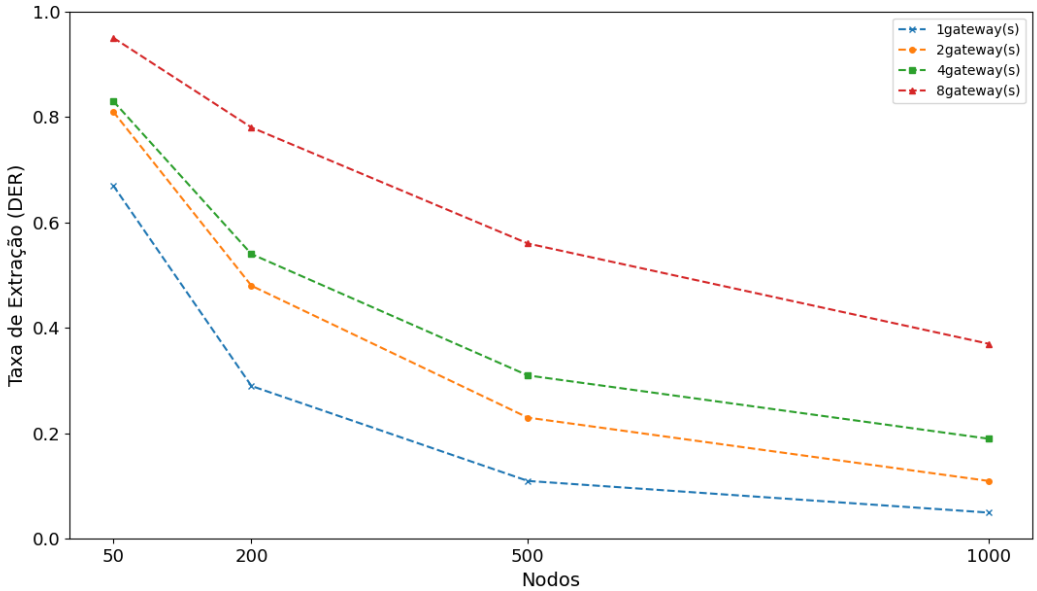
Para algumas aplicações de cidade inteligente, precisamos ter uma DER próxima de 1, o que é ideal para evitar muitas perdas em aplicações que exigem alta precisão. Observando os gráficos, apenas configurações com baixa densidade de nodos demonstram uma DER próxima de 1. As medidas para 1, 2 e 4 gateways a partir de 200 nodos

Figura 29 – Colisões para um intervalo de mensagem de 500s



Fonte: Elaborado pelo Autor

Figura 30 – DER para um intervalo de mensagem de 60s



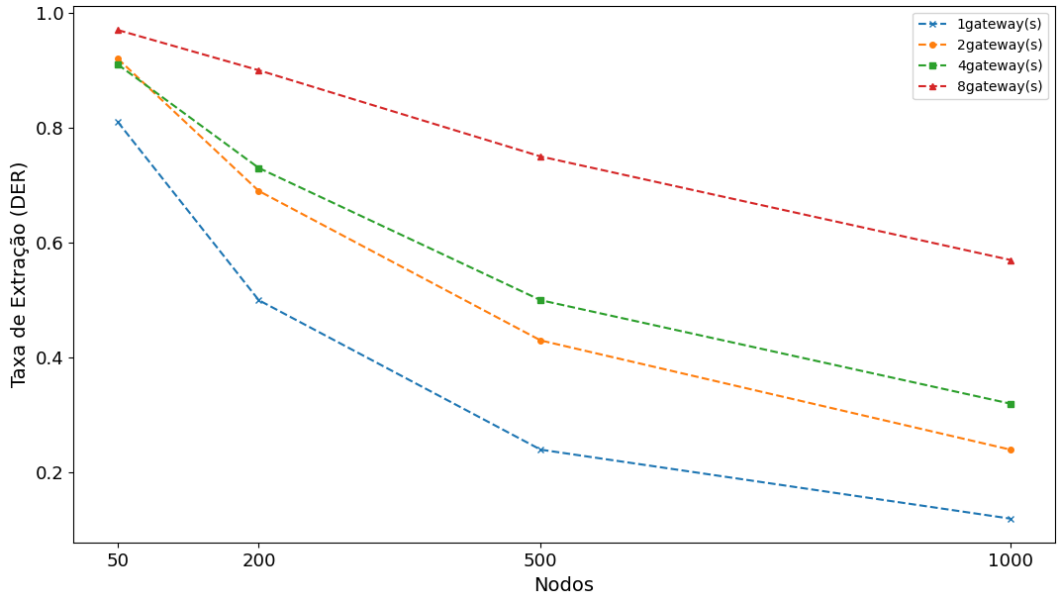
Fonte: Elaborado pelo Autor

já apresentam perdas significativas de pacotes, o que resulta na inviabilidade de certas aplicações IoT.

