

Universidade Federal do Estado do Rio de Janeiro  
Centro de Ciências Exatas e Tecnologia  
Curso Bacharelado em Sistemas de Informação



Autores: Igor Fernandes da Cruz  
Roberto Assakura

Orientador: Carlos Alberto Vieira Campos

Julho/2013

Universidade Federal Do Estado Do Rio de Janeiro  
Centro de Ciências Exatas e Tecnologia  
Curso Bacharelado em Sistemas de Informação

Igor Fernandes da Cruz

Roberto Assakura

Avaliação de Desempenho de Protocolos de Roteamento de Redes DTN Oportunísticas

Universidade Federal Do Estado Do Rio de Janeiro  
Centro de Ciências Exatas e Tecnologia  
Curso Bacharelado em Sistemas de Informação

## **DEDICATÓRIA**

A Deus, o que seria de nós sem a fé que temos nele.

Aos nossos familiares que, com muito carinho e apoio, não mediram esforços para que chegássemos a esta etapa de nossas vidas.

Universidade Federal Do Estado Do Rio de Janeiro  
Centro de Ciências Exatas e Tecnologia  
Curso Bacharelado em Sistemas de Informação

## **AGRADECIMENTOS**

Ao professor Carlos Alberto Campos, pela paciência na orientação e incentivo que tornaram possível a conclusão desta monografia.

À professora Leila Andrade, pelo convívio, pelo apoio, pela compreensão e pela amizade.

Ao aluno de doutorado de Pittsburgh, PJ Dillon, pela sua contribuição através do projeto BubbleRap desenvolvido para o ambiente do ONE.

A todos os professores da Escola de Informática Aplicada da UNIRIO, que foram tão importantes em nossa vida acadêmica e no desenvolvimento desta monografia.

Aos amigos e colegas, pelo incentivo e pelo apoio constante.

## SUMÁRIO

LISTA DE FIGURAS.....	vi
LISTA DE TABELAS.....	viii
LISTA DE ABREVIATURAS.....	ix
1. INTRODUÇÃO .....	1
2. REFERENCIAL TEÓRICO.....	3
3. TRABALHOS RELACIONADOS.....	26
4. SIMULAÇÃO DOS PROTOCOLOS DE ROTEAMENTO DTN EM UM CENÁRIO COM CARACTERÍSTICAS REAIS.....	28
5. AVALIAÇÃO DOS RESULTADOS OBTIDOS.....	44
6. CONCLUSÃO E TRABALHOS FUTUROS .....	52
REFERÊNCIAS BIBLIOGRÁFICAS.....	55
APÊNDICE A.....	60
APÊNDICE B.....	61
APÊNDICE C.....	62

## LISTA DE FIGURAS

2.1.	Exemplo de uma rede sem fio utilizando o padrão IEEE 802.11 .....	4
2.2.	Exemplo do uso e funcionamento de uma rede VANET .....	6
2.3.	Exemplo da transmissão de mensagens em uma rede MANET.....	7
2.4.	Exemplo de uma rede de sensores.....	8
2.5.	Exemplo do uso da rede de sensores no monitoramento de atividades sísmicas em um vulcão.....	9
2.6.	Exemplo de uma rede DTN rural .....	11
2.7.	Exemplo de uma Drive-thru Internet .....	12
2.8.	Exemplo de rede PSN .....	13
2.9.	Exemplo de um grafo de rede social.... ..	20
2.10.	Ilustração do algoritmo BubbleRap .....	25
4.1.	Tela da interface gráfica do ONE.....	31
4.2.	Visualização do log dos eventos da simulação no ONE.....	32
4.3.	Exemplo do uso da tecnologia Bluetooth em redes Ad hoc.....	35
4.4.	Mapa da área escolhida do Centro da cidade do Rio de Janeiro.....	39
4.5.	Snapshot do simulador ONE durante a realização de uma simulação utilizando mapa do centro da cidade.....	40

## LISTA DE FIGURAS (cont.)

6.1.	Análise da influência da variação do raio de alcance dos nós na taxa de entrega para mensagens de 10 Kbytes.....	45
6.2.	Análise da influência da variação do raio de alcance dos nós no atraso médio para mensagens de 10 Kbytes.....	46
6.3.	Análise da influência da variação do raio de alcance dos nós na sobrecarga para mensagens de 10 Kbytes.....	47
6.4.	Análise da influência da variação do raio de alcance dos nós na taxa de entrega para mensagens de 100 Kbytes .....	49
6.5.	Análise da influência da variação do raio de alcance dos nós no atraso médio para mensagens de 100 Kbytes.....	50
6.6.	Análise da influência da variação do raio de alcance dos nós na sobrecarga para mensagens de 100 Kbytes.....	51

## **LISTA DE TABELAS**

2.1.	Métricas de centralidade da rede social mostrada na Figura 2.9.....	20
4.1.	Classes de dispositivos Bluetooth .....	35
4.2.	Tabela da configuração dos nós em cada simulação.....	43



## LISTA DE ABREVIATURAS

CLI	Command Line Interface
CRAWDAD	Community resource for archiving data at Dartmouth
CSMA/CA	Carry Sense Multiple Access/ Collision Avoid
DTN	Delay Tolerant Networks
IDE	Integrated Development Environment
IP	Internet Protocol
IPN	Interplanetary Internet
LAN	Local Area Networks
MANET	Mobile Ad hoc Networks
MSC	Mobile Switching Center
ONE	Opportunistic Network Environment
OSM	Open Street Map
PAN	Personal Area Network
POI	Points of interest
PSN	Pocket Switched Networks
SPMBM	Shortest Path Map-Based Movement
SSID	Service Set Identifier
VANET	Vehicular Ad hoc Networks
WiFi	Wireless Fidelity
WKT	Well-Known Text
WLAN	Wireless Local Area Networks

# 1. INTRODUÇÃO

A área de redes sem fios nos últimos anos vem sendo foco de diversos estudos e trabalhos que vem permitindo sua evolução e a criação de novas tecnologias ou abordagens. Esses estudos tem como objetivo reduzir os efeitos físicos e naturais sofridos por esse tipo de comunicação, possibilitar que áreas inóspitas possam se comunicar e tentar reduzir custos de implementação de redes. As redes sem fios são divididas em diversos modelos de implementação e configuração, dentre eles temos: redes locais sem fio, redes ad hoc e redes tolerantes a atrasos. Abordaremos brevemente esses modelos de redes nos tópicos seguintes e explicaremos maiores detalhes sobre o foco desse trabalho, que vem a ser as redes tolerantes a atrasos e a desconexões (DTN – Disruption/Delay Tolerant Networks).

## 1.1 Objetivo

O objetivo deste trabalho é de analisar a influência das métricas sociais no desempenho de redes tolerantes a atraso e desconexões. Para atingir este objetivo, utilizamos três protocolos de roteamento em DTN: um de replicação mais tradicional, o Epidêmico[15], que usa múltiplas cópias de mensagens para encaminhamento, o Spray and Wait[14] que diminui o número de cópias a ser enviado e guarda parte das mensagens em cada nó e, por fim, um protocolo que utiliza métricas sociais no seu algoritmo de encaminhamento, o BubbleRap[2].

Para a realização deste trabalho, foi criado um cenário que usa uma área real do centro da cidade do Rio de Janeiro escolhida para nossas simulações e para tal foi extraído um mapa da referida área a partir do projeto OpenStreetMap [11].

Como resultado obtido apresentamos uma avaliação de desempenho do protocolo BubbleRap em relação aos dois outros protocolos que utilizam métricas tradicionais, e que são nativos do simulador ONE (Opportunistic Network Environment).

## **1.2 Relevância do trabalho**

O interesse de nosso trabalho é verificar até que ponto o uso de um protocolo que utiliza métricas e estruturas sociais em seus algoritmos de roteamento se traduz em otimização da performance na entrega de mensagens nas redes DTN.

## **1.3 Estrutura do restante da monografia**

O Capítulo 2 apresenta a fundamentação teórica sobre redes sem fio, suas variações, redes tolerantes a atrasos, os protocolos de roteamento empregados e as redes sociais.

O Capítulo 3 descreve alguns trabalhos relacionados com redes tolerantes a atrasos e desconexões, redes sociais e análise de desempenho de protocolos.

O Capítulo 4 apresenta a ferramenta de simulação usada no trabalho, o processo de configuração do ambiente de simulação, configuração dos cenários usados e aproximações necessárias para obter resultados pretendidos.

O Capítulo 5 apresenta os resultados obtidos através de experimentação e transformados em informações utilizando ferramenta estatística, e a avaliação referente as métricas de desempenho utilizadas.

O Capítulo 6 descreve o que foi concluído ao término deste trabalho e sugestões para desenvolvimento de trabalhos futuros.

## 2. REFERENCIAL TEÓRICO

Neste capítulo apresentamos os principais conceitos utilizados ao longo do trabalho, entre os quais abordaremos as redes locais sem fio, as redes ad hoc, as redes tolerantes a atraso, os protocolos de roteamento das redes tolerantes a atraso, os protocolos baseados em contexto social e as redes sociais.

### 2.1. Redes locais sem fio

São redes que possibilitam a seus usuários conectividade entre eles e ao mundo externo através de uma infraestrutura híbrida, ou seja, trabalhando com parte cabeada e sem fio ao mesmo tempo. Os pacotes de dados, denominação dada às mensagens criadas em qualquer tipo de rede, devem ser encaminhados a uma estação base antes de serem entregues a qualquer outro ponto interno ou externo da rede sem fio.

Dentre as redes móveis sem fio temos as redes locais sem fio (Wireless Local Area Network – WLAN), que utilizam o padrão IEEE 802.11 e são usadas para estabelecer comunicação entre elementos dentro de uma LAN e, para que seja possível essa comunicação, um protocolo de acesso ao meio compartilhado é usado (CSMA/CA – *Carry Sense Multiple Access / Collision Avoid*), como exemplo de rede sem fio utilizando IEEE 802.11 temos a Figura 2.1 [38].

O CSMA/CA é um protocolo usado para analisar se o meio está sendo usado no momento que um nó deseja encaminhar pacote de dados a um destino, fazendo com que seja evitada a colisão dos mesmos no uso de um meio compartilhado durante a transmissão. Este modelo de rede é constituído por pontos de acesso e hospedeiros sem fios. Os hospedeiros sem fios podem ser *laptops*, impressoras e *tablets*, ou seja, qualquer elemento que possua a capacidade de utilizar a tecnologia 802.11 e se comunicar a uma estação-base.

Um ponto de acesso vem a ser uma estação base, sua funcionalidade é fazer com

que os elementos “associados” a ela e que compartilham o mesmo SSID (*Service Set Identifier*) consigam encaminhar pacotes de dados entre si e também podendo, quando o mesmo possuir acesso a algum provedor de Internet, ter acesso a meios externos. O termo “associado” significa que o hospedeiro móvel está no raio de alcance de transmissão da estação base. É de responsabilidade da estação-base realizar o encaminhamento das mensagens entre os nós e implementar políticas de segurança para acesso ao meio.

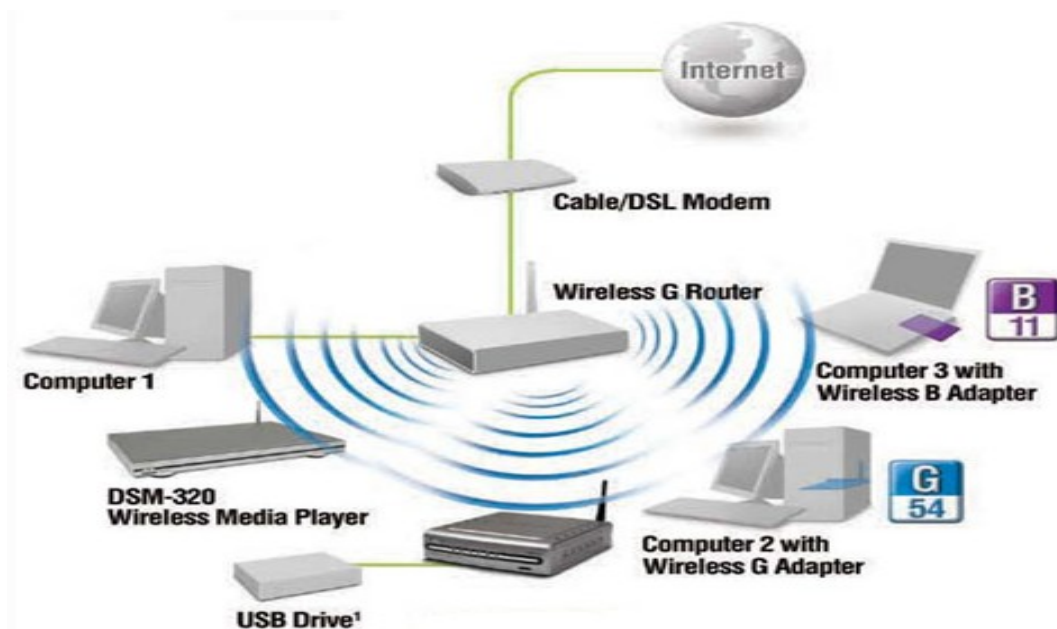


Figura 2.1 – Exemplo de uma rede sem fio utilizando o padrão IEEE 802.11 [36]

## **2.2. Redes ad hoc**

Ad hoc é uma abordagem de rede sem fio que opera sem uma infraestrutura pré-estabelecida. A mesma é composta por um conjunto de nós que estabelecem comunicações ponto a ponto, sendo que esse modo opera de forma distinta das redes infraestruturadas, pois para o funcionamento destas é necessário que haja um elemento centralizador responsável por controlar a rede.

Apesar da rede ad hoc estar em estudos a muito tempo, ainda encontramos desafios a serem superados tais como: problemas de conservação de energia, variação da topologia no decorrer do tempo, instabilidade nos enlaces de comunicação e suporte a escalabilidade.

Destacamos outras abordagens dentro do conceito de redes ad hoc: VANETs (redes ad hoc veiculares), MANETs (redes ad hoc móveis), redes oportunísticas, Bluetooth e redes de sensores.

As VANETs [33] são uma extensão da rede ad hoc que tem como foco principal viabilizar a transmissão de dados entre veículos utilizando abordagem ad hoc, ou seja, sem o uso de infraestrutura de comunicação ou ainda com o auxílio de uma infraestrutura ao longo das vias e rodovias. Na Figura 2.2 temos um exemplo do uso de uma VANET, onde podemos ver a utilização de antenas de curto alcance e os enlaces de comunicação que são estabelecidos quando há uma curta proximidade entre os carros ou através dos rádios instalados no topo dos postes ao longo das vias. Além disso, esses rádios podem possibilitar a comunicação com redes externas.

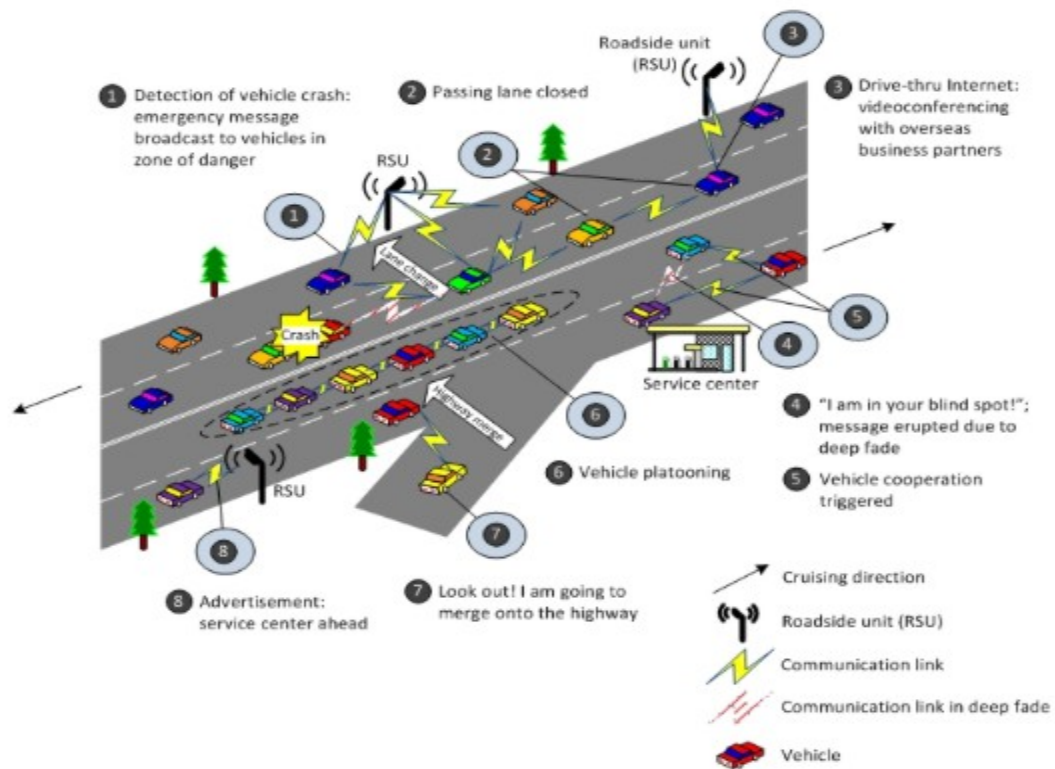


Figura 2.2 – Exemplo do uso e funcionamento de uma rede VANET [33]

As redes oportunísticas ou tolerantes a atrasos e a interrupções (DTN-Disruption/Delay Tolerant Network) são uma classe de MANETs que tem como característica os nós estarem distantes uns dos outros e serem tolerantes a atraso e a desconexões. Nessa variação de redes ad hoc, na maior parte do tempo, os nós não possuem nenhuma conexão fim a fim entre a origem e o destino, pois seus nós encontram-se isolados.

