COPPE/UFRJ

ESCASSEZ DE RECURSOS EM REDES TOLERANTES A ATRASOS E INTERRUPÇÕES

Raphael Melo Guedes

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: José Ferreira de Rezende

Rio de Janeiro

Dezembro de 2009

ESCASSEZ DE RECURSOS EM REDES TOLERANTES A ATRASOS E INTERRUPÇÕES

Raphael Melo Guedes

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. José Ferreira de Rezende, Dr.

Prof. Célio Vinicius Neves de Albuquerque, Ph.D.

Prof. Aloysio de Castro Pinto Pedroza, Dr.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2009

Guedes, Raphael Melo

Escassez de Recursos em Redes Tolerantes a Atrasos e Interrupções/Raphael Melo Guedes. – Rio de Janeiro: UFRJ/COPPE, 2009.

XIV, 98 p.: il.; 29, 7cm.

Orientador: José Ferreira de Rezende

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2009.

Referências Bibliográficas: p. 90 – 98.

1. Delay Tolerant Networks. 2. Escassez de recursos. 3. Protocolos de Roteamento em DTNs. I. Rezende, José Ferreira de. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Dedico este trabalho

à minha família.

Agradecimentos

Agradeço aos meus pais Antônio e Lúcia, minhas irmãs Ana e Camile, ao Sebastião e a meu sobrinho Bruno pelo apoio.
Aos colegas do GTA: André, Carina, Celso, Chenrique, Danilo, Kleber, Marcel, Natália, Reinaldo, Tibério, e todos os demais; pela ajuda, paciência e amizade.
Aos colegas de faculdade: Daniel, Diego e Felipe que me ajudaram muito.
Aos profs. do GTA em especial ao professor José Rezende pela orientação e ajuda.
E claro agradeço ao Criador.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ESCASSEZ DE RECURSOS EM REDES TOLERANTES A ATRASOS E INTERRUPÇÕES

Raphael Melo Guedes

Dezembro/2009

Orientador: José Ferreira de Rezende

Programa: Engenharia Elétrica

As redes tolerantes a atrasos e interrupções (RTAIs) têm se tornado cada vez mais importantes no âmbito das pesquisas em redes avançadas. Estas redes possuem como principal característica a robustez à falta de conectividade fim-a-fim. Para isso, as soluções desenvolvidas realizam o armazenamento intermediário e o encaminhamento oportunista das mensagens geradas. Entretanto, por serem voltadas para cenários com dispositivos móveis, um dos problemas que podem afetar o seu desempenho é a restrição de recursos como energia, taxa de transmissão e capacidade de armazenamento. Este trabalho propõe alguns modelos de economia de energia, tais como políticas ou modos de gerenciamento de energia que trabalham sobre os períodos de funcionamento dos nós, com o objetivo de aumentar o tempo operacional da rede. Adicionalmente, propõe um tipo de encaminhamento que determina o número de mensagens a serem transmitidas a partir do histórico de encontros dos nós, diminuindo o número de mensagens injetadas na rede, contudo mantendo uma taxa de entrega satisfatória.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

RESOURCE CONSTRAINT IN DELAY/DISRUPTION/DISCONNECT TOLERANT NETWORKS

Raphael Melo Guedes

December/2009

Advisor: José Ferreira de Rezende

Department: Electrical Engineering

Delay/Disruption/Disconnect Tolerant Networks (DTNs) have become increasingly important in the context of advanced networks research. These networks have as main important feature the robustness against the lack of an end-to-end connection. Thus, existing solutions make use of intermediate storage and opportunistic forwarding to deliver end-to-end messages. However, since they are mainly focused on scenarios with mobile devices, one of the problems that can affect their performance is resource constraint such as energy, transmission rate and storage space.This work proposes some models of energy saving such as policies or power management. These power managements work on periods of operation of the nodes, in order to increase the uptime of the network. Additionally proposes a type of forwarding that determinate the number of messages to be transmitted from the history of encounters, reducing the number of messages injected in the network and maintaining a satisfactory delivery rate.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ADUs	*Application Data Units*, p. 11
CHANTS	*CHAllenged NeTworkS*, p. 1
DD	*Destination-Dependent*, p. 19
DI	*Destination-Independent*, p. 19
DTNRG	*Delay Tolerant Network Research Group*, p. 1
DTN	*Delay/Disruption/Disconnect Tolerant Networks*, p. 1
EDAQ	*Earliest Delivery with All Queues*, p. 25
EDLQ	*Earliest Delivery with Local Queuing*, p. 25
ED	*Earliest Delivery*, p. 25
ICN	*Intermittently Connected Networks*, p. 1
IRTF	*Internet Research Task Force*, p. 1
LEFO	*LEss FOrwarded*, p. 48
LHOC	*Less HOp Count*, p. 48
MED	*Minimum Expected Delay*, p. 20
MEED	*Minimum Estimated Expected Delay*, p. 20
MOFO	*Evict most forwarded first*, p. 28
MULE	*Mobile Ubiquitous LAN Extensions*, p. 7
NS-2	*Network Simulator*, p. 32