4.2.3 Variação da simulação outdoor com ADR ativo

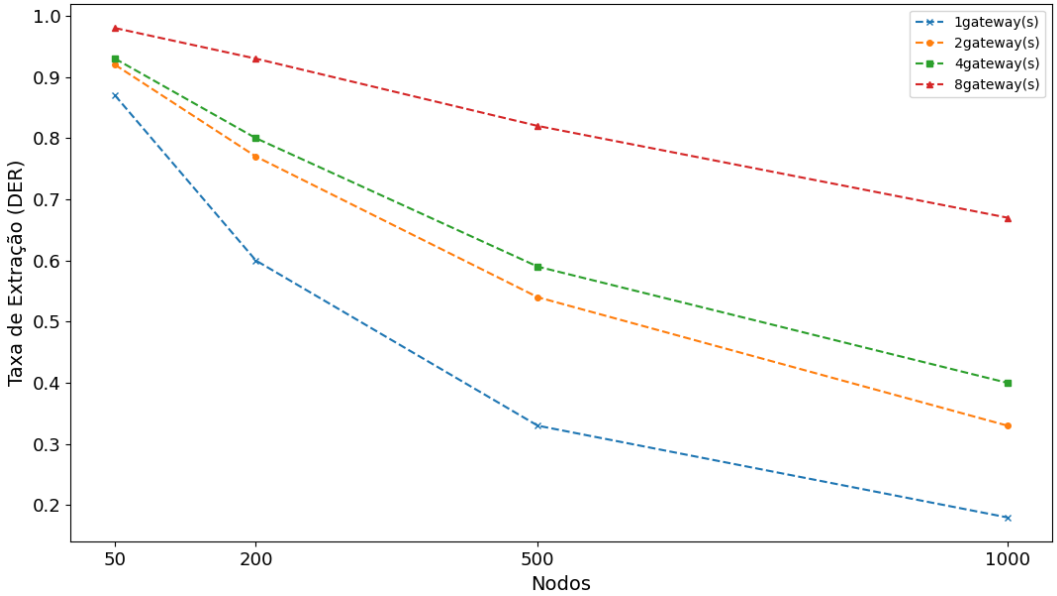
Nesta seção, demonstraremos a variação das simulações utilizando o algoritmo ADR ativo para comparar com as medições anteriores sem o ADR e observar se há alguma mudança na DER ou nas colisões de pacotes.