As trocas de mensagens ocorrem quando há proximidade entre os nós e um contato é estabelecido. Durante este contato os nós encaminham as mensagens que estão armazenadas em seus *buffers*. Assim, as mensagens vão sendo encaminhadas por nós na

rede até que sejam entregues aos seus respectivos destinos. Para isso, é necessário o uso de protocolo de roteamento ou encaminhamento. A Figura 2.3 mostra a transmissão de mensagens em uma rede MANET, explicando visualmente como uma mensagem que foi criada no notebook, seria encaminhada até o smartphone.

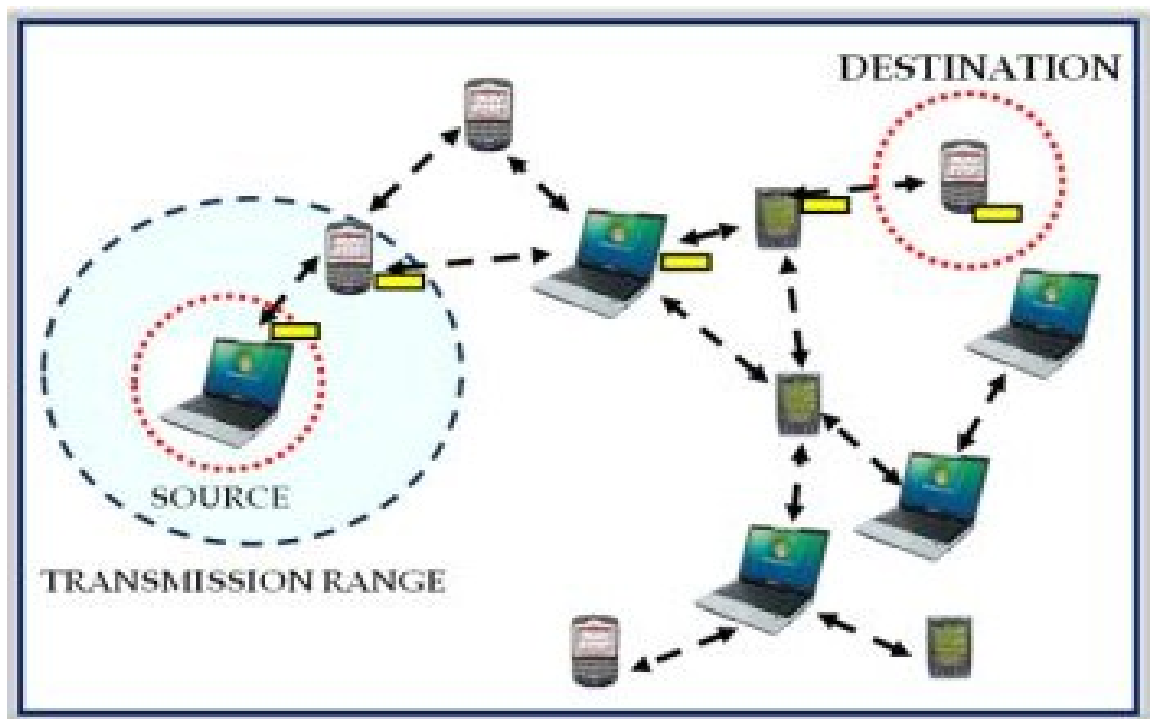


Figura 2.3 – Exemplo da transmissão de mensagens em uma rede MANET [35].

As redes Bluetooth podem ser um tipo de redes ad hoc que trabalham com um raio de alcance de 1 a 100 metros, dependendo da classe a que pertence o dispositivo. Detalharemos mais sobre essa tecnologia que foi escolhida para nossa simulação no



Capítulo 4 e na Tabela 4.3.

As redes de sensores [8] são uma especialização das redes ad hoc que trabalham em curto alcance. Esse tipo de rede é composta por sensores e estações base que se comunicam entre si por meio de *transceivers*, assim como podemos ver na Figura 2.4. Esta tecnologia tem uma vasta aplicação, podendo ser usada para o monitoramento de ambiente, monitoração industrial, detecção de presença, análises sísmicas, rastreamento de animais, pessoas e objetos e na área da saúde.

Os sensores são compostos de uma unidade de controle, de *transceivers* sem fio e de uma fonte de energia.

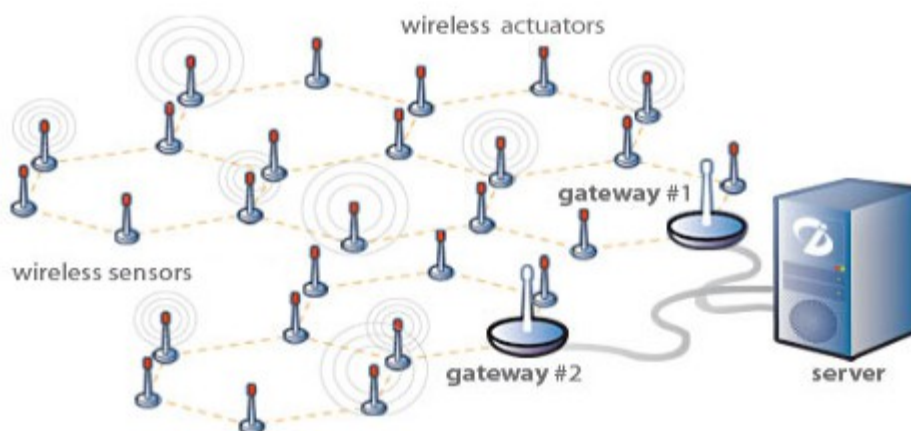


Figura 2.4 – Exemplo de uma rede de sensores [38]

A função do sensor é coletar os dados monitorados e encaminhá-los ao nó sorvedouro, de forma que os dados obtidos possam ser tratados por algum usuário, como podemos ver na Figura 2.5, onde temos um exemplo de aplicação de rede de sensores, que é usada, nesse caso, para monitorar atividades sísmicas em um vulcão. Na Figura 2.5 é possível ver que os sensores (em cor azulada) estão espalhados ao longo do vulcão monitorando os dados de variação de temperatura para que esses mesmos dados possam ser enviados para um nó sorvedouro ou estação base (dispositivo de cor vermelha).

As estações bases operam de forma similar às das redes VANETs, podendo possuir saídas para redes externas ou gateways de Internet. O número de elementos nesta arquitetura depende da aplicação para as quais os sensores serão empregados, podendo chegar a ordem de grandeza de milhares de nós presentes na rede. Para a transmissão de dados dos nós sensores até o sorvedouro também é necessário o uso de um protocolo de roteamento dos dados.

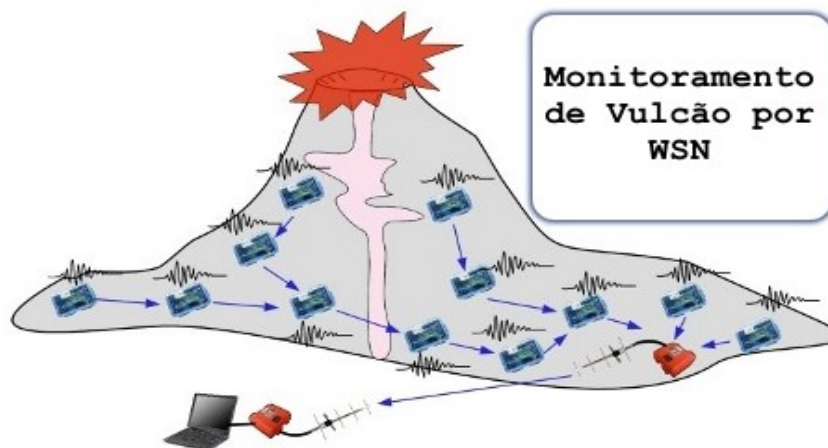


Figura 2.5 – Exemplo do uso da rede de sensores no monitoramento de atividades sísmicas em um vulcão [37]

### 2.3. Redes DTN

Essa abordagem de rede vem a ser uma classe de MANETs. O estímulo para o desenvolvimento desta abordagem surgiu quando se tentou adaptar protocolos baseados em TCP na criação das redes IPN (Interplanetary Internet) [44], pois protocolos baseados em TCP não se comportam da forma esperada em ambientes com longos atrasos, de longas distâncias e com desconexões frequentes.

O trabalho inicial na área [24] foi desenvolvido por Vint Cerf [43] e outros pioneiros no desenvolvimento da Internet. Naquele momento, foi desenvolvido um modelo de rede que levava em conta as características peculiares desta tecnologia, tais como atrasos significativos nas transmissões de dados e problemas na geração de pacotes com erros na transmissão.

A partir do trabalho desenvolvido nas redes IPN, Kevin Fall apresentou, em uma conferência no ano de 2003, um trabalho [25] sobre a motivação da criação das redes tolerantes a atrasos, sendo chamada de redes DTN. Este trabalho apresentou algumas adaptações das redes IPN para projetar um modelo de redes que funcionasse em ambientes hostis e sem infraestrutura, em ambientes terrestres. A partir de então, diversos trabalhos foram apresentados por pesquisadores em conferências e alguns grupos de discussões foram criados a fim de debater e desenvolver novas técnicas para essa nova abordagem.

As aplicações para uma DTN são as mais diversas possíveis. A mesma poderia ser usada em ambientes militares, em situações extremas nas quais nenhum canal de comunicação esteja operacional, em situações de calamidade pública nas quais toda infraestrutura tenha sido afetada por intempéries ou para o trabalho de exploração de uma nova área geográfica, num primeiro momento em que haja carência de tecnologias de comunicação de dados.

Têm surgido algumas propostas no desenvolvimento de redes que possibilitam conectividade tolerante a atrasos para Internet ou lidam com desconexões, como a rede esparsa em ambiente rural mostrada na Figura 2.6, que é um exemplo de DTN que vem sendo utilizada como uma opção de baixo custo de acesso à Internet em áreas remotas.

Essas áreas, representadas nessa figura como Região 2, estão distantes dos grandes centros com acesso à Internet e não existe conectividade fim-a-fim permanente entre o nó origem e o nó destino durante a sessão de comunicação estabelecida. Assim sendo, para superar estes obstáculos foram usados mensageiros, ou diríamos melhor, pontos de acesso móveis, como ônibus ou outro transporte equipado com um ponto de acesso, e um meio de armazenamento [39].

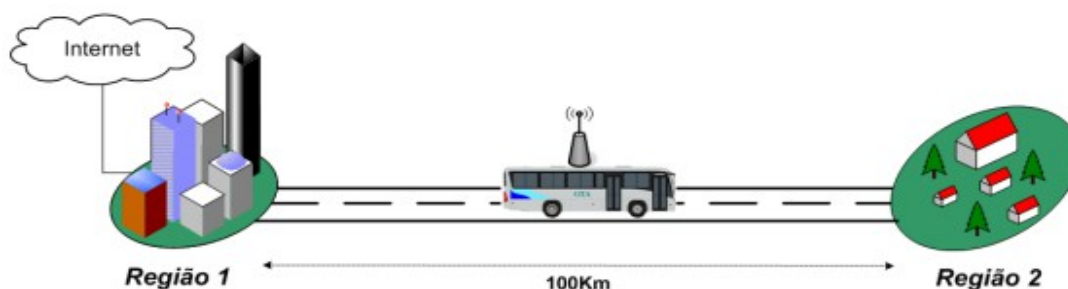


Figura 2.6 - Exemplo de uma rede DTN rural [39].

Foi desenvolvido também o projeto de *Drive-thru Internet*, uma rede híbrida que permite o acesso à Internet para veículos em velocidades de até quase 200 km/h.

A idéia desse projeto é fornecer *hot spots* ao longo do caminho, dentro de uma cidade ou até mesmo em auto-estradas. Estes *hot spots* precisam ser colocados de uma forma que um veículo obterá acesso por WLAN, ainda que por um período de tempo relativamente curto. Se os pontos de acesso estiverem localizados em áreas de descanso, o motorista pode passar lentamente ou até mesmo parar, prolongando, assim, o período de conectividade.

Um ou mais pontos de acesso localmente interligados formam a chamada ilha de conectividade que pode prestar serviços locais, bem como acesso à Internet.

Várias dessas ilhas de conectividade ao longo de uma estrada ou na mesma área

geográfica podem ser interligados e cooperar para fornecer acesso à rede com conectividade intermitente para uma área maior.

Este tipo de rede é considerada híbrida (vide Figura 2.7), pois tem aspectos de desconexões que devem ser gerenciados por protocolos específicos, quando o veículo passa de um ponto a outro de acesso à Internet [40].

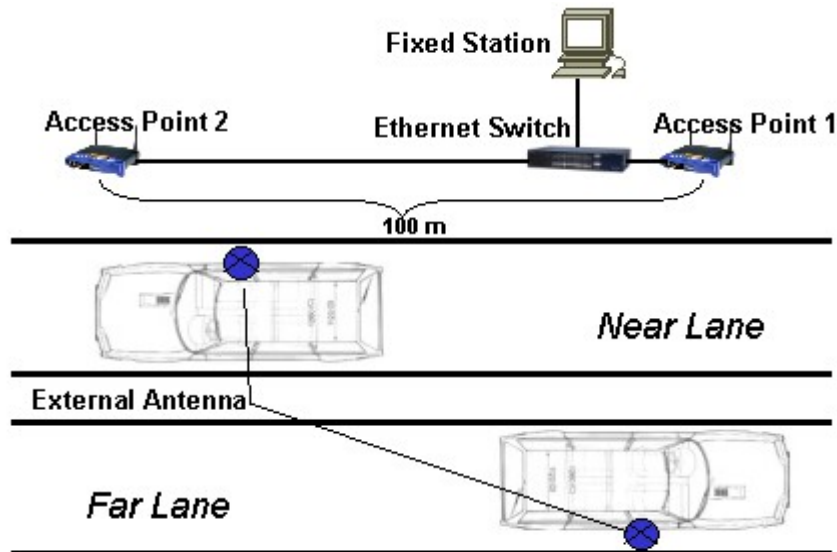


Figura 2.7 – Ilustração de uma *Drive-thru Internet* [40].

### 2.3.1. Redes PSN

A ideia de Redes de Comutação Portáteis (PSN – *Pocket Switched Networks*) foi introduzida primeiramente no ano de 2005 em [22], onde os autores descrevem uma rede oportunística composta de nós móveis (indivíduos carregando um *smartphone*, PDA ou similares), e recentemente vêm sendo consideradas um tipo especial de redes DTN.

O interesse em redes PSN veio da observação de que um crescente número de aparelhos de comunicação pessoal móvel adotava vários tipos de interfaces sem fio, permitindo não somente comunicação com uma infraestrutura de rede celular, mas

também com redes sem fio *peer-to-peer*. Como exemplos disso temos as tecnologias WiFi e Bluetooth.

Este modelo de redes oportunísticas pode ter grande dimensão e é composto basicamente de indivíduos móveis que possuem interfaces sem fio que permitam trocar mensagens diretamente entre esses indivíduos, como podemos ver na Figura 2.8.

Enquanto os componentes e protocolos das redes oportunísticas e, em particular, redes PSN (a maioria com WiFi ou Bluetooth), já estão relativamente amadurecidos, as aplicações para esses tipos de redes ainda têm muito para ser exploradas.

