Jamming-aware Routing in Military Wireless Sensor Networks

João Tiago Henriques Montez

Dissertação para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador(es): Professor Doutor António Manuel Raminhos Cordeiro Grilo
Professor Doutor João Paulo Baptista de Carvalho

Júri

Presidente: Professor Nuno Cavaco Gomes Horta
Orientador: Professor Doutor António Manuel Raminhos Cordeiro Grilo
Vogal: Professor Doutor Carlos Manuel Ribeiro Almeida

Outubro 2016

Agradecimentos

Ao meu orientador, Professor Doutor António Grilo, e coorientador, Professor Doutor João Paulo Carvalho, um profundo agradecimento pela disponibilidade e assistência na resolução dos problemas que surgiram ao longo do desenvolvimento deste projeto.

Um agradecimento especial ao Major de Transmissões Lopes pela cooperação na elaboração dos testes efetuados com o *jammer*.

Resumo

As ameaças presentes no campo de batalha estão em constante modificação, sendo que se tornaram mais diversificadas e perigosas ao longo da História. Recentemente, tem-se intensificado o uso de engenhos explosivos ativados à distância por sinais de rádio-frequência, como forma de ataque a forças destacadas no combate contra o terrorismo, pelo que este é um problema com o qual as forças terrestres se defrontam atualmente.

A guerrilha, assim como o terrorismo, são ameaças imprevisíveis, sem fronteiras e que podem deflagrar a qualquer momento e em qualquer lugar. Os grupos guerrilheiros utilizam frequentemente engenhos explosivos, como forma de ataque, sendo a proteção contra os mesmos essencial, num teatro de operações.

Os *jammers* surgem como uma solução para a proteção da força contra explosivos ativados por meios rádio, já que estes aparelhos conseguem impedir as comunicações *wireless* no seu raio de ação. Por outro lado, as redes de sensores são também um elemento útil no teatro de operações, visto que permitem vigilância de grandes áreas e por longos períodos de tempo. No entanto, é necessário ter em conta que a comunicação entre nós da rede pode ser afetada pela utilização de *jammers*, quer acidentalmente por parte de forças amigas, quer propositadamente por parte de forças inimigas.

É neste âmbito que surge o presente projeto, que tem como objetivo minimizar a disrupção da rede de sensores devido à utilização de *jammers*, utilizando uma técnica baseada na alteração das rotas de encaminhamento de tráfego. Para isso, desenvolveu-se uma extensão ao protocolo *Destination Sequenced Distance Vector* (DSDV).

Para a escolha das rotas, o algoritmo utilizado baseia-se na posição geográfica dos nós e do *jammer* e tem como objetivo desviar as rotas, quando possível, por nós suficientemente distantes em relação ao *jammer*.

Neste trabalho não é elaborado nenhum algoritmo de deteção do *jammer*, já que se supõe que os nós conhecem a sua localização e que têm um algoritmo de deteção da posição geográfica do mesmo, sendo apenas elaborado um novo algoritmo de *routing*.

O algoritmo utilizado mostrou-se mais eficiente na situação descrita, tendo-se diminuído as perdas em 20% em relação ao DSDV original, em versão *manpack*, e em 30% para montagens veiculares.

Palavras-chave: Redes de sensores sem fios, empastelamento, protocolos de encaminhamento

Abstract

The Threats present in the battlefield are constantly changing, becoming more diversified and dangerous throughout History. Recently, the use of explosive devices, remotely activated by radio frequency signals, as a form of attack to deployed forces in the combat against terrorism, has been intensified. This is therefore a problem that land forces have to face nowadays.

Guerrilla, as well as terrorism, is an unpredictable threat, without frontiers, that can happen anywhere at any time. Guerrilla groups often use explosive devices as form of attack. Protection against these attacks is essential in a theater of operations.

Jammers appear as a solution to the protection of force against radio activated explosives, since these apparatus can prevent wireless communications within their action range. On the other hand, sensor networks are also a useful element in the battlefield, since they enable surveillance of large surfaces for long periods of time. However, it is necessary to have in consideration that network nodes communication may be affected by the use of jammers, either accidentally by friendly forces, or purposely by enemy forces.

This project focus this area, with the objective of minimizing the disruption of sensors network owing to the use of jammers, using a technique based on adapting the traffic routes. For this purpose, an extension of Destination Sequenced Distance Vector (DSDV) protocol has been used.

For routes selection, the algorithm used is based on nodes and jammer geographic position and has the objective of deviating routes, when possible, keeping them far enough from the jammer.

This thesis does not develop any algorithm for jammer detection, assuming that it is in place, running in the sensor network, allowing the nodes to know its position. It was only elaborated a new routing algorithm.

Simulations show that the developed algorithm was efficient in the target situations having lost less 20% than the original DSDV for used jammers, in manpack version, and 30% less for vehicles assemblies.

Keywords: Wireless Sensor Network, jamming, routing protocol

Conteúdo

Lista de Tabelas

Lista de Figuras

Lista de Símbolos

Simbolos gregos

γ Fator de perda de pacotes.

γ_{o0} Atenuação devido a gases.

γ_{w0} Atenuação devido ao vapor de água.

Simbolos romanos

$(\frac{C}{N})_{cip}$ Relação Sinal-Ruído.

A Atenuação.

A_0 Atenuação em espaço livre.

A_{At} Atenuação atmosférica.

A_{Obs} Atenuação de Obstáculo.

b_{rf} Largura de banda.

d Distância.

d_0 Distância de referência.

$f_{[MHz]}$ Frequência.

G_E Ganho de emissão.

G_R Ganho de receção.

K_B $1.38 \cdot 10^{-23} J/K$ (Constante de Boltzmann).

l Diâmetro.

N_0 Ruído Térmico.

P_E Potência de emissão.

P_{Ruido} Potência do ruído.

P_{Rx} Potência de receção.

P_{Sinal} Potência do sinal.

P_{Tx} Potência de transmissão.

PL *Perdas de pacotes (Path Loss)*.

PL_0 Referência para perda de pacotes(*Reference Path Loss*) (Friis(d_0)).

$T_{[K]}$ Temperatura (Kelvin).

X_g Log-normal Shadowing.

Q *Convex Hull*.

Subscritos

i, j, k Índices computacionais.

x, y, z Componentes Cartesianos.

Glossário

ANF	*Ambient Noise floor*.
AODV	*Ad-hoc On-Demand Distance Vector Routing*.
AOMDV	*Ad-hoc On-demand Multipath Distance Vector Routing*.
CL	*Centroid Localization*.
CSMA	*Carrier Sense Multiple Access*.
CST	*Carrier Sensing Time*.
DAO	*DODAG Destination Advertisement Object*.
DIO	*DODAG Information Object*.
DIS	*DODAG Information Solicitation*.
DODAG	*Destination Oriented Direct Acyclic Graph*.
DSDV	*Destination Sequenced Distance Vector Routing*.
IAMP	*Interference Activity Aware Multi-path Routing Protocol*.
IED's	Artefactos Explosivos Improvisados.
IETF	*Internet Engineering Task Force*.
IoT	*Internet of Things*.
JAID	*Jamming Avoidance Itinerary Design algorithm*.
JSS	*Jamming Signals Strength*.
LBR	*LowPAN Border Router*.
LLNs	*Low Power and Lossy Networks*.
MAC	*Medium Access Control*.
MA	Agentes móveis.
MCCL	*Minimum-covering-circle*.
PE	Elemento de processamento .
PSR	*Packet Send Ratio*.
RPL	*Ripple Routing Protocol*.
RREPs	*Route Reply*.
RREQ	*Route Request*.

RSSI *Received Signal Strength Indicator*.

SNR *signal-to-noise ratio*.

Capítulo 1

Introdução

A guerrilha e o terrorismo são duas ameaças imprevisíveis nos teatros de operações atuais e, por isso, difíceis de combater.

A tática de guerrilha consiste em usar pequenas forças com o objetivo de desgastar forças mais numerosas, recorrendo a emboscadas. Esta é uma técnica de resistência, utilizada por grupos pequenos, com grande capacidade de ocultação e mobilidade, que surpreendem o adversário em pontos críticos da sua organização, sendo que os alvos prioritários são, normalmente, os comandantes.

Por outro lado, o terrorismo tem o mesmo princípio de atuação que a guerrilha. No entanto, tem como objetivo criar o pânico e subjugar o lugar afetado pelo medo e pelo terror. Este é praticado à escala global, não tem fronteiras e não faz distinção entre Forças Armadas e civis, o que o torna ainda mais imprevisível e difícil de combater.

Nestes grupos é frequente a utilização de engenhos explosivos improvisados (*Improvised Explosive Devices*, IED's) como forma de ataque, sendo que estes dispositivos são, normalmente, ativados por meios rádio.

Tanto a guerrilha, como o terrorismo são atos levados a cabo por indivíduos portadores de equipamento ligeiro, o que os torna bastante móveis e lhes permite ter grande facilidade em atacar por zonas de difícil acesso e difíceis de monitorizar.

Este tipo de ameaças contra as forças destacadas são constantes em missões de Apoio à Paz. Desta forma, torna-se essencial melhorar a segurança das áreas circundantes a pontos críticos, como é o caso de aquartelamentos e torres de comunicações, ou zonas onde a passagem pode ser facilmente obstruída como, por exemplo, pontes.

A utilização de Redes de Sensores sem Fios pode ser uma forma de solucionar este problema. Estas consistem numa grande quantidade de nós de sensores distribuídos por uma determinada área, que permitem a vigilância desta durante 24 horas por dia.

Os nós são, geralmente, pequenos, têm um baixo custo, necessitam de pouca energia, têm pouca capacidade de processamento e comunicação limitada. É o fato de apresentarem um consumo energético reduzido que os torna ideais para objetivos específicos, como a vigilância, sendo capazes de operar por longos períodos de tempo, partindo do princípio que as suas funções se encontram desenhadas

por forma a maximizar a eficiência energética.

1.1 Motivação e definição do problema

As Redes de Sensores sem Fios permitem detetar, classificar e localizar elementos hostis em tempo útil, mesmo dentro de edifícios ou em condições meteorológicas difíceis, como se pode observar na figura 1.1. No entanto, manter a segurança de uma rede de sensores pode ser uma tarefa desafiante.

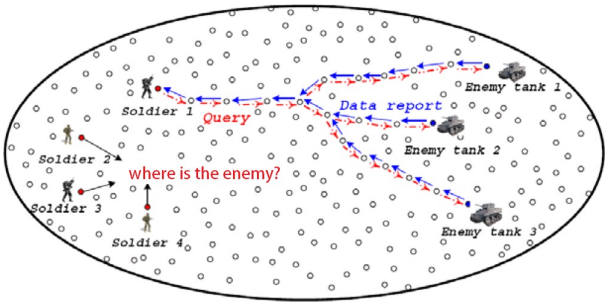


Figura 1.1: Exemplo de utilização de redes de sensores. O soldado utiliza a rede de sensores para localizar o carro de combate [1].

O inimigo consegue facilmente atacar comunicações *wireless*, tanto acedendo a comunicações entre sensores, ligando os seus próprios dispositivos e correndo em modo de monitorização, como impedindo a comunicação entre sensores, através do envio de um sinal de alta potência na mesma gama de frequência (*jamming*).

No primeiro caso, a solução passa essencialmente por cifrar o sinal ou por controlar o acesso de novos nós à rede, por exemplo, utilizar chaves de acesso cada vez que um novo nó se tenta conectar à rede, obrigando a que apenas nós com conhecimento da chave de acesso se possam conectar.

A utilização de *jammers* para ataques a sistemas de telecomunicações não é recente, no entanto, a sua utilização em teatros de operações tem-se tornado bastante frequente com o aparecimento de IED's utilizados tanto na tática de guerrilha, como em atos terroristas. Um exemplo disso é o caso da intervenção no Iraque, em que este tipo de ameaças são constantes. Os *jammers* têm a capacidade de bloquear a receção do sinal dos engenhos explosivos durante o período de tempo em que estes se encontrem no raio de ação dos primeiros.

As comunicações entre nós de sensores, essenciais na vigilância de áreas defensivas, podem ser afetadas indireta ou diretamente através de *jammers*. Indiretamente, caso uma força no terreno utilize *jammers* como um meio de proteção da força, para bloquear o sinal dos IED's. Neste caso, pode ser uma força amiga a bloquear a rede sensores. Diretamente, caso um grupo armado pretenda desativar a Rede de Sensores sem Fios, por forma a conseguir penetrar as defesas opositoras.

Em qualquer das situações de utilização de *jammers*, a eficácia da rede de sensores é comprometida, tornando-se essencial encontrar uma solução para melhorar as comunicações.

Tanto no caso da guerrilha, como no terrorismo, é de esperar que as forças tenham dificuldade em adquirir armamento de grandes dimensões, pelo que apenas dispõem de armamento ligeiro. Conse-

quentemente, é de esperar que, em caso de utilização de *jammers*, estes correspondam maioritariamente à versão *manpack*, como o representado na figura 1.2.

Importa referir que as missões de contra-guerrilha são, normalmente, efetuadas em terrenos inóspitos e, consequentemente, por forças apeadas, sendo a versão *manpack* do *jammer* a utilizada nestes casos, o que pode provocar disrupção da rede de sensores por parte de forças amigas.

Figura 1.2: Exemplo de *jammer* (versão *Manpack*).

No caso de montagens veiculares, espera-se que sejam utilizadas em viaturas ligeiras com grande mobilidade (figuras 1.3 e 1.4).

Figura 1.3: Exemplo de *jammer* (versão veicular).

Figura 1.4: Veículo militar com equipamento de comunicações.

1.2 Objetivos

A utilização de Redes de Sensores sem Fios por parte das Forças Armadas tem como objetivo prever eventuais ameaças e melhorar o tempo de resposta. Uma falha na comunicação desta rede pode pôr em causa uma operação militar.

Este trabalho tem como objetivo melhorar a eficácia da rede de sensores quando a comunicação entre nós é impedida pela utilização de *jammers*, quer seja por parte da nossa força, quando estes são

utilizados como proteção da força contra IED's, quer quando este é utilizado pelo inimigo, com intenção de quebrar a comunicação na Rede de Sensores sem Fios.

Por forma a cumprir o objetivo principal e testar as suas capacidades, torna-se necessário:

- Efetuar testes de alcance com o *jammer*.
- Definir um modelo de propagação adequado que permita inserir o *jammer* em simulação.
- Desenvolver um algoritmo de routing que permita minimizar a disrupção da rede de sensores quando esta é afetada por um *jammer*.
- Efetuar testes ao algoritmo desenvolvido.
- Analisar os resultados obtidos.

Após a resolução do projeto, é de esperar que o algoritmo desenvolvido permita diminuir a disrupção provocada pelo *jammer* na Rede de Sensores sem Fios.

1.3 Contribuições

Este projeto resultou nas seguintes contribuições:

- Algoritmo de *routing* baseado no DSDV para o tornar mais resiliente a *jamming*.
- Modelo de simulação NS-3 de *jamming* em redes sem-fios.
- Avaliação do impacto de *jammers* no desempenho de Redes de Sensores sem Fios, assumindo integração de um sistema de localização do *jammer*.

1.4 Organização do Documento

O capítulo 2 consiste no Enquadramento do tema na área científica e revisão do Estado da Arte. Na secção 2.1 são explicados alguns conceitos de Redes de Sensores sem Fios e na secção 2.2 são apresentados alguns dos principais algoritmos de *routing*. Na secção 2.3.1 são apresentados diversos tipos de *jammers* e na secção 2.3.2 são explicados alguns algoritmos de deteção e localização do *jammer*. Por fim, são descritas algumas contramedidas *anti-jamming* na secção 2.3.3.

No capítulo 3 é descrito todo o trabalho desenvolvido, sendo que na secção 3.1 é descrito o algoritmo de encaminhamento utilizado. Os resultados dos testes para modelação do *jammer* são apresentados na secção 3.2, resultados estes que permitiram elaborar o modelo numérico abordado na secção 3.3. Na secção 3.4 é descrita a implementação em NS-3.

Na primeira parte do capítulo 4 é testado o desempenho do DSDV na presença de *jamming* e os resultados relativos à versão *manpack* do *jammer* são apresentados na secção 4.2. Os resultados relativos à montagem veicular do *jammer* são apresentados na secção 4.3.

No capítulo 5 apresentam-se as conclusões, sendo sugerido algum trabalho a desenvolver no futuro, ao longo da secção 5.1.

Capítulo 2

Enquadramento do tema na área científica e revisão do estado da arte

Neste capítulo são abordados alguns conceitos básicos acerca de Redes de Sensores sem Fios, na secção 2.1, são descritos alguns protocolos de *routing*, com interesse para o caso em estudo, na secção 2.2 e alguns aspetos acerca de *jamming*, na secção 2.3. São apresentados diversos tipos de *jammers*, na secção 2.3.1, métodos de deteção, localização e *tracking* do *jammer*, na secção 2.3.2 e algumas contramedidas para *jamming* na secção 2.3.3.

2.1 Redes de Sensores sem Fios

As Redes de Sensores sem Fios podem ser definidas como conjuntos de dispositivos ou nós sensores que, para além de poderem funcionar de forma isolada e autónoma, têm a capacidade para se agruparem e formarem redes sem suporte de infraestrutura externa, permitindo-lhes colaborar e/ou enviar dados para estações externas, através de nós de interface especiais designados *sink*. Têm como objetivo monitorizar um conjunto de fenómenos físicos.

O *hardware* utilizado numa rede de sensores depende da aplicação a que se destina. O custo, tamanho e consumo de energia são normalmente os fatores decisivos.

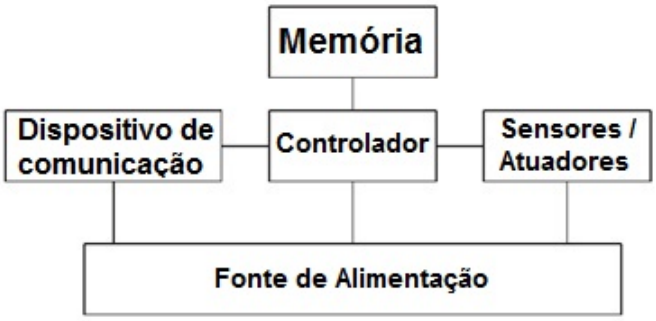


Figura 2.1: Componentes que constituem um nó [2].

Um nó básico de uma rede de sensores é constituído por cinco componentes principais (figura 2.1):

- Controlador – processa os dados recebidos.
- Memória – armazena os programas e dados intermédios.
- Sensores e atuadores – são os responsáveis por recolher os dados e são a ligação com o ambiente que os rodeia.
- Dispositivo de comunicação – ligação ente nós, requer um dispositivo para enviar e receber dados através de *wireless*.
- Fonte de alimentação – Normalmente baterias, no entanto, pode ser possível retirar energia do ambiente, por exemplo através de painéis solares.

A figura 2.2 ilustra um módulo XBee868Pro. Este consiste num módulo de comunicação rádio que pode ser usado para estabelecer a ligação entre nós sensores.

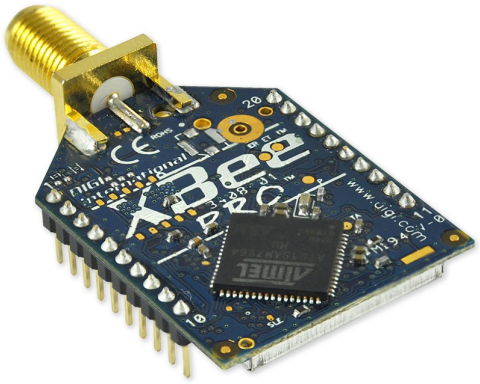


Figura 2.2: Exemplo de um módulo de comunicação usado em Redes de Sensores sem Fios.

A comunicação é o principal elemento no estudo em causa e, para que esta aconteça, é necessário que existam tanto transmissor como recetor no nó. Os dispositivos responsáveis por converter os bits que saem do microcontrolador em ondas rádio e o inverso são os *transceivers*.

Transmitir e receber ao mesmo tempo, em meios *wireless*, é impraticável na maior parte dos casos, sendo normalmente utilizadas comunicações *half-duplex*.

O facto de os nós de sensores serem fortemente dependentes de baterias resulta numa limitação da potência transmitida e, portanto, da distância de comunicação entre o emissor e o recetor. Por este motivo, nem sempre é possível efetuar comunicação direta entre o emissor e o *sink*, quando a área geográfica a cobrir apresenta uma grande dimensão.

A utilização da rede de sensores em edifícios sofre o mesmo efeito, apesar das distâncias serem mais curtas, por existir uma elevada atenuação, devido aos obstáculos. Para fazer frente as estas limitações, uma solução é utilizar transmissões de curto alcance entre nós, os quais integram a funcionalidade de *relays*. Assim, os pacotes de dados efetuam múltiplos saltos desde o emissor até ao *sink*. Este conceito é especialmente atrativo para redes de sensores, uma vez que os próprios nós servem

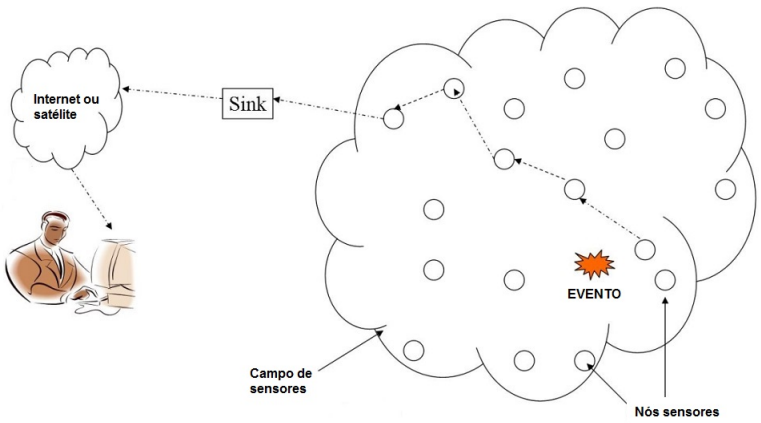


Figura 2.3: Exemplo de comunicação em múltiplos saltos entre o emissor e o *sink* [3].

de estações de retransmissão, sem necessidade de adicionar *hardware* extra, como se pode observar na figura 2.3.

Em redes que efetuam múltiplos saltos, os nós são responsáveis por garantir a comunicação entre o emissor e o *sink*, sendo que cada nó tem que decidir a que vizinho enviará os pacotes de dados que não lhe são destinados. Para que cada nó saiba para qual dos nós vizinhos deve enviar os pacotes de dados, existem os protocolos de *routing*.

2.2 Protocolos de *Routing*

A regra de encaminhamento mais simples consiste em enviar os pacotes recebidos para todos os vizinhos. Enquanto o nó emissor e o recetor estiverem na mesma rede, é garantido que os pacotes chegam ao destino. Para evitar ciclos, um nó deve enviar apenas pacotes que nunca tenha visto e estes devem conter data de expiração. Em alternativa, os pacotes podem ser enviados de forma aleatória e esperar que eventualmente cheguem ao destino. No entanto, ambos os métodos são pouco eficientes, visto que têm um atraso elevado e baixa performance.

Em muitos protocolos são utilizadas tabelas de *routing* que listam o vizinho mais apropriado para cada destino do pacote, tendo como parâmetro o custo de enviar o pacote por esse vizinho. A construção e manutenção destas tabelas de *routing* é tarefa dos algoritmos de *routing*, utilizando protocolos de *routing* para trocar informação entre si.

Os protocolos de *routing* podem ser classificados como centrados nos nós, centralizados nos dados, com conhecimento da localização ou baseados na qualidade de serviço [4].

A maioria dos protocolos de *routing* para redes *Ad-hoc* são protocolos centrados nos nós em que o destino é especificado com base em endereços numéricos dos nós. Em redes de sensores, a comunicação centrada nos nós não é comum. Nas redes de sensores é mais comum encontrar protocolos centrados nos dados, em que o *sink* envia pedidos para certas regiões e espera que os sensores nessas regiões lhe enviem os dados, muitas vezes agregados. No caso em que são centrados nos dados, estes são enviados por todos os sensores dessa região, o que provoca redundância significativa.