Figura 31 – DER para um intervalo de mensagem de 300s



Fonte: Elaborado pelo Autor

Figura 32 – DER para um intervalo de mensagem de 500s



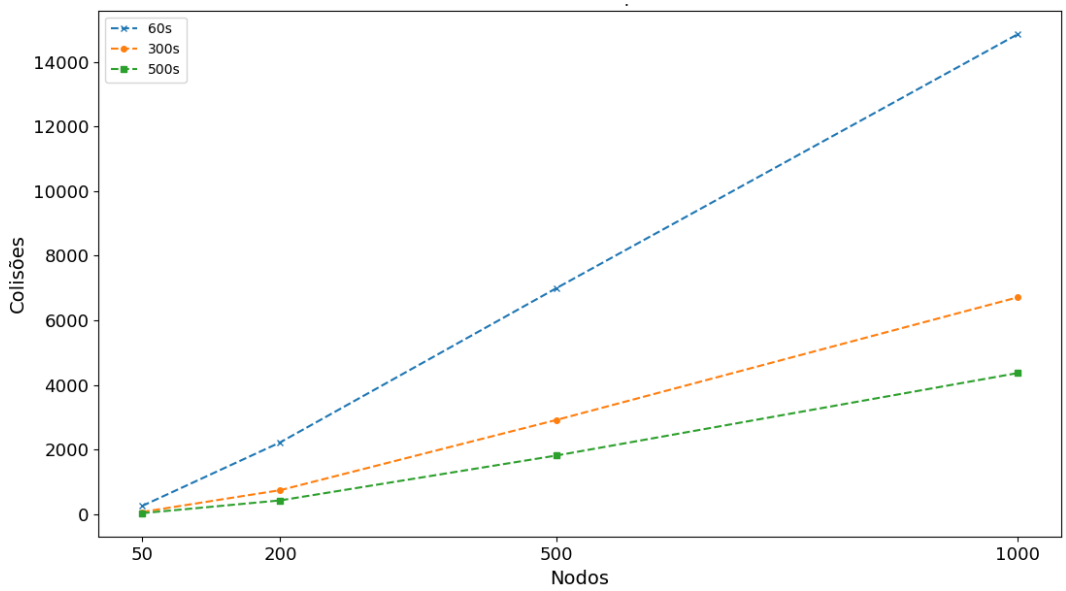
Fonte: Elaborado pelo Autor

As configurações das simulações foram as mesmas do experimento anterior, para intervalos de mensagem de 60, 300 e 500 segundos. Nas simulações, foram alterados alguns parâmetros no arquivo .ini para ativar o ADR, conforme exemplo do (FLORA-DOCUMENTATION, 2022).

Na Figura 33, podemos observar que, mesmo com a utilização de apenas um gateway, o número de colisões aumenta drasticamente à medida que o intervalo entre as mensagens diminui. Especificamente, nos casos de intervalos de 60 segundos, 300 segundos

e 500 segundos entre as mensagens, não foram encontradas diferenças significativas em relação aos experimentos sem a utilização do ADR (Adaptive Data Rate). O gráfico consolidado apresenta o comportamento das colisões para esses três cenários de intervalo de mensagem, indicando uma tendência clara de aumento das colisões à medida que a frequência de mensagens transmitidas é maior. Os valores de DER também permaneceram os mesmos, o que pode indicar que a configuração de fator de espalhamento 12, potência de transmissão 14 e largura de banda de 125kHz seria a mais adequada para uma rede com alta densidade de nodos.