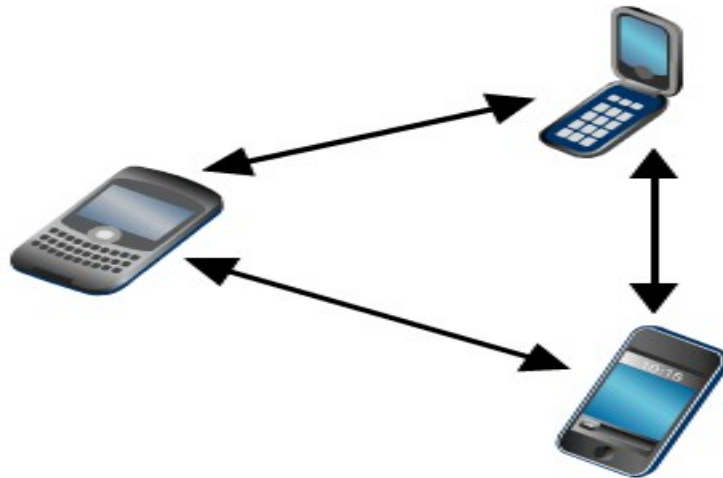


Figura 2.8 – Exemplo de rede PSN [42]

### 2.3.2. Protocolos de Roteamento em DTN

No ambiente DTN, os nós móveis estabelecem contatos entre os nós e, neste momento, as mensagens são transferidas e armazenadas em *buffer*, aguardando um próximo contato para serem encaminhadas, caso o nó destinatário não seja o alvo da mensagem.

O maior desafio no processo de roteamento em DTN é com a redução de atrasos nas entregas de mensagens e o aumento da taxa de entrega de mensagens na rede. E isso

requer tanto uma otimização do número de cópias de mensagens circulando na rede, quanto da estratégia usada para selecionar as mensagens a serem enviadas durante o contato com outro nó. Com respeito ao número de cópias de uma mensagem circulando na rede, pode ser observado que, quanto maior o número delas, maior são as chances para o destino obter contato com um nó carregando uma cópia da mensagem. Contudo, quanto maior o número de cópias, maior o número de mensagens circulantes, aumentando a chance da mensagem ser descartada por falta de espaço nos *buffers* dos dispositivos.

Algumas técnicas foram propostas na literatura para o encaminhamento de mensagens, sendo que o que há de comum entre elas é o fato das mesmas trabalharem com roteamento baseado em replicação. Estes protocolos possibilitam uma melhor taxa de entrega em relação a protocolos que trabalham com roteamento baseado em encaminhamento [24]. Estes tipos de protocolos serão melhor explicados a seguir.

#### **2.3.2.1. Protocolo Epidêmico**

O Protocolo Epidêmico [15] foi a primeira proposta desenvolvida para o cenário das DTNs e baseia-se na disseminação de cópias das mensagens a encaminhar a todos os elementos da rede, que ainda não a tenham recebido. Este modelo inunda rapidamente a rede com cópias das mensagens, sendo este o protocolo que garante a melhor taxa de entrega com menor atraso, mas com custos muito elevados. Este protocolo é também usado para garantir que as bases de dados distribuídas consigam manter a sua sincronização. A estratégia usada para o reencaminhamento de mensagens é muito eficiente em encontros oportunistas, mas, em situações realistas, o custo de recursos utilizados é significativo. Os novos protocolos baseados nesta abordagem fazem uso de heurísticas para decidir se é feita a entrega ou não de uma cópia aos diversos nós que entrarem em contato.

### 2.3.2.2. Protocolo Spray and Wait

Os protocolos baseados em epidemia permitem um grande número de mensagens disponíveis na rede, fato que faz com que haja um grande número de recursos gastos em cópias que serão desnecessárias quando uma dessas cópias chega ao destino. Um novo paradigma é tentar obter o melhor dos algoritmos de inundação, limitando a quantidade de cópias disseminadas na rede.

No protocolo Spray and Wait [14], isso é realizado por meio de duas fases distintas. Uma primeira fase (Spray) em que o dispositivo que gera uma mensagem decide o número de cópias ( $L$  cópias) que vão ser distribuídas pela rede e copia as  $L$  cópias para os  $L$  dispositivos que encontra. Quando uma unidade ou nó recebe a cópia, o protocolo entra na segunda fase, i.e. fase de espera (Wait). Nesta segunda fase, o dispositivo aguarda por contactar o destinatário e entregar a mesma por transmissão direta.

Uma outra versão deste mesmo protocolo, o Spray & Wait binário [14], estipula que, na primeira fase, em vez de entregar  $L$  cópias da mensagem aos  $L$  nós com que entrar em contato, o dispositivo deverá entregar somente  $L/2$  cópias das mensagens que detém aos  $L/2$  dispositivos que encontrar, até que um nó receba apenas uma mensagem. Nesse momento, esse nó passa para o modo de espera (Wait) até que a mensagem tenha o seu TTL expirado ou a mesma ser entregue ao destinatário por transmissão direta. Para ambas as alternativas, o tempo de entrega das mensagens está inteiramente relacionado com o tempo usado na 2ª fase do processo. Os autores deste segundo método o indicam como o mais eficiente das várias versões apresentadas, porque usa um número limitado de cópias e processa o roteamento das mensagens em duas fases distintas, evitando a típica sobrecarga na rede, comum em protocolos tipo Epidêmico, que enviam múltiplas cópias.



### **2.3.2.3. Protocolos baseados em métricas sociais**

Nesse trabalho, o nosso foco de atenção será um tipo específico de redes oportunísticas, chamadas de PSN [22], nas quais os nós que formam as redes são indivíduos carregando aparelhos portáteis de comunicação, como os *smartphones*. Essa classe de rede é muito interessante sob o ponto de vista de modelo de mobilidade, já que os nós da rede são carregados por pessoas, por isso a caracterização de mobilidade de nós PSN é equivalente a da mobilidade humana, a qual está sendo bastante investigada em campos como a sociologia, a engenharia de transporte, o planejamento urbano e assim por diante. O que é de nosso interesse em particular é tentar compreender as interações sociais entre indivíduos e seus padrões de mobilidade. Entender esses relacionamentos é fundamental para, por exemplo, otimizar o desempenho de protocolos de roteamento sociais, projetar primitivas de comunicação sociais, projetar e otimizar serviços inovadores para aplicações de redes sociais móveis.

Destacamos como interesse principal do nosso trabalho um protocolo que usa métricas sociais para decisões de encaminhamento de mensagens, o protocolo BubbleRap[2]. Descreveremos mais detalhes sobre ele após discorrer sobre a teoria envolvida em seu funcionamento, como veremos nos tópicos seguintes.

## **2.4. Redes sociais**

“Uma rede social é uma estrutura social feita de indivíduos e é um conjunto de conexões entre indivíduos representando algum tipo de interdependência social, como amizade, parentesco, interesses comuns e assim por diante.” [8]

As redes sociais, que também foram originalmente definidas no campo da sociologia como medida de capital social (i.e., o valor que um indivíduo obtém por ser parte de uma estrutura social), agora se tornam uma ferramenta para explicar fenômenos complexos em diferentes campos científicos, como na antropologia, biologia, economia,

ciência da computação e geografia.

A análise de redes sociais foi usada na epidemiologia para ajudar a entender como diferentes padrões de mobilidade humana ajudam ou inibem o dispersar de doenças infecciosas, tal qual HIV em uma população, como também pode ser usada para estimar a medida de influência de líderes de opinião em espalhar ideias inovadoras entre uma população. Dentro dessa análise, o estudo de estruturas sociais tornou possível a identificação de propriedades fundamentais, como a presença de nós “socialmente centrais” (nós líderes) e a existência de comunidades dentro de uma grande rede social. Sendo assim, descrevemos a seguir algumas métricas, utilizadas nas redes de computadores, definidas num grafo de rede social considerando o tipo de rede PSN.

#### **2.4.1. As métricas para redes sociais**

A seguir, as métricas mais relevantes para caracterizar a estrutura dos grafos de redes sociais, que podem ser usadas para identificar os nós mais importantes numa rede social ou para identificar uma comunidade dentro de uma grande estrutura social.

Primeiramente introduzimos uma classe de métrica destinada a medir a “centralidade social” dos nós da rede e depois uma para medir o grau de clusterização observado na rede. Devemos observar que as métricas definidas aqui podem ser aplicadas tanto para o grafo inteiro da rede social ou para parte dele.

##### **2.4.1.1. Métricas de centralidade**

As métricas de centralidade se destinam a medir a importância de um nó dentro de um grande grafo. Quando aplicadas às redes sociais, as métricas de centralidade são usadas para identificar nós socialmente influentes. A seguir introduzimos algumas métricas de centralidade e discutimos como elas têm sido usadas para otimizar o desempenho de

protocolos de roteamento sociais para PSNs. Usamos  $G = (V, E)$  para denotar um grafo de rede social com conjunto de nós  $V$  e conjunto de arestas  $E$  (não-direcionadas) e assumimos que  $G$  é conexo.

Uma métrica de centralidade bem simples é a centralidade de grau do nó (*node degree centrality*), isto é, o número de arestas em  $E$  que incidem em um nó. Intuitivamente, os nós com grau mais alto tem mais laços sociais e ainda serão melhor conectados socialmente do que os nós com grau mais baixo. Contudo, foi observado que geralmente a centralidade de grau de um nó não é suficiente para capturar a “importância social” de um nó dentro da rede, pois pode haver, por exemplo, um nó que é parte de uma comunidade isolada dentro de uma rede maior que pode ter uma centralidade de grau relativamente alta, por ter muitas conexões com outros membros de sua comunidade, mas contudo ter pouca “influência social”, devido ao isolamento da comunidade a que pertence [8].

Uma segunda métrica largamente usada é a centralidade de intermediação (*betweenness centrality*), que mede a fração de caminhos mais curtos entre dois nós na rede que passam por um nó específico. Temos então,

$$CI(u) = \sum_{s \neq u \neq t \in V} \frac{\delta_{st}(u)}{\delta_{st}}$$

onde  $\delta_{st}$  é o número de caminhos mínimos do nó  $s$  ao nó  $t$  em  $G$  e  $\delta_{st}(u)$  é o número desses caminhos que passam pelo nó  $u$ . O valor acima pode ser normalizado dividindo-se por  $(n-1)(n-2)/2$ , que representa o número total de pares de nós em  $G$  que não incluem  $u$ , onde  $n$  é a cardinalidade do conjunto de nós  $V$ .

A centralidade de intermediação pode ser usada para medir o quão eficiente é um nó em disseminar informação dentro da rede social: se um nó  $u$  com alta intermediação for removido da rede, a informação vai levar mais tempo para se propagar, já que muitos caminhos mais curtos não existiriam mais após a remoção do nó  $u$ , desde que estivessem sendo usados os caminhos mais curtos.

A terceira métrica é baseada na noção topológica de proximidade, a centralidade de proximidade (*closeness centrality*), que mede o quão perto um nó está de outros nós da rede. Dentro do contexto de redes sociais, a distância entre dois determinados nós da rede é definida em termos de número de saltos no grafo da rede social e é chamada de distância geodésica. Formalmente, a distância geodésica  $dG(u,v)$  entre quaisquer dois nós em  $G$  é definida como a medida do número de saltos do menor caminho que conecta  $u$  a  $v$  em  $G$ . A centralidade de proximidade [8] é definida como:

$$CP(u) = \frac{1}{\sum_{t \in V - \{u\}} dG(u, t)}$$

A centralidade de proximidade nos dá o inverso do valor do somatório das distâncias geodésicas do nó  $u$  em relação aos outros nós de  $G$ , em outras palavras, pode ser usada para medir o quanto uma mensagem gerada no nó  $u$  vai levar para alcançar todos os outros nós na rede social.

Um exemplo de um grafo de rede social é mostrado na Figura 2.9 e a métrica de centralidade correspondente a cada nó é dada na Tabela 2.1 [5]. Os nós mais centrais, de acordo com as diferentes métricas, estão destacados na mesma figura.

O nós com o valor mais alto de centralidade de grau estão destacados na cor verde (nós 1, 3 e 4) e o nó com o valor mais alto de centralidade de intermediação e, por acaso, também com o valor de centralidade de proximidade mais alto (nó 5) está destacado na cor azul.

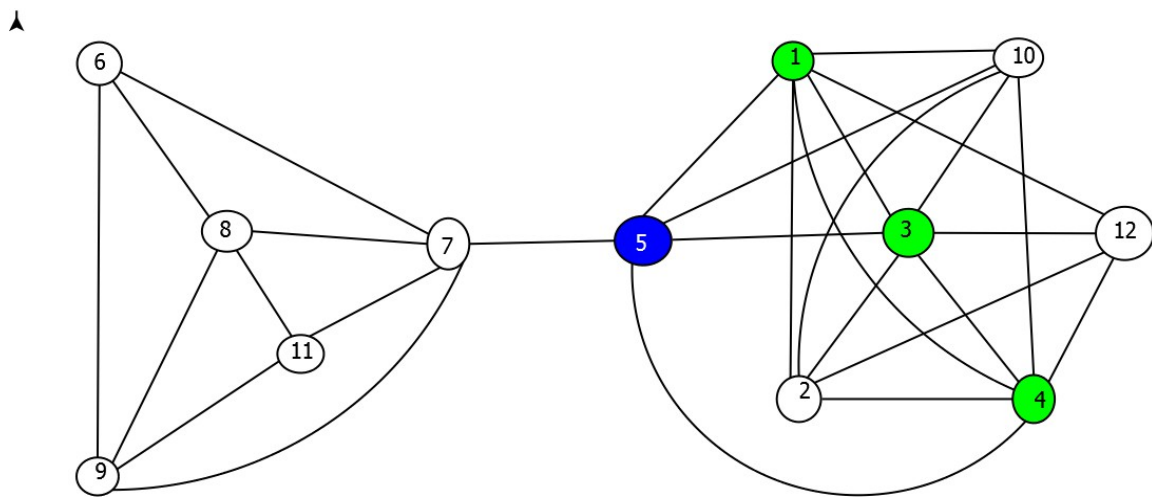


Figura 2.9 Exemplo de um grafo de rede social [5].

Nó	Centralidade de Grau	Centralidade de Intermediação	Centralidade de Proximidade
1	6	3,75	0,050
2	5	0,25	0,038
3	6	3,75	0,048
4	6	3,75	0,050
5	5	30	0,059
6	3	0	0,033
7	5	28,33	0,053
8	4	0,33	0,038
9	4	0,33	0,038
10	5	1,5	0,048
11	3	0	0,037
12	4	0	0,037

Tabela 2.1 Métricas de centralidade da rede social mostradas na Figura 2.9

As métricas de centralidade têm sido usadas nos projetos de protocolos de roteamento para PSNs [8]. A ideia delineada aqui é que os nós relativamente mais centrais no grafo de rede social seriam os preferidos para carregar cópias de mensagens para os nós relativamente menos centrais, já que estes têm uma possibilidade maior de entrar em contato com muitos outros nós na PSN. Exemplos de protocolos que exploram essa ideia são SimBet [20] e BubbleRap [2]. Ambos utilizam a centralidade de intermediação (combinada com outras medidas) para identificar os nós centrais na rede.

#### 2.4.1.2. Métrica de “clusterização”

A métrica de “clusterização” tem por objetivo medir o grau com que os nós de um grafo tendem a se aglomerar. Por sua vez, os aglomerados (*clusters*) de nós podem ser identificados como comunidades dentro de uma grande rede social.

Antes de nos referirmos a definição dessa métrica, valhe ressaltar a definição de clique, da Teoria dos Grafos: “Clique ou grafo completo é um grafo, ou subgrafo, em que seus vértices são interligados ou adjacentes dois a dois; de forma que o caminho mais curto entre quaisquer dois vértices  $v$  e  $w$  é a aresta  $(v,w)$ .” [45].