Nos protocolos com conhecimento da localização, os nós conhecem a sua posição geográfica. A localização pode ser utilizada para melhorar a performance do protocolo de *routing* e providenciar tipos de serviços diferentes.

Nos protocolos *routing* baseados em qualidade de serviço, o rácio de envio de dados, a latência e consumo de energia são os principais aspetos considerados. Para uma boa qualidade de serviço, o protocolo deve ter um rácio alto de dados enviados, baixa latência e baixo consumo de energia.

Os protocolos *routing* são também classificados com base no local onde são iniciados, destino ou fonte. Quando são iniciados na fonte, os caminhos de *routing* são iniciados por imposição da fonte e começam no nó fonte. A fonte avisa quando os dados estão disponíveis e começa o envio. Quando se trata de um protocolo com iniciação no destino, o caminho é iniciado a partir do destino.

Outra classificação possível é baseada na arquitetura da Rede de Sensores sem Fios. Algumas redes de sensores são compostas por nós homogéneos e outras por nós heterogéneos. Podemos classificar os protocolos conforme eles operem numa topologia uniforme ou hierárquica. Nas topologias uniformes, todos os nós são tratados de forma igual. Na topologia hierárquica, alguns nós têm mais peso que outros. A topologia hierárquica tem muitas vantagens como escalabilidade e eficiência de energia, porque as rotas são fáceis de gerir.

As redes de sensores podem ser destinadas a diversas aplicações, podem conter nós móveis, topologias complexas e a densidade dos nós depende da aplicação a que se destinam. Para que possa servir um conjunto tão variado de condições, existem alguns aspetos a ter em conta, quando se projeta a rede.

A rede de sensores deve ser tolerante a falhas, isto é, quando ocorre uma falha num dos nós, a rede deve ser capaz de continuar a funcionar sem interrupções. Deve ser escalável, permitindo que sejam adicionados mais nós quando necessário e deve conseguir adaptar-se a alterações provocadas por fatores externos. No momento da definição da arquitetura das redes deve-se ter em conta os custos de produção associados aos nós, uma vez, que, podem ser necessários em grandes quantidades. O local onde a rede vai funcionar deve ser tido em conta, esta pode destinar-se a monitorizar diversos locais, desde o interior de edifícios até ao fundo do mar.

Uma das barreiras mais complicadas de ultrapassar é consumo de energia, o tempo de vida do nó depende da duração da bateria. Para isso, o protocolo de *routing* tem de ser o mais eficiente possível, minimizando o número de mensagens trocadas.

De seguida, são exemplificados alguns protocolos de *routing* com características especificas que podem ser úteis para otimizar a comunicação entre os nós da rede de sensores na presença de um *jammer*.

O livro [2] exemplifica e explica alguns protocolos de *routing*, sendo que no âmbito do trabalho a desenvolver, alguns deles mereceram especial atenção e serão descritos nos seguintes subcapítulos.

2.2.1 *Destination Sequenced Distance Vector Routing*

O *Destination Sequenced Distance Vector Routing* (DSDV) é um protocolo a considerar. Este protocolo consiste essencialmente em manter atualizada uma tabela de *routing* que contem os custos de cada caminho, e envia os dados pelo caminho com custo menor [5, 6].

O DSDV mede o custo do caminho em termos distância do nó emissor ao nó *sink*. Por sua vês, esta distância é medida pelo número de saltos (*hops*) que cada nó dá desde que parte do nó emissor até ao nó *sink*.

O DSDV foi criado para redes *ad-hoc* móveis. Em redes de sensores, normalmente simplifica-se a construção da rotas, apenas sendo construída a rota para o *sink*. Esta simplificação surge pelo fato dos nós sensores apenas necessitarem de comunicar com o *sink*, sendo dispensável a existência de rotas para outros nós sensores.

Cada nó altera a sua tabela de *routing* periodicamente ou imediatamente, quando são detetadas alterações à topologia da rede ou existe informação nova significante. Cada nó envia por *broadcasting* ou *multicasting* um pacote de atualização da tabela, o pacote começa com a métrica um para os nós ligados diretamente, o que indica que estes estão a um salto de cada nó que recebe o pacote. Por sua vez, o nó vizinho incrementa a métrica e reenvia o pacote. Este processo repete-se até que cada nó receba uma cópia do pacote com a respetiva métrica. Os nós recebem vários pacotes, sendo que entre pacotes com o mesmo destino, é preferido o de menor métrica e os outros são descartados.

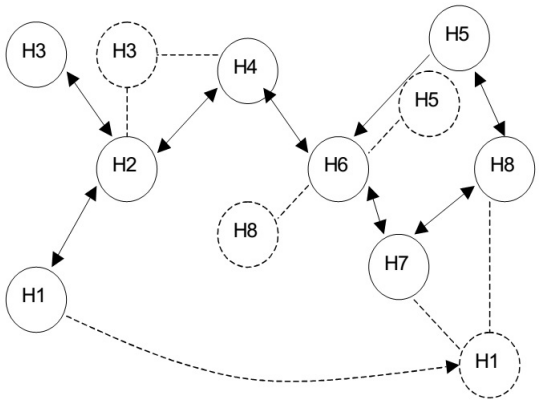


Figura 2.4: Exemplo da alteração da rede com o protocolo DSDV [6].

Dest	Next Hop	Metric	Seq.No.	Install
H1	H4	3	S406_H1	T001_H6
H2	H4	2	S128_H2	T001_H6
H3	H4	3	S564_H3	T001_H6
H4	H4	1	S710_H4	T002_H6
H5	H7	3	S392_H5	T001_H6
H6	H6	0	S076_H6	T001_H6
H7	H7	1	S128_H7	T002_H6
H8	H7	2	S050_H8	T002_H6

Figura 2.5: Tabela de *routing* do nó H6 no instante inicial [6].

A figura 2.4 mostra as alterações da rede durante o movimento dos nós móveis. A figura 2.4 é o exemplo da tabela de *routing* do nó H6 no momento antes ao movimento. A coluna *sequence number* da tabela representa o momento em que o caminho foi criado, assim, é possível determina quais são os

caminhos mais antigos e descarta-los, quando é recebida nova informação, no momento de atualização das tabelas. A adição desta coluna permitiu a eliminação de ciclos.

2.2.2 *Ad-hoc On-Demand Distance Vector Algorithm*

O *Ad-hoc On-Demand Distance Vector Algorithm* (AODV) surge como melhoria do DSDV, este protocolo é utilizado em ZigBee, contrariamente aos restantes protocolos apresentados, este é um protocolo reativo ([7, 8]). Os protocolos reativos esperam que seja necessária a comunicação por parte de um sensor para que as rotas sejam criadas. Os protocolos em que as rotas estão em constante atualização, são denominados de protocolos pró-ativos. Neste caso, os nós que não se encontrem em caminhos ativos, não mantêm nenhuma informação de *routing* nem participam em alterações periódicas das tabelas de *routing*.

Os nós não necessitam de descobrir e manter rotas para outros nós, enquanto os dois não necessitarem de comunicar, a não ser que este nó precise e oferecer os seus serviços como nó intermédio para a comunicação de outros dois.

Quando os nós móveis têm interesse em comunicar, cada nó pode tomar conhecimento dos nós vizinhos, através de várias técnicas, incluindo o envio de mensagens de ”olá”. As tabelas de *routing* dos nós sem vizinhos, são organizadas para otimizar o tempo de resposta a movimentos locais e providenciar respostas rápidas aos pedidos de formação de novas rotas. Os principais objetivos do algoritmo são:

- Enviar mensagens para descobrir vizinhos apenas quando necessário.
- Distinguir a gestão de conexões locais da manutenção geral da topologia.
- Disseminar informação acerca de alterações em conexões locais aos nós móveis que necessitem da informação.

O processo de descoberta do caminho é iniciado quando o nó fonte precisa de comunicar com outro nó, para o qual não tem informação na sua tabela. Cada nó mantém dois contadores, um número de sequência do nó e uma identificação de envio. A fonte inicia a descoberta do caminho através do envio de uma mensagem de pedido de rota (*Route Request* - RREQ). O RREQ contém identificadores que evitam o *looping* do pedido.

Cada nó que recebe o RREQ mantém a referência da fonte e o último *sequence number* conhecido para a fonte, à semelhança do DSDV. a referência da fonte é utilizado para manter informação acerca da rota de regresso até à fonte e o *sequence number* especifica o quanto recente é a rota.

Enquanto o RREQ viaja desde a fonte para os vários destinos, define automaticamente o caminho inverso de todos os nós para a fonte como é demonstrado na figura 2.6.

Eventualmente, o RREQ chegará a um nó que possua uma rota para o destino, esse nó determina a atualidade da rota comparando os números de sequência para o destino. Caso o número do pedido seja maior do que o número da sua rota, o nó não deve utilizar a sua rota como resposta, continuando o processo anterior até que encontre um nó que contenha o número de sequência maior ou igual.

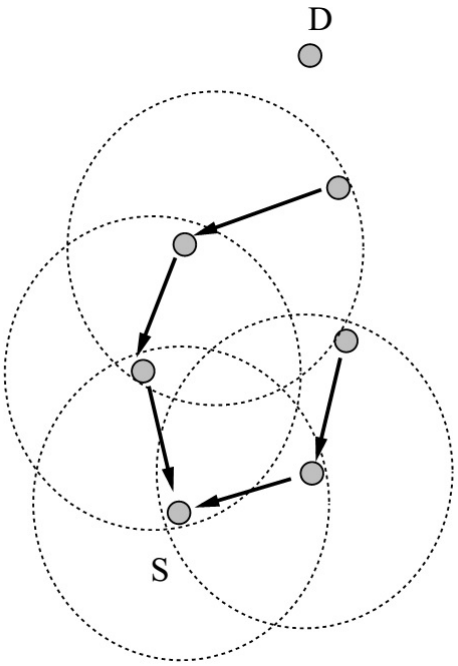


Figura 2.6: Formação do caminho inverso [8].

Assim que é atingido o destino, este envia uma mensagem de resposta para a fonte, pelo caminho inverso e cada nó guarda informação que permite definir a rota (figura 2.7).

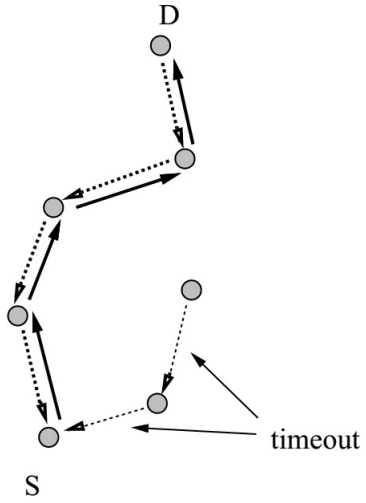


Figura 2.7: Formação do caminho [8].

O RREQ mantem um temporizador que elimina a mensagem ao fim de 3000 ms caso o pedido não atinja o destino.

2.2.3 *Ad-hoc On-Demand Multipath Distance Vector Algorithm*

O protocolo *Ad-hoc On-demand Multipath Distance Vector Routing* (AOMDV) [9] é uma extensão do AODV, que permite evitar quebras de ligação nos caminhos.

As entradas da tabela de *routing* contém a informação acerca do próximo salto, assim como, a contagem dos saltos correspondente. Todos os saltos seguintes contém o mesmo número de sequência, o que ajuda a manter o controle da rota. Para cada destino, os nós mantêm a contagem do número de saltos, que é usado como contagem máxima de saltos para cada caminho e é usada para enviar avisos de rota do destino.

A inexistência de ciclos é garantida por um nó, ao aceitar caminhos alternativos para o destino,apenas se estes tiverem uma contagem de saltos menor, que a de referência para esse destino.

Como é usado o número de saltos com maior contagem, a contagem de saltos de referência não se altera para o mesmo número de sequência. Quando é recebida uma rota para um destino com número de sequência maior, a lista de próximos saltos e a lista de saltos de referência são reiniciados.

O AOMDV pode ser usado para descobrir quebras nos nós ou quebras nas ligações das rotas. A ideia principal consiste em criar vários caminhos, entre emissor e recetor, durante o processo de criação das rotas.

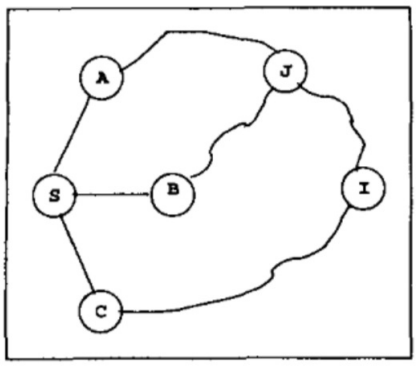


Figura 2.8: Exemplificação de 3 rotas criadas com recurso ao AOMDV [8].

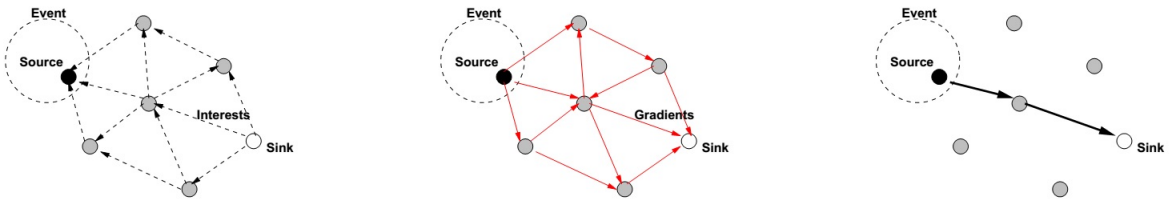
Com auxílio da figura 2.2.3 é possível explicar o processo de descoberta de quebra nas rotas. O nó S envia um pacote para I, sendo que A, B e C são vizinhos. O pacotes são enviados pelos três caminhos criados. J retransmite os pacotes recebidos, sendo que apenas um dos pacotes recebidos do caminho A ou B foi retransmitido. I recebe um pacote vindo de C e dois vindos de J. Uma vez que recebe apenas um pacote de J, existe uma quebra na ligação entre S-A-J ou S-B-J.

2.2.4 Protocolo *Directed Diffusion*

Outro protocolo que poderá ser útil é o *Directed Diffusion*, tem como objetivo estabelecer n caminhos eficientes entre uma ou mais fontes até ao destino. O protocolo *Directed Diffusion* é centralizado nos dados ([10]), ou seja, não necessita de conhecer a identificação dos nós para a comunicação, o que lhe interessa são os dados, sendo que, o nó que obtém esses dados não é significativo.

A figura 2.9 exemplifica o funcionamento do protocolo *Directed Diffusion*.

Suponhamos que o utilizador da rede tem como objetivo localizar um soldado numa região remota do campo de batalha. O utilizador subscreve ”localização do soldado”, especificada por um conjunto de



(a) Interest propagation

(b) Initial gradients set up

(c) Data delivery along reinforced path

Figura 2.9: Exemplo de funcionamento do protocolo *Directed Diffusion* [10].

atributos, os sensores ao longo da rede procuram informação acerca do soldado.

A aplicação do utilizador envia os dados através duma lista de atributos que caracterizam a tarefa a desenvolver. Os atributos descrevem a informação desejada especificando o tipo de sensores e possivelmente a localização destes. O nó do utilizador torna-se o *sink*.

O pedido de informação propaga-se de vizinho para vizinho até chegar ao nó na região especificada. A grande vantagem do protocolo *Directed Diffusion* é que todos os nós tomam conhecimento da tarefa, assim todos os nós guardam e interpretam o pedido, em vez de apenas o encaminharem.

Os nós, ao receberem o pedido, guardam a informação dos vizinhos que o requereram, para cada um destes vizinhos definem um gradiente. O gradiente representa as duas direções que os dados correspondentes a um pedido podem seguir e se o seu estado está ativo ou inativo. Depois de definido o gradiente o nó reenvia o pedido para os seus vizinhos.

Os sensores indicam quais são os dados que podem gerar, publicando um conjunto de atributos apropriados. Os sensores que satisfizerem os requisitos são ativados e começam a recolher dados.

Os nós intermédios coletam os dados enquanto estes se propagam até ao *sink*, estes dados são utilizados para diferentes objetivos para diferentes níveis de difusão. O mecanismo de difusão nuclear utiliza os dados capturados para suprimir mensagens duplicadas e prevenir *loops*. Pode existir agregação de dados, ou seja, os dados capturados por vários sensores podem ser fundidos numa única resposta.

O envio de dos dados no protocolo *Directed Diffusion* divide-se em duas fases. Na primeira fase os dados são marcados como dados de exploração e são enviados para todos os nós vizinhos que tenham gradientes correspondentes. Nesta fase o nó vai escolhendo um caminho preferencial, tendo por base os tempos de entrega das mensagens e reforça o peso desse caminho. Na segunda fase os dados são enviados apenas pelos caminhos reforçados, assim é possível diminuir a redundância.

Os dados exploratórios são enviados periodicamente para que possam ser mantidos os caminhos atualizados.

2.2.5 *Ripple Routing Protocol*

Outro dos protocolos de *routing* que tem características úteis é o *Ripple Routing Protocol* (RPL), este protocolo merece especial atenção, uma vez que se prevê que constituirá a base da *Internet* das

Coisas (*Internet of Things* - IoT), sendo definido pela *Internet Engineering Task Force* (IETF) ([11]). O RPL destina-se a redes de baixa potência e possíveis perdas de pacotes (*Low Power and Lossy Networks* - LLNs) e tem como objectivo criar um *Destination Oriented Direct Acyclic Graph* (DODAG) utilizando funções, conjuntos de métricas e restrições.

O processo de construção do grafo começa no nó raiz ou *LowPAN Border Router* (LBR), que é configurado como administrador do sistema. Podem existir múltiplos *roots* configurados no sistema. O RPL especifica um conjunto de mensagens novas ICMPv6 para alterar informação relacionada com o grafo. Estas mensagens são *DODAG Information Solicitation* (DIS), *DODAG Information Object* (DIO) e *DODAG Destination Advertisement Object* (DAO).

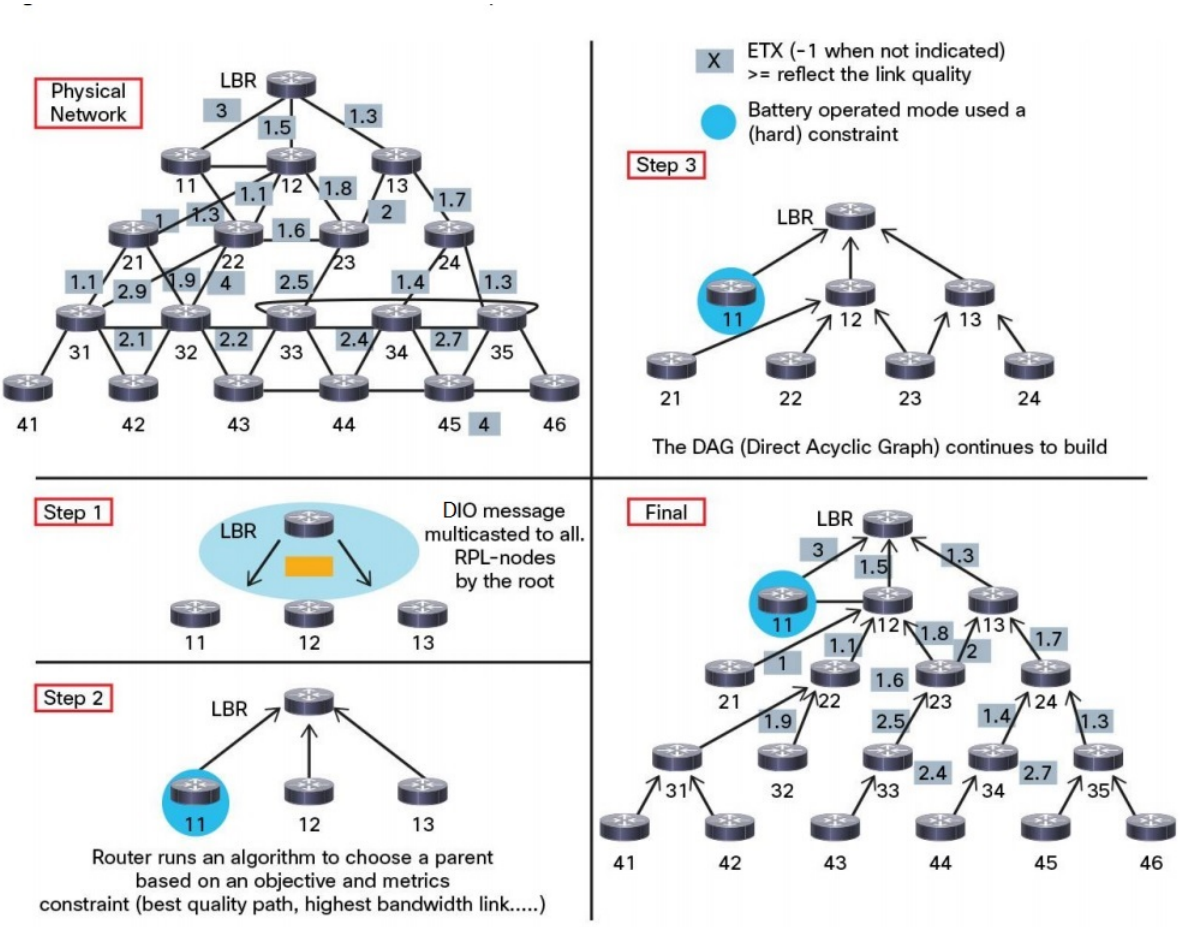


Figura 2.10: Processo de construção do grafo em RPL [12].

A raiz começa a enviar informação acerca do grafo utilizando mensagens DIO. Os nós vizinhos recebem e processam as mensagens DIO de vários nós e tomam decisões baseadas em regras, como a função objetivo, características DAG e custo do caminho para decidir se se juntam ao grafo. Assim que um nó se junta ao grafo passa a ter um caminho definido para si e o nó raiz passa a ser o nó ”pai”. O nó define um *ranking* para si próprio que indica as coordenadas do nó na hierarquia do grafo.

Este processo repete-se até que o grafo esteja completo, como está representado na figura 2.10.

As mensagens DIS são utilizadas pro-ativamente pelos nós para solicitar informação, dos nós vizinhos, acerca do grafo para que estes atinjam um estado estável.

O RPL permite a criação de múltiplas topologias de *routing* graças ao conceito de instâncias DODAG identificadas por ID. A ideia é construir e identificar múltiplos grafos na mesma topologia física. Podemos então definir vários caminhos de acordo com os objetivos de otimização e as restrições impostas.

Um nó pode ter apenas um identificador associado a um grafo, mas pode ter vários identificadores para diferentes grafos simultaneamente. Isto permite a construção de múltiplas topologias de *routing* numa rede, por exemplo, o tráfego menos crítico pode ser enviado por nós alimentados por baterias e o tráfego mais crítico pode seguir caminhos com menor latência.

A criação de múltiplos grafos permite que a rede crie novos grafos quando deteta a presença de um *jammer* nas imediações e encontrar alternativas de *routing*.

A capacidade do protocolo de *routing* reparar a topologia de *routing* quando uma falha ocorre é essencial. O RPL tem mecanismos para reparar os grafos quando ocorre uma falha na ligação, sendo necessário ter cuidados para que não seja ativado um *reset* em condições de transição.

O RPL especifica duas técnicas, que se complementam em natureza e ação. Quando a falha numa ligação ou nó vizinho é detetada e quando o nó não tem outro *router* na direção ascendente, é acionada uma reparação local (figura 2.11) para que seja rapidamente encontrado um novo ”pai”. Esta reparação é local e não tem influência em todo o grafo. Enquanto ocorrem estas reparações a forma do grafo pode desviar-se da ótima, a certo ponto pode ser necessário reconstruir todo o grafo, graças a um mecanismo que se chama ”reparação global”.