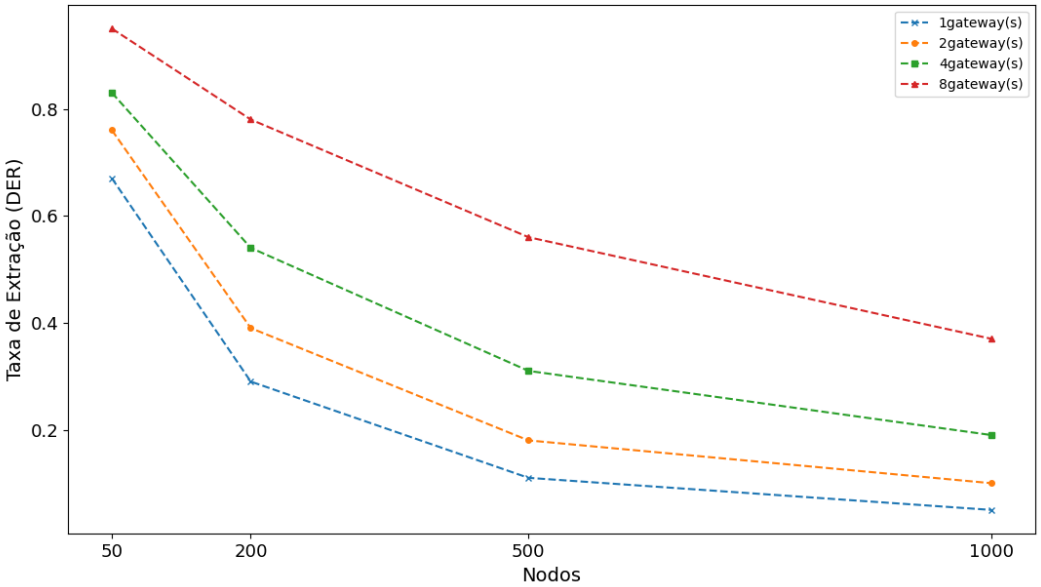
Figura 33 – Colisões para 1 gateway



Fonte: Elaborado pelo Autor

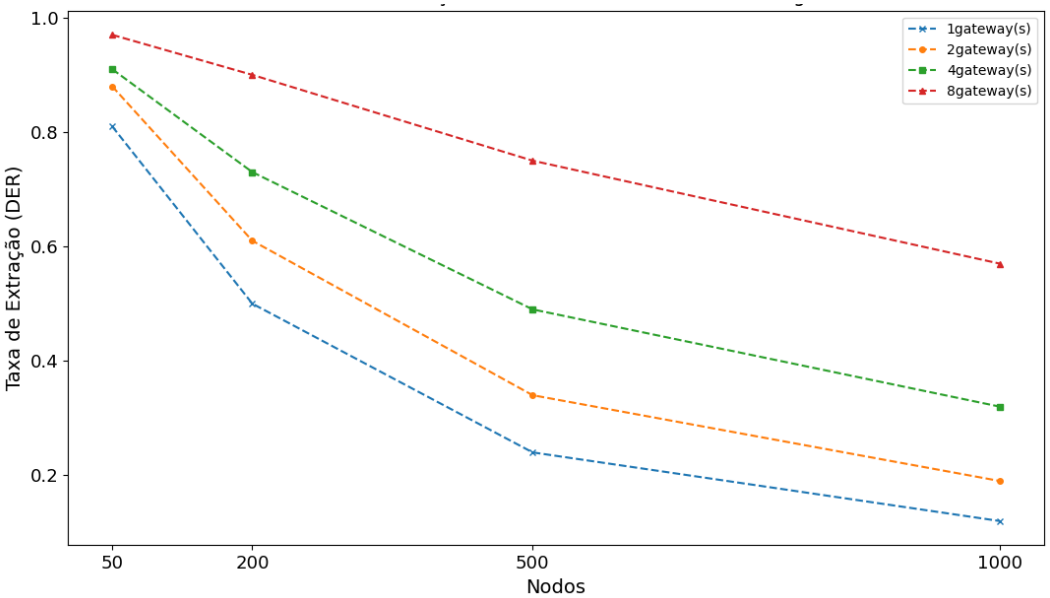
É evidente que o ADR tem como objetivo maximizar a eficiência espectral e a economia de energia. No entanto, no caso avaliado, aparentemente o algoritmo optou por selecionar o fator de espalhamento máximo e a potência máxima. Com isso, os resultados foram semelhantes aos experimentos sem ADR, pois estavam igualmente configurados. Em relação à DER, no caso de um intervalo de mensagem de 300 segundos para 2 e 4 gateways, quando o ADR estava ativo, houve uma piora na entrega de pacotes. Isso demonstra que, apesar de reduzir o consumo de energia em algum momento, acaba afetando a entrega de pacotes.

Figura 34 – DER para um intervalo de mensagem de 60s com ADR



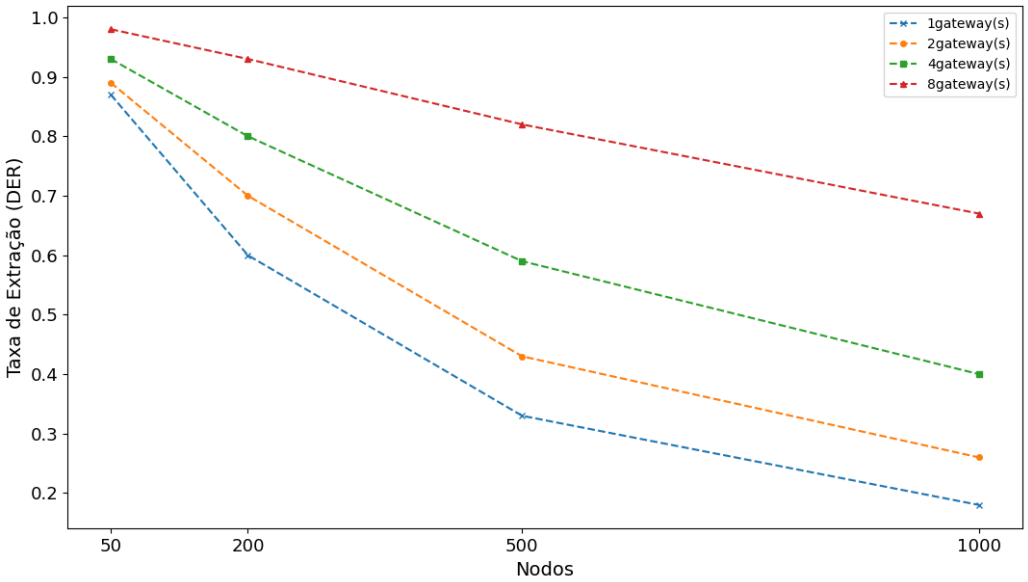
Fonte: Elaborado pelo Autor

Figura 35 – DER para um intervalo de mensagem de 300s com ADR



Fonte: Elaborado pelo Autor

Figura 36 – DER para um intervalo de mensagem de 500s com ADR



Fonte: Elaborado pelo Autor

5 CONCLUSÕES

Neste trabalho, buscou-se construir um cenário de simulação com base em medições reais de um ambiente próximo ao IFSC Campus São José. Para isso, realizaram-se parametrizações de dois modelos de perda de percurso e diversas mudanças nos parâmetros de simulação, permitindo que o ambiente simulado se aproximasse do real.