O coeficiente de “clusterização” de um nó  $u$  mede a capacidade dos vizinhos dele em formar um clique. Segundo [18], formalmente, o coeficiente de “clusterização” do nó  $u$  é definido como:

$$CC(u) = \frac{2|Eu|}{deg(u)(deg(u)-1)}$$

onde  $Eu$  é o conjunto de arestas do conjunto  $E$  tal que ambos pontos finais são vizinhos do nó  $u$  em  $G$ . Nessa fórmula,  $deg(u)(deg(u)-1)/2$  representa o número de possíveis

arestas entre dois vizinhos quaisquer do nó  $u$ , onde  $\deg(u)$  é o grau do nó  $u$ . Assim, o coeficiente de “clusterização” representa a fração de arestas existentes entre vizinhos de  $u$  sobre o número total de tais possíveis arestas. O coeficiente de “clusterização” igual a 1 indica que os vizinhos de  $u$  formam um clique (“clusterização” máxima), enquanto um coeficiente de “clusterização” igual a 0 indica que os vizinhos de  $u$  estão arrumados como uma estrela com centro em  $u$  (“clusterização” mínima).

Embora as métricas de “clusterização” têm por objetivo medir o nível de aglomeração em uma rede social, o uso delas, por si só, não serve para identificar comunidades dentro da rede. Achar comunidades em uma rede social é uma tarefa desafiadora a começar pela definição de comunidade, que não é única, e pela complexidade computacional dos algoritmos de detecção de comunidades, que é, em geral, NP-difícil.

Intuitivamente, uma comunidade é um grupo de entidades sociais, dentro de uma grande rede social, que se associam umas às outras por laços sociais. Contudo, transformar essa noção intuitiva em uma definição formal não é tão simples. Como não existe um consenso na própria definição de comunidade, o conjunto de algoritmos de detecção de comunidades proposto na literatura também é muito variado. A seguir, mencionaremos uma definição bem simples de comunidade que é usada na literatura de redes oportunísticas.

A definição de comunidade [19] é: “Um conjunto de nós  $C \subseteq V$  pode ser definido como uma comunidade se um conjunto de arestas internas a  $C$  formar um clique. Mais formalmente, o conjunto  $C$  é uma comunidade se, e somente se, o número de arestas internas for definida como  $|C| (|C| - 1) / 2$ .”

Note que essa definição leva em conta somente o número de arestas internas para definir uma comunidade, enquanto a razão entre o número de arestas internas e externas não é considerada. Assim, qualquer subconjunto de nós  $V' \subseteq V$  em um grafo completo  $G = (V, E)$  pode ser considerado como uma comunidade de acordo com essa definição. Essa definição de comunidade é bem estrita em afirmar que cada membro da comunidade deve possuir uma laço social com todos os outros membros.

### 2.4.2. Protocolo BubbleRap

O BubbleRap [2] é um protocolo de roteamento baseado em regras sociais, como por exemplo, o fato de um nó pertencer a uma determinada comunidade ou o fato de um nó ser mais popular em relação a outros nós. A maior popularidade de um nó está diretamente relacionada com a tendência que um nó tem de se encontrar com outros nós.

O protocolo utiliza dois algoritmos de detecção de comunidade (*community detection*): K-clique e Weighted Network Analysis, que identificam comunidades dentro de um conjunto de nós do grafo social. Também utiliza uma abordagem pré-calculada da métrica de centralidade de grau, isto é, o *global rank*, que é a centralidade de grau considerada do ponto de vista da comunidade global, e o *local rank*, que é a centralidade de grau dentro da comunidade do nó destino.

No protocolo BubbleRap [2], a centralidade de intermediação é usada para achar nós “pontes” que carregam as mensagens até a comunidade do nó destino, uma vez que cada nó pertence a ao menos a uma comunidade.

Uma vez apontado para a direção da comunidade correta, o roteamento das mensagens é feito somente para nós de dentro da própria comunidade do nó destino, com base na avaliação da centralidade de grau local.

No pseudocódigo do Algoritmo 1, a função LabelOf(node w), sendo w um nó qualquer, representa uma forma de detecção de comunidade através da identificação por Label, com cada nó sendo marcado por um único Label, que identifica a comunidade a qual o nó w pertence.

Esse mesmo algoritmo é a versão simplificada do algoritmo do BubbleRap, que foi ampliada e implementada em nosso trabalho e é baseado no conceito de comunidade e nas métricas de centralidade de grau global e de centralidade de grau local, esta última considerada quando se está dentro da comunidade do nó destino.

Na nomenclatura usada no algoritmo do BubbleRap a centralidade de grau global é chamada de GlobalRank e a centralidade de grau local é chamada de LocalRank.



Algoritmo 1. Pseudocódigo do algoritmo BubbleRap na versão simplificada

```
begin

foreach EncounteredNode_i do

    if ( LabelOf (currentNode) == LabelOf (destination) ) then

        if ( LabelOf (EncounteredNode_i) == LabelOf (destination) ) and
            ( LocalRankOf (EncounteredNode_i) > (LocalRankOf (currentNode) )
        then
            EncounteredNode_i.addMessageToBuffer ( message )
        else
            if ( LabelOf (EncounteredNode_i) == LabelOf (destination) ) or
                ( GlobalRankOf (EncounteredNode_i) > GlobalRankOf (currentNode) )
            then
                EncounteredNode_i.addMessageToBuffer (message)

    end

end
```

O processo de encaminhamento inicia quando se deseja enviar uma mensagem de um nó origem (*Source*) a outro nó, chamado de nó destino (*Destination*) como vemos na Figura 2.10.

O nó atual “borbulha” a mensagem caminho acima na árvore hierárquica da centralidade de grau da comunidade global até encontrar um nó que pertença a subcomunidade do nó destino (círculo grande em azul claro na Figura 2.10). A partir daí, o algoritmo usa a centralidade de grau local do nó para “borbulhar” <sup>1</sup> a mensagem novamente, agora já dentro da subcomunidade do nó destino, caminho acima da árvore de centralidade de grau local até achar o nó destino ou a mensagem expirar (ver ilustração desses itens na Figura 2.10).

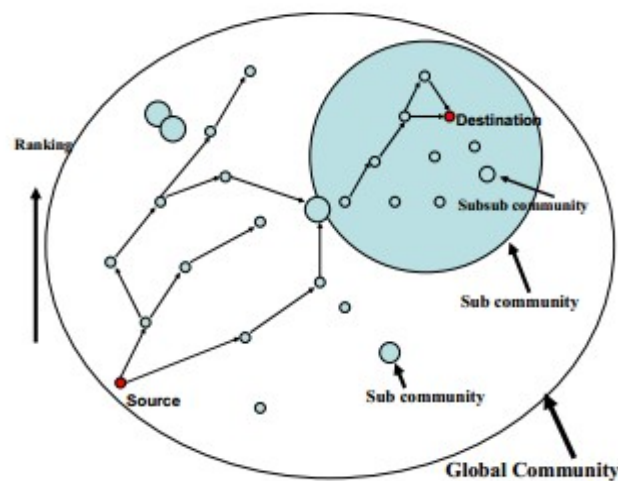


Figura 2.10 - Ilustração do algoritmo BubbleRap [2]

<sup>1</sup> O termo “borbulhar” significa passar a mensagem de um nó de grau menor para um nó de grau maior com o qual entra em contato e assim por diante.

### 3. TRABALHOS RELACIONADOS

O protocolo Spray and Focus [30] supera as desvantagens apresentadas pelos algoritmos baseados em inundação, como o Epidêmico, em termos de sobrecarga de mensagens na rede. E, embora protocolos de replicação controlada, como o Spray and Wait, gerem e distribuam mensagens eficientemente em termos de desempenho, em alguns cenários é necessário muita mobilidade nos nós da rede para atingirem esse objetivo.

O protocolo Spray and Focus apresenta uma solução ótima para o desafio de encaminhamento de mensagens nas redes oportunísticas ou DTN. Esse protocolo leva em consideração dois fatores: primeiro, o número de cópias a serem usadas para cada mensagem, chamado de  $L$ , cujo cálculo do valor ótimo nesse caso não é tão trivial como o Spray and Wait que usa o valor  $L$  ou  $L/2$  (no caso do “binary” Spray and Wait) e o segundo fator são os “*timers*”, que ficam registrados nos últimos contatos dos nós. Esse último fator é usado constantemente no cálculo usado pelo utilitário de tomada de decisões durante a fase “Focus”, diferentemente da fase “Wait” do protocolo Spray and Wait que encaminha mensagens usando o conceito de transmissão direta, sem tomada de decisão.

Na apresentação desse protocolo foi mostrado que a combinação do conceito de replicação controlada e de encaminhamento baseado em um utilitário leva a um roteamento eficiente nessas redes desafiadoras.

Em [28] é apresentada uma abordagem de avaliação de desempenho do protocolo PeopleRank, nos modos centralizado e distribuído, usando, para tal, traces de mobilidade real e interações sociais dos nós (informações do Facebook, por exemplo).

Similarmente à ideia do algoritmo do PageRank [29], que se utiliza de um “Random Walk” no grafo da World Wide Web, onde os nós desse grafo são as páginas da Web e as arestas são os links entre as páginas, o PeopleRank também baseia seus algoritmos de encaminhamento em um *ranking* dos nós. O algoritmo do primeiro se caracteriza por aumentar o valor do *rank* do nó quanto mais *links* este possuir com outras páginas ou com outros nós.

Seguindo um raciocínio análogo para o caso do PeopleRank, em que o nó é uma pessoa, quanto mais ligações sociais possuir, maior será o valor do seu *rank*.

O PeopleRank utiliza um fator probabilístico em sua fórmula e esse fator determina a intensidade da relação social entre os nós em questão. Como exemplo, poderia ser definido um valor desse fator igual a 1 (valor máximo), para uma relação de parentesco, ou, ainda, definir esse fator em 0.5, para expressar a relação de amizade, a qual seria mais forte do que uma relação de “interesses em comum”, que teria então um valor de 0.2. Visto que o valor mínimo desse fator foi definido como sendo zero. Notamos que, no algoritmo do PeopleRank, quanto mais frequentemente dois nós se encontram, mais rápido aumentam as suas classificações no *ranking* social.

Segundo esse artigo [28], o desempenho de protocolos como o PeopleRank é determinado por dois fatores conflitantes, como atraso na entrega de mensagens e a sobrecarga de mensagens na rede, e, no mesmo trabalho, foi feita uma análise comparativa de desempenho com outros protocolos sociais que usam em seus algoritmos as métricas centralidade de intermediação e centralidade de grau.

Em [32], os autores investigam, através de simulação, como a inclusão de elementos com características de movimentação especiais influenciam no desempenho de uma DTN sem nenhuma infraestrutura adicional de *hardware* e *software*. E, na avaliação realizada, foi mostrado que todos os quatro protocolos (Epidêmico, Spray and Wait, PROPHET [4] e MaxProp [23]) foram influenciados positivamente, tendo mostrado melhoras nas métricas de desempenho, como sobrecarga, atraso médio, probabilidade de entrega e número de saltos para as mensagens entregues durante a simulação com nós especiais. Além disso, foi feito um levantamento da participação de cada nó na entrega das mensagens. Destacaram-se entre os resultados obtidos das métricas, os valores do protocolo MaxProp, que obteve os melhores resultados por ser desenvolvido para uma rede veicular.

## 4. SIMULAÇÃO DOS PROTOCOLOS DE ROTEAMENTO DTN EM UM CENÁRIO COM CARACTERÍSTICAS REAIS

Neste capítulo é apresentada uma avaliação comparativa de desempenho dos protocolos Epidêmico e Spray and Wait em comparação com o protocolo BubbleRap, que utiliza métricas sociais.

Como nosso enfoque nesse trabalho é em movimentos de pedestres, ou seja, de pessoas, investigamos como a variação dos raios de alcance do Bluetooth dos dispositivos móveis pode ser explorada para melhorar a performance nas mesmas redes.

Sobre a avaliação realizada, mostramos nos resultados obtidos das simulações como foi que o tamanho do *buffer*, que foi definido nos cenários de simulação como sendo de 4 Mbytes ou 128 Mbytes, influenciou na mudança de comportamento das métricas de desempenho em cada protocolo, variando-se a carga de dados nas mensagens da rede e variando-se o raio de alcance dos nós.

### 4.1. Métricas de desempenho

Usamos as seguintes métricas de desempenho para avaliar os protocolos:

- i) Taxa de entrega, que contabiliza o total de mensagens entregues em relação ao total de mensagens que foram enviadas. Esta métrica permite analisar a eficiência do protocolo na entrega de mensagens;
- ii) Sobrecarga de mensagens, definida como sendo o resultado do total de mensagens geradas na rede menos o total de mensagens entregues, dividido pelo total de mensagens entregues. Foi escolhida essa métrica porque ela traduz, de maneira geral, o quanto de recursos dos nós estão sendo consumidos, como por exemplo o quão crítico está o armazenamento de mensagens nos *buffers* dos nós;
- iii) Atraso médio na entrega de mensagens, definido como sendo o somatório dos atrasos

de todas as mensagens entregues dividido pelo número total de mensagens entregues. Por essa métrica identificamos o tempo médio gasto pelos protocolos para entregar as mensagens ao destino.

#### **4.2. Modelos de mobilidade sintéticos usados no ONE.**

O modelo de movimento utilizado pelos nós pedestres da simulação é o Shortest Path Map-Based Movement model (SPMBM), que é uma versão avançada de modelo de movimento baseado em mapas. Inicialmente, este modelo define aleatoriamente o posicionamento dos nós e os pontos do mapa que serão designados como destinos para os mesmos. Em seguida, ele aplica o algoritmo Dijkstra [27] de melhor caminho, para escolher por qual caminho o nó vai seguir no mapa. Quando o nó chega ao destino, ele aguarda por um momento e um outro destino é definido para o mesmo. Normalmente, todos os lugares do mapa tem igual probabilidade de serem escolhidos como o próximo destino.

#### **4.3. Ferramentas utilizadas para a simulação**

Para que fosse possível alcançar nosso objetivo, foram usadas algumas ferramentas *opensource* que nos permitiram a configuração do cenário de simulação, a realização da própria simulação, o ajuste do mapa de uma área real da cidade usada na simulação, a geração de matrizes de pontos com base nos resultados obtidos do simulador, o tratamento estatístico dos resultados obtidos na simulação, a geração dos gráficos comparativos de desempenho.

Especificamos e descrevemos cada uma dessas ferramentas a seguir.

#### 4.3.1. ONE

O ambiente de simulação ONE [9] foi escolhido por ser um *software* livre e possuir ferramentas para configuração de diversos cenários de simulação de maneira rápida e prática. O simulador ONE foi escrito em Java e permite importar dados de mobilidade de *traces* do mundo real ou de outros geradores de mobilidade. Ele pode produzir um grande número de relatórios da movimentação dos nós, dos repasse de mensagens e das estatísticas gerais.

O ONE é um ambiente de simulação que tem a capacidade de gerar movimentação dos nós a partir de diferentes modelos de mobilidade, fazer roteamento de mensagens entre os nós utilizando protocolos de roteamento DTN e visualização da movimentação dos nós em tempo real através da interface gráfica.

Na realização deste trabalho, foram usados ambientes Windows e Unix, em máquinas diferentes, pois ao longo do desenvolvimento do trabalho, notamos que seria mais otimizado, em relação ao tempo, realizarmos as simulações através de *scripts* de CLI (*Command Line Interface*) ao invés da interface gráfica. Foram utilizados *scripts* do Windows e do Unix com extensões *.bat* e *.sh*, respectivamente, para automatizar o processo de simulação, o que nos permitiu dar início a diversas *threads* simultâneas e em máquinas diferentes.

Entre os protocolos de roteamento que o ONE possui de forma nativa, o Spray and Wait e o Epidêmico são usados em nosso trabalho. Além destes, o ONE permite a simulação com outros protocolos desenvolvidos externamente e que podem ser importados, o que nos possibilitou o uso do BubbleRap[2]. Isso foi feito a partir das contribuições de implementação do aluno de doutorado da Universidade de Pittsburgh, PJ Dillon[3].

O ONE permite a inclusão de arquivos de mapas no formato WKT, sendo que WKT é uma linguagem de marcação para representar vetores geométricos, inclusive mapas.

A Figura 4.1 mostra um exemplo da interface gráfica do ONE. Nela é possível identificar a área de simulação, o painel lateral com os nós, o painel inferior com os logs

de eventos e o painel superior com o controle da simulação. Ainda nessa figura podemos ver os nós em sua posição dentro da área de simulação. Cada nó é identificado à sua direita pela letra do grupo que pertence e sua numeração dentro deste grupo. Acima da tela é mostrado o cronômetro que registra o tempo de simulação, os botões de controle do simulador, centralizado na tela acima, e, mais à direita da tela, é mostrado o controle de atualizações da interface gráfica, a escala da imagem que será mostrada na tela e o botão de captura de imagens.