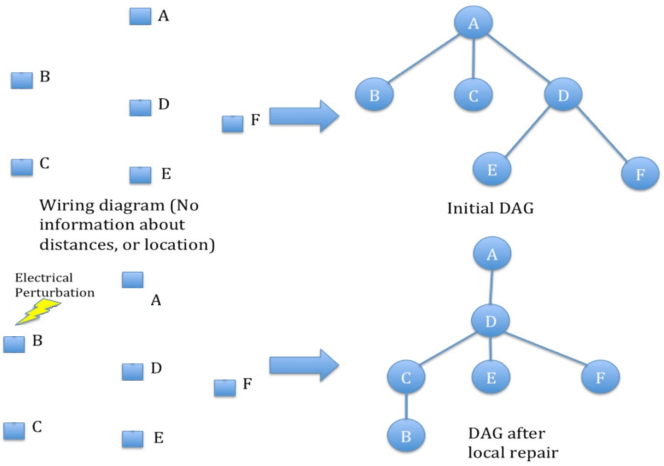


Figura 2.11: Exemplo de reparação local em RPL depois de uma perturbação [11].

A reparação global pode ser ativada apenas pela raiz, é uma técnica de otimização que tem custos elevados. A reparação global implica tráfego de controlo adicional na rede, cada nó tem de voltar a correr a função de seleção dos ”pais”.

Em LLNs, especialmente quando os nós são muito limitados em termos energéticos, é essencial limitar o tráfego de controlo na rede. A maioria dos protocolos de *routing* verificam o funcionamento dos nós periodicamente, validando a adjacência dos nós e mantendo as tabelas de *routing* atualizadas. Este processo é dispendioso em LLNs onde os recursos são escassos.

O RPL utiliza um mecanismo adaptado, o ”*trickle timer*”, este mecanismo controla o rácio de envio das mensagens DIO. Certos eventos são considerados como inconsistências na rede, como por

exemplo, *loops*, junção e saída de nós da rede.

O intervalo de tempo do *trickle timer* aumenta enquanto a rede se estabiliza, o que resulta em menos mensagens DIO enviadas na rede. Quando é detetada uma inconsistência na rede, os nós fazem *reset* ao *trickle timer* e as mensagens DIO voltam a ser enviadas com maior frequência.

A frequência de envio das mensagens DIO dependem da estabilidade da rede, por outras palavras, quanto mais estável for a rede menos mensagens do RPL são enviadas.

Uma vez que o objetivo é aplicar o RPL em redes de sensores militares, a segurança da rede é um fator que não pode ser descorado. A segurança é critica, mas a sua implementação em LLNs pode tornar-se impossível, pelo que não se justifica incluir segurança sofisticada na implementação do RPL. Pode-se usar segurança na camada ligação para preencher os requisitos necessários sem que seja necessário recorrer à segurança em RPL.

2.2.6 *Algoritmo de routing selecionado*

O impacto de um *jammer* é idêntico para todos os casos apresentados, com exceção do AODV. Ao serem afetadas pelo *jammer*, as Redes de Sensores sem Fios são obrigadas a redefinir as suas rotas, sendo que este processo de redefinição da rotas reinicia constantemente com a movimentação do *jammer*, aumentando o *overhead* e o atraso no envio dos dados. O AODV, por ser um protocolo reativo, apenas altera as rotas caso o *jammer* impossibilite a comunicação no momento em que esta for necessária.

Para a execução deste trabalho, optou-se pela utilização de um protocolo de *routing* proativo. Os protocolos proativos são vantajosos por terem um atraso no envio de dados esporádicos menor.

De entre os protocolos de *routing* estudados, o RPL seria o protocolo mais vantajoso a utilizar, no entanto, não existe implementação do mesmo em NS-3. Este fato, levou à utilização de uma versão simplificada do DSDV em que apenas o *sink* gera atualizações das rotas. Sendo que, esta alteração torna o DSDV mais parecido ao RPL, embora lhe continuem a faltar alguns mecanismos que tornam o RPL mais eficiente.

2.3 *Jamming*

Nesta secção são abordados alguns tipos de *jammers* (secção 2.3.1), métodos de deteção, alguns métodos de localização e *tracking* do *Jammer* (secção 2.3.2) e são apresentadas algumas contramedidas *anti-jamming* a nível das camadas física, MAC e Rede (secção 2.3.3).

2.3.1 Tipos de *Jammers*

Os *jammers* são aparelhos que têm essencialmente como principal objetivo inibir comunicações via radio.

Na guerra eletrónica são utilizados essencialmente quatro tipos de *jammers* ([13]), *spot jammer*, *sweep jammer*, *barrage jammer* e *reactive jammer*.

O *spot jammer* é um *jammer* que sabe exatamente a frequência rádio da rede alvo, atacando-a apenas nessa mesma frequência. Necessita de menos energia para operar, sendo o *jammer* mais eficiente e eficaz. No entanto, tem a desvantagem de a rede poder mudar de frequência (*channel surfing/ frequency hopping*) para evitar *jamming*.

O *sweep jammer*, contrariamente ao *spot jammer*, não sabe a frequência alvo, conhece o espectro de frequências mais provável e efetua saltos de frequência por esse espetro periodicamente ou aperiodicamente. Afeta a rede temporariamente, é menos eficaz e eficiente que o *spot jammer* mas pode atacar mais que uma rede e impõe restrições de liberdade ao salto de frequências por parte da rede alvo.

O *barrage jammer* cobre uma grande largura de banda do espectro rádio ao mesmo tempo. Este tipo de sistema deixa pouca margem à rede alvo para evitar *jamming* e permite bloquear várias redes ao mesmo tempo, requer uma potência muito elevada para manter a densidade espectral de *jamming*.

Os três modelos descritos até agora são modelos de *jammers* ativos, estes tentam bloquear o canal impedindo que circule tráfego. Os *jammers* ativos são normalmente mais eficazes por manterem o canal sempre ocupado, no entanto são métodos mais fáceis de detetar.

Uma abordagem alternativa consiste na utilização de métodos reativos.

O *reactive jammer* tem como objetivo afetar o envio e receção de mensagens, este mantém-se inativo durante o período em que não pressente transferência de mensagens, mas começa a transmitir sinais rádio assim que pressente atividade no canal. Este tipos de *jammers* têm a vantagem de serem mais difíceis de detetar.

Para além dos *jammers* indicados, que atacam a camada física, pode-se bloquear a comunicação através da ligação de dados enviando pacotes corrompidos e por violação dos protocolos.

O inimigo pode também construir um *jammer* para a camada MAC, utilizando um dispositivo *wireless* que simplesmente ignore o protocolo *Medium Access*. Por exemplo, usando o dispositivo para enviar pacotes repetidamente. Como consequência, todos os dispositivos com distância X irão pensar que o canal está ocupado e recusar a transmissão de dados.

2.3.2 Deteção, Localização e *Tracking* do *Jammer*

Existem várias métodos para detetar *jamming*. Nos protocolos MAC (*Media Access Control*), como o CSMA (*Carrier Sense Multiple Access*), cada nó espera que o acesso fique livre para que possa enviar os seus pacotes. O tempo médio que cada nó tem de esperar até que o acesso fique livre e disponível para transmitir é o CST (*Carrier Sensing Time*). O CST é calculado a partir do tempo em que o nó está disponível para transmitir e do tempo em que o nó tem o acesso livre para transmitir os seus pacotes. Na presença de um *jammer* o canal pode estar constantemente ocupado.

Os nós fixam uma margem a partir do valor do CST, caso esta margem seja excedida o nó considera que está a sofrer *jamming*. Este método apenas pode ser aplicado a redes que usem protocolo MAC baseado em deteção de disponibilidade e é incapaz de identificar ataques à camada física. A escolha da margem é um processo pouco preciso e computacionalmente pesado para ser processado no nó.

O rácio do número de pacotes enviados por um nó durante um período de tempo pelo número de pacotes que supostamente deveriam ser enviados nesse mesmo período de tempo é definido por PSR (*Packet Send Ratio*). Uma quebra neste rácio pode indicar a presença de um *jammer*, no entanto não a garante por si só.

O rácio de pacotes entregues, é calculado mantendo a contagem das confirmações de receção com sucesso e o número total de pacotes enviados pelo nó [14, 15].

Çakiroğlu and Özcerit [16] defendem que a energia consumida em excesso por um nó pode indicar a presença de um *jammer*, pode ser calculada através da descida de tensão da bateria, multiplicando o quadrado do seu valor pelo tempo da medição e dividindo pela resistência do nó. O *jammer* força o sensor a manter-se em *BACKOFF* mesmo se estiver em modo IDLE, causando um consumo de energia excessivo. No entanto existem diversos motivos pelos quais o nó pode gastar energia em excesso, como seja o aumento de tráfego nesse período.

A variação da potência do sinal recebido é um modo de deteção de *jamming*, a presença deste afeta a relação sinal-ruído. Se ao longo do tempo o nó coletar a potência recebida de dados legítimos, permite o cálculo de um desvio padrão para a receção de um sinal legítimo, assim podemos deduzir que quando a receção de um sinal não se encontrar dentro dos limites do valor de desvio padrão esse nó se encontra nas imediações de um *jammer*.

O indicador de potência do sinal recebido (*Received Signal Strength Indicator* - RSSI) permite a medição de potência do sinal numa ligação rádio e pode ser usado para localização, estimativa da qualidade do sinal e controle de potência. Este indicador é suportado pela maioria dos *transceivers* sem custos adicionais. As medições através do RSSI são muito fáceis de usar e têm consumos de energia muito baixos comparados com outros métodos.

Os métodos principais de processamento são baseados em aproximações estatísticas ou em técnicas geométricas de triangulação. O RSSI é utilizado em diversas aplicações, incluindo *tracking*. Em redes de sensores, os sistemas de *tracking* que utilizam técnicas rádio podem ser baseadas em medições de RSSI.

Os algoritmos de *tracking* baseados em medições de RSSI são normalmente compostos por dois passos:

- Utilização das medições de RSSI para estimar a distância entre dois nós que usam canais conhecidos ou por métodos de calibração *offline*. A calibração *offline* pode ser baseada numa tabela conhecida entre valores de RSSI medidos e distâncias. O segundo passo consiste em aplicar métodos estatísticos ou geométricos para obter a localização a partir da distância estimada.

- Aplicação de métodos estatísticos ou geométricos para obter a localização a partir da distância estimada.

O RSSI tem especial interesse para deteção se for utilizado um *reactive jammer* para atacar a rede. Caso a rede seja afetada por este tipo de *jammers*, a potência do sinal recebido sofrerá um aumento devido ao ruído introduzido. Com o RSSI é possível estimar a potência do sinal recebido no preciso momento em que existe comunicação, permitindo verificar se este valor se encontra dentro dos

parâmetros normais. Caso o valor seja demasiado elevado indica que poderemos estar na presença de um *jammer*. Obter o valor da potência do sinal quando não existe comunicação será inútil neste caso, uma vez que, o *jammer* está inativo neste período.

Os sistemas de *tracking* baseados em medições de RSSI têm vários desafios relacionados com a natureza do RSSI. As medições do RSSI são muito afetadas pelas variações dos meios *wireless*. Em caso de utilização em edifícios, as reflexões nas paredes resultam em múltiplas interferências no sinal recebido. Por sua vez, estas interferências podem levar a elevados erros de *tracking*.

É absolutamente necessário obter uma calibração precisa dos valores de RSSI em função da distância. As medições de RSSI são muito sensíveis, principalmente a longas distâncias, uma vez que o seu valor atenua em função desta. O valor do RSSI é também muito mais sensível a interferências quando os nós se encontram a longas distâncias.

A sensibilidade das medições de RSSI em relação ao meio e à distância podem ser diminuídas utilizando conhecimento anterior. O conhecimento prévio acerca das dimensões do ambiente permitem uma otimização das definições da potência de transmissão dos nós e uma melhor conversão do valor do RSSI. A informação acerca da velocidade de deslocamento permitem melhorar a taxa de transmissão [17].

Esta técnica pode ser adaptada para deteção de *jamming* uma vez que o RSSI nestas condições tem características específicas [18]. Com o intuito de perceber a forma como a potência do sinal do *jammer* afeta a comunicação na rede de sensores foram efetuados alguns testes que podem ser observados na secção 3.

Chimankar and Nandedkar [18] e Liu et al. [19] propõem um sistema de localização do *jammer* baseado na potência do sinal daquele, este sistema permitirá conhecer a posição dos sensores nos limites da zona afetada e, calculando a posição central a estes nós, aproximar a posição do *jammer*. A estrutura deste sistema é mostrado na figura 2.12.

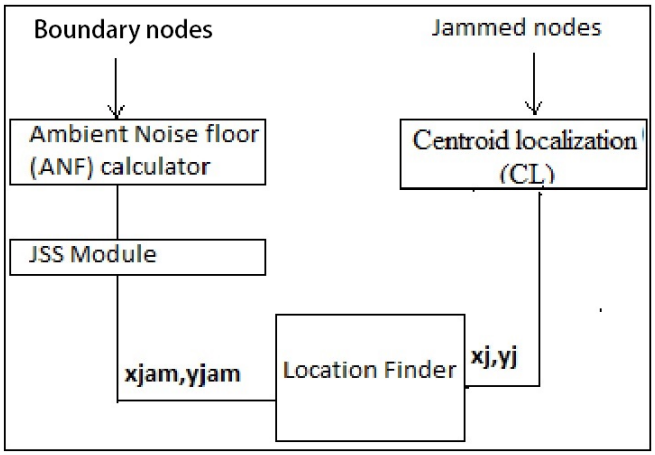


Figura 2.12: Sistema de localização de *jammers*. [18]

Para a localização do *jammer* é necessário ter conhecimento prévio da potência do seu sinal para diversas distâncias, este conhecimento permite conhecer uma série de possíveis localizações, que mesmo não sendo exatas, estarão próximas desta.

Conforme demonstra a figura 2.12 o sistema proposto é constituído por um calculador de ruído de fundo *Ambient noise floor - (ANF)*, por um módulo de medição de potência do sinal do *jammer* (JSS), por um módulo de localização central e um localizador.

O ANF é a soma de todo o ruído em espaço livre, incluindo o sinal do *jammer*. O ANF *calculator* calcula o valor do sa e o sc em que o sa é o ruído do ambiente quando apenas o sinal do *jammer* está ativo e o sc é o valor do ruído quando o sinal do *jammer* e da comunicação estão ambos presentes.

Quanto mais alto for o valor do ANF maior é a potência o sinal.

O módulo JSS calcula as coordenadas x e y dos nós no limite da zona afetada.

$$JSS(x_j, y_j) = \frac{x_1, x_2, x_3 ... x_n}{n}, \frac{y_1, y_2, y_3 ... y_n}{n} \tag{2.1}$$

Em que x_i e y_i são as coordenadas x_i e y_i dos limites respetivamente.

O módulo de localização central calcula as coordenadas dos nós centrais x_{jam} e y_{jam} da mesma forma que o módulo JSS calcula o dos limites.

Por ultimo, a média de cada um dos valores de x e y permitirá prever a localização do *jammer*.

$$AVG((x_j, y_j), (x_j, y_j)) = \frac{x_j, x_{jam}}{2}, \frac{y_j, y_{jam}}{2} \tag{2.2}$$

Este método está sujeito a obter desvios consideráveis quando os sensores têm uma distribuição pouco uniforme.

Yanqiang et al. [20] propõem outra solução para localização dos *jammers*, a utilização do algoritmo *Minimum-covering-circle* (MCCL). MCCL utiliza o conhecimento geométrico do círculo e estima o centro do círculo como a posição do *jammer*.

Quando ocorrem ataques de *Jammers* deixa de existir comunicação entre sensores na zona afetada, no entanto, existem várias técnicas desenvolvidas que podem permitir a comunicação mesmo entre estes nós. Para a implementação do MCCL assume-se que apenas os nós que se encontram no limite da zona afetada podem efetuar comunicação.

Este algoritmo não tem capacidade de detetar ataques de *jammers*. Para que possa ser aplicado este tem de ser detetado utilizando um dos métodos apresentados anteriormente.

Com o intuito de estimar a posição do *jammer*, o algoritmo MCCL começa por colecionar as coordenadas de todos os nós afetados. Assumindo que existem n nós afetados que conseguem enviar informação de localização, com notação:

$$Q = (X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n) \tag{2.3}$$

Podemos definir vários passos em MCCL com base nesta informação.

1. Definir o menor polígono que contem todos os pontos de Q (*Convex Hull*).

$$CH(Q) = (P_1, P_2, ..., P_m) \tag{2.4}$$

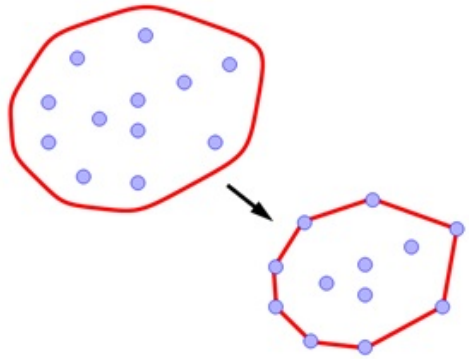


Figura 2.13: Convex hull.

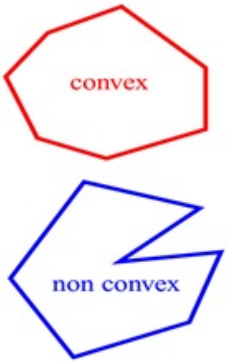


Figura 2.14: exemplo não convexo.

2. Calcular o diâmetro l de $CH(Q)$, para cada P_i e P_j, P_I e P_j são pontos simetricamente opostos, em que o ponto médio de l definido por O_1 é uma aproximação do centro do circulo. $\frac{|l|}{2}$ é o raio do circulo. Se $d(P_v, O_1) \leq \frac{|l|}{2}$, ($v \in 1, 2, ..., m, v$ diferente i, j), O_1 é a posição estimada para o *jammer*. Caso não se confirme continuamos para o passo 3.

3. Calcular a distância $d(P_u, l)$entre cada vértice de $CH(Q) = (P_1, P_2, ..., P_m)$ e l, onde $P_u \in P1, P2, ...,$ Pm, e $u \in 1, 2, ..., m, u \neq i, j$. Encontrar a distância máxima $d(P_u, l)$ e o u correspondente passa a ser tratado por k.

4. Definir o plano bissetor perpendicular a l e P_iP_k respetivamente, sendo que, de seguida estes 2 bissetores em trissetores em O_2 como se pode ver na figura 6. A distância $d(P_i, O_2)$ entre P_i e O_2 passa a ser definida por R_2.

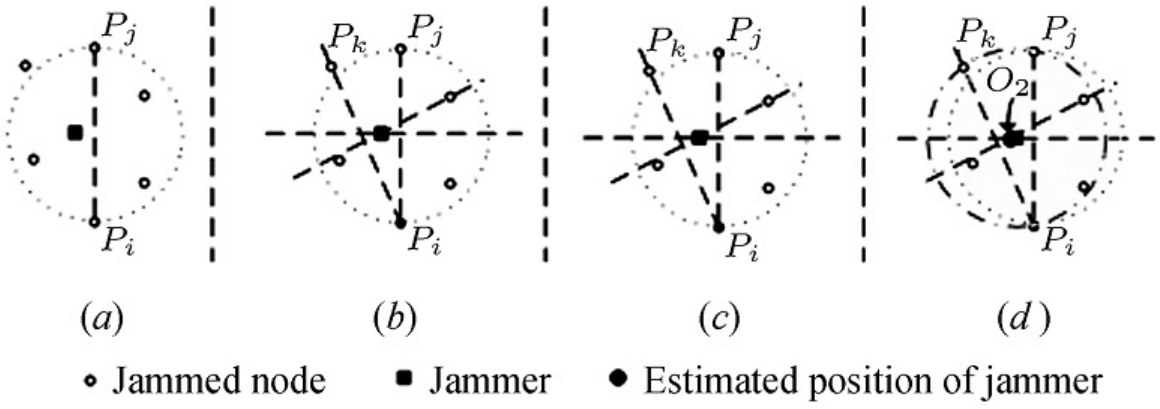


Figura 2.15: Algoritmo MCCL. [20]

5. Se $d(P_v, O_2) \leq R_2$, $v \in 1, 2, ..., m, v \neq i, j, k$, O_2 é a posição estimada para o *jammer* e o algoritmo termina. Caso exista k' que satisfaça a condição $d(P_{k'}, O_2) > R_2$, $k' \in 1, 2, ..., m, k' \neq i, j, k$, então substitui-se P_k por $P_{k'}$. Repete-se os passos 4 e 5 até que não exista k'.

   O MCCL termina dando O_2 como posição estimada para o *jammer*.

Para a elaboração deste trabalho assume-se a existência de um mecanismo de deteção do *jammer*, ainda que este não seja implementado. Esta informação é essencial para que se possam criar

algoritmos de *routing* baseados na distância geográfica entre *jammer* e nós.

2.3.3 Contramedidas *anti-jamming*

Nesta secção são exemplificadas algumas contramedidas *anti-jamming* possíveis de implementar a nível das camadas fisica, MAC e rede.

Camada Física

Frequency Hopping e *Spatial Retreats* são duas técnicas utilizadas como contramedida para evitar *jamming* na camada física.

A primeira estratégia, *Frequency Hopping* consiste em efetuar constantemente saltos na frequência durante a comunicação. Ambos os dispositivos devem conhecer qual a próxima frequência a ser utilizada e qual o momento em que mudar. Esta medida funciona caso o *jammer* bloqueie apenas uma frequência (*spot jammer*). Os rádios estão constantemente a modificar a frequência em que operam, mesmo que não exista ameaça.

A segunda estratégia apresentada é *Spatial Retreats*. Esta estratégia é uma opção para nós moveis, que consiste essencialmente em mover os nós para uma posição segura, quando o *jammer* é detetado. A chave para o sucesso deste método é a decisão da posição para que os nós se devem mover e a coordenação dos seus movimentos.

Camada MAC

W. Xu [21] apresenta duas contramedidas para evitar jamming na camada MAC: *Channel Surfing* e *Spatial Retreats*, já apresentada como defesa para a camada física.

A primeira estratégia apresentada é *Channel Surfing*. Tipicamente, os dispositivos rádio utilizam um canal único para comunicar. Quando existe uma interferência que bloqueia o primeiro canal, a solução lógica passa por migrar para outro canal. A ideia de *Channel Surfing* é motivada por, uma técnica comum na camada física, saltos de frequência.

Esta técnica funciona se o *jammer* bloquear apenas um canal de cada vez e se o *jammer* não se fizer passar por um membro da rede, ou seja, se não conhecer nenhuma chave de rede. É necessário que os nós conheçam o canal alternativo antes de sofrerem *jamming*, uma vez, que, após serem bloqueados deixam de conseguir comunicar.

É também necessário que os nós tenham capacidade de detetar o *jammer*. Uma vez detetado o *jammer*, o nó contacta os seus vizinhos através do canal alternativo para que estes sejam informados da nova política de canal. Cada dispositivo deve manter um canal alternativo ativo para os seus vizinhos.

A figura 2.16 demonstra a utilização do canal alternativo por parte nós afetados pelo *jammer*. Os restantes nós comunicam co canal normal.

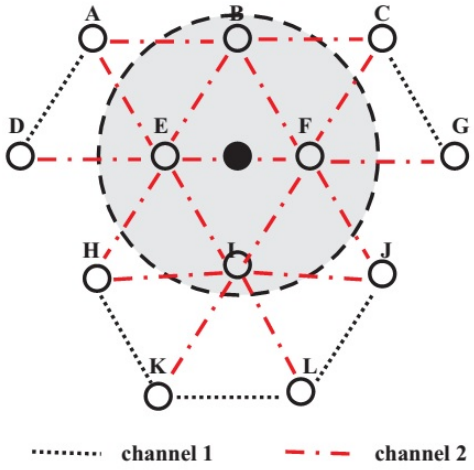


Figura 2.16: Exemplo de *Channel Surfing*. [21]

Camada Rede

Pretende-se com o presente trabalho, melhor o desempenho de uma Rede de Sensores sem Fios, quando esta é afetada por um *jammer*, desenvolvendo um algoritmo de *routing*. Tendo em atenção o objetivo do trabalho, o estudo de contramedidas a nível da camada Rede tem especial importância.