As simulações de teste demonstraram que um dos modelos se aproximou muito do ambiente real, tornando-o a escolha para realizar as análises das possíveis aplicações no cenário. Utilizando as ferramentas descritas, foi possível simular diversos cenários e analisar o desempenho da rede com diferentes configurações, obtendo sucesso nos resultados. Foram avaliados cenários com e sem o ADR, com alta densidade de nodos e com configurações de 1, 2, 4 e 8 gateways. É relevante mencionar que os resultados relacionados ao ADR foram bastante semelhantes aos obtidos sem a utilização do mesmo. Portanto, sugere-se realizar uma investigação mais aprofundada dessa questão em trabalhos futuros. Uma alternativa possível para uma densidade maior de nodos seria a redução da frequência de mensagens.

Com um aumento no número de gateways, haveria uma possibilidade de suportar algo próximo a 200 nodos, mas isso é algo a ser avaliado em trabalhos futuros. Além disso, o problema das colisões que ocorreram no experimento físico também é algo a ser avaliado, uma vez que em outros trabalhos semelhantes não foram registradas essas ocorrências. Outras opções para pesquisas futuras incluem a continuação do experimento indoor para a parametrização de um modelo indoor, que foi iniciado para complementar este trabalho; o uso de inteligência artificial para calcular os modelos de propagação de rádio, como proposto no artigo (IMAI; KITAO; INOMATA, 2019) que utiliza redes neurais; realizar testes com diferentes fatores de espalhamento e potência de transmissão, aumentando o número de pontos de coleta; e realizar testes com os dispositivos para avaliar o consumo de bateria.

REFERÊNCIAS

BARBIROLI, M. et al. On the performance evaluation of lora networks: A simulation study. *Sensors*, v. 19, n. 9, p. 1980, 2019. 32

BEDAQUE, D. Testes de desempenho de enlaces ponto a ponto da tecnologia lora. IFSC Campus São José, 2016. 38, 48, 52

BOR, M. C. et al. Do lora low-power wide-area networks scale? In: *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (MSWiM '16), p. 59–67. ISBN 9781450345026. Disponível em: https://doi.org/10.1145/2988287.2989163. 14, 18, 19, 20, 30, 56

BOR, M. C. et al. Do lora low-power wide-area networks scale? In: *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (MSWiM '16), p. 59–67. ISBN 9781450345026. Disponível em: https://doi.org/10.1145/2988287.2989163. 18

BORGHETTI, A.; PAOLINI, M.; POLESE, M. Performance analysis of lora networks in a smart city environment. *IEEE Transactions on Smart Grid*, v. 10, n. 2, p. 2222–2232, 2019. 32

CHAUDHARI, B. S.; ZENNARO, M. *Lpwan Technologies for IOT and M2M applications*. [S.l.]: Academic Press, 2020. 16, 17

CHAUDHARI, B. S.; ZENNARO, M.; BORKAR, S. Lpwan technologies: Emerging application characteristics, requirements, and design considerations. *Future Internet*, v. 12, n. 3, 2020. ISSN 1999-5903. Disponível em: https://www.mdpi.com/1999-5903/12/3/46. 16

DATASHEET-IGT200. 2019. https://www.khomp.com/wp-content/uploads/2019/02/ITG_200_-_PT_v4.pdf. Acessado em : 10-06-2023. 34

DATASHEET SMW-SX1276M0. 2019. https://d229kd5ey79jzj.cloudfront.net/1239/Transceiver_LoRaWAN_SMT_SMW-SX1276M0.pdf. Acessado em : 01-07-2023. 34

ELDEFRAWY, M. et al. Formal security analysis of lorawan. *Computer Networks*, v. 148, p. 328–339, 2019. ISSN 1389-1286. Disponível em: https://www.sciencedirect.com/science/article/pii/S1389128618306145. 22