Na Figura 4.2, podemos ver o log dos eventos gerados durante o processo de simulação. Esses eventos podem ser filtrados através controle dos eventos de log na caixa de seleção ao lado esquerdo da tela, assim permitindo uma melhor análise do que ocorre durante a simulação.

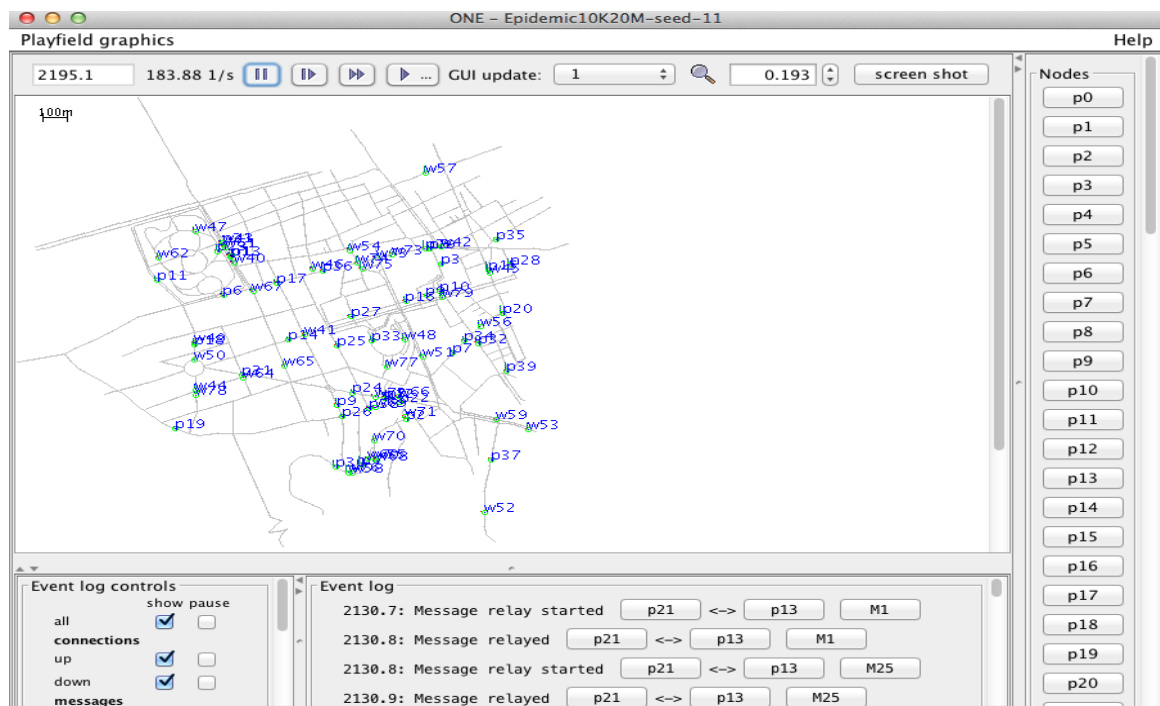


Figura 4.1. Tela da interface gráfica do ONE.



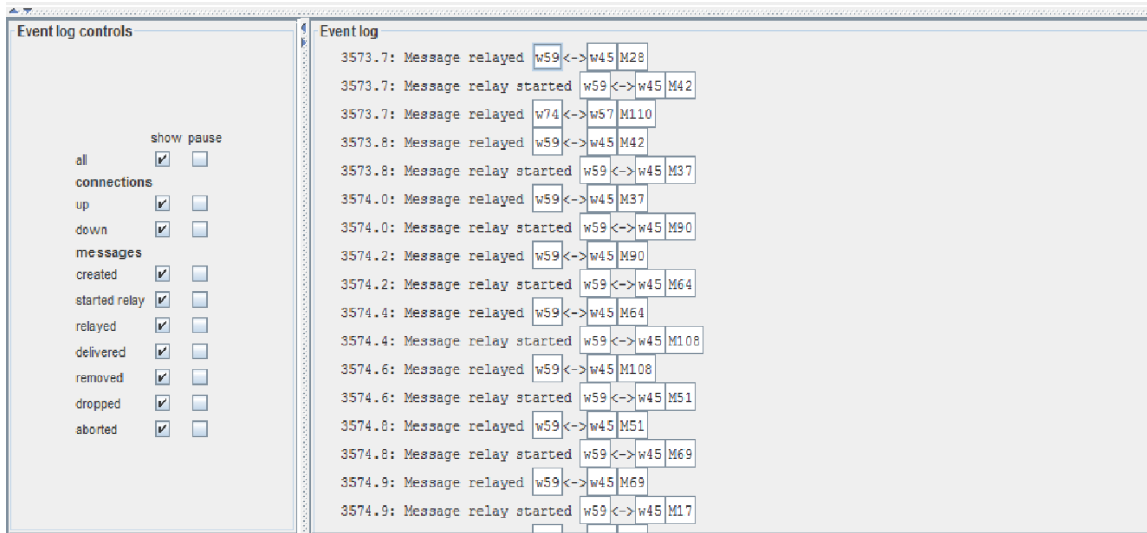


Figura 4.2. – Visualização do log dos eventos da simulação de redes DTN no ONE.

#### 4.3.2. NetBeans 7.2

O NetBeans 7.2 [12] é um *software* livre e de código aberto. Permite o desenvolvimento rápido e fácil de aplicações em Java para *desktop*, *mobile* e *web*, além de possuir uma grande comunidade de usuários e desenvolvedores em todo mundo. Este ambiente foi utilizado para a configuração dos parâmetros de simulação e os ajustes dos protocolos utilizados na simulação. O mesmo foi usado neste trabalho para desenvolver os aplicativos que lêem os diretórios contendo os relatórios no formato texto (com extensão TXT), para o cálculo das métricas envolvidas nas simulações e para a criação de arquivos formatados para o *software* estatístico R.

#### 4.3.3. OpenStreetMap

O projeto OpenStreetMap (OSM) [11] permite criar e distribuir dados geográficos gratuitamente, para a confecção de mapa e para a criação de arquivos na extensão XML com as coordenadas geográficas de um mapa. Para ter acesso ao *site*, é necessário criar um conta, colocar o endereço do local de interesse no pesquisador do *site* e selecionar o desejado. Após aparecer o mapa, devemos exportar o mesmo na extensão XML, logo em seguida o mapa é criado. O objetivo do OSM é disponibilizar um mapa gratuito e editável por qualquer pessoa que conheça uma região e esteja disposta a contribuir com informações da mesma. Além disso, o projeto permite que usuários construam os mapas de maneira colaborativa.

#### 4.3.4 Osm2wkt

O padrão do mapa utilizado pelo ONE utiliza a extensão WKT. Como foi necessário usar uma área geográfica para a qual não possuíamos um mapa nessa extensão, utilizamos o *software opensource* Osm2wkt[16], o qual é baseado em JAVA. Este *software* foi uma contribuição de Cristoph P. Mayer e foi executado nos ambientes Unix e Linux para a conversão do mapa utilizado nas simulações.

#### 4.3.5. Ferramenta R

A ferramenta R [13] é um *software* livre utilizado para o tratamento estatístico de dados e geração de gráficos. O mesmo pode ser executado sobre várias plataformas como Unix e Windows. O *software* R possui uma IDE amigável e permite a importação de dados para tratamentos estatísticos e a execução de *scripts* de comando do R. Assim, o mesmo foi utilizado neste trabalho para aproximar os dados obtidos das simulações por uma distribuição t-Student e plotar os gráficos comparativos das métricas de desempenho dos

protocolos analisados.

#### **4.3.6. Programas desenvolvidos**

Os programas desenvolvidos para esse trabalho encontram-se descritos nos Apêndices A, B e C.

#### **4.4. Tecnologias escolhidas para os cenários de simulação**

Bluetooth [10] é uma especificação industrial para áreas de redes pessoais sem fio (Wireless personal area networks – PANs). O Bluetooth provê uma maneira de conectar e trocar informações entre dispositivos como telefones celulares, notebooks, computadores, impressoras, câmeras digitais e consoles de videogames digitais, através de uma frequência de rádio de curto alcance. As especificações do Bluetooth foram desenvolvidas e licenciadas pelo "Bluetooth Special Interest Group".

Bluetooth é um padrão de comunicação primariamente projetado para baixo consumo de energia com baixo alcance. Dependendo da potência, o alcance varia de 1 metro a 100 metros, como podemos ver na Tabela 4.1. Sua tecnologia é baseada em *microchips* transmissores de baixo custo em cada dispositivo. Ele possibilita a comunicação desses dispositivos uns com os outros quando estão dentro do raio de alcance. Os dispositivos usam um sistema de comunicação via rádio, por isso não necessitam estar na linha de visão um do outro, e podem estar até em outros ambientes, contanto que a transmissão recebida seja suficientemente potente.

Dispositivos que possuem um amplificador de potência na transmissão têm uma sensibilidade de recepção melhorada, e existem antenas altamente otimizadas que normalmente alcançam distâncias de 1 km em linha de visão livre, usando o padrão Bluetooth classe 1 [21].

Classe	Potência máxima permitida (mW/dBm)	Alcance (Aproximadamente)
<b>Classe 1</b>	100 mW (20 dBm)	até 100 metros
<b>Classe 2</b>	2.5 mW (4 dBm)	até 10 metros
<b>Classe 3</b>	1 mW (0 dBm)	~ 1 metro

Tabela 4.1 Classes de dispositivos Bluetooth

Uma configuração típica de rede que usa padrão Bluetooth opera com um nó mestre e até sete nós escravos ativos na troca de mensagens, denominada de *piconet*, possuindo uma limitação de 255 nós estacionados (inativos). Quando interligamos um certo número de *piconets* temos então uma *scatternet*, a qual pode suportar a comunicação entre mais do que oito dispositivos.

A função do nó mestre é controlar e gerenciar os nós escravos na troca de mensagens e comutar os nós que não estão trocando mensagens para o estado inativo, com intuito de reduzir o consumo de energia. Na Figura 4.3, podemos ver um exemplo do uso tecnologia Bluetooth em redes ad hoc.

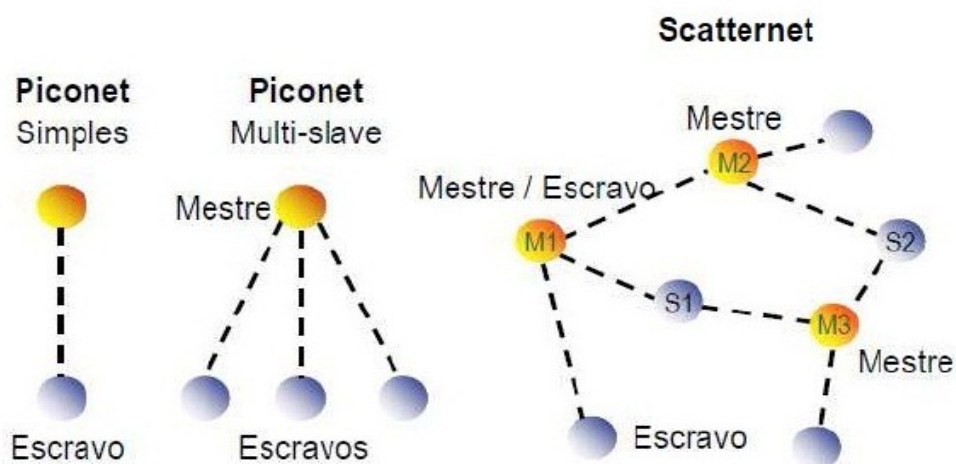


Figura 4.3 – Exemplo do uso da tecnologia Bluetooth em redes ad hoc [35]

#### 4.5. Cenários utilizados para as simulações

Em todas as simulações, temos 80 nós com configurações comuns, definidas no Capítulo 4.6 e na Tabela 4.2, que utilizam o mesmo modelo de mobilidade e trocam mensagens entre si. O comportamento dos nós é equivalente ao deslocamento natural de um pedestre. Para alcançar o objetivo deste trabalho, alguns parâmetros sofreram variação ao longo das simulações, esses parâmetros são: tamanho do *buffer*, raio de alcance dos nós e o tamanho da mensagem gerada por cada nó.

O ONE acessa as configurações de parâmetros de simulação comuns a todos os protocolos a partir do arquivo chamado `default_settings.txt`, localizado no pacote padrão do projeto ONE, mas pode acessar também outro arquivo (opcional) para cada protocolo específico. Logo, para cada protocolo utilizamos um arquivo de configuração `Nome_do_protocolo_settings.txt`, no qual incluímos os parâmetros desejados referentes ao protocolo, os quais vão sobrepor os valores já definidos no `default_settings.txt`, caso já existam.

Por questões de ganho de tempo de simulação, dividimos a simulação em 3 partes, sendo cada qual referente a um protocolo, cada um em uma máquina diferente, como descrito a seguir:

- i) simulação do protocolo Epidêmico[15], executado pelo arquivo `simscrip_t_Epidemic.sh`, sendo este um *script* de comandos do Unix, cujo código está referenciado no Apêndice A;
- ii) simulação do protocolo Spray and Wait [14], executado pelo arquivo `simscrip_t_SprayAndWait.bat`, sendo este um *script* de comandos do Windows, cujo código está referenciado no Apêndice A;
- iii) simulação do protocolo Distributed BubbleRap [2], executado pelo arquivo `simscrip_t_DistributedBubbleRap.bat`, sendo este um *script* de comandos do Windows, cujo código está referenciado no Apêndice A.

#### 4.6. Configurações de cenário comuns a todas as simulações feitas

O conteúdo do arquivo de configuração do ONE (`default_settings.txt`) é descrito a seguir.

Foi considerado um tempo total de simulação de 4 horas (14400 s), tomando por base o tempo que um nó pedestre leve para caminhar ao longo de todo o perímetro externo da área escolhida para a simulação do centro do Rio de Janeiro, com a velocidade média de 1 m/s. Teríamos a seguinte conta: se a velocidade média dele fosse de 0.5 m/s, teríamos 8000 m de perímetro dividido por 0.5 m/s, que seria igual a 16000 segundos. Fizemos então, uma aproximação do tempo, arredondando para 4 horas ou 14400 segundos, como sendo o tempo suficiente para um pedestre percorrer esse perímetro.

O tempo de *warm up*, i.e., tempo que o ONE começa a considerar os dados para a simulação, depois de iniciado o movimento, foi de 600 s (10 minutos). O *warm up* em um trabalho de simulação é o tempo que o ambiente da simulação sofre alterações importantes até sua “estabilização”. Após esse tempo, já estão começando a ser geradas mensagens, já houve movimentação inicial dos nós pela área de simulação e, conseqüentemente, houve também alguma troca de mensagens, sem ter havido, no entanto, perdas por enchimento do *buffer*, o que deve ser contabilizado.

Foram usadas 10 sementes (o *default* é 0) do gerador de números pseudo-aleatórios para o modelo de mobilidade do ONE, o que permitirá a geração de 10 rodadas de simulação para cada conjunto de parâmetros investigados.

A simulação foi feita usando 80 nós pedestres que caminham por todas as vias do mapa. A velocidade dos nós pedestres, por imposição do simulador ONE, só pode ser definida em forma de intervalo, como foi feito no trabalho, a mínima de 0,1 m/s e a máxima de 1,5 m/s. Como artifício poderia ser fixada a velocidade mínima igual a velocidade máxima como um valor único.

O TTL (*Time to Live*) é definido como o tempo que uma mensagem ou pacote pode ficar na rede antes de ser descartado. O TTL das mensagens geradas pelos nós foi definido em 120 minutos, ou seja, metade do tempo de simulação. Não foi nosso interesse nesse trabalho avaliar um TTL infinito, isto é, igual ou maior que o tempo de simulação total.

Todos os nós utilizam interfaces Bluetooth com velocidade de transmissão de 250kBps, equivalente a 2 Mbits. E a influência do raio de alcance da interface Bluetooth foi avaliada variando-se o mesmo de 2m, 5m, 10m, 20m, 30m, 40m, 60m, 80m.

O tamanho de *buffer* de mensagens variando em 4 Mbytes e 128 Mbytes. E o tamanho das mensagens foi definido como 10 Kbytes (baixa carga) ou 100 Kbytes (alta carga), com mensagens sendo criadas a cada 30 s em média. No conjunto de 80 nós da simulação, isso significa que, após o *warm up* de 600 s, já tenham sido criadas aproximadamente 20 mensagens.