B. Kan et al. [22] apresenta o *Interference Activity Aware Multi-path Routing Protocol* (IAMP), este consiste numa variação do AOMDV apresentado na secção 2.2.3.

Os autores utilizam informação dinâmica do *jammer* específica para minimizar o impacto sobre a disponibilidade dos caminhos. Esta abordagem permite descartar os caminhos mais afetados pelo *jammer*. A interferência é utilizada como medida para a dinâmica do jammer e esta métrica é utilizada em conjunto com a contagem dos saltos como métrica para a seleção do caminho.

Com intenção de evitar o excesso de *broadcasts* e incorporar as dinâmicas do *jammer* para escolher os caminhos mais fiáveis, no IAMP, é utilizada uma estratégia para descobrir a rota baseada em prioridades, na qual, é adicionada uma métrica de prioridade aos candidatos. É adicionada uma prioridade mais alta ao RREQ para candidatos com interferência mais baixa.

Com a utilização deste mecanismo, o nó menos afetado pelo *jammer* tem maior probabilidade de definir o caminho critico. No entanto, nem todas as interferências detetadas são provocadas por *jammers*, a própria comunicação em simultâneo, por parte dos nós, pode causar interferência nos nós vizinhos, o que aumentaria o tempo de entrega dos pacotes desviando-os por rotas alternativas sem que existisse necessidade.

Muraleedharan et al. [23] apresenta um algoritmo baseado em sistemas de formigas e em inteligência de enxames, este algoritmo modela o comportamento social de insetos coletivos, como formigas e abelhas. O sistema de formigas é uma evolução da inteligência de enxames, é um algoritmo revolucionário com características únicas, como robustez, resolução de problemas distribuídos, versatilidade e capaz de resolver problemas distribuídos. O sistema de formigas resolve qualquer problema de otimização convexo.

O sistema adapta-se a redes com alterações ambientais. Os agentes do sistema comunicam, quer seja direta ou indiretamente na resolução de um problema distribuído. Os agentes movem-se em direção à solução ótima e comunicam diretamente partilhando conhecimento com os seus vizinhos.

O primeiro grupo de agentes atravessam os nós de forma aleatória, e uma vez que cheguem ao destino, estes depositam feromonas nos caminhos como meio de comunicação indireta com as outras formigas. O número de feromonas deixadas pelas formigas anteriores aumenta a probabilidade de a mesma rota ser escolhida na mesma iteração. As feromonas evaporam com o tempo, o que previne soluções não otimizadas de dominarem inicialmente.

Este sistema tem capacidade de evitar problemas na rede, por exemplo, se um nó perder a energia, os agentes deixam de o atravessar e são adicionados novos caminhos, para evitar esse nó. Assim, a comunicação continua sem o sensor degradado. Estes agentes garantem a rota otimizada para o destino, utilizando recursos limitados e aprendendo a ambiente da rede.

Inicialmente, a capacidade de computação exigida é elevada, mas esta, cai drasticamente assim que os agentes aprendem a rede e o ambiente.

Cada agente mantém o conjunto de nós que visitou numa lista. A formigas visitam os nós da Rede de Sensores sem Fios independentemente do número de saltos e nunca revisitam um nó durante a mesma viagem. A atualização das tabelas é feita apenas no final da viagem.

Mpitziopoulos et al. [24] apresenta um algoritmo que evita o itinerário afetado pelo *jammer*, (*Jamming Avoidance Itinerary Design algorithm*, JAID). O algoritmo JAID contém um processo de escolha de caminhos baseado em grafos. Durante o processo de escolhas do caminho, cada nó recebe um custo, que é função das perdas da ligação.

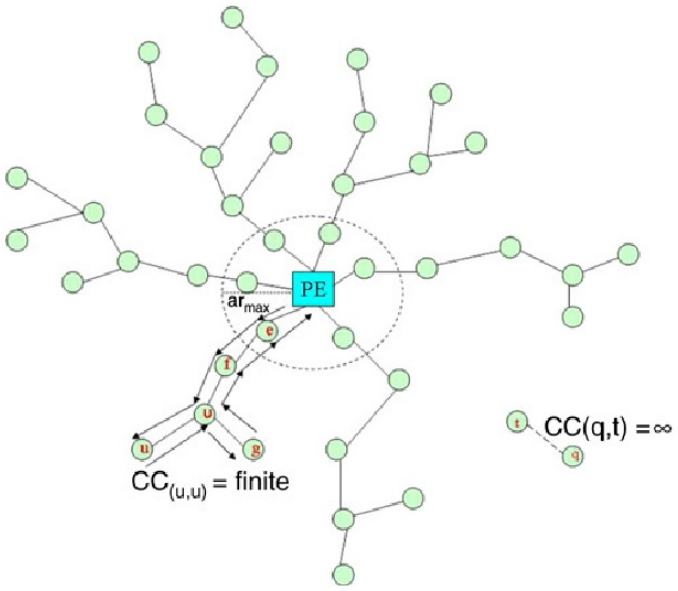


Figura 2.17: Construção dos itinerários pelo JAID. [24]

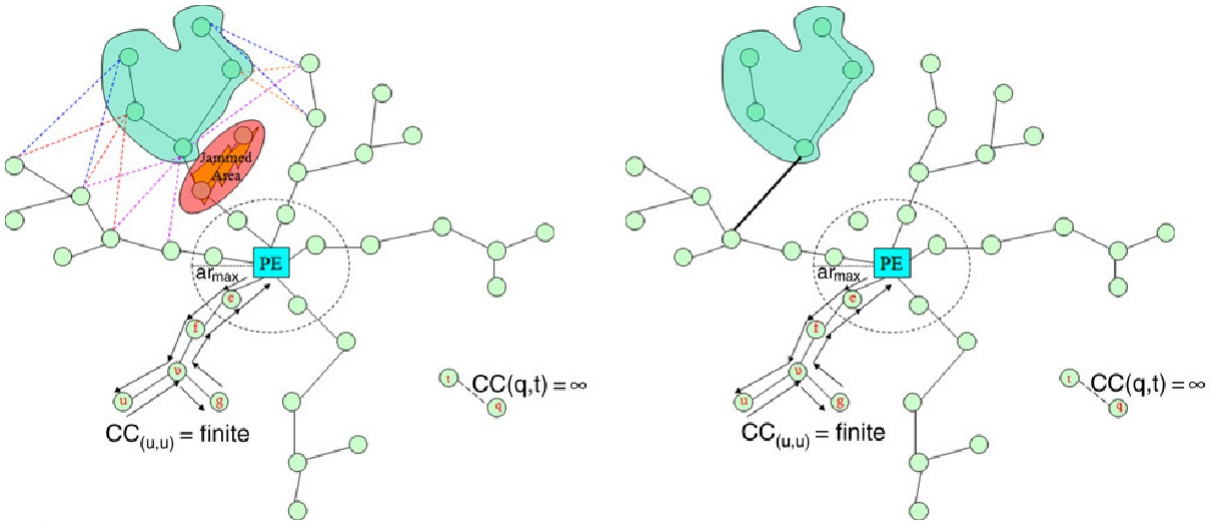
A execução do algoritmo JAID compreende-se em três fases:

Na fase inicial o elemento de processamento (PE), conecta-se a todos os nós ao seu alcance (figura 2.17). Estes nós representam o ponto inicial dos itinerários dos agentes móveis (MA). Definindo o

número de nós igual ao número de MA.

Na segunda fase, os novos nós são adicionados às árvores formadas inicialmente. Durante o processo de estabelecimento da conexão, devem ser tidas em conta duas regras básicas: o candidato não deve ter estabelecida nenhuma ligação com outra árvore e o nó candidato deve ter uma ligação a outro nó que providencie um caminho para o PE. Estas duas regras garantem que as ligações são estabelecidas a partir dos nós mais próximos do PE, para os mais afastados.

A terceira fase, é executada durante um evento de interferência rádio ou ataque por *jamming*. Primeiro, é utilizado um algoritmo de mapeamento da área afetada pelo *jammer* [25] e são identificadas as árvores, com alcance para os nós no perímetro dessa zona, que perdem a ligação (figura 2.18(a)).



(a) Área jammed e ligação a partir da árvore cortada.

(b) Seleção do caminho a partir da árvore cortada.

Figura 2.18: Escolha do caminho pelo JAID. [24]

O algoritmo JAID escolhe ligar as árvores, com a ligação perdida, a nós ao alcance, por forma, a diminuir o custo total da formação de itinerários (figura 2.18(b)).

No caso de o número de nós afetados pelo *jammer* ser maior que 20%, do número total de nós, então, o algoritmo JAID limita-se a reconstruir todos os itinerários de raiz, assegurando-se que os novos caminhos não cruzam a área afetada pelo *jammer*.

Este algoritmo de *routing* não tem em consideração a movimentação do *jammer*, seria útil prever o sentido de deslocamento deste, para não criar rotas alternativas que não estejam prestes a ser afetadas.

Capítulo 3

Descrição do Trabalho

O trabalho consiste, essencialmente, na criação de uma variante do protocolo DSDV, que permitam obter uma melhoria de pacotes recebidos no *sink*, em relação ao protocolo DSDV original, quando a Rede de Sensores é afetada por um *Jammer*.

Neste capítulo, na secção 3.1, será descrito o algoritmo de encaminhamento utilizado. Na secção 3.2 serão apresentados os resultados obtidos nos testes que analisam a potência recebida do *jammer* a diferentes distâncias.

Na secção 3.3 é descrito o modelo de propagação do *jammer* que, posteriormente, será inserida no simulador.

O trabalho foi elaborado essencialmente em ambiente de simulação, pelo que será elaborada uma pequena abordagem ao simulador utilizado na secção 3.4, assim como a definição dos parâmetros de simulação necessários para tornar a simulação próxima da realidade.

3.1 Algoritmo de Encaminhamento Utilizado

O algoritmo de Encaminhamento criado teve como base o DSDV. Numa primeira fase simplificou-se o DSDV, para que este apenas criasse as rotas para o *Sink*, sendo este o único a gerar *updates* e diminuindo o *overhead* da Rede de Sensores sem Fios.

O DSDV, tal como muitos outros algoritmos de encaminhamento, utiliza uma política de custos como métrica para escolha da rota. Sendo que, esta consiste no menor caminho, em termos de *hops*, até ao *sink*.

Antes de chegar ao algoritmo utilizado, testaram-se três opções distintas:

- Numa primeira abordagem, optou-se por adicionar uma métrica temporal, para além da métrica de custos, que atuava como escolha da rota em caso de igualdade do número de *hops*. Cada nó guardava o instante temporal em que era afetado pelo *jammer* e, durante a verificação do menor caminho para o *Sink*, era também verificada esta métrica temporal. Nos caso de caminhos com o mesmo tamanho até ao *Sink*, optava-se por escolher os nós que tivessem sido afetados pelo *jammer* há mais tempo, no máximo de 20 minutos.

Com esta métrica dava-se preferência a caminhos por nós que já tivessem sido afetados, estando estes na zona contrária ao sentido de deslocamento do *jammer*.

- Numa segunda fase, testou-se o mesmo algoritmo já apresentado anteriormente, mas com possibilidade de optar por um caminho mais longo num salto. Este algoritmo foi descartado por não garantir a inexistência de ciclos.
- Por último, testou-se o mesmo algoritmo com uma métrica de distância, em vez de métrica temporal. Nesta solução, optou-se por escolher o caminho com maior distância ao *jammer*, em caso de igualdade de saltos até ao *sink*.

Estes algoritmos obtiveram melhores resultados em testes de simulação que o DSDV original, no entanto, a melhoria em termos de pacotes recebidos, foi sempre menor que 10%.

Uma vez que os resultados obtidos não foram satisfatórios, optou-se por usar um algoritmo em que a métrica é a distância ao *jammer*. Para este algoritmo é necessário conhecer previamente a posição do *jammer*. O algoritmo foi criado tendo como suposição que a posição geográfica do *jammer* seria obtida por um dos algoritmos apresentados no Capítulo 2 e, por este motivo, não foi implementado nenhum algoritmo de localização do *jammer* em NS-3.

A implementação do algoritmo consistiu em substituir a métrica de escolha da rota do DSDV, que consiste apenas na escolha do caminho com menor número de *hops*, por outra que tenha em atenção tanto o número de *hops*, como a distância dos nós ao *jammer*. A implementação está de acordo com o diagrama da figura 3.1.

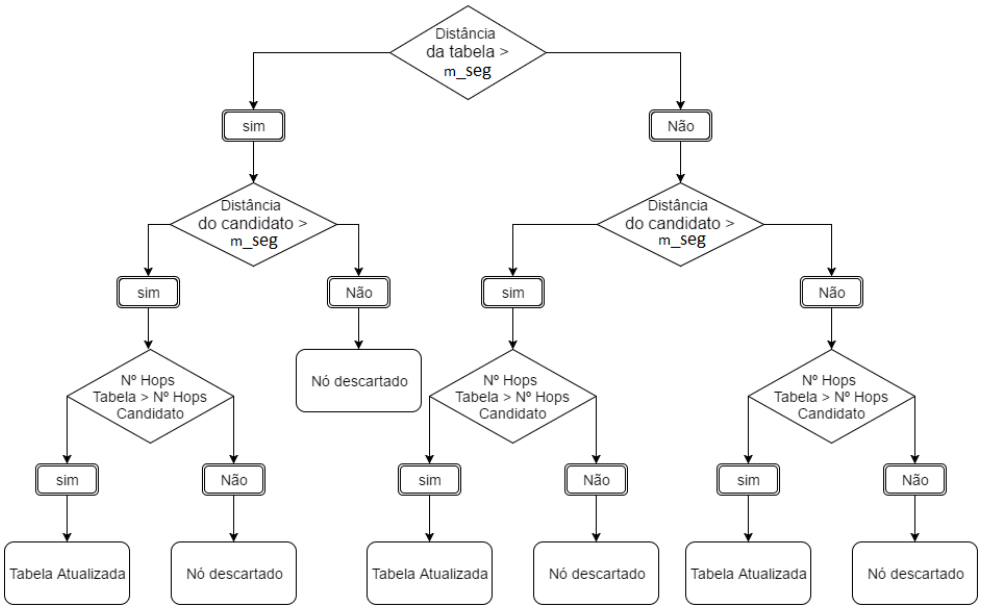


Figura 3.1: Diagrama do algoritmo implementado.

Na solução implementada dá-se preferência à distância ao *jammer* como escolha da rota. Caso seja possível o algoritmo encaminha os pacotes por rotas em que os sensores estão afastados do *jammer* numa distância superior à margem de segurança (m_seg). Inicialmente verifica-se se a distância do nó, já presente na tabela, ao *jammer* é superior à margem de segurança. Caso se verifique, é testada a

distância do *jammer* ao nó candidato a fazer parte da rota e são comparados os números de *hops*, até ao destino, entre o nó já presente na tabela e o nó candidato. Por fim, verificadas as duas condições e sendo o número de *hops* do nó candidato menor, este substitui o nó antigo na tabela de *routing*. Por outro lado, caso não se verifique a segunda condição ou a redução no número de *hops* o nó candidato é descartado.

Caso não se verifique a primeira condição (distância do nó já presente na tabela de *routing* superior à margem de segurança), se a distância ao *jammer* for maior que a margem de segurança e o número de *hops* menor ou igual que os do nó inserido na tabela, procede-se à atualização da tabela. Caso contrário o nó candidato é descartado.

Por último, se não for possível obter um nó com uma distância superior à margem de segurança, opta-se por utilizar o DSDV, ou seja, apenas a escolha por número de *hops*.

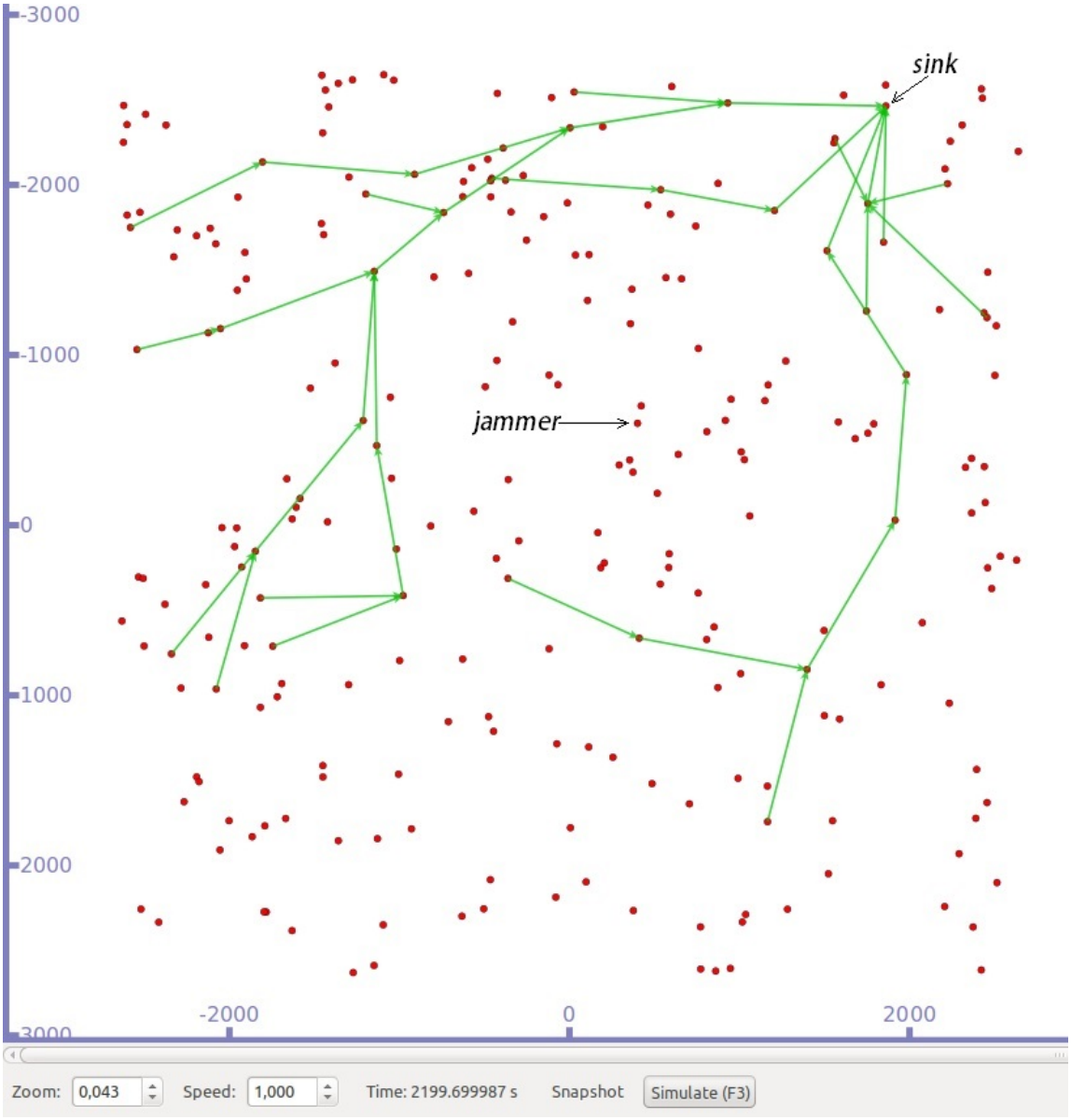


Figura 3.2: Simulação do algoritmo desenvolvido, no momento em que é aplicada a contramedida ao *jammer*.

A figura 3.2 representa uma simulação em NS-3, onde é aplicada a contramedida descrita. Como se pode observar, quando possível, as rotas são desviadas por nós que se encontram a uma distância superior à margem de segurança.

Importa referir que, como se considerou a utilização de *jammer* em versão *manpack*,a velocidade de deslocamento é baixa e, por este motivo, o protocolo DSDV se comporta bastante bem quando a atualização das rotas se dá em períodos de tempo muito curtos. Com períodos de atualização curtos a Rede de Sensores sem Fios reconstrói as tabelas de *routing* constantemente, dando pouco tempo ao *jammer* para afetar as rotas estabelecidas antes que sejam criadas alternativas.

Apesar de a diminuição do período de atualização das rotas ser uma boa solução, é muito dispendiosa em termos de gasto de energia por parte dos sensores e de *overhead* provocado pelos pacotes de atualização.

Relativamente à dimensão dos pacotes enviados, não existe qualquer alteração em relação ao tamanho dos pacotes transmitidos, utilizando o protocolo DSDV original ou o algoritmo de *routing* desenvolvido, uma vez, que no algoritmo criado apenas se acrescenta a distância do sensor ao *jammer* como métrica e, estando os nós sensores estáticos, esta pode ser previamente conhecida.

3.2 Testes para modelação do *Jammer*

Com intuito de estabelecer um modelo de propagação para o *Jammer*, estabeleceu-se contacto com o Major de Transmissões Lopes, do Regimento de Transmissões, que, por sua vez, se disponibilizou para realizar testes recorrendo a um analisador de espectros.

Durante os testes, o *jammer* foi ligado com uma potência de emissão de -25 dB e foram registados os valores medidos pelo analisador de espectro para várias distâncias. [1] Apesar de -25 dB não ser a potência máxima do *jammer*, é possível, a partir desta, extrapolar para valores diferentes.

A tabela 3.1 mostra os valores obtidos para a potência do sinal a várias distâncias do *jammer* estando este a emitir a uma potência de -25 dB. Mostra também a extrapolação obtida para diferentes potências de emissão. Estes dados foram obtidos utilizando um analisador de espectro na faixa de 869.4-869.65 MHz. Os testes foram realizados num espaço amplo, ainda que possam ter existido reflexões devido à presença de edifícios nas imediações, e em condições atmosféricas de aguaceiros ligeiros.

A utilização desta largura de banda deve-se ao fato de esta ser a banda de funcionamento do módulo XBee868Pro, que é bastante utilizado em redes de sensores.

Durante a execução dos testes, o *jammer* não estava a emitir na sua potência máxima. Em operações militares, a potência de emissão do *jammer* está diretamente relacionada com a necessidade de proteção da força e autonomia da bateria, sendo que, para operações de longa duração, é recomendável utilizar potências mais baixas. A extrapolação para diferentes valores de potência de emissão pretende cobrir variadas situações possíveis, uma vez que o *jammer* permite a regulação da potência de emissão.

[1] O *Jammer* utilizado durante os testes não pode ser indicado por questões de confidencialidade.

Tabela 3.1: Potência recebida (dB) em função da distância para diferentes potências de emissão.

distância (m)	Pe = -25 dB	Pe = -20 dB	Pe = -15 dB	Pe = -10 dB	Pe = -5 dB	Pe = 0 dB
3.00	-25,20	-20,20	-15,20	-10,20	-5,20	-0,20
5.00	-29,34	-24,34	-19,34	-14,34	-9,34	-4,34
7.50	-32,45	-27,45	-22,45	-17,45	-12,45	-7,45
10,00	-38,86	-33,86	-28,86	-23,86	-18,86	-13,86
15,00	-43,86	-38,86	-33,86	-28,86	-23,86	-18,86
20,00	-57,80	-52,80	-47,80	-42,80	-37,80	-32,80
25,00	-62,76	-57,76	-52,76	-47,76	-42,76	-37,76
30,00	-65,78	-60,78	-55,78	-50,78	-45,78	-40,78
35,00	-71,42	-66,42	-61,42	-56,42	-51,42	-46,42
40,00	-74,78	-69,78	-64,78	-59,78	-54,78	-49,78

As extrapolações foram obtidas tendo como base a equação 3.1 para o modelo Logarítmico de Perdas de Percurso (*Log - Distance path loss model*) de rádio propagação:

$$\begin{aligned} PL &= PL_0 + 10 \cdot \gamma \cdot \log(\frac{d}{d_0}) + X_g \\ PL &= P_{Tx} - P_{Rx} \end{aligned} \tag{3.1}$$

- $P_{Tx} =$ Potência Transmitida
- $P_{Rx} =$ Potência Recebida
- Distância de referência - $d_0 = 1m$
- $PL_0 = Friis(d_0)$
- Log-normal Shadowing: $X_g = N(\sigma)$
- Fator de perda de pacotes: γ
- No modelo de Friis para atenuação em espaço livre: $\gamma = 2$ e $X_g = 0$
- Para propagação com atenuação sem ser em espaço livre: tipicamente $\gamma > 2$

Como se pode observar, a potência recebida aumenta proporcionalmente à potência emitida, uma vez que todos os restantes fatores são constantes nos testes efetuados.