ESPRESSIF Systems ESP32-WROOM-32U Datasheet. 2019. https://octopart.com/datasheet/esp32-wroom-32u-espressif+systems-89180214. Acessado em : 20-06-2023. 35

FLORA-DOCUMENTATION. 2022. https://flora.aalto.fi/. Acessado em : 13-12-2022. 31, 61

GRIVA, A. I. et al. Lora-based iot network assessment in rural and urban scenarios. *Sensors*, v. 23, n. 3, 2023. ISSN 1424-8220. Disponível em: https://www.mdpi.com/1424-8220/23/3/1695. 54

HATA, M. Empirical formula for propagation loss in land mobile radio services. *IEEE Transactions on Vehicular Technology*, IEEE, v. 29, n. 3, p. 317–325, 1980. 32

HAXHIBEQIRI, J. et al. A survey of lorawan for iot: From technology to application. *Sensors*, v. 18, n. 11, 2018. ISSN 1424-8220. Disponível em: https://www.mdpi.com/1424-8220/18/11/3995. 23, 24

IMAI, T.; KITAO, K.; INOMATA, M. Radio propagation prediction model using convolutional neural networks by deep learning. In: *2019 13th European Conference on Antennas and Propagation (EuCAP)*. [S.l.: s.n.], 2019. p. 1–5. 65

INET-DOCUMENTATION. 2022. https://inet.omnetpp.org/docs/. Acessado em : 03-12-2022. 25

KIM, H. et al. A simulation framework for performance analysis of multi-interface and multi-channel wireless networks in inet/omnet++. In: *Proceedings of the 2010 Spring Simulation Multiconference*. San Diego, CA, USA: Society for Computer Simulation International, 2010. (SpringSim '10). ISBN 9781450300698. Disponível em: https://doi-org.ez130.periodicos.capes.gov.br/10.1145/1878537.1878643. 25, 26, 27

LINKA, H. et al. Path loss models for low-power wide-area networks: Experimental results using lora. In: . [S.l.: s.n.], 2018. 47, 57

LORA AT Command Manual. 2020. https://s3-sa-east-1.amazonaws.com/robocore-lojavirtual/1239/SMART_LoRa_AT.Command_v1.1_en_v0.5.pdf. Acessado em : 26-06-2023. 35

NODERED. 2023. https://nodered.org/. Acessado em : 20-05-2023. 34

RAPPAPORT, T. S. *Comunicações Sem Fio: Princípios e Práticas*. 2. ed. Porto Alegre: Bookman, 2009. 32, 50, 52

RAZA, U.; KULKARNI, P.; SOORIYABANDARA, M. Low power wide area networks: An overview. *IEEE Communications Surveys & Tutorials*, v. 19, n. 2, p. 855–873, 2017. 16, 17

ROBOCORE-DEVKIT. 2021. https://www.robocore.net/lorawan/iot-devkit-lorawan. Acessado em : 22-05-2023. 34

SANTOS, E. J. dos. *Análise do impacto de técnicas de diversidade na comunicação entre dispositivos LoRa/LoRaWAN*. Florianópolis: [s.n.], 2019. Trabalho de Conclusão de Curso submetido ao Departamento de Engenharia Elétrica e Eletrônica da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia Eletrônica. Orientador: Richard Demo Souza. 49

SEMTECH. *LoRa® and LoRaWAN®: A Technical Overview*. 1. ed. [S.l.], 2019. Disponível em: https://lora-developers.semtech.com/uploads/documents/files/LoRa_and_LoRaWAN-A_Tech_Overview-Downloadable.pdf. 18, 20, 21, 22, 23, 24, 30

SLABICKI GOPIKA PREMSANKAR, M. D. F. M. Adaptive configuration of lora networks for dense iot deployments. 2018. Disponível em: https://flora.aalto.fi/resources/slabicki-2018-noms.pdf. 14, 20, 29, 30

THETHINGSNETWORK. 2023. https://www.thethingsnetwork.org/. Acessado em : 10-06-2023. 34

VARGA, A.; LTD, O. *Omnet++ Simulation Manual*. 1. ed. [S.l.], 2021. Disponível em: https://doc.omnetpp.org/omnetpp/SimulationManual.pdf. 14, 26, 28

YASCARIBAY, G. et al. Performance evaluation of communication systems used for internet of things in agriculture. *Agriculture*, v. 12, n. 6, 2022. ISSN 2077-0472. Disponível em: https://www.mdpi.com/2077-0472/12/6/786. 29, 32