Foi usado um modelo de mobilidade sintético ShortPathMapBasedMovement (SPMBM) que, inicialmente, coloca os nós em lugares aleatórios, como foi explicado no Capítulo 4.2.

O mapa escolhido para a simulação deste trabalho representa uma área real do centro da cidade do Rio de Janeiro de aproximadamente 4.000.000 m<sup>2</sup>. O mapa abrange a área do Campo de Santana ao Passeio Público. A Figura 4.4 mostra a área e a região destacada correspondente aos limites do mapa da simulação, obtido a partir do OpenStreetMap da área do centro da cidade. Com o trecho do mapa selecionado, foi usada uma extensão do ONE que permite a conversão de dados de localização do formato do OSM para o formato WKT, sendo então nomeado de cidade3.osm.wkt. O parâmetro de tamanho do modelo de movimento foi ajustado para 3000 X 3000 (comprimento e largura em metros), para que nenhum dos caminhos possíveis para a circulação de pedestres ficasse fora dos limites das coordenadas definidas no mapa.

A Figura 4.5 mostra um *snapshot* do simulador ONE durante a realização de uma simulação do nosso trabalho. Nesta figura é possível identificar a área de simulação, em que foi utilizado um mapa ajustado do centro da cidade do Rio de Janeiro, o painel lateral com os números dos nós e as posições dos nós nos caminhos do mapa num determinado momento da simulação. Na parte de baixo da mesma figura podemos ver o log dos eventos que estão ocorrendo na simulação. E o tempo decorrido da simulação logo na parte de cima, junto do painel de controle da simulação.





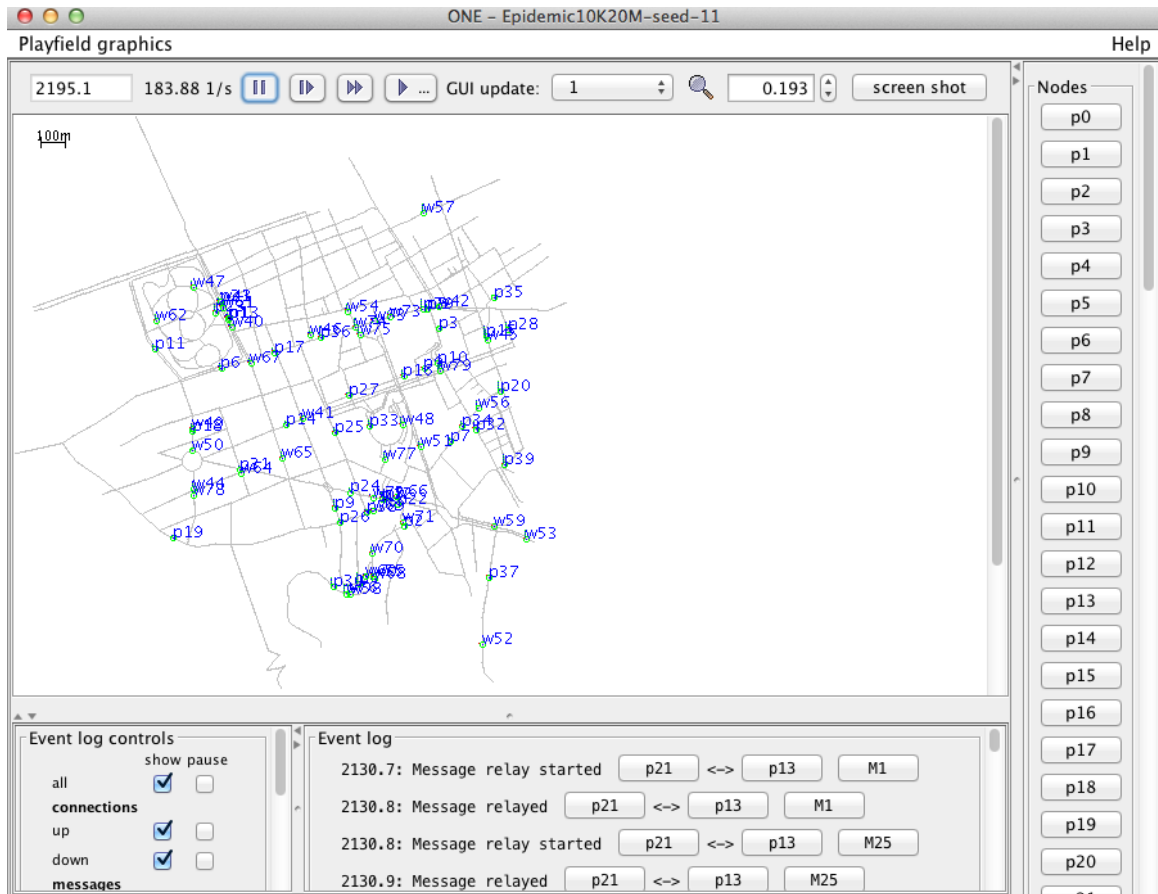


Figura 4.5 – *Snapshot* do simulador ONE durante a realização de uma simulação utilizando mapa do centro da cidade.

#### 4.8. Configuração de Cenário para o Protocolo Epidêmico

Para simular o protocolo Epidêmico no cenário da Figura 4.4 foi usado um *script* de comandos do Unix chamado `simscripEpidemic.sh`. A configuração específica para o protocolo Epidêmico foi feita no arquivo `epidemic_settings.txt`. Nesse arquivo ficaram definidos o nome do cenário de acordo com o tipo de protocolo de roteamento a ser seguido e os parâmetros do protocolo de roteamento em si.

#### **4.9. Configuração de Cenário com o Protocolo Spray And Wait**

Para simular o protocolo SprayAndWait no cenário da Figura 4.4 foi usado um *script* de comandos do Windows chamado `simscript_SprayAndWait.bat`.

A configuração específica para o protocolo Spray And Wait foi feita no arquivo `sw_settings.txt`. Nesse arquivo foram definidos: o nome do cenário de acordo com o tipo de protocolo de roteamento a ser seguido (Spray And Wait), os parâmetros do protocolo de roteamento em si e o número de cópias usadas no roteamento.

#### **4.10. Configuração de Cenário com o Protocolo Distributed BubbleRap**

Para simular o protocolo Distributed BubbleRap no cenário da Figura 4.4 foi usado um *script* de comandos do Windows chamado `simscript_DistributedBubbleRap.bat`. A configuração específica para o protocolo Distributed BubbleRap foi feita no arquivo `dbr_settings.txt`. Nesse arquivo foi definida a abordagem da versão distribuída do algoritmo BubbleRap, que usa uma solução pelo método de Divisão e Conquista em 2 passos, cada um com seus respectivos algoritmos:

- 1o.passo) `KcliqueCommunityDetection`, para detectar se o nó seguinte pertence a comunidade do nó destino. O valor de  $K$  pode variar de 1 a 7 (para  $K=7$ , se verifica a presença de uma única comunidade) e o valor de  $K=6$  é considerado um limiar ótimo, como podemos observar em [6]. Logo, usamos  $K=6$ . Esse algoritmo também foi usado por permitir a detecção de comunidades superpostas, ou seja, mesmo que essas comunidades contenham nós que também pertençam a outras comunidades, conforme o artigo [6].
- 2º passo) `CwindowCentrality`, para avaliar a centralidade de intermediação e centralidade de grau do nó avaliado. Esse algoritmo leva em consideração os contatos dos nós em janelas de tempo pré-estabelecidas, otimizadas para evitar a contabilização dos dados de contatos em um tempo muito longo, o que tornaria o

grafo de agregação de contatos muito poluído e os seus nós perderiam a heterogeneidade. Ou seja, todos os nós se tornariam muito “centrais”, devido ao fato desse grafo de agregação de contatos se aproximar cada vez mais de um grafo *full mesh*, perdendo o significado de centralidade para os nós. E também evitar uma janela de tempo muito pequena quando os valores das centralidades dos nós ainda não são significativos, por não ter havido sequer contatos entre os nós [7].

Para cada Protocolo (Epidemic[15], Spray and Wait[14] e Distributed BubbleRap[2]) procedemos dois ciclos de simulação: o primeiro ciclo com mensagens de 10 Kbytes, que chamamos de cenários de baixa carga de dados, e o segundo ciclo com mensagens de 100 Kbytes, que chamamos de cenário de alta carga de dados. A Tabela 4.2 mostra a configuração dos nós em cada simulação.

As configurações comuns a todas as simulações são o tempo total de simulação de 14400 s (4 horas), com um tempo de *warmup* de 600 s e 10 sementes aleatórias (*seeds*). A geração de mensagens na rede foi definida para todas as simulações com uma mensagem nova criada no intervalo de 25 a 35 segundos (randomicamente e a cada 30 segundos em média). Todos os nós tem a capacidade de gerar e receber mensagens e o TTL de mensagens de qualquer simulação ficou em 120 minutos (a metade do tempo total da simulação).

Foi definido para todos os casos de simulação a geração de um relatório do ONE chamado de MessageStatsReport, pois esse relatório contém os valores de métricas diversas, como mensagens criadas, mensagens entregues, latência média e sobrecarga de mensagens.

Parâmetro	Simulação 1	Simulação 2	Simulação 3	Simulação 4
Tipo de nó	Pedestres	Pedestres	Pedestres	Pedestres
Quantidade de nós	80	80	80	80
<i>Buffer</i> (Mb)	4	128	4	128
Tamanho das mensagens (Kb)	10	10	100	100
Raios de transmissão (m)	2,5,10,20,30,40,60,80	2,5,10,20,30,40,60,80	2,5,10,20,30,40,60,80	2,5,10,20,30,40,60,80
Velocidade de transmissão (Mb/s)	2	2	2	2
Modelo de movimento	SPMBM	SPMBM	SPMBM	SPMBM
Velocidade do nó -min a máx (m/s)	0,1 a 1,5	0,1 a 1,5	0,1 a 1,5	0,1 A 1,5

Tabela 4.2 Tabela da configuração dos nós em cada simulação.

## 5. AVALIAÇÃO DOS RESULTADOS OBTIDOS

Os dados obtidos das métricas de desempenho dos protocolos foram lidos do relatório gerados pelo ONE (MessageStatsReport) e transformados em informações pelos aplicativos em Java que desenvolvemos chamados LeReports e PreparaMatrix, descritos no Apêndice B. A partir dessas informações foram gerados os gráficos comparativos do capítulo corrente através do *script* do *software* R que também desenvolvemos chamado *plotaGraficoMatrix*, descrito no Apêndice C.

Para a plotagem dos gráficos, foi usada uma aproximação por uma distribuição de t-Student, com um nível de confiança de 95%, para os resultados obtidos das 10 rodadas para cada conjunto de parâmetros do cenário de simulação.

Com o objetivo de avaliar o desempenho dos protocolos, primeiramente apresentaremos esse desempenho sob a influência de uma baixa carga de dados na rede, na qual o tamanho das mensagens é de 10 KB. Depois apresentaremos a mesma avaliação de desempenho sob a influência de uma carga de dados maior, na qual as mensagens são de 100 KB, ou seja, 10 vezes o tamanho das mensagens anteriores.

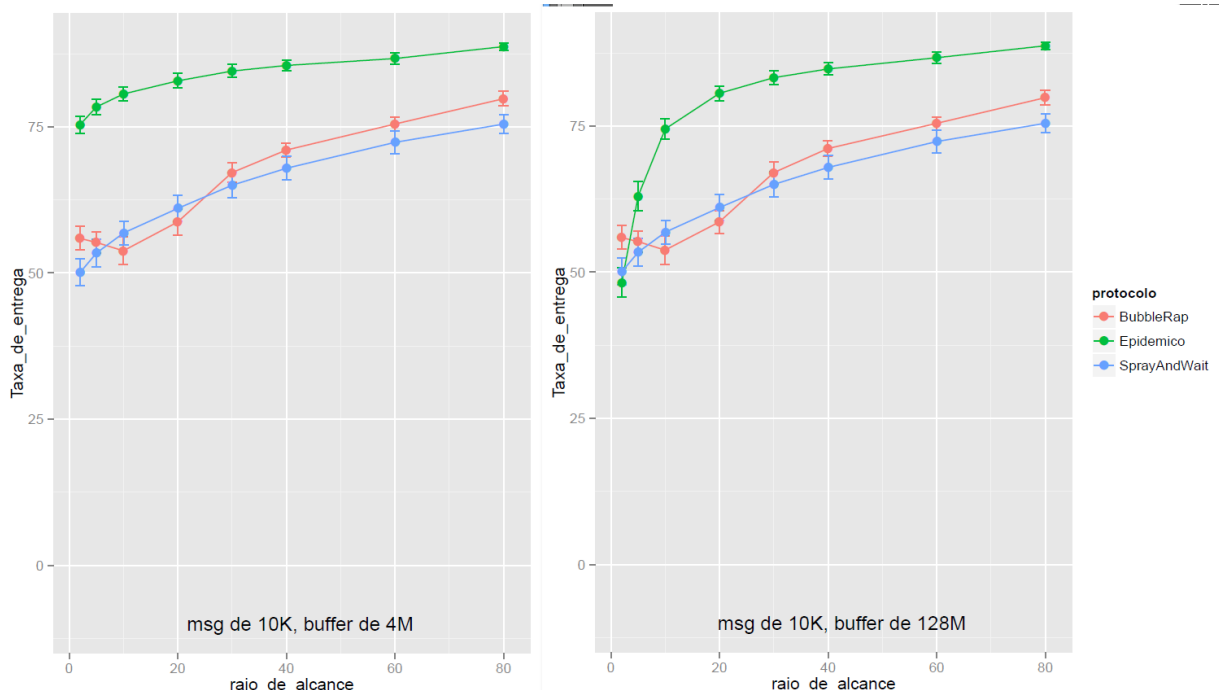
### 5.1. Cenários de baixa carga

Na Figura 6.1, o Epidêmico apresentou os melhores resultados dentre as taxas de entrega obtidas, por ser um protocolo baseado em inundação com múltiplas cópias, superando o Spray and Wait, que tem um número de cópias limitado ( $L$ ), em aproximadamente 30% para todos os raios de alcance acima de 5 m.

O BubbleRap, embora tenha começado com a taxa de entrega baixa, mostrou uma eficiência crescente para os raios de alcance do Bluetooth acima de 30 m e ultrapassou as taxas de entrega do Spray and Wait.

No cenário de baixa carga de dados, o aumento da capacidade de buffer de 4 para 128 Mbytes, além de não proporcionar melhora na taxa de entrega para os protocolos

analisados, ainda piorou a taxa de entrega do Epidêmico para os raios menores que 20 m.



a. buffer de 4 Mbytes.

b. buffer de 128 Mbytes.

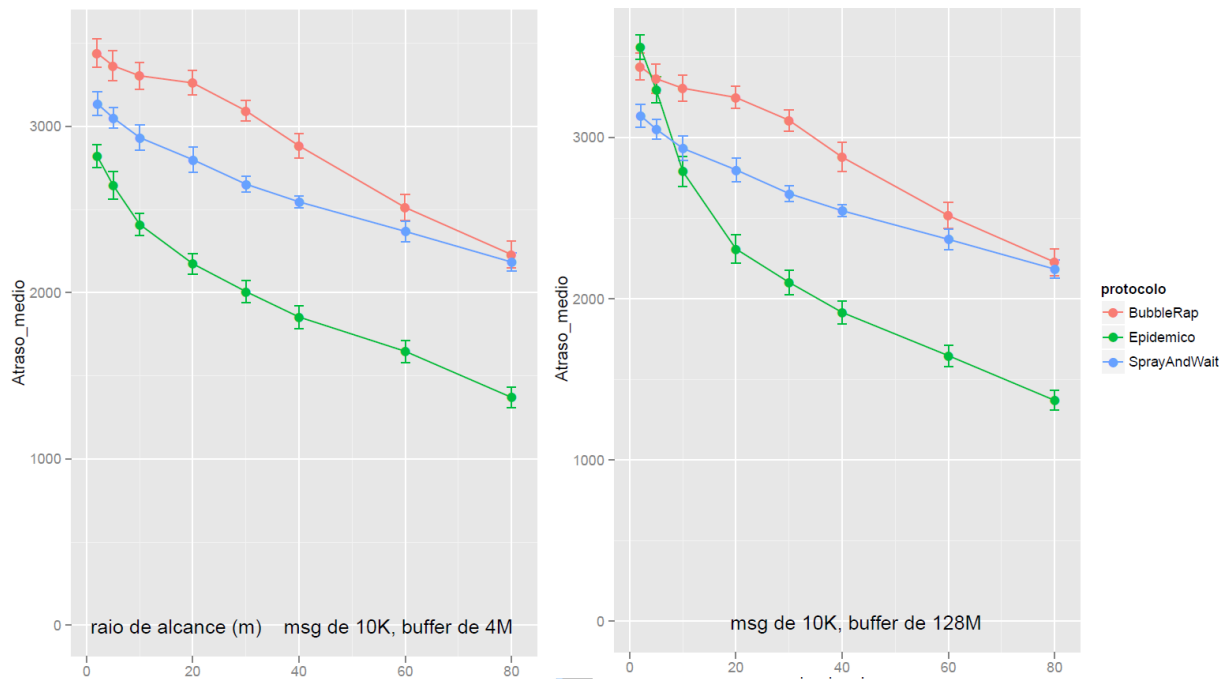
Figura 6.1 Análise da influência do *buffer* na taxa de entrega com variação do raio de alcance dos nós com baixa carga.