Na figura 3.3 estão representados graficamente os valores obtidos em teste e a reta de tendência $y = -1.3889x - 23.767$ gerada automaticamente pelo Excel.

Como era de esperar por observação da equação 3.1, o gráfico correspondente aos testes efetuados aproxima-se a uma reta. A partir dos valores testados é possível calcular o valor de γ. PL_0 depende

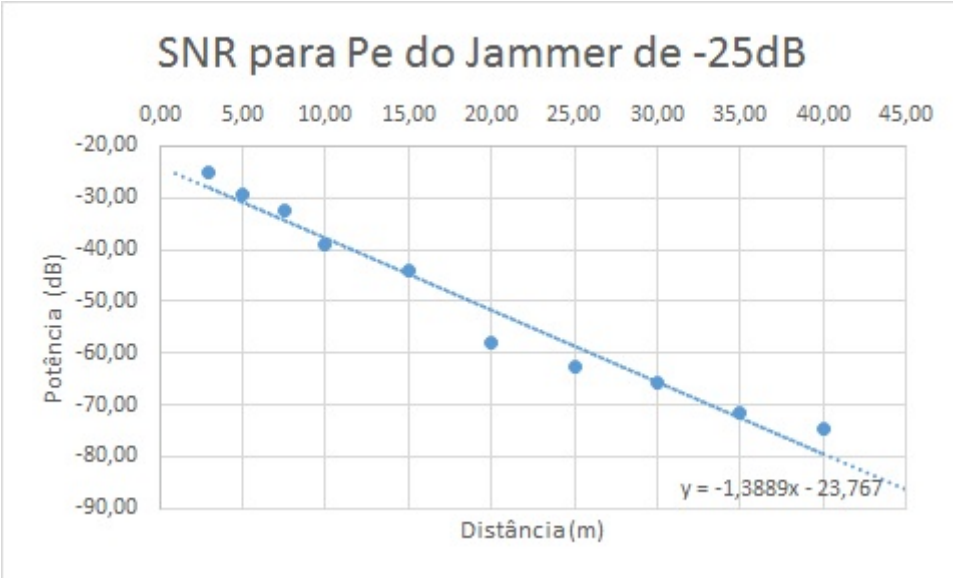


Figura 3.3: Potência recebida em teste em função da distância.

do valor da potência recebida e da potência de emissão para uma distância d_0 de 3 m, tendo-se obtido o valor de 0,2 dB.

Após a obtenção do valor de PL_0, utilizou-se a potência de emissão e receção para uma distância de 35 m para obter o valor de PL e, consequentemente, γ, uma vez que esta potência recebida está mais próxima da reta de tendência gerada pelo Excel. O γ obtido tem o valor de $4,33$.

O valor de γ obtido indica que o sinal sofreu atenuações diversas, o que era expectável, visto que as condições atmosféricas não eram as ideais, para além da presença do fenómeno de reflexão. Note-se que o γ é 2 para o modelo de propagação em espaço livre.

A obtenção do valor de γ aliado ao facto dos valores obtidos em testes se aproximarem de uma reta (figura 3.3), permite afirmar que os valores da potência de receção são extrapoláveis para distâncias maiores, utilizando na equação 5 o valor de γ calculado.

Após a análise dos dados, e admitindo que o módulo a utilizar será o XBeePro868 [26], é possível definir a distância do *jammer* aos nós, por forma a que estes consigam comunicar.

O fabricante do módulo XBeePro868 define um alcance máximo de 40 Km para comunicação, uma largura de banda do módulo de 24 kbit/s e uma sensibilidade na receção de 1%, para uma potência recebida de -112 dBm.

O Fabricante disponibiliza também um *white paper*, onde demonstra testes de alcance efetuados com o módulo XBeePro868.

Os testes de alcance foram efetuados utilizando antenas com ganho de 2,1 dBi. Estes concluem que uma atenuação adicional de 11 dB, para uma distância de 40 km, é o valor aceitável para conseguir comunicar, tendo obtido, com sucesso, 96,9% dos pacotes (Tabela 3.2). É importante referir que, durante a elaboração destes testes, os nós não sofriam qualquer atenuação devido a obstáculos.

Os valores disponibilizados pelo fabricante serão úteis para a definição do cenário, permitindo, através de cálculo, obter valores para a distância entre nós que irá possibilitar a comunicação du-

Tabela 3.2: Testes de alcance do XBeePro868 [26].

868 MHz Dipole Antenna at 40 km	
Variable Attenuation	Percentage of Good Packets
0 dB	100%
9 dB	100%
10 dB	100%
11 dB	96.9%
12 dB	63%
13 dB	32%
15 dB	5%
20 dB	0%

rante a presença do *jammer*. A potência do sinal do *jammer* será somada ao ruído térmico do circuito, por forma a descobrir qual a redução da distância entre nós que permita anular o efeito deste ruído adicional e manter a comunicação destes. Desta forma, os valores obtidos dependerão da distância entre o *jammer* e os nós.

3.3 Modelo do Rádio dos Sensores

O NS-3 não contém nenhum módulo de simulação do *jammer* nas versões mais recentes, sendo necessário criar um modelo matemático que permita inseri-lo na simulação. Tendo como base os valores fornecidos pelos testes dos fabricantes do módulo XBeePro868 e os resultados obtidos dos testes com o *jammer*, é possível criar o seguinte modelo.

A Relação Sinal-Ruído (*signal-to-noise ratio*, SNR) é dada pela fórmula 3.2.

$$(\frac{C}{N}) = P_{Sinal} - P_{Ruido}\,. \tag{3.2}$$

Tendo a equação 3.3 para a potência do sinal e a equação 3.4 para o ruído térmico:

$$P_{Sinal} = P_E + G_E + G_R - A\,. \tag{3.3}$$

Em que:

- $P_E[dB] =$ Potência de emissão.
- $G_E[dB] =$ Ganho de emissão.
- $G_R[dB] =$ Ganho de receção.
- $A[dB] =$ Atenuações.

$$N_0 = K_B{\cdot}\, T_{[K]}{\cdot}\, b_{rf}\,. \tag{3.4}$$

Em que:

- $N_0 =$ Ruído Térmico
- $K_B = 1.38 \cdot 10^{-23} J/K$ (Constante de Boltzmann)
- $T_{[K]} = 290k$
- $b_{rf} = 24kb/s$ (Largura de banda do módulo XBeePro868)

Obtém-se:

$$(\frac{C}{N}) = P_E + G_E + G_R - 10LOG(N_0) - A\,. \tag{3.5}$$

A atenuação é dada por:

$$A = A_0 + A_{At} + A_{Obs}\,. \tag{3.6}$$

Em que:

- $A_0[dB] =$ Atenuação em espaço livre.
- $A_{At}[dB] =$ Atenuação atmosférica.
- $A_{Obs}[dB] =$ Atenuação de Obstáculo.

A atenuação em espaço livre é dada por:

$$A_0 = 32.4 + 20LOG(d_{[km]}) + 20LOG(f_{[MHz]})\,. \tag{3.7}$$

A atenuação da atmosfera é dada por:

$$A_{At} = d * (\gamma_{o0} + \gamma_{w0})\,. \tag{3.8}$$

Em que:

- $d =$ Distância da ligação.
- $\gamma_{o0} =$ Atenuação devido a gases.
- $\gamma_{w0} =$ Atenuação devido ao vapor de água.

Utilizando os valores fornecidos pelo fabricante:

- $P_E = -5\,dB$
- $G_E = G_R = 2.1\,dBi$
- $d = 40\,km$
- $A_{Obs} = 11\,dB$

Em que 11 dB foi o valor considerado como razoável para a atenuação de obstáculos, obtido nos testes do fabricante.

Para a atenuação devido a gases (γ_{o0}) e devido ao vapor de água (γ_{w0}) utilizou-se $0.0075\,dB/Km$ e $0.003\,dB/Km$, respetivamente. Estes valores são considerados valores típicos para Portugal continental.

O resultado obtido através dos cálculos para a relação Sinal-Ruído foi de 24.7383 dB, sendo este o valor mínimo que a ligação terá de ter para que a comunicação entre os sensores possa existir.

Tendo em consideração os valores obtidos nos testes com o *jammer*, é possível obter o valor da relação Sinal-Ruído da ligação somando a potência de receção, provocada pelo *jammer*, como sendo um valor de ruído, adicional ao ruído térmico. A adição de ruído obriga a que exista uma redução de distâncias entre nós da Rede de Sensores, para que se mantenha a relação Sinal-Ruído, garantindo a continuidade da comunicação.

A introdução de ruído adicional na ligação provoca uma redução na relação Sinal-Ruído da ligação. A redução da distância entre nós permite diminuir as atenuações e, consequentemente, compensar o decaimento do valor da relação Sinal-Ruído.

Em operações militares é pouco comum ocorrerem deslocamentos de forças apeadas em áreas abertas, visto que, por regra, estes ocorrem por linhas de água, zonas de vegetação densa, vales e áreas edificadas. Para simular as irregularidades no terreno, optou-se pela utilização do modelo Logarítmico de Propagação Para Três Distâncias (*Three Log Distance Propagation Loss Model*). Este modelo já está implementado em NS-3 e está localizado em *src/propagation/model/propagation-loss-model.cc*.

O valor de $\gamma = 4.33$, obtido na equação 3.1, foi aumentado para distâncias superiores a 500 m e 1000 m, usando-se $\gamma = 5$ e $\gamma = 6$, respetivamente. Este modelo necessita de dois valores de referência: a distância, d_0, e a potência recebida, P_r, correspondente à distância d_0. Foram utilizados dois dos valores obtidos em testes com o *jammer*, $d_0 = 10\,m$ e $P_r = -13.86\,dB$. Note-se que, para a escolha dos valores, se teve em atenção a aproximação à reta de tendência.

O modelo de propagação utilizado para a Rede de Sensores sem Fios foi o mesmo que para o *jammer*, *Three Log Distance Propagation Loss Model*, com os mesmos valores de γ e o mesmo ponto de referência.

3.4 Implementação em NS-3

Todo o projeto será desenvolvido em ambiente de simulação, recorrendo ao NS-3. O NS-3 é um simulador de redes cujos modelos estão implementados em C++. A implementação em NS-3 está dividida por camadas, como se pode observar na figura 3.4.

O código base do NS-3 está maioritariamente organizado na diretoria *src* e pode ser descrito pelo diagrama da fig 3.4. Em geral, cada módulo apenas depende dos módulos que, na figura, se encontram abaixo dele.

O *core* é comum a todos os simuladores e consiste no modelo do hardware e ambiente, sendo que a

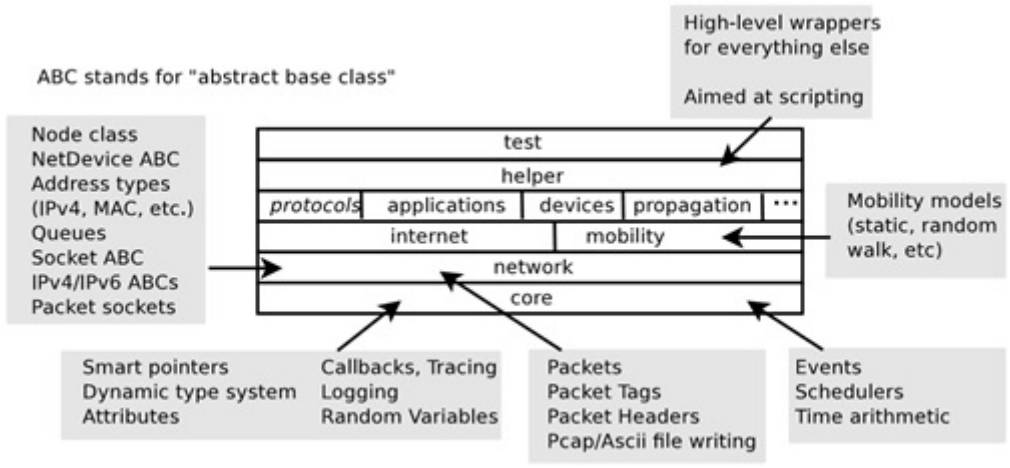


Figura 3.4: Estrutura de implementação do NS-3 [27].

implementação está em src/core. Os pacotes são objetos fundamentais na simulação de redes e estão implementados em src/*network*. Estes dois módulos de simulação servem de base para qualquer tipo de rede e não apenas para Redes de Sensores.

Os módulos acima descritos são comuns a qualquer simulação, sendo que, são sempre utilizados, mesmo para redes ou dispositivos diferentes.

As alterações necessárias ao código base do NS-3 dar-se-ão maioritariamente a nível da camada físico, para a simulação do *jammer* e a nível da camada Rede, para a adaptação do protocolo DSDV.

O NS-3 contem um gerador de variáveis *random* no seu código original. Estas varáveis são geradas tendo como base sementes, que, por defeito, são sempre iguais. As variáveis *random* são providenciadas através da instância ns3::RandomVariableStream.

Por defeito, as simulações em NS-3 utilizam sementes fixas, pelo que é de esperar resultados idênticos em todas as variáveis *random* para diferentes simulações.

Durante as simulações é necessário ter em atenção se os valores obtidos devido a valores *random* são determinísticos. Para este efeito, neste trabalho, serão elaborados vários testes em condições idênticas, mas com sementes diferentes, testando assim diferentes situações que possam ser provocadas num cenário real.

A adaptação do protocolo DSDV para melhorar a comunicação numa Rede de Sensores na qual parte dos nós são afetados por um *Jammer* apenas é possível com um modelo de propagação do *Jammer* conhecido.

3.4.1 Inserção do *Jammer*

Numa primeira fase, é essencial definir todas as configurações do NS-3 de acordo com o cenário definido.

Os nós da rede de sensores foram configurados de acordo com as especificações definidas pelo fabricante do módulo XBeePro868.

Para que exista comunicação entre nós, é necessário que a potência recebida no nó seja, em condições ideais, de pelo menos -112 dBm. No entanto, em condições reais existe sempre ruído e reflexões, pelo que optou-se por definir este valor em -80 dBm, dando assim uma margem de segurança.

De acordo com o teste de distância efetuado, tanto o ganho de antena de transmissão como de receção é de 2.1 dB, sendo este o valor usado.

Estando na presença de um *jammer*, a potência de transmissão deve ser a máxima, por forma a que os nós da Rede de Sensores sem Fios sejam menos afetados. A potência de emissão foi definida com o valor de 25.0206 dBm e a frequência de transmissão é de 868 MHz.

Para o caso em estudo, supõe-se que já existe um algoritmo de deteção do *jammer* em funcionamento, assim, fazendo uso deste, é possível aumentar a energia de transmissão apenas quando um *jammer* é detetado por algum do elementos da Rede de Sensores. Este método permite diminuir os gastos energéticos e aumentar o período de funcionamento da mesma.

A definição destes parâmetros dá-se a nível da camada física, em *src/wifi/model/yans-wifi-phy.cc*.

O *jammer* atua na camada física, por isso será inserido em *src/wifi/model/yans-wifi-phy.cc*.

A modelação do *jammer* consiste essencialmente em adicionar à simulação um nó sem grandes capacidades. Neste nó é apenas instalado o módulo rádio e o modulo mobilidade. O nó mover-se-á pela Rede de Sensores sem Fios simulando um *jammer*, transportado por uma força em deslocamento no campo de batalha.

Sempre que se dá uma transmissão a nível físico, são verificadas as distâncias entre emissor e recetor, assim como, a distância entre o *jammer* e o recetor.

Através da distância entre emissor e recetor e entre o *jammer* e o recetor e do modelo de propagação obtido em testes com o *jammer*, pode-se calcular a potência emitida pelo *jammer* que é recebida (como ruído) nos nós.

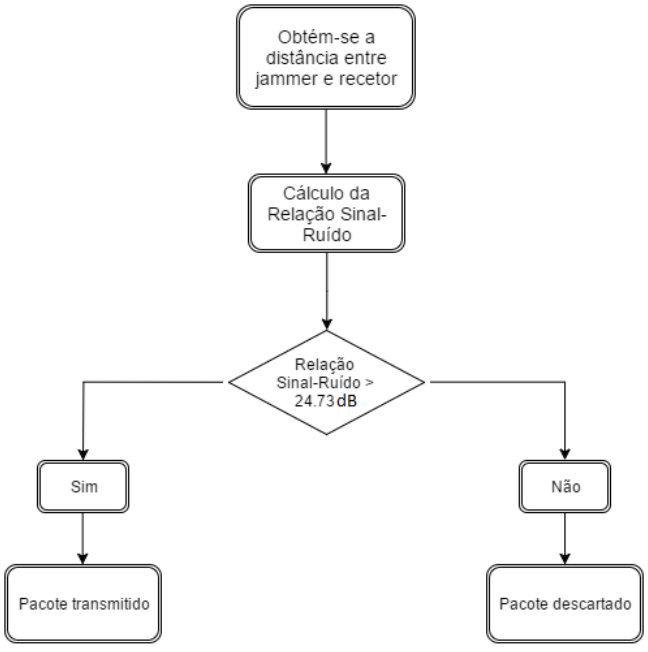


Figura 3.5: Fluxograma explicativo da implementação do *jammer* em NS-3.

É necessário determinar a relação Sinal-Ruído da ligação, sendo que a receção do pacote é efetuada caso a relação Sinal-Ruído seja favorável. Caso contrário, o pacote é perdido.

O fluxograma da figura 3.5 representa a implementação do *jammer*, sendo que, durante o cálculo da Relação Sinal-Ruído é tido em conta o diferente γ para as três diferentes zonas do modelo Logarítmico de Propagação Para Três Distâncias, somada como ruído e a verificação da relação Sinal-Ruído resultante.

Observando a figura 3.6, pode-se verificar o resultado do código inserido, no qual, como esperado, os nós mais próximos do *jammer* não conseguem receber dados, apenas os conseguem enviar para nós que não são afetados.

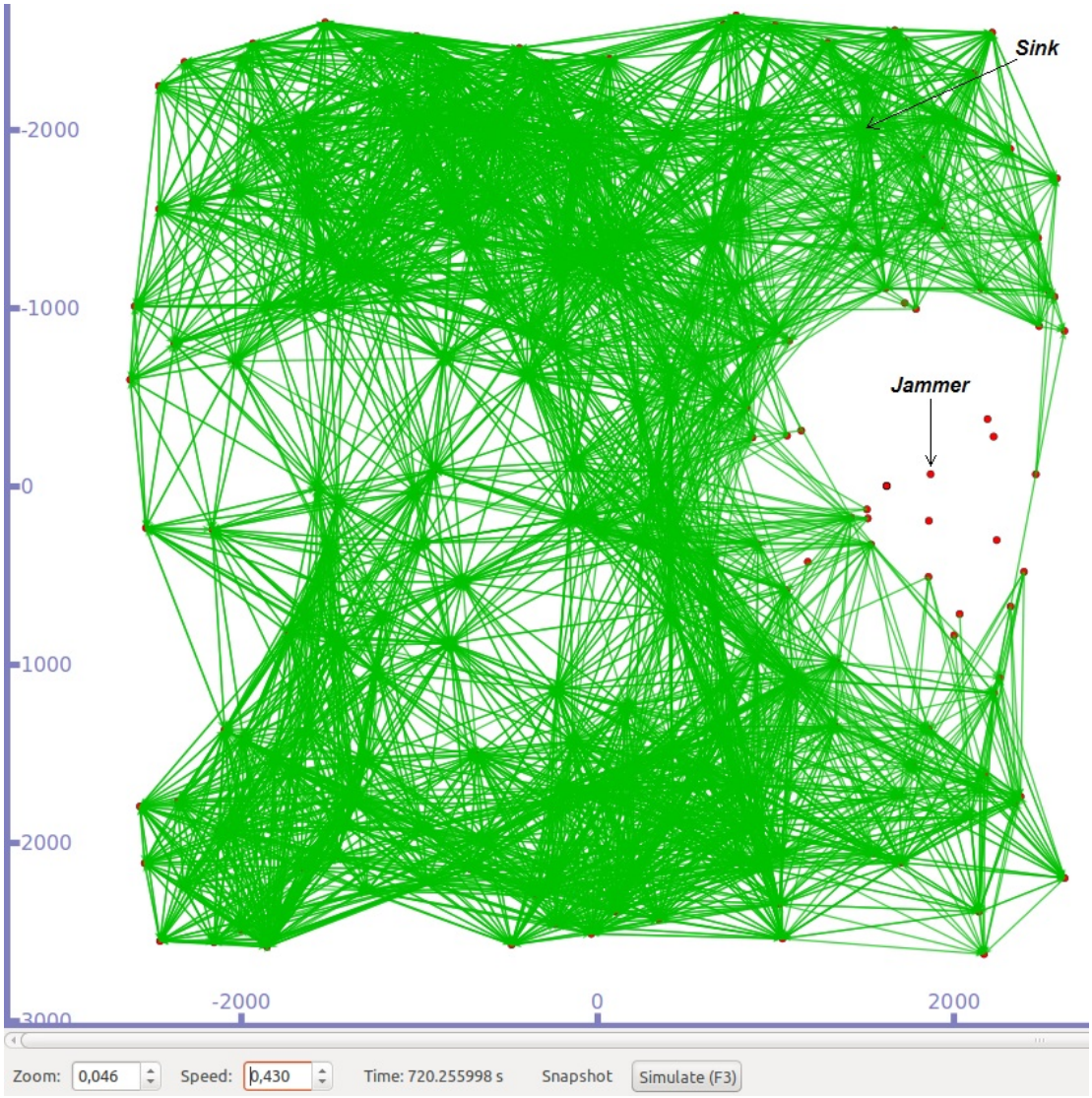


Figura 3.6: Simulação *jammer* em NS-3 durante a atualização das rotas.

Na figura 3.7, pode-se observar a quebra nas rotas, do canto inferior direito, provocada pela movimentação do *jammer*.