Na Figura 6.2, para todos os três protocolos, os resultados mostraram que quanto maior fica o raio de alcance do dispositivo de Bluetooth, mais decresce o valor do atraso médio. Isso pode ser explicado pelo fato de que, quando se aumenta o raio de alcance, a rede se torna mais conectada, facilitando a entrega das mensagens.

E nessa mesma figura, pode ser observado que o protocolo Epidêmico gerou os menores atrasos médios na entrega de mensagens entre os três protocolos analisados. Em

segundo lugar está o Spray and Wait, no qual o atraso médio das mensagens também é decrescente, mas não em grau tão acentuado quanto o Epidêmico. Por fim, os maiores atrasos médios foram gerados pelo BubbleRap, se recuperando somente para os raios maiores de 60m, quando ficou com valores próximos aos do Spray and Wait.

Deve-se notar que, para esse cenário de baixa carga, o Epidêmico sofreu influência negativa do aumento do tamanho do *buffer*, aumentando o atraso médio para os raios menores que 30 m.



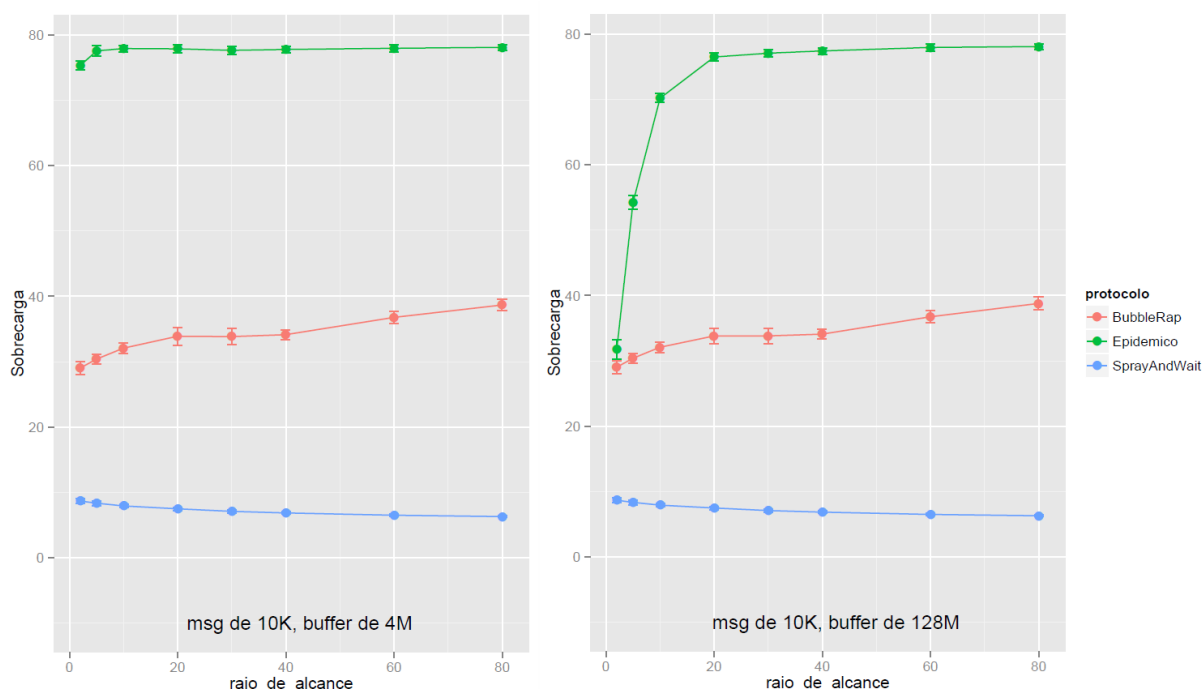
a . buffer de 4 Mbytes.

b . buffer de 128 Mbytes.

Figura 6.2 Análise da influência do *buffer* no atraso médio com variação do raio de alcance dos nós com baixa carga.

Na Figura 6.3, o protocolo Spray and Wait apresentou uma sobrecarga de oito mensagens inicialmente, chegando a seis para o maior raio de alcance,. O BubbleRap iniciou com sobrecarga de 30 mensagens e ficou com valores de sobrecarga média de 35 mensagens. Já o Epidêmico extrapolou, chegando a quase 80 mensagens, a não ser em raios menores que 20m em que a sobrecarga diminuiu.

Pela análise dos gráficos comparativos das métricas em geral, o aumento dos *buffers* de 4 Mbytes para 128 Mbytes, para simulações de baixa carga de dados, não proporcionou melhora na performance, a não ser nos raios menores que 20m e no protocolo Epidêmico.



a . buffer de 4 Mbytes.

b . buffer de 128 Mbytes.

Figura 6.3 Análise da influência do *buffer* na sobrecarga com variação do raio de alcance dos nós com baixa carga.



Podemos observar que, para os cenários de baixa carga de dados, o *buffer* de 128 Mbytes teve um comportamento similar ao de um *buffer* infinito, caso em que não haveria a perda de mensagens por transbordo do *buffer*; não sendo portanto, recriadas mensagens por motivo de descarte.

## 5.2. Cenários de alta carga

Para contornar qualquer possibilidade de limitação não programada pelo algoritmo, procuramos delimitar o raio de alcance do Bluetooth das simulações em 80 m, pois um raio de alcance acima desse valor não seria realista, se considerarmos o local de deslocamento dos pedestres no Centro do Rio e as interferências do próprio ambiente.

Acima de 80 m de raio de alcance dos nós, haveria um crescimento muito grande da ocorrência de contatos com vários outros nós, aumentando a carga na rede muito além da ocorrida nos raios de alcance menores.

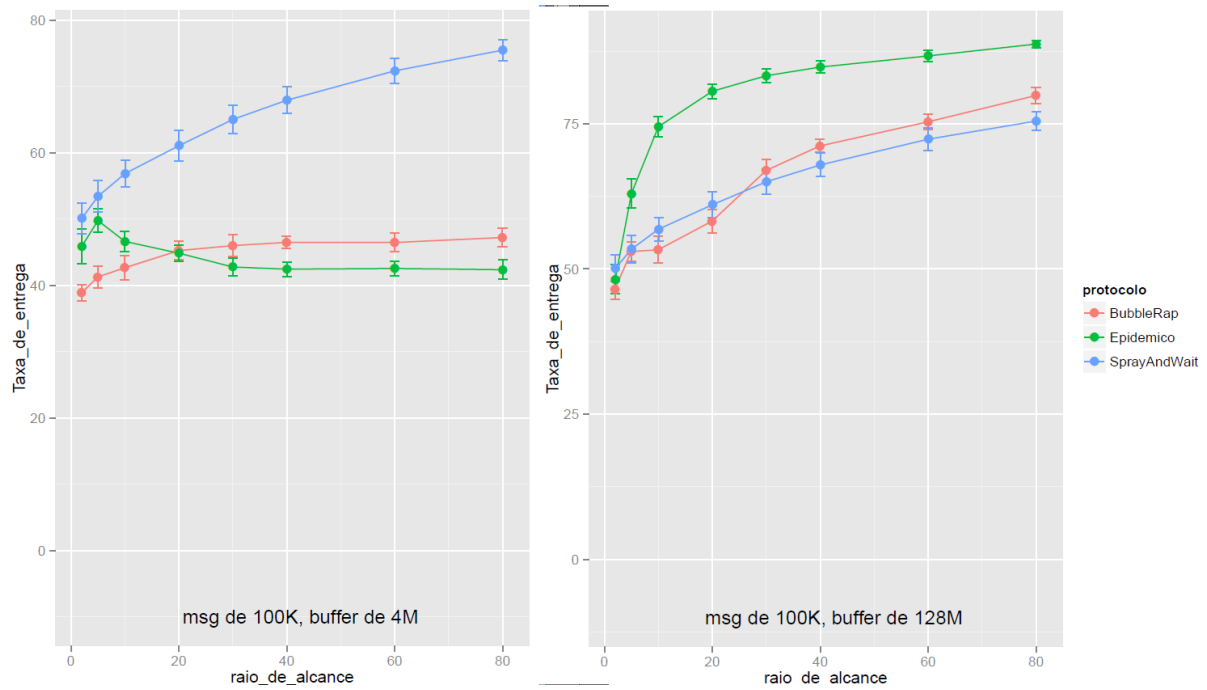
Nos gráficos da Figura 6.4, obtidos das simulações de alta carga (com mensagens de 100 Kbytes) notamos uma grande mudança de comportamento da taxa de entrega dos três protocolos, quando o *buffer* varia de 4 para 128 Mbytes.

O Spray and Wait apresenta uma taxa de entrega crescente conforme aumentamos o raio de alcance, chegando a ultrapassar os valores do Epidêmico e do BubbleRap, para o *buffer* de 4 Mbytes.

O Epidêmico se estabiliza em aproximadamente 42% para o *buffer* de 4 Mbytes, devido ao enchimento do *buffer*, que se tornou pequeno para a alta carga de mensagens. Já para o *buffer* de 128 Mbytes o Epidêmico atinge as melhores taxas de entrega para todos os raios de alcance conforme era esperado.

Enquanto isso, o BubbleRap mudou o seu comportamento com o aumento do *buffer*. Para o *buffer* de 4 Mbytes e para os raios de alcance acima de 20 m, apresentou taxas de entrega superiores ao Epidêmico, chegando a ter valores de quase 50 % (vide

Figura 6.4 a). Já para o *buffer* de 128 Mbytes, o mesmo BubbleRap superou o Spray and Wait, para raios de alcance acima de 30 m (Figura 6.4 b).

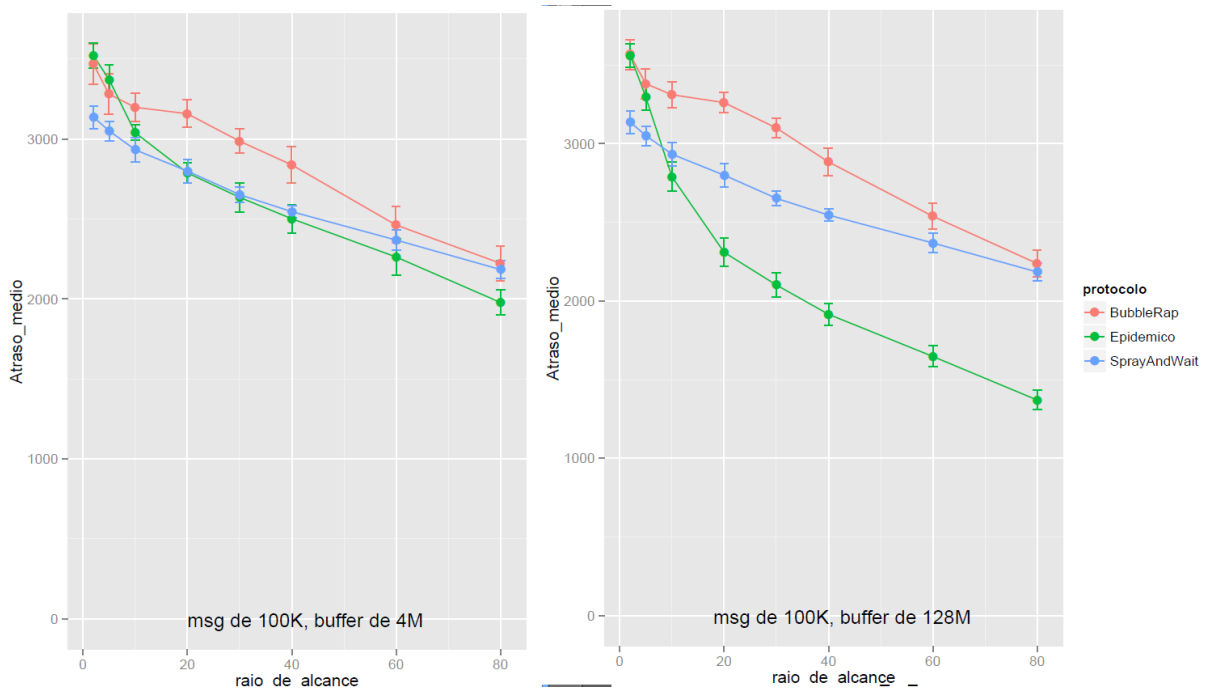


a . buffer de 4 Mbytes.

b . buffer de 128 Mbytes.

Figura 6.4 Análise da influência do *buffer* na taxa de entrega com variação do raio de alcance dos nós com alta carga.

Na Figura 6.5, o protocolo BubbleRap apresenta um atraso médio próximo aos outros dois protocolos para o *buffer* de 4 Mbytes, no entanto, o Epidêmico sofre uma otimização quando há mais espaço em *buffer* (*buffer* de 128 Mbytes), mostrando atrasos médios bem menores, como vemos nas Figuras 6.5 (a) e (b).



a . buffer de 4 Mbytes.

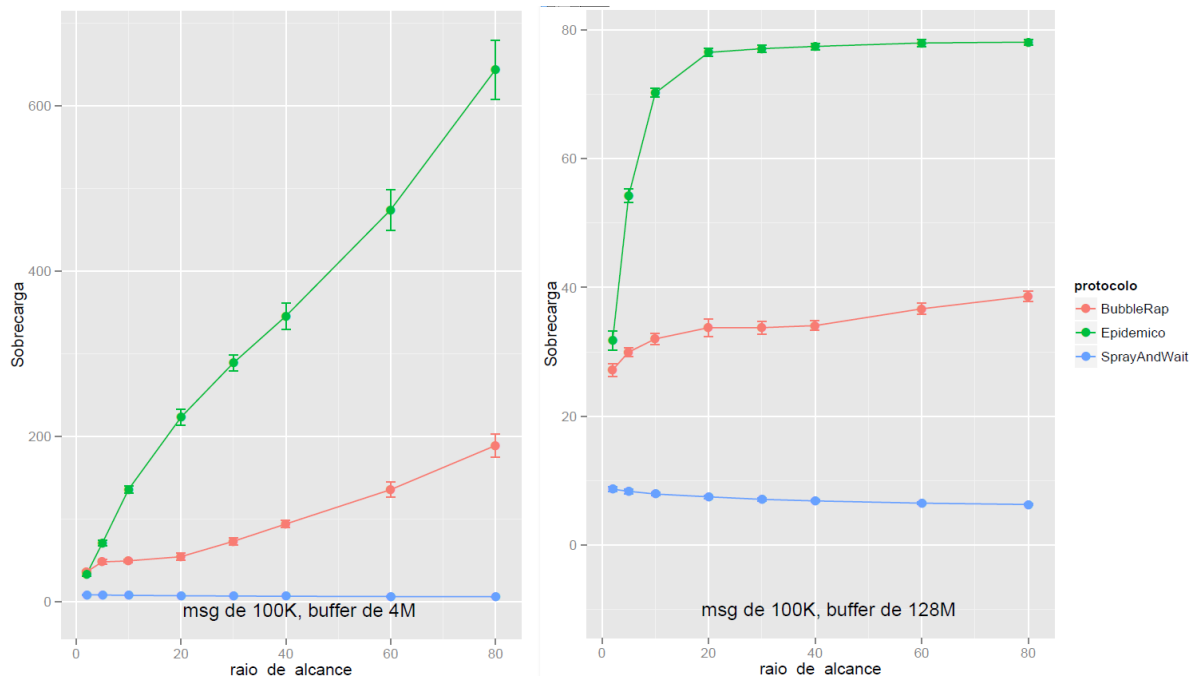
b . buffer de 128 Mbytes.

Figura 6.5 Análise da influência do *buffer* no atraso médio com variação do raio de alcance dos nós com alta carga.

A variação do tamanho de *buffer* provoca mudanças significativas para a sobrecarga de mensagens no protocolo Epidêmico. Com o *buffer* menor (de 4 Mbytes), ele apresenta uma sobrecarga linearmente crescente com o aumento do raio de alcance, como pode ser visto na Figura 6.6 (a). Sua característica de replicação implica em que, mesmo em situações de enchimento de *buffer*, as mensagens geradas e as suas cópias enviadas para os nós que se encontram no raio de alcance do dispositivo ultrapassam em muito o número de mensagens descartadas, provocando uma sobrecarga em toda a rede, porque provavelmente sejam originadas mais mensagens para substituir as que foram descartadas. Nesse cenário, o BubbleRap fica com uma sobrecarga bem baixa (100 mensagens em média) e a sobrecarga menor é a do Spray and Wait, conforme o

esperado, pois limita o número de cópias.

Já para o *buffer* de 128 Mbytes Figura 6.6 (b), o Epidêmico se estabiliza com valor de sobrecarga razoável, mas, ainda assim, a sobrecarga é aproximadamente 100% a mais do que o protocolo BubbleRap e 1000% (aproximadamente) a mais em relação ao Spray and Wait, que apresentou em todos os resultados sobrecarga próxima de seis, conforme o esperado, pela definição do parâmetro L, que limitou as cópias de mensagens em 6.



a . buffer de 4 Mbytes.

b . buffer de 128 Mbytes.

Figura 6.6 Análise da influência do *buffer* na sobrecarga com variação do raio de alcance dos nós com alta carga.

## 6. CONCLUSÃO E TRABALHOS FUTUROS

Notamos claramente que o protocolo Spray and Wait é o único que não apresenta mudança no comportamento das métricas de desempenho com a variação do tamanho de *buffer*, nem mesmo quando aumentamos de baixa carga para alta carga de dados. Assim, para os cenários de simulação analisados aqui, não se pode afirmar que houve mudanças no desempenho desse protocolo em virtude do aumento do *buffer*.

Nos cenários de baixa carga, o protocolo BubbleRap também não sofreu alterações no seu comportamento em função da variação do tamanho do *buffer*. Já o protocolo Epidêmico sofreu uma certa piora nas métricas para os raios de alcance menores que 20 m.

Para os cenários de alta carga e com pouco espaço de sobra nos *buffers* dos nós (com *buffer* de 4 Mbytes), o protocolo Epidêmico sobrecarregou a rede com um grande volume de mensagens, tornando-se então muito desvantajoso sob o ponto de vista dessa métrica (sobrecarga), em relação aos outros dois protocolos. Isso pode ser explicado pelas próprias características do seu algoritmo de encaminhamento, baseado em inundação com múltiplas cópias, além do fato que seu algoritmo mantém as cópias das mensagens na rede, mesmo depois de serem entregues.

Para os cenários de alta carga, mas com espaço razoável nos *buffers* dos nós (com *buffer* de 128 Mbytes), destacamos que o BubbleRap se estabilizou com uma média de 35 mensagens de sobrecarga na rede, mostrando ser muito mais vantajoso do que o Epidêmico, com cerca de 80 mensagens, só perdendo para o Spray and Wait, por ser este último um protocolo que limita o número de cópias a serem encaminhadas, apresentando sempre sobrecarga baixíssima, com valores próximos a seis mensagens (o mesmo valor definido para o fator limitador  $L$ ) em todas as simulações, mesmo nos cenários de baixa carga.

Por fim, o BubbleRap mostrou ser, sem dúvida nenhuma, um protocolo que gera pouca sobrecarga na rede, com taxa de entrega mais próxima do Epidêmico do que o

Spray and Wait, apesar de que o atraso médio gerado e visto nesses cenários não seja o seu ponto forte.

Talvez o cenário utilizado nas simulações não tenham sido tão adequados ao uso do protocolo BubbleRap que possui seus algoritmos baseados em métricas sociais. Poderiam ter sido utilizados mais dados de redes sociais, como os do FaceBook, e traces reais (do projeto CRAWDAD) que podem ser explorados para decisões melhores de encaminhamento, pois assim seriam fornecidas informações de comunidades e de popularidade de um nó dentro e fora dessas comunidades.

## 6.1. Trabalhos futuros

Até aqui, nesse trabalho, testamos e avaliamos o desempenho de protocolos que utilizam a replicação, com o cuidado de não sobrecarregar a rede excessivamente, ou mesmo os que utilizam métricas sociais para melhorar a eficiência no encaminhamento de mensagens.

Dentro dessa mesma linha de pesquisa, podemos vislumbrar a evolução de muitos tipos de protocolos de encaminhamento em redes DTN, porém, mais especificamente, poderiam ser desenvolvidos futuramente testes de novos protocolos sempre buscando indícios que confirmem e comprovem a influência das métricas sociais na melhoria do desempenho dessas redes DTN.

Com a evolução dos dispositivos portáteis com acesso às redes sociais e paralelamente com o aumento da capacidade dos *buffers* dedicados aos protocolos, deve-se considerar em trabalhos futuros a utilização das métricas sociais como um dos fatores determinantes para o encaminhamento de mensagens (similares ao que o protocolo estudado BubbleRap se propõe a usar).

Podemos destacar primeiramente a comparação de protocolos em função das métricas sociais, em segundo lugar, a análise da parametrização dos protocolos testados, como o parâmetro K do algoritmo KcliqueCommunityDetection do BubbleRap, ou o

parâmetro  $L$ , que limita o número de cópias a serem repassadas no protocolo Spray and Wait.

Este trabalho nos deixa uma reflexão sobre como os padrões de comunicação social, que venham representados por um Grafo Social, podem afetar o comportamento das redes DTN oportunísticas móveis e como essas informações sociais e de mobilidade (nosso enfoque nesse trabalho é em movimentos de pedestres ou de pessoas) podem ser exploradas para melhorar a performance nas mesmas redes.

Podem ser usados, em futuras simulações, *traces* de mobilidade reais junto com seus respectivos dados de relacionamento social, como comunidades, interesses sociais, popularidade dos nós e grafos sociais. Uma opção a ser explorada seriam as simulações com dados de experimentos reais, selecionados entre os dados de redes sem fio disponíveis no projeto CRAWDAD [17], para estudar características individuais de padrões de mobilidade humana ou mesmo os padrões de seus contatos.

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] A. Mitibaa, M. May, M. Ammar, C. Diot, “Peoplerank , combining social and contact information for opportunistic forwarding”, World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE International Symposium, San Francisco CA, Pages 1-5, 2012.
- [2] P. Hui, J. Crowcroft, E. Yoneki, “BUBBLE Rap, Social Based Forwarding in Delay Tolerant Networks”, MobiHoc '08, Hong Kong, SAR, China, Pages 241-250, 2008.
- [3] P.Dillon, “Software Contributions to the ONE Simulator”, disponível em: <https://github.com/knightcode/the-one-pitt> , acessado em 06/2013.
- [4] A.Lindgren, A.Doria, and O.Schelén, “Probabilistic Routing in intermittently connected networks”, SIGMOBILE Mobile Comput. Comm. Rev., vol. 7, no. 3, pp. 19-20, 2003.
- [5] Marsden P, Egocentric and sociocentric measures of network centrality. Social Networks 24, 407-422, 2002.
- [6] Gergely Palla, Imre Derényi, Illés Farkas, Tamás Vicsek, “Uncovering overlapping community structure of complex networks in nature and society”, Nature, Vol: 435, Pages 814-818, 2005.
- [7] Theus Hossman, Thrassyvoulos Spyropoulos, Frank Legendre, “Know Thy Neighbor: Towards Optimal Mapping of Contacts to Social Graphs for DTN Routing”, IEEE INFOCOM, Pages 1-9, 2010.
- [8] Paolo Santi, MOBILITY MODELS FOR NEXT GENERATION WIRELESS NETWORKS/ AdHoc Vehicular and Mesh Networks, John Willey & Sons – ISBN:



9781119992011.

[9] A. Keränen and J.Ott, “The ONE - the opportunistic network environment simulator” disponível em: <http://www.netlab.tkk.fi/tutkimus/dtn/theone/>, acessado em 06/2013.

[10] Bluetooth, disponível em: <http://pt.wikipedia.org/wiki/Bluetooth>, acessado em 06/2013.

[11] Projeto OpenStreetMap, disponível em: <http://www.openstreetmap.org>, acessado em 06/2013.

[12] NetBeans 7.2.1, disponível em: <https://netbeans.org/>, acessado em 06/2013.

[13] *Software* R, disponível em: <http://www.r-project.org/>, acessado em 06/2013.

[14] Thrasyvoulos Spyropoulos, Konstantinos Psounis, Cauligi S. Raghavendra, “Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks”, *SIGCOMM'05, Philadelphia, PA, USA, Pages 252-259, 2005*.

[15] Amin Vahdat and David Becker, “Epidemic Routing for Partially-Connected Ad Hoc Networks”, Department of Computer Science, Duke University, Durham, UK, pages 1-14, 2000.

[16] A ferramenta Osm2wkt, disponível em: <http://www.tm.kit.edu/~mayer/osm2wkt/>, acessada em 06/2013.

[17] Team C 2011, disponível em: <http://crawdad.cs.dartmouth.edu/index.php>, acessada em 06/2013.

- [18] Watts D and Strogatz S, “Collective Dynamics of Small-World networks”, *Nature*, Pages 440- 442, 1998.
- [19] Panagiotis Pantazopoulos, Merkourios Karaliopoulos, and Ioannis Stavrakakis, “On the Local Approximations of Node Centrality in Internet Router-level Topologies“, 7th International Workshop on Self-Organizing Systems (IFIP IWSOS'13), Palma de Mallorca, Spain, pages 1-12, 2013.
- [20] Daly and Haar M, “Social network analysis for routing in disconnected delay tolerant MANETs”. *Proceedings of ACM MobiHoc*, Quebec, Montreal, Canada, pp 32-40, 2007.
- [21] Classes de Bluetooth, disponível em:  
[http://www.bluegiga.com/WT41\\_Long\\_Range\\_Bluetooth\\_Module](http://www.bluegiga.com/WT41_Long_Range_Bluetooth_Module), acessado em 06/2013.
- [22] P.Hui, A. Chaintreau, , et. al., “Pocket switched networks and human mobility in conference environments”. In *Proc. WDTN '05, SIGCOMM*, Philadelphia, PA, USA, Pages 244-251, 2005.
- [23] J. Burgess, B.Gallagher, D.J. and Levine, B.N. MaxProp: Routing for vehicle-based disruption-tolerant networks. In *Proc. Of the IEEE INFOCOM*, pages, 1-11,2006.
- [24] Athanasios Vasilakos, Yan Zhang, Thrasyvoulus V. Spyropoulos, *Delay Tolerant Networks – Protocols and Applications*, 2012, CRC Press, ISBN: 978-1-4398-1108-5.
- [25] Kevin Fall , “A Delay Tolerant Netowrok Architecture for Challenged Internets”, *SIGCOMM '03*, Karlsruhe, Germany, pages 27-34, 2003.
- [26] James F. Kurose, Keith W. Ross, *Redes de computadores e a Internet*, 3a. Edição.

[27] Algoritmo Dijkstra, disponível em: [http://en.wikipedia.org/wiki/Dijkstra's\\_algorithm](http://en.wikipedia.org/wiki/Dijkstra's_algorithm), acessado em 05/2013.

[28] A. Mitibaa, M. May, C. Diot, M. Ammar. “PeopleRank: Social Opportunistic Forwarding”, IEEE INFOCOM, San Diego, CA, USA, pages 1-5, 2010

[29] S. Brin and L. Page. “The anatomy of a large-scale hypertextual web search engine”. Comput. Netw. Syst. 30(1-7): Elsevier, Pages 107-117, 1998.

[30] Thrasyvoulos Spyropoulos, Konstantinos Psounis, Cauligi S. Raghavendra, “Spray and Focus: Efficient Mobility – Assisted Routing for Heterogeneous and Correlated Mobility”, IEEE PerCom WorkShops, White Plains, NY, USA, , Pages 79-85, 2007.

[31] Steven H. Strogatz, “Exploring complex networks”, Nature, pages 268-276, 2001.

[32] Tadeu L. Villares, Carlos Alberto V. Campos, Aline C. Viana, “A Influência de Nós Especiais na Entrega de Mensagens nas Redes Tolerantes a Atrasos e Interrupções”, WRA'13, DF, Brasil, pages 44-57, 2013.

[33] VANETs, disponível em: [http://bbcr.uwaterloo.ca/~htcheng/Current\\_Projects.html](http://bbcr.uwaterloo.ca/~htcheng/Current_Projects.html) , acessado em 06/2013

[34] Bluetooth, disponível em: <http://ugfredes.files.wordpress.com/2011/05/bluetooth.jpg> , acessado em 06/2013

[35] MANETs , disponível em: <http://www.mdpi.com/1424-8220/11/4/3652> , acessado em 06/2013.

[36] WiFi, disponível em: <http://hanns.no.comunidades.net/index.php?pagina=1613126523> , acessado em 06/2013.

[37] Sensores, disponível em: [http://www.vivasemfio.com/blog/category/1\\_tecnologias/r\\_z\\_tecnologias/wireless\\_sensor\\_network](http://www.vivasemfio.com/blog/category/1_tecnologias/r_z_tecnologias/wireless_sensor_network), acessado em 06/2013.

[38] Sensores, disponível em: <http://www.inetsis.es/br/products.html>, acessado em 06/2013.

[39] Carina T. de Oliveira, Marcelo D. D. Moreira, Marcelo G. Rubinstein, Luís Henrique M. K. Costa e Otto Carlos M. B. Duarte, “Redes Tolerantes a Atrasos e Desconexões”, SBRC 2007, pages 233-256, 2007.

[40] Jörg Ott, Dirk Kutscher; Applying DTN to Mobile Internet Access: An Experiment with HTTP; Technical Report TR-TZI-050701, Technologiezentrum Informatik, Universität Bremen; July 2005.

[41] Ott, J. e Kutscher, D. , Exploiting regular hot-spots for drive-thru Internet. Em Kommunikation in Verteilten Systemen (KiVS), ISBN 978-3-540-24473-8, p. 218–229, 2005.

[42] Redes PSN, disponível em: <http://www.shigs.co.uk/static/presentations/wirelessdays2012/index.html#/step-5> , acessado em 06/2013.

[43] Vinton Cerf, disponível em : [http://www.livinginternet.com/i/ii\\_cerf.htm](http://www.livinginternet.com/i/ii_cerf.htm) , acessado em 06/2013.

[44] Interplanetary Internet, disponível em: <http://ipnsig.org/> , acessado em 06/2013.

[45] Clique, disponível em : [http://www.dimap.ufrn.br/~dario/arquivos/Cap2\\_Grafos-2001.pdf](http://www.dimap.ufrn.br/~dario/arquivos/Cap2_Grafos-2001.pdf) , acessado em 06/2013.

## Apêndice A

Pacotes desenvolvidos nesse trabalho para as simulações dos protocolos no ONE.

`simscript_NomeDoProtocolo` (.bat e .sh): esses *scripts* de comandos (em CLI) do Windows e do Unix executam as simulações dos 3 protocolos no ONE, a saber, `simscript_DistributedBubbleRap`, `simscript_Epidemic`, `simscript_SprayAndWait`.

## **Apêndice B**

Pacotes desenvolvidos nesse trabalho para a interface de dados entre o ONE e a ferramenta R.

LeReports: programa desenvolvido na linguagem Java, com aproximadamente 200 linhas de código, para a leitura de dados de saída dos Reports do ONE que faz a interface com o aplicativo PreparaMatrix1, gerando arquivos de texto para esse propósito.

PreparaMatrix1: programa desenvolvido na linguagem Java, com aproximadamente 100 linhas de código, para o tratamento dos dados contidos nos arquivos de texto gerados pelo aplicativo LeReports e preparando o arquivo de entrada no formato próprio da ferramenta R, ou seja, arquivos que contém as matrizes de pontos a serem plotadas.

## Apêndice C

Pacote desenvolvido nesse trabalho para a geração de gráficos na ferramenta R.

`plotaGraficoMatrix`: *script* de comandos do *software* R desenvolvido, com aproximadamente 100 linhas de código, para geração de gráficos comparativos das diversas métricas de desempenho da rede DTN.