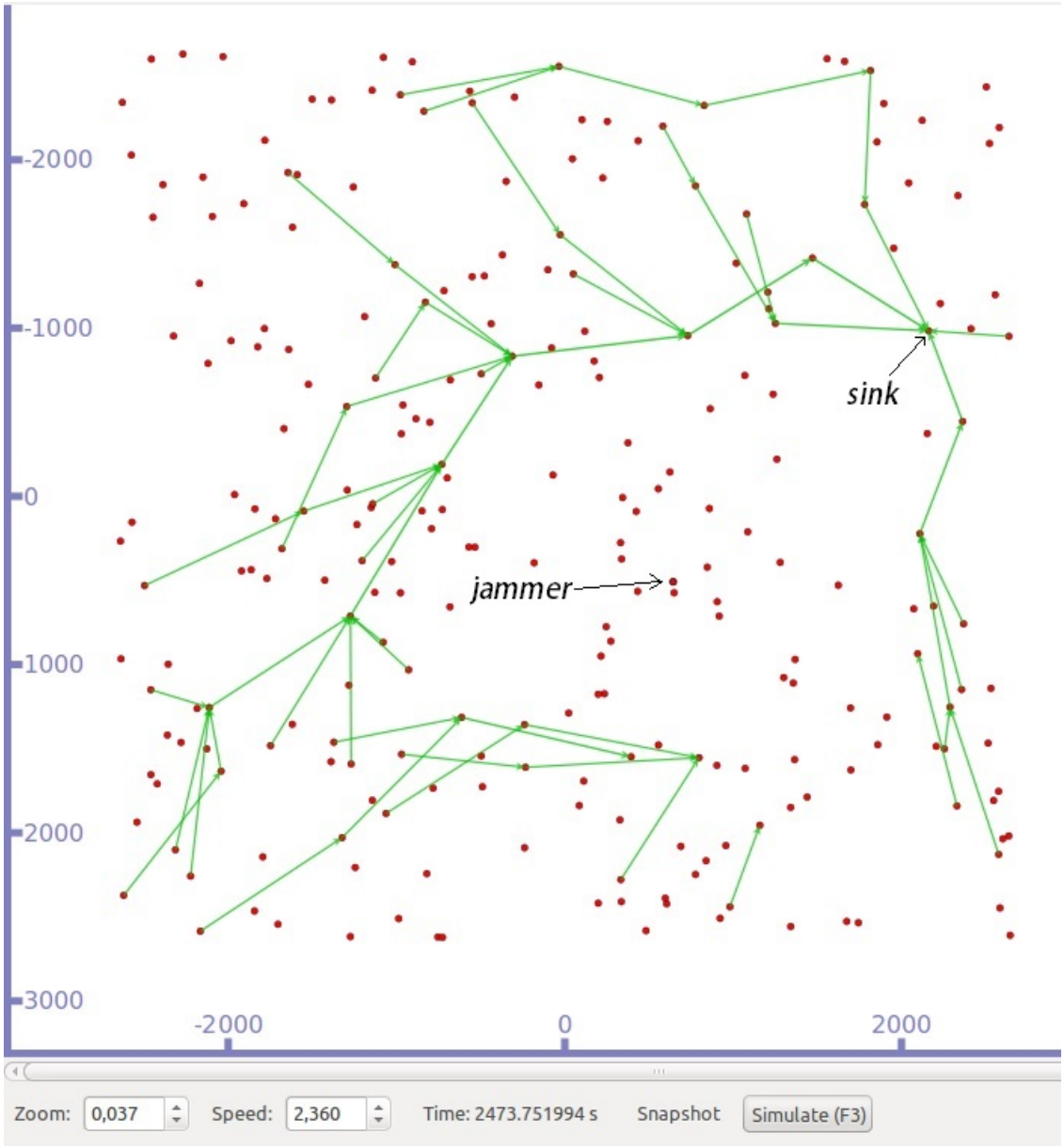


Figura 3.7: Simulação *jammer* sem contra-medida.

Capítulo 4

Resultados

Neste capítulo serão apresentados testes, obtidos em simulação, e respetivas análises, através de gráficos comparativos entre os três casos em estudo: DSDV sem qualquer interferência, DSDV com a interferência de um *jammer* e o DSDV modficado com a contramedida desenvolvida (com interferência de um *jammer*).

A implementação do *jammer* e do algoritmo de *routing* desenvolvido em NS-3 permitiu elaborar testes ao desempenho do DSDV na presença de *jamming*, sendo estes descritos na secção 4.1.

Nas secções 4.2 e 4.3 são apresentados os resultados obtidos em testes ao algoritmo desenvolvido em versão *manpack* e em montagem veicular, respetivamente.

4.1 Desempenho do DSDV na Presença de *Jamming*

Todas as simulações executadas tiveram como base os testes práticos efetuados e o modelo de propagação obtido a partir desses testes.

Uma vez que para o trabalho desenvolvido se pressupõe a existência um algoritmo de localização do *jammer* já implementado, foi disponibilizada, pelo Professor António Grilo, a implementação de uma base de dados que guarda a identificação dos nós associados ao endereço IP dos mesmos. A utilização desta base de dados permitiu aceder diretamente à camada física e obter a distância de cada nó sensor ao *jammer* sem que existisse a necessidade de criar um algoritmo de localização.

Importa referir que para o trabalho efetuado se considerou que o algoritmo de deteção do *jammer*, teria uma exatidão de 100%, pelo que os testes executados foram feitos em condições ótimas. É possível que os resultados obtidos fossem piores em caso de erro na deteção do *jammer*, uma vez que o algoritmo utilizado depende desta informação.

Inicialmente, testou-se o DSDV ainda sem o algoritmo de encaminhamento modificado, com intenção de verificar o bom funcionamento da Rede de Sensores sem Fios. Importa referir que a utilização do algoritmo de encaminhamento desenvolvido apenas faz sentido para redes em que exista um número elevado de nós. Caso existam poucos nós na rede, a probabilidade de que todos os nós sejam afetados, durante a passagem do *jammer*, é bastante elevada.

Para efeitos de simulação, optou-se por espalhar 250 nós sensores, de forma aleatória, numa área quadrangular com 28,09 km^2. Numa operação tática, os nós tanto poderiam ser lançado por meio aéreo, como colocados no local por meio terrestre. Para este trabalho utilizou-se apenas um *sink*.

O protocolo DSDV pode ser utilizado nos casos em que os nós sensores são móveis, no entanto, neste caso, consideraram-se nós estáticos, cenário típico de uma Rede de Sensores sem Fios.

Na tabela A.1 estão representados os valores obtidos para 10 testes à rede de Sensores sem Fios, quando esta não sofre qualquer ação do *jammer* e não são aplicadas quaisquer contramedidas. Cada um dos testes foi executado em condições idênticas, mas com sementes diferentes. Desta forma, são alterados todos os parâmetros aleatórios da rede, como é o caso da posição geográfica dos nós sensores.

Cada simulação teve a duração de 4100 s, sendo que, até aos 200 s, apenas ocorreram as atualizações das rotas. Assim, a rede tem tempo de se adaptar antes de ocorrer a transmissão de dados.

A transmissão de dados começa a partir dos 200 s, no entanto, os dados da tabela A.1 apenas foram contabilizados após 500 s. Tomou-se esta opção para manter a concordância com as simulações com o *jammer*, apresentadas na tabela A.5, uma vez que este apenas afeta a rede após os 500 s.

Em teste, foi possível observar que o raio de ação do *jammer* era, normalmente, inferior a 1000 m, pelo que se considerou a margem de segurança 1400 m. Os 400 m adicionais permitem ao *jammer* mover-se dentro dessa área sem que chegue a afetar as rotas. Este valor adicional na distância permite aumentar o período de atualização das tabelas de *routing*, mantendo um consumo de energia e *overhead* baixos.

Como referido anteriormente, o protocolo DSDV comporta-se bastante bem caso o período de atualização das rotas seja muito curto e a velocidade de deslocamento do *jammer* suficientemente lenta.

No caso em estudo, optou-se por considerar uma versão *manpack* do *jammer*, considerando-se a velocidade de deslocamento de 2 m/s. Apesar de o protocolo DSDV se comportar bem para períodos de atualização curtos, o *overhead* provocado pelas atualizações é muito maior.

Uma vez que a margem entre o alcance do *jammer*, observado na maioria das transmissões, é de 1000 m e a distância ao *jammer* utilizada na contramedida é de 1400 m, podemos considerar que existe uma margem de 400 m para que o *jammer* se movimente, antes que afete a rede e seja necessária uma nova atualização das rotas. Assim, considerou-se um período de atualização de rotas de 120 s como uma boa opção, permitindo ainda uma margem de segurança para os nós sensores que sejam afetados a distâncias maiores.

Na camada aplicação, utilizou-se o protocolo UDP com intervalo de envio de pacotes de 7 s para cada nó sensor. O primeiro pacote é enviado num período de tempo aleatório entre 1 a 7 s após os 200 s de simulação, para evitar que todos os nós tentem enviar ao mesmo tempo e exista colisão de pacotes. Em caso real, é pouco provável que todos os nós enviem dados exatamente ao mesmo tempo, pelo que esta opção se considerou adequada. O período de tempo aleatório entre os primeiros 7s de transmissão é controlado por sementes, que definem todos os campos aleatórios da simulação. Estas são alteradas para todas as simulações.

Na tabela A.2 estão representadas a média e o desvio padrão dos valores obtidos nas 10 simulações executadas e representadas na tabela A.1. Como seria de esperar, as perdas são bastante baixas quando não existem interferências externas de grande relevância, como é o caso de um *jammer*. Em todas as simulações foram obtidas perdas inferiores a 0,05%, sendo a média de 0,03% e desvio padrão de 0,02.

Os valores médio e máximo para o tempo desde que o pacote é enviado, a partir do nó sensor, até ao *sink* foi de 5,93 ms e de 2.01 s respetivamente. O desvio padrão foi de 0,62 para o tempo médio e 0,01 para o tempo máximo. Ambos os valores foram considerados razoáveis e dentro dos parâmetros expectáveis.

Por último, o valor de transmissões totais a nível físico será, posteriormente, utilizado como forma de comparação para o consumo energético da Rede de Sensores sem Fios.

Após a elaboração dos testes ao DSDV, inseriu-se o *jammer* e voltou a repetir-se as 10 simulações, com sementes iguais às anteriores, por forma a garantir que as condições de simulação sejam idênticas.

O *jammer* começa a afetar a Rede de Sensores sem Fios aos 500 s, 300 s após o início da transmissão de dados. Todas as simulações foram elaboradas por forma a que o *jammer* nunca afetasse o *sink*, uma vez que, se este for afetado, não existe forma do algoritmo desenvolvido conseguir manter as comunicações (num cenário real, seria aconselhável colocar vários nós *sink* em posições suficientemente distantes, por forma a aumentar a resiliência da rede).

O *jammer* atravessa a Rede de Sensores sem Fios em linha reta, maioritariamente de forma transversal ao sentido de comunicação entre os nós sensores.

Após cada atualização das rotas da Rede de Sensores sem Fios, existe um período de tempo, até à atualização seguinte, em que a rede está vulnerável à interferência causada pela movimentação do *jammer*. Durante o período de atualização, os nós que fazem parte das rotas podem ser afetados instantes depois, se o *jammer* se aproximar destes.

O ruído causado pela aproximação do *jammer* provoca quebras nas ligações entre nós sensores, que, por sua vez, resultam no aumento da perda de pacotes da aplicação.

Os resultados obtidos para as simulações com o *jammer* em versão *manpack* são apresentados na tabela A.3.

A tabela A.4 contém os valores de média e desvio padrão referentes aos 10 resultados demonstrados na tabela A.3.

Como se pode observar, em média, foram perdidos 11,28% dos pacotes enviados, sendo que o desvio padrão foi de 0.86, mais elevado do que o obtido sem interferências. O tempo médio de receção dos pacotes também foi mais elevado, cerca de 1 ms, com desvio padrão de 0,55. O tempo máximo foi de 11,22 s, valor que subiu 9,21 s em relação ao obtido anteriormente.

Em relação ao tempo máximo de entrega dos pacotes, não se verifica um aumento em todos os casos, mas verifica-se na maioria dos casos. Importa referir que, neste caso, a variação entre os valores obtidos para os diferentes testes é bastante acentuado, enquanto que sem interferências se mantinha mais uniforme.

Apesar de todo o trabalho ter sido orientado para o caso em que é utilizada uma versão *manpack*

do *jammer*, decidiu aplicar-se também o algoritmo desenvolvido a um *jammer* montado em viatura.

Importa referir que um *jammer* utilizado em montagem veicular não tem tantas limitações energéticas como a versão *manpack*, pelo que o ruído provocado na transmissão de pacotes será maior. Utilizou-se 10 W como potência de emissão do *jammer* em montagem veicular. O aumento da potência de emissão tem como consequência o aumento do raio de ação onde os efeitos do *jammer* são sentidos.

A velocidade de deslocamento do *jammer* é outro aspeto a ter em conta. Estando este montado num veículo, é lógico que a velocidade de deslocamento deste corresponda à da viatura. Em mato, podemos assumir que a velocidade de deslocamento de uma viatura é da ordem dos 20 km/h (5,56 m/s).

Neste caso, as simulações têm uma duração de 1700 s, uma vez que o *jammer* demora menos tempo a atravessar a Rede de Sensores sem Fios. O tempo de início do envio de pacotes mantém-se igual ao caso anterior.

O estudo do *jammer* em montagem veicular tem como objetivo perceber o comportamento do algoritmo num caso de estudo diferente, pelo que todos os restantes parâmetros de simulação que não dizem diretamente respeito ao *jammer* foram mantidos.

A tabela A.5 representa os dados obtidos para a simulação do protocolo DSDV, quando os nós sofrem interferências causadas por um *jammer* em montagem veicular.

A simulação foi efetuada com o *jammer* a atravessar a Rede de Sensores sem Fios em linha reta e com velocidade constante. Por isso, é de esperar que o *jammer* percorra 667,2 m em 120 s (período de atualização da rede).

Tendo presente que o alcance do *jammer* é maior devido ao aumento da potência de emissão e que a distância ao *jammer* utilizada no algoritmo se mantém, é de esperar que a percentagem de perda de de pacotes também aumente. A distância de 400 m adicionais, utilizado no algoritmo, fica aquém dos 667,2 m percorridos pelo *jammer* em 120 s, pelo que os nós sensores são mais afetados.

Os resultados obtidos confirmam o aumento relativo da perda de pacotes, assim como o tempo médio de receção dos pacotes. O tempo máximo de entrega dos pacotes diminuiu consideravelmente em relação aos valores obtidos no caso em que o *jammer* era utilizado em versão *manpack*.

A tabela A.10 contém os valores de média e desvio padrão referentes aos 10 resultados demonstrados na tabela A.5.

Como era de esperar, ambos os valores de média e desvio padrão subiram consideravelmente em relação à simulação com a versão *manpack* do *jammer*.

Os valores de média e desvio padrão relativos ao tempo médio que cada pacote demora desde o nó sensor até ao *sink* aumentou em ambos, ainda que de forma pouco acentuada. A média e desvio padrão do tempo máximo que o *sink* demora a receber cada pacote diminui consideravelmente em relação ao caso anterior.

Os testes efetuados permitiram verificar e validar o problema existente. A perda de pacotes, após a inserção do *jammer* na simulação, subiu em mais de 10%. Assim sendo, pode-se verificar se o algoritmo criado leva a uma redução destas perdas. É preciso notar que a percentagem de perdas depende do número de nós afetados, ou seja, depende do raio de alcance do *jammer* e da área coberta pela rede:

uma rede maior corresponderá a uma percentagem de perdas menor, pois há menor percentagem de nós afetados.

4.2 Resultados para *jammer* tipo *manpack*

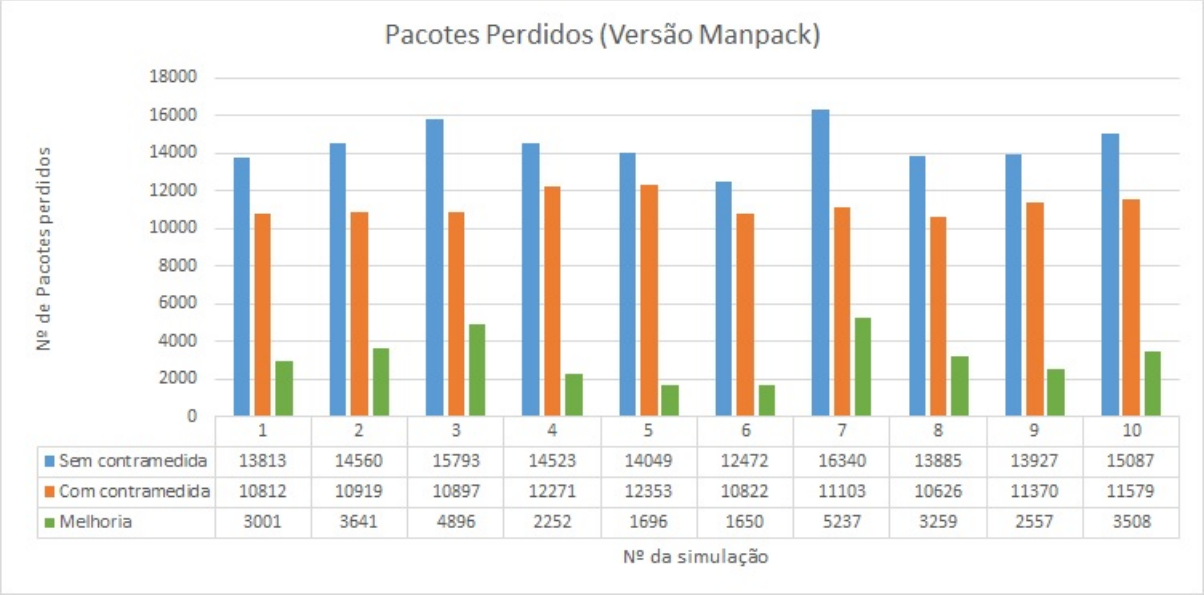
O algoritmo de *routing* desenvolvido foi testado em 10 situações distintas, tal como no caso dos testes apresentados na secção 4.1, sendo que as sementes utilizadas em cada uma das simulações (1 a 10) correspondem às usadas para o mesmo número de simulação, em qualquer um dos casos.

Na tabela A.7 estão representados os resultados obtidos nos testes com o algoritmo de *routing* criado.

Como se pode observar, a simulação que obteve um maior número de pacotes perdidos (medição nº 5), ficou aquém do melhor resultado obtido com o protocolo DSDV original. Este fato permite concluir que todas as medições obtiveram melhores resultados com a utilização do algoritmo proposto.

A tabela A.8 contém a média e desvio padrão dos valores obtidos na simulação da tabela A.7.

O gráfico da figura 4.1 permite observar, em simultâneo, a quantidade de pacotes perdidos quando a rede de sensores sofre a interferência de um *jammer*, durante as simulações em que não foi aplicada qualquer contramedida nas simulações em que foi utilizado o algoritmo de encaminhamento desenvolvido.



	1	2	3	4	5	6	7	8	9	10
Sem contramedida	13813	14560	15793	14523	14049	12472	16340	13885	13927	15087
Com contramedida	10812	10919	10897	12271	12353	10822	11103	10626	11370	11579
Melhoria	3001	3641	4896	2252	1696	1650	5237	3259	2557	3508

Figura 4.1: Pacotes Perdidos (Versão *Manpack*).

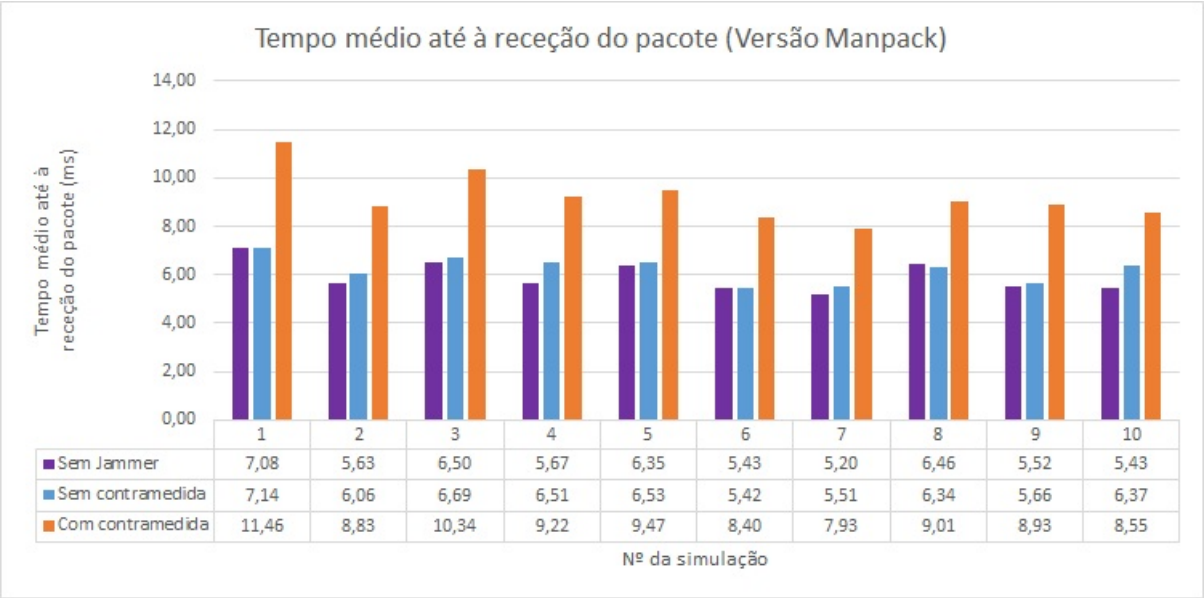
A melhoria apresentada no gráfico corresponde à diferença de pacotes perdidos obtidos em ambos os casos. Esta melhoria é positiva em todas as simulações, o que permite concluir que a contramedida aplicada obtém melhores resultados que o protocolo DSDV original. Em média, esta melhoria foi de 3169 pacotes com desvio padrão de 1216.

O gráfico da figura B.1 representa o valor relativo para a perda de pacotes, tanto das simulações sem qualquer contramedida, como das simulações com o DSDV modificado, assim como a melhoria relativa

dos pacotes perdidos, em relação ao número total de pacotes perdidos sem qualquer contramedida. Como se pode observar, na medição 3 e 7 conseguiram-se resultados superiores a 30%, enquanto que no pior caso, simulação 5, se obteve 12.07 %.

Em média, conseguiu-se obter uma redução de 21,57% dos pacotes perdidos, com um desvio padrão de 0.07. Estes valores confirmam o sucesso do algoritmo implementado.

O gráfico representado na figura 4.2 representa o valor de tempo médio entre emissão de um pacote, até que este seja recebido no *sink*, para cada uma das 10 simulações.



	1	2	3	4	5	6	7	8	9	10
Sem Jammer	7,08	5,63	6,50	5,67	6,35	5,43	5,20	6,46	5,52	5,43
Sem contramedida	7,14	6,06	6,69	6,51	6,53	5,42	5,51	6,34	5,66	6,37
Com contramedida	11,46	8,83	10,34	9,22	9,47	8,40	7,93	9,01	8,93	8,55

Figura 4.2: Tempo médio até à receção do pacote (Versão *Manpack*).

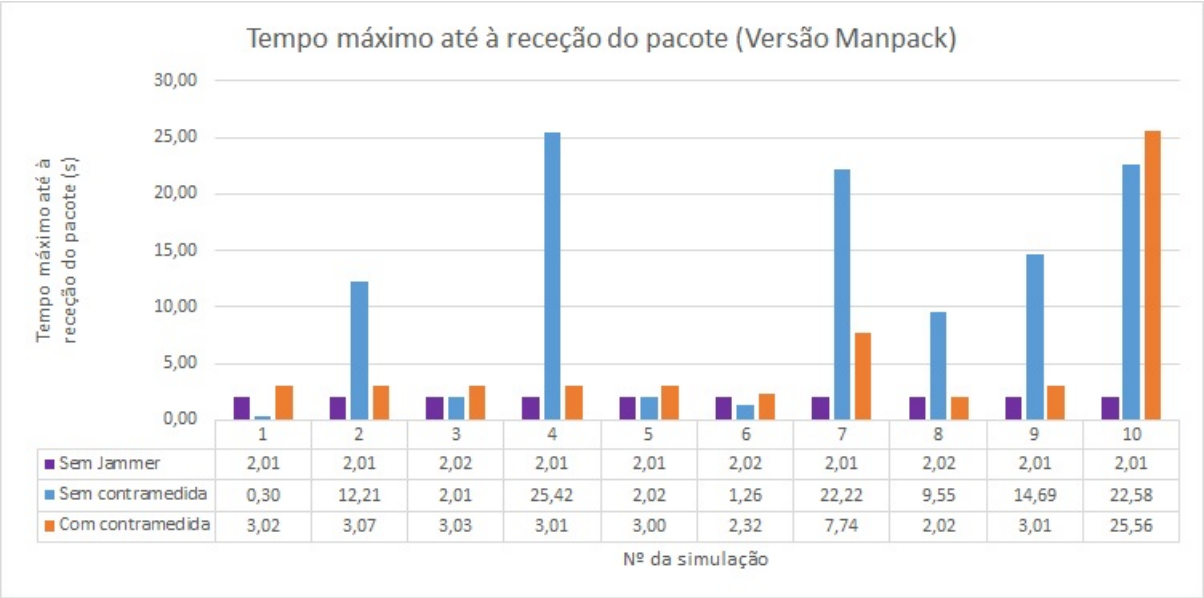
Como se pode observar, o tempo médio de receção de pacotes é muito menor nas simulações em que não existia qualquer interferência, aumentando ligeiramente nas simulações em que não foi aplicada qualquer contramedida e aumentando consideravelmente com a contramedida utilizada.

O aumento do tempo médio é facilmente justificado pelo aumento do número de saltos. O *jammer* provoca quebra nas ligações com menor número de saltos, obrigando a que, quando este causa interferências, sejam criadas rotas alternativas com maior número de saltos. Quando é aplicada a contramedida, esta tem como objetivo desviar as rotas do *jammer*, mesmo resultando num maior número de saltos. Por este motivo, é de esperar o aumento no tempo médio. Apesar do atraso na receção, este método permite receber mais pacotes no *sink*.

Em média, obteve-se 5,93 ms para as simulações sem *jammer*, com desvio padrão de 0,62, 6,22 ms para as simulações sem contramedida, com desvio padrão de 0,55, e 9,21 ms, para simulações com contramedida ativa, com desvio padrão 1,02.

O gráfico da figura 4.3 representa o tempo máximo registado entre a emissão de um pacote até que este é recebido no *sink*, para as diversas simulações.

Os valores médios do tempo máximo obtidos foram de 2,01 s, com desvio padrão de 0,01, para as simulações sem *jammer*, 11,22 s, com desvio padrão de 9,75, para simulações com a interferência do *jammer* e 5,58 s, com desvio padrão de 7,20, para simulações em que foi aplicada a contramedida.

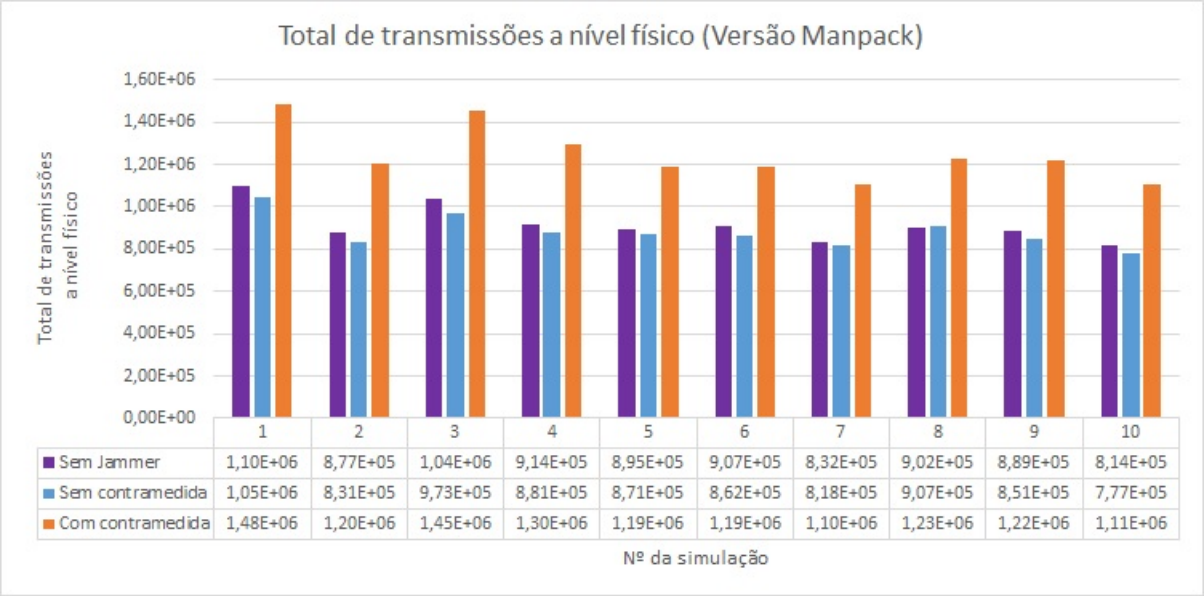


	1	2	3	4	5	6	7	8	9	10
Sem Jammer	2,01	2,01	2,02	2,01	2,01	2,02	2,01	2,02	2,01	2,01
Sem contramedida	0,30	12,21	2,01	25,42	2,02	1,26	22,22	9,55	14,69	22,58
Com contramedida	3,02	3,07	3,03	3,01	3,00	2,32	7,74	2,02	3,01	25,56

Figura 4.3: Tempo máximo até à receção do pacote (Versão *Manpack*).

Note-se que o desvio padrão é bastante acentuado e os valores irregulares.

O gráfico apresentado na figura 4.4 representa o total de transmissões a nível físico durante o tempo de simulação.

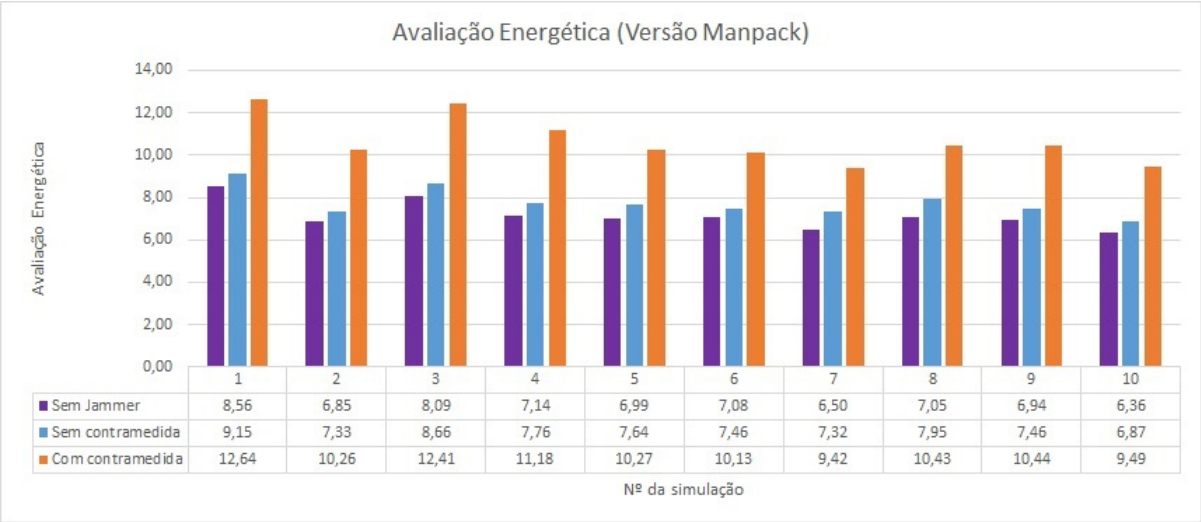


	1	2	3	4	5	6	7	8	9	10
Sem Jammer	1,10E+06	8,77E+05	1,04E+06	9,14E+05	8,95E+05	9,07E+05	8,32E+05	9,02E+05	8,89E+05	8,14E+05
Sem contramedida	1,05E+06	8,31E+05	9,73E+05	8,81E+05	8,71E+05	8,62E+05	8,18E+05	9,07E+05	8,51E+05	7,77E+05
Com contramedida	1,48E+06	1,20E+06	1,45E+06	1,30E+06	1,19E+06	1,19E+06	1,10E+06	1,23E+06	1,22E+06	1,11E+06

Figura 4.4: Total de transmissões a nível físico (Versão *Manpack*).

Em média, foram efetuadas $9,16 \times 10^5$ transmissões a nível físico, com desvio padrão de $8,64 \times 10^4$, para simulações sem interferência do *jammer*. No caso das simulações sem contramedida, foram efetuadas $8,82 \times 10^5$, com desvio padrão de $7,80 \times 10^4$, enquanto que, para simulações em que foi aplicada a contramedida, foi obtido o valor de $1,25 \times 10^6$, com desvio padrão de $1,30 \times 10^5$.

A informação sobre do número total de transmissões a nível físico permite elaborar uma análise indireta acerca do consumo energético. Esta análise, que está representada no gráfico da figura 4.5, consiste numa avaliação relativa, através de uma grandeza adimensional, obtida a partir da divisão do número de transmissões a nível físico pelo número total de pacotes recebidos, com sucesso, no *sink*.



	1	2	3	4	5	6	7	8	9	10
Sem Jammer	8,56	6,85	8,09	7,14	6,99	7,08	6,50	7,05	6,94	6,36
Sem contramedida	9,15	7,33	8,66	7,76	7,64	7,46	7,32	7,95	7,46	6,87
Com contramedida	12,64	10,26	12,41	11,18	10,27	10,13	9,42	10,43	10,44	9,49

Figura 4.5: Avaliação energética entre os diversos casos simulados (Versão *Manpack*).

Como se pode observar no gráfico, a contramedida utilizada é a mais dispendiosa em termos energéticos. O valor médio obtido foi de 7,76, com desvio padrão de 0,68, para o DSDV sem qualquer contramedida, foi de 7,76, com desvio padrão de 0,68, para o DSDV com interferência de *jammer* e de 10,67, com desvio padrão de 1,1, para a contramedida utilizada.

Após a análise dos resultados, conclui-se que o algoritmo utilizado tem vantagem em termos de pacotes recebidos, quando na presença de um *jammer*, no entanto, o *overhead* e gasto energético da rede é maior. Por este motivo, o algoritmo deve apenas ser aplicado quando é efetivamente detetado um *jammer*. A implementação utilizada resulta num algoritmo idêntico ao do protocolo DSDV quando não é detetado nenhum *jammer*.

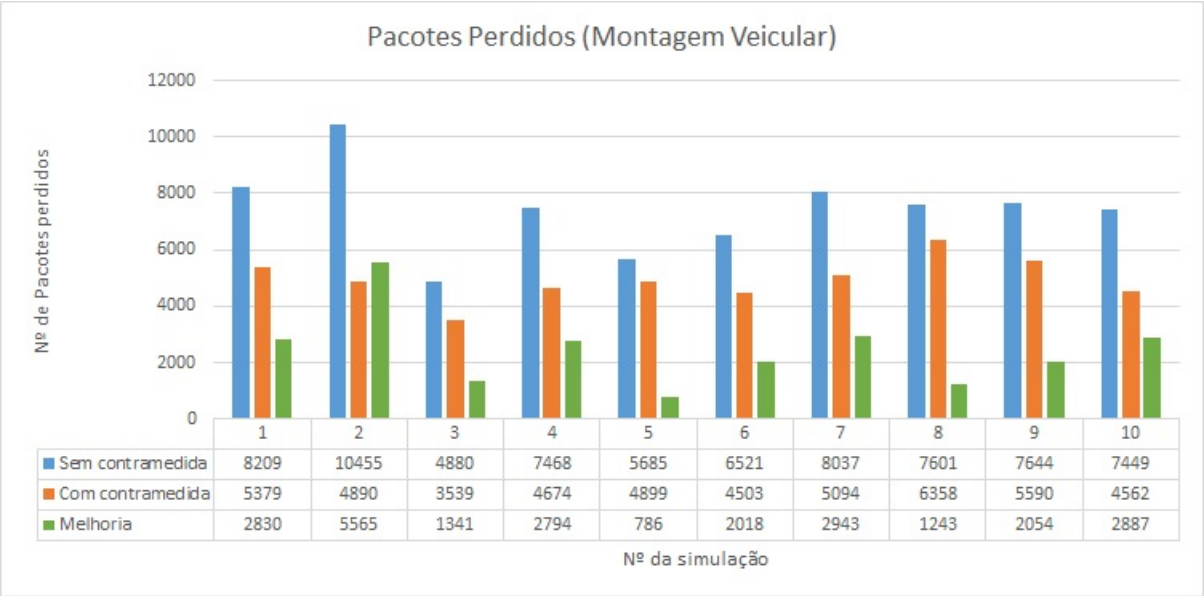
4.3 Resultados para *jammer* em montagem veicular

À semelhança dos testes realizados para o *jammer* utilizado em versão *manpack*, o algoritmo criado foi testado em 10 situações distintas para a utilização do *jammer* em montagem veicular. As sementes e movimentação do *jammer* ao longo da Rede de Sensores sem Fios foram semelhantes às utilizadas nos testes da secção 4.2.

Na tabela A.9 estão representados os resultados obtidos nos testes com o algoritmo de *routing* criado.

A tabela A.10 contém a média e desvio padrão dos valores obtidos na simulação da tabela A.9.

O gráfico da figura 4.6, representa a diferença de pacotes recebidos entre a simulação em que não foi aplicada qualquer contramedida e a simulação em que foi implementado o algoritmo de *routing* desenvolvido.



	1	2	3	4	5	6	7	8	9	10
Sem contramedida	8209	10455	4880	7468	5685	6521	8037	7601	7644	7449
Com contramedida	5379	4890	3539	4674	4899	4503	5094	6358	5590	4562
Melhoria	2830	5565	1341	2794	786	2018	2943	1243	2054	2887

Figura 4.6: Pacotes Perdidos (Montagem Veicular).

Importa referir que o tempo de simulação é menor no caso em que o *jammer* é usado em montagem veicular, por isso, o valor total de pacotes enviados é menor e, consequentemente, também os valores de pacotes perdidos são menores.

Em média, perderam-se 7394 pacotes, com desvio padrão de 1511, para as simulações sem contramedida, enquanto que para as simulações em que foi utilizado o algoritmo de *routing* desenvolvido, a média foi de 4948, com desvio padrão de 746. Sendo que a média da melhoria foi de 2446 pacotes, com desvio padrão de 1341.

O gráfico da figura B.2 representa o valor relativo de perda de pacotes. Estes valores permitem uma melhor comparação com o gráfico representado na figura B.1. A melhoria representa o valor da diferença entre pacotes perdidos, observado no gráfico 4.6, em relação ao número total de pacotes perdidos sem qualquer contramedida.

Os valores relativos de perdas observados no gráfico são mais elevados que os valores obtidos para a versão *manpack* do *jammer*. Assim, como seria de esperar, a Rede de Sensores sem Fios é mais afetada pelo *jammer* em montagem veicular, pois a sua velocidade de deslocamento e potência de emissão são mais elevadas.

Obteve-se um valor médio de perdas de 17,32%, com desvio padrão de 3,54, sem qualquer contramedida e uma média de 11,59%, com desvio padrão de 11,42, para simulações em que foi utilizado o algoritmo de *routing* desenvolvido.

No entanto, apesar de os valores de perdas serem mais elevados, a melhoria relativa obtida foi mais elevada, o que permite concluir que a diferença entre pacotes recebidos sem qualquer contramedida e com o algoritmo desenvolvido é mais acentuado quando é usado um *jammer* em montagem veicular. Apesar do algoritmo desenvolvido ter tido como base a utilização do *jammer* em versão *manpack*, a sua utilização mostrou-se mais vantajosa contra montagens veiculares, tendo-se obtido uma média 31,6%, com desvio padrão de 11,42.

O gráfico representado na figura 4.7 consiste no tempo médio entre envio e receção de um pacote no *sink*, para cada uma das 10 simulações. São representados os tempos médios de entrega do pacote, tanto para a utilização do protocolo DSDV com interferência de um *jammer*, como para o método desenvolvido.

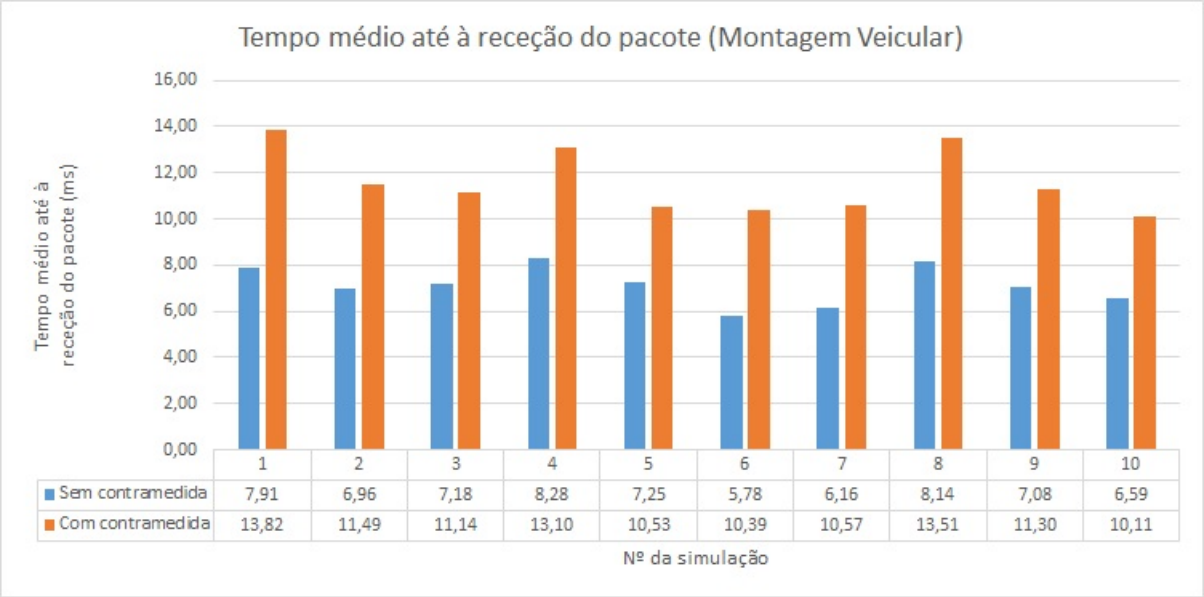


Figura 4.7: Tempo médio até à receção do pacote (Montagem Veicular).

Os valores médios do valor de tempo médio obtido para a entrega de cada pacote, para as 10 simulações, foram de 7,33 ms, com desvio padrão de 0,82, para as simulações em que foi utilizado o protocolo DSDV com interferência do *jammer* e de 11,6 ms, com desvio padrão de 1,38, para as simulações em que foi utilizado o algoritmo desenvolvido.

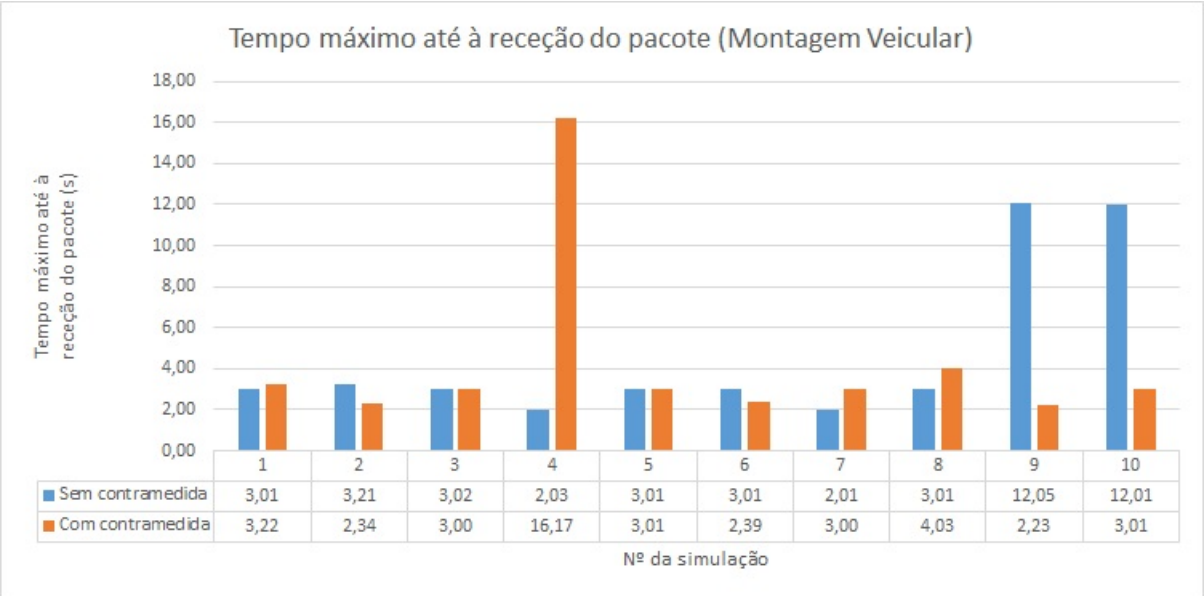


Figura 4.8: Tempo máximo até à receção do pacote (Montagem Veicular).

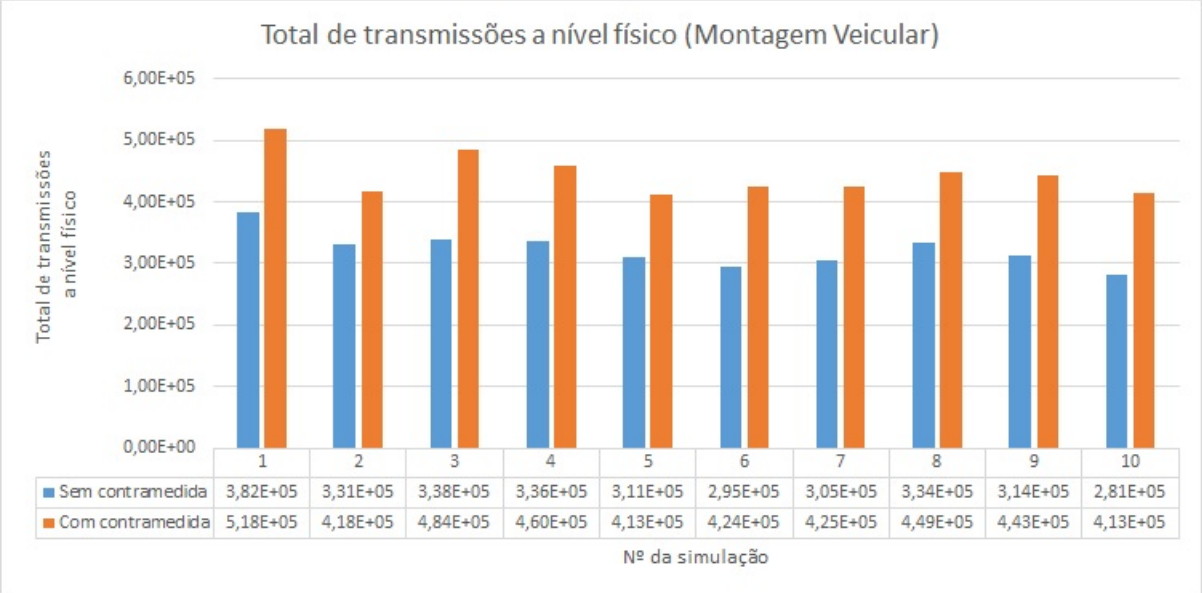
Uma vez que apenas é contabilizado o tempo médio dos pacotes que são, efetivamente, recebidos,

podemos explicar este aumento no tempo de receção com o aumento da distância percorrida pelo pacote, quando é utilizado o algoritmo desenvolvido, à semelhança da conclusão obtida para a figura do gráfico 4.2.

O gráfico representado na figura 4.8 consiste no tempo máximo registado entre a emissão de um pacote até que este é recebido no *sink*, para as simulações em que se considerou o *jammer* em montagem veicular. À semelhança dos valores obtidos no gráfico representado na figura 4.3, os valores obtidos são bastante irregulares, pelo que nada se pode concluir.

Os valores médios de tempo máximo obtidos foram de 4,64 s, com desvio padrão de 3,92, para simulações em que se utilizou o protocolo DSDV com a interferência do *jammer* em montagem veicular, e de 4,24 s, com desvio padrão de 4,22, para simulações em que se utilizou o algoritmo de *routing* desenvolvido.

O gráfico da figura 4.9 representa o total de transmissões a nível físico durante o período de transmissão de dados.



	1	2	3	4	5	6	7	8	9	10
Sem contramedida	3,82E+05	3,31E+05	3,38E+05	3,36E+05	3,11E+05	2,95E+05	3,05E+05	3,34E+05	3,14E+05	2,81E+05
Com contramedida	5,18E+05	4,18E+05	4,84E+05	4,60E+05	4,13E+05	4,24E+05	4,25E+05	4,49E+05	4,43E+05	4,13E+05

Figura 4.9: Total de transmissões a nível físico (Montagem Veicular).

Em média, foram efetuadas $3,23 \times 10^5$ transmissões a nível físico, com desvio padrão de $2,83 \times 10^4$, para simulações em que foi utilizado o protocolo DSDV com interferência do *jammer*, e foram, em média, efetuadas $4,45 \times 10^5$, com desvio padrão de $3,44 \times 10^4$, para simulações em que foi utilizada a contramedida desenvolvida.

À semelhança do gráfico da figura 4.5, o gráfico da figura 4.10 permite, utilizando a informação do gráfico da figura 4.9, elaborar uma análise indireta acerca do consumo energético, por pacote recebido com sucesso no *sink*.

A utilização do algoritmo de *routing* desenvolvido torna-se mais dispendiosa em termos energéticos, o que coincide com o obtido para a versão *manpack* do *jammer*. A média dos valores obtidos foi de 9,17, com desvio padrão de 0,97 para a utilização do DSDV com interferência do *jammer* e de 11,79, com desvio padrão de 0,94, para a utilização do algoritmo de *routing* desenvolvido.

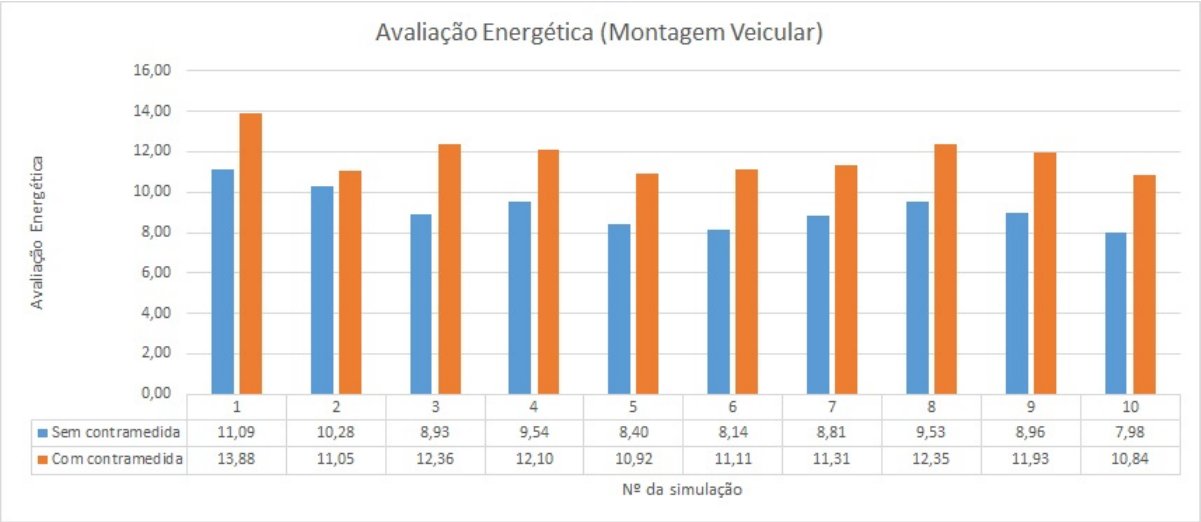


Figura 4.10: Avaliação energética entre os diversos casos simulados (Montagem Veicular).

Os resultados obtidos demonstram a melhoria em termos de pacotes recebidos, quando uma Rede de Sensores sem Fios sofre a ação de um *jammer*, tendo-se conseguido bastantes simulações com resultados superiores a 30%, chegando a 53,23% na simulação 2, com o *jammer* em montagem veicular.

Importa ter em conta que os resultados foram obtidos considerando que a deteção do *jammer* era exata, podendo estes valores descer ligeiramente em caso de desvios na deteção do *jammer*.

Capítulo 5

Conclusões

O objetivo principal desta dissertação consiste em desenvolver um algoritmo de *routing* que melhore o desempenho, em termos de pacotes recebidos, de uma Rede de Sensores sem Fios quando esta é afetada por um *jammer*.

Sentiu-se necessidade de estudar diversos protocolos de *routing*, para que um deles servisse de base e pudesse ser alterado para cumprir os objetivos definidos. Concluiu-se que o melhor protocolo a usar seria o RPL. No entanto, não existe nenhuma implementação do mesmo em NS-3, pelo que se optou por utilizar uma versão modificada do DSDV em que apenas são criadas as rotas para o *sink*. Esta aproximação tornou o DSDV mais próximo do RPL.

Antes de proceder à implementação do algoritmo de *routing*, foi necessário estudar e efetuar testes com um *jammer*, por forma a obter um modelo de propagação mais próximo da realidade.

Considerou-se que o modelo de propagação a utilizar seria o *Three Log Distance Propagation Loss Model*, em que os γ utilizados foram de 4.33 , 5 e 6, para os primeiro, segundo e terceiro intervalos de distâncias, respetivamente. Justificou-se o incremento de γ pelo fato das operações militares se darem em zonas de terrenos irregulares ou em áreas edificadas.

Foram realizadas simulações em ns-3, por forma a avaliar o desempenho do algoritmo desenvolvido.

O *jammer* foi inserido no modelo de simulação, representado por um nó sem nenhumas capacidades, que se move pela rede de sensores e permite usar a sua distância ao nó recetor como parâmetro. Caso não sejam cumpridos os valores mínimos para comunicação, de acordo com o modelo definido, o pacote é perdido.

O algoritmo desenvolvido foi testado tanto para o *jammer* em versão *manpack*, como para uma versão veicular, sendo os resultados obtidos comparados com o protocolo DSDV sem qualquer contramedida. Este algoritmo consiste essencialmente na definição de um raio de segurança ao *jammer*, que os pacotes evitam, se possível. No caso estudado, definiu-se esta distância de segurança como 1400 m.

O trabalho foi concluído com sucesso, tendo-se obtido cerca de 20% e 30% de melhoria para o *jammer* em versão *manpack* e em montagem veicular, respetivamente. No entanto verificou-se um aumento no *overhead* da rede, assim como na energia despendida, por pacote recebido no *sink*. Posto

isto, o algoritmo é bastante útil quando, efetivamente, existe a presença de um *jammer*, sendo que o seu comportamento se assemelha ao do DSDV quando o *jammer* se afasta.

5.1 Trabalho futuro

Os testes efetuados tiveram como pressuposto que a localização do *jammer* era conhecida, sendo utilizada 100% de exatidão para a localização deste. Como trabalho futuro, seria de especial interesse realizar um algoritmo de deteção do *jammer* e voltar a testar os cenários considerados.

O algoritmo de localização do *jammer* poderia ter como base a potência do sinal recebido e sabendo a interferência causada pelo *jammer*, estimar a distância a que este se encontra dos nós. Este método poderá ser influenciado pela interferência de outros sensores no sinal recebido, pelo que se torna essencial estudar aprofundadamente o efeito, tanto da interferência causada pelo *jammer*, como outras interferências externas que influenciem o método.

Obstáculos no terreno, tais como elevações do terreno e edifícios, podem dificultar a deteção através deste método, pelo que seria interessante dar capacidade à rede de utilizar situações passadas para prever casos futuros (aprendizagem automática).

Outra possibilidade de trabalho a realizar, passa por estudar o efeito de utilizar mais que um *sink* na rede de sensores, e mais que um *jammer*.

Seria ainda útil experimentar sistemas com antenas inteligentes (*smart antennas*), que podem optar por ser omnidirecionais ou favorecer uma determinada direção.

Referências

[1] H. Jeon, K. Park, D.-J. Hwang, and H. Choo. Sink-oriented dynamic location service protocol for mobile sinks with an energy efficient grid-based approach. *Sensors*, 9(3):1433–1453, 2009.

[2] H. Karl and A. Willig. *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, 2007.

[3] A. W. Khan, A. H. Abdullah, M. H. Anisi, and J. I. Bangash. A comprehensive study of data collection schemes using mobile sinks in wireless sensor networks. *Sensors*, 14(2):2510–2548, 2014.

[4] R. V. Biradar, V. Patil, S. Sawant, and R. Mudholkar. Classification and comparison of routing protocols in wireless sensor networks. *Special Issue on Ubiquitous Computing Security Systems*, 4(2):704–711, 2009.

[5] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. 24(4):234–244, 1994.

[6] G. He. Destination-sequenced distance vector (dsdv) protocol. *Networking Laboratory, Helsinki University of Technology*, pages 1–9, 2002.

[7] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. 2003.

[8] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. 2003.

[9] S. R. Biradar, K. Majumder, S. K. Sarkar, and P. C. Performance evaluation and comparison of aodv and aomdv. *International Journal on Computer Science and Engineering*, 2010.

[10] F. Silva, J. Heidemann, R. Govindan, and D. Estrin. Directed diffusion. *USC/Information Sciences Institute, Tech. Rep. ISI-TR-2004-586*, 2004.

[11] J. Vasseur, N. Agarwal, J. Hui, Z. Shelby, P. Bertrand, and C. Chauvenet. Rpl: The ip routing protocol designed for low power and lossy networks. *Internet Protocol for Smart Objects (IPSO) Alliance*, 36, 2011.

[12] S. G. L. M. Infrastructure. A standardized and flexible ipv6 architecture for field area networks. 2011.

[13] S. Misra, R. Singh, and S. Mohan. Information warfare-worthy jamming attack detection mechanism for wireless sensor networks using a fuzzy inference system. *Sensors*, 10(4):3444–3479, 2010.

[14] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: attack and defense strategies. *IEEE network*, 20(3):41–47, 2006.

[15] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. pages 46–57, 2005.

[16] M. Çakiroğlu and A. T. Özcerit. Jamming detection mechanisms for wireless sensor networks. page 4, 2008.

[17] G. Blumrosen, B. Hod, T. Anker, D. Dolev, and B. Rubinsky. Enhancing rssi-based tracking accuracy in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 9(3):29, 2013.

[18] A. S. Chimankar and V. Nandedkar. Effective approach for localizing jammers in wireless sensor netork. 2013.

[19] H. Liu, Z. Liu, Y. Chen, and W. Xu. Localizing multiple jamming attackers in wireless networks. pages 517–528, 2011.

[20] S. Yanqiang, W. Xiaodong, and Z. Xingming. Jammer localization for wireless sensor networks. *Chinese Journal of Electronic*, 20(CJE-4):735–738, 2011.

[21] W. Xu. *Defending wireless networks from radio interference attacks*. ProQuest, 2007.

[22] B. Kan and J. Fan. A novel jamming-aware metric for mhwn routing.

[23] R. Muraleedharan and L. A. Osadciw. Jamming attack detection and countermeasures in wireless sensor network using ant system. In *Defense and Security Symposium*, pages 62480G–62480G. International Society for Optics and Photonics, 2006.

[24] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou. Jaid: An algorithm for data fusion and jamming avoidance on distributed sensor networks. *Pervasive and Mobile Computing*, 5(2):135–147, 2009.

[25] A. D. Wood, J. A. Stankovic, and S. H. Son. Jam: A jammed-area mapping service for sensor networks. In *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*, pages 286–297. IEEE, 2003.

[26] D. I. Inc. *XBee®/XBee-PRO® RF Modules*.

[27] *NS-3 Network Simulator*.

Anexo A

Tabelas de resultados obtidos

Tabela A.1: Validação do protocolo DSDV através de 10 medições com sementes diferentes.

Teste ao DSDV sem interferência do *jammer*

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos (%)	Tempo médio (ms)	Tempo máximo (s)	Tx total (Físico)
1	127997	128060	63	0,05%	7,08	2,01	1,10E+06
2	128042	128054	12	0,01%	5,63	2,01	8,77E+05
3	128026	128060	34	0,03%	6,50	2,02	1,04E+06
4	128045	128062	17	0,01%	5,67	2,01	9,14E+05
5	128000	128048	48	0,04%	6,35	2,01	8,95E+05
6	128058	128058	0	0,00%	5,43	2,02	9,07E+05
7	128012	128054	42	0,03%	5,20	2,01	8,32E+05
8	127993	128054	61	0,05%	6,46	2,02	9,02E+05
9	128036	128061	25	0,02%	5,52	2,01	8,89E+05
10	128028	128073	45	0,04%	5,43	2,01	8,14E+05

Tabela A.2: Média e Desvio Padrão dos valores obtidos na tabela A.1.

Teste ao DSDV sem interferência do *jammer*

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos(%)	Tempo médio(ms)	Tempo máximo(s)	Tx total (Físico)
Média	128023	128058	34	0,03%	5,93	2,01	9,16E+05
Desvio Padrão	22,35	6,70	20,97	0,02	0,62	0,02	8,64E+04

Tabela A.3: Validação do protocolo DSDV, quando sofre a interferência de um *jammer* (versão *man-pack*), através de 10 medições com sementes diferentes.

	Teste ao DSDV com interferência do *jammer*						
nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos (%)	Tempo médio (ms)	Tempo máximo (s)	Tx total (Físico)
1	114247	128060	13813	10,79%	7,14	0,30	1,05E+06
2	113494	128054	14560	11,37%	6,06	12,21	8,31E+05
3	112267	128060	15793	12,33%	6,69	2,01	9,73E+05
4	113539	128062	14523	11,34%	6,51	25,42	8,81E+05
5	113999	128048	14049	10,97%	6,53	2,02	8,71E+05
6	115586	128058	12472	9,74%	5,42	1,26	8,62E+05
7	111714	128054	16340	12,76%	5,51	22,22	8,18E+05
8	114169	128054	13885	10,84%	6,34	9,55	9,07E+05
9	114134	128061	13927	10,88%	5,66	14,69	8,51E+05
10	112986	128073	15087	11,78%	6,37	22,58	7,77E+05

Tabela A.4: Média e Desvio Padrão dos valores obtidos na tabela A.3.

	Teste ao DSDV com interferência do *jammer*						
nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos(%)	Tempo médio(ms)	Tempo máximo(s)	Tx total (Físico)
Média	113613	128058	14444	11,28%	6,22	11,22	8,82E+05
Desvio Padrão	1098,23	6,70	1099,01	0,86	0,55	9,75	7,80E+04

Tabela A.5: Validação do protocolo DSDV quando sofre a interferência de um *jammer* (montagem veicular) através de 10 medições com sementes diferentes.

	Teste ao DSDV com interferência do *jammer*						
nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos (%)	Tempo médio (ms)	Tempo máximo (s)	Tx total (Físico)
1	34479	42688	8209	19,23%	7,91	3,01	3,82E+05
2	32230	42685	10455	24,49%	6,96	3,21	3,31E+05
3	37806	42686	4880	11,43%	7,18	3,02	3,38E+05
4	35218	42686	7468	17,50%	8,28	2,03	3,36E+05
5	36999	42684	5685	13,32%	7,25	3,01	3,11E+05
6	36172	42693	6521	15,27%	5,78	3,01	2,95E+05
7	34646	42683	8037	18,83%	6,16	2,01	3,05E+05
8	35087	42688	7601	17,81%	8,14	3,01	3,34E+05
9	35046	42690	7644	17,91%	7,08	12,05	3,14E+05
10	35240	42689	7449	17,45%	6,59	12,01	2,81E+05

Tabela A.6: Média e Desvio Padrão dos valores obtidos na tabela A.5.
Teste ao DSDV com interferência do *jammer*

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos(%)	Tempo médio(ms)	Tempo máximo(s)	Tx total (Físico)
Média	35292	42687	7394	17,32%	7,13	4,64	3,23E+05
Desvio Padrão	1511,93	3,01	1511,59	3,54	0,82	3,92	2,83E+04

Tabela A.7: Validação do algoritmo de *routing* criado através de 10 medições com sementes diferentes.
Teste ao DSDV Modificado

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos (%)	Tempo médio (ms)	Tempo máximo (s)	Tx total (Físico)
1	117248	128060	10812	8,44%	11,46	3,02	1,48E+06
2	117135	128054	10919	8,53%	8,83	3,07	1,20E+06
3	117163	128060	10897	8,51%	10,34	3,03	1,45E+06
4	115791	128062	12271	9,58%	9,22	3,01	1,30E+06
5	115695	128048	12353	9,65%	9,47	3,00	1,19E+06
6	117236	128058	10822	8,45%	8,40	2,32	1,19E+06
7	116951	128054	11103	8,67%	7,93	7,74	1,10E+06
8	117428	128054	10626	8,30%	9,01	2,02	1,23E+06
9	116691	128061	11370	8,88%	8,93	3,01	1,22E+06
10	116494	128073	11579	9,04%	8,55	25,56	1,11E+06

Tabela A.8: Média e Desvio Padrão dos valores obtidos na tabela A.7.
Teste ao DSDV Modificado

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos(%)	Tempo médio(ms)	Tempo máximo(s)	Tx total (Físico)
Média	116783,20	128058,40	11275,20	8,80%	9,21	5,58	1,25E+06
Desvio Padrão	614,07	6,70	614,23	0,48	1,02	7,20	1,30E+05

Tabela A.9: Validação do protocolo DSDV através de 10 medições com sementes diferentes.

Teste ao DSDV Modificado

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos (%)	Tempo médio (ms)	Tempo máximo (s)	Tx total (Físico)
1	37309	42688	5379	12,60%	13,82	3,22	5,18E+05
2	37795	42685	4890	11,46%	11,49	2,34	4,18E+05
3	39147	42686	3539	8,29%	11,14	3,00	4,84E+05
4	38012	42686	4674	10,95%	13,10	16,17	4,60E+05
5	37785	42684	4899	11,48%	10,53	3,01	4,13E+05
6	38190	42693	4503	10,55%	10,39	2,39	4,24E+05
7	37589	42683	5094	11,93%	10,57	3,00	4,25E+05
8	36330	42688	6358	14,89%	13,51	4,03	4,49E+05
9	37100	42690	5590	13,09%	11,30	2,23	4,43E+05
10	38127	42689	4562	10,69%	10,11	3,01	4,13E+05

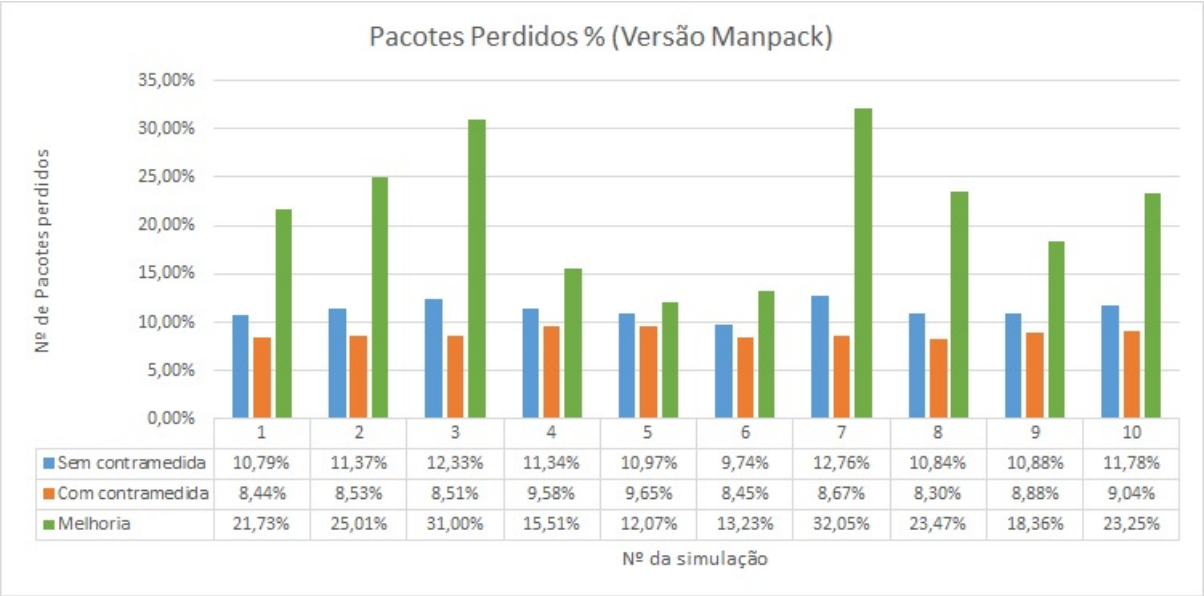
Tabela A.10: Média e Desvio Padrão dos valores obtidos na tabela A.5.

Teste ao DSDV Modificado

nº medição	Pacotes recebidos	Pacotes enviados	Pacotes perdidos	Pacotes perdidos(%)	Tempo médio(ms)	Tempo máximo(s)	Tx total (Físico)
Média	37738,40	42687,20	4948,80	11,59%	11,60	4,24	4,45E+05
Desvio Padrão	746,64	3,01	746,88	1,75	1,38	4,22	3,44E+04

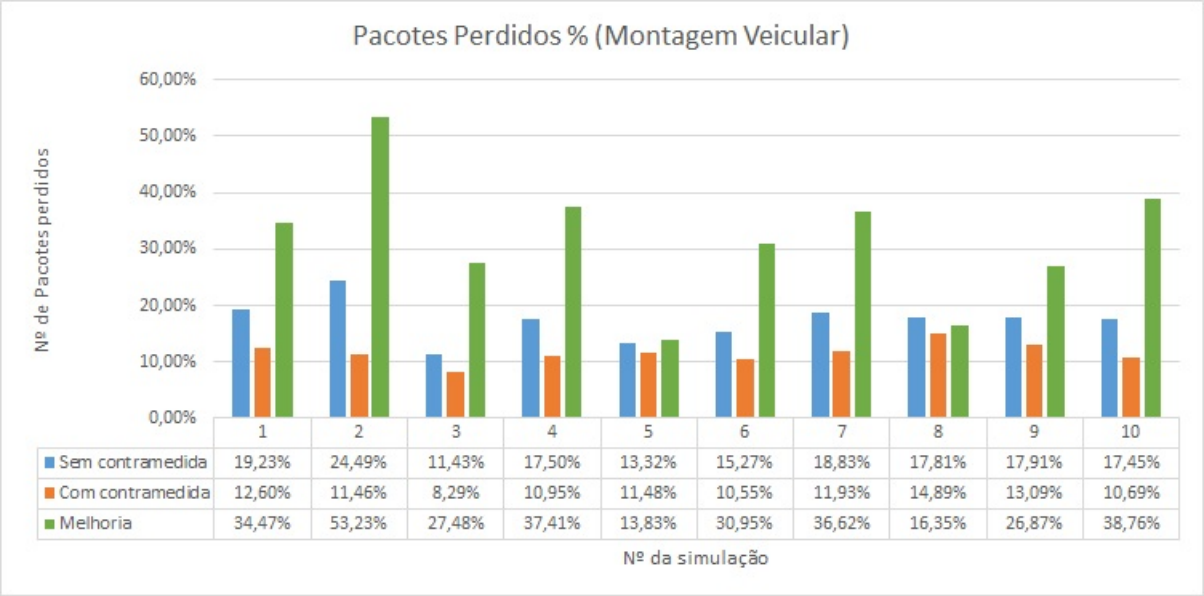
Anexo B

Gráficos de perdas em percentagem



	1	2	3	4	5	6	7	8	9	10
Sem contramedida	10,79%	11,37%	12,33%	11,34%	10,97%	9,74%	12,76%	10,84%	10,88%	11,78%
Com contramedida	8,44%	8,53%	8,51%	9,58%	9,65%	8,45%	8,67%	8,30%	8,88%	9,04%
Melhoria	21,73%	25,01%	31,00%	15,51%	12,07%	13,23%	32,05%	23,47%	18,36%	23,25%

Figura B.1: Pacotes Perdidos % (Versão *Manpack*).



	1	2	3	4	5	6	7	8	9	10
Sem contramedida	19,23%	24,49%	11,43%	17,50%	13,32%	15,27%	18,83%	17,81%	17,91%	17,45%
Com contramedida	12,60%	11,46%	8,29%	10,95%	11,48%	10,55%	11,93%	14,89%	13,09%	10,69%
Melhoria	34,47%	53,23%	27,48%	37,41%	13,83%	30,95%	36,62%	16,35%	26,87%	38,76%

Figura B.2: Pacotes Perdidos % (Montagem Veicular).