PDUs *Protocol Data Units*, p. 12

PROPHET *Probabilistic Routing Protocol using History of Encounters and transitivity*, p. 19

PSN *Pocket Switched Networks*, p. 14

RTAIs *Redes Tolerantes a Atrasos e Interrupções*, p. 1

SNC *Sámi Network Connectivity*, p. 8

SWIN *Shared Wireless Infostation Model*, p. 27

SeNDT *Sensor Networking with Delay Tolerance*, p. 7

TIER *Technology and Infrastructure for Emerging Regions*, p. 8

Wi-Fi *Wireless Fidelity*, p. 9

Capítulo 1

Introdução

1.1 Motivação

Uma premissa básica para o bom funcionamento de redes "convencionais" [1], como as redes cabeadas e até mesmo as redes *ad hoc*, é a existência de uma conectividade fim-a-fim entre os terminais que desejam se comunicar. Na maioria dos protocolos de roteamento, o sucesso na descoberta de uma rota e, portanto, o início de uma comunicação, só é possível caso exista um caminho entre a fonte e o destino. Entretanto, existem ambientes denominados "desafiadores" onde nem sempre é possível garantir essa conectividade fim-a-fim, ou por ela nunca existir ou por ser intermitente.

Esta dificuldade tem estimulado grupos de pesquisa na procura de soluções para este problema, como o DTNRG (*Delay Tolerant Network Research Group*) [2] criado pelo IRTF (*Internet Research Task Force*) [3]. A essas redes, que levam em consideração longos atrasos e frequentes desconexões, dá-se o nome de DTN (*Delay/Disruption/Disconnect Tolerant Networks*) que mais recentemente foram denominadas de várias formas, como: Redes Oportunistas (*Opportunistic Networks*) [4], Redes com Desafios (CHANTS) *CHAllenged NeTworkS* [5], Redes com Conectividade Eventual (*Episodically-Connected Networks*), Redes Intermitentemente Conectadas (ICN - *Intermittently Connected Networks*) [6] ou ainda Redes com Desafios de Conectividade (*Connectivity-Challenged Networks*). Ao longo deste trabalho, será utilizado o termo em português RTAIs para Redes Tolerantes a Atrasos e Interrupções.

As principais características das RTAIs estão relacionadas às desconexões e aos

longos atrasos na entrega de mensagens. Estes atrasos podem chegar a ordem de horas, até mesmo dias. Com relação às desconexões, estas podem ocorrer pela alta mobilidade dos nós, que provoca constantes mudanças na topologia da rede; por péssimas condições de comunicação; por economia de recursos, como em sensores sem fio que "dormem" para poupar energia, etc.. Estes eventos podem resultar em uma conectividade intermitente da rede durante um período, ou ainda, pode ser que um caminho entre a origem e o destino nunca chegue a ficar completamente estabelecido [7].

Ainda que uma conectividade fim-a-fim, em uma RTAI, seja rara de ocorrer entre dois nós (fonte e destino), os protocolos de roteamento destas redes, tiram vantagem de enlaces temporais casuais. Ou seja, os protocolos aproveitam-se de enlaces criados pelo encontro do nó com algum vizinho. Deste modo, os nós, após trocarem alguma informação referente ao funcionamento do protocolo utilizado, repassam a mensagem que tenham acordado em transmitir. Explora-se a mobilidade dos próprios nós na tentativa de entregar uma mensagem ao nó de destino [1]. Logo, quando deseja-se enviar uma mensagem em uma RTAI, esta é armazenada em *buffer* e encaminhada nó a nó desde a origem até o destino, de acordo com os enlaces disponíveis. Por esta razão, os protocolos de roteamento em RTAIs pertencem a uma classe chamada *store-carry-and-foward*.

Quando dois nós entram em contato, o desejável é que cada um deles obtenha mensagens novas. Entretanto, mesmo que os nós possuam *buffer* suficiente, o tempo desse contato pode ser curto com relação ao tempo despendido na transmissão de cada mensagem, tornando-se impossível a troca de todas as mensagens desejadas. Desta forma, a difusão das mensagens pela rede fica prejudicada, consequentemente o nó é obrigado a permanecer mais tempo com a mensagem armazenada e, este procedimento pode conduzir à saturação do espaço de armazenamento e finalmente ao descarte. Assim, o correto funcionamento da rede depende da capacidade de armazenamento dos seus nós, da capacidade de transferência de dados durante o tempo de contatos entre eles e da cooperação entre os componentes [8, 9].

O gasto de energia também pode representar um problema em RTAIs. Como a principal característica dessas redes é a mobilidade dos seus nós, supõe-se que tais nós sejam dispositivos restritos em energia. Logo é muito importante evitar desperdícios

com transmissões desnecessárias, pois quando a energia de um nó acaba, todas as mensagens em seu *buffer* são perdidas.

Um vasto número de protocolos de roteamento foram e são propostos para contornar dificuldades nestes tipos de ambientes, considerado hostil. Dentre estas propostas, a grande meta é alcançar uma taxa de entrega satisfatória. Porém, outra grande preocupação é a quantidade de sobrecarga (*overhead*[1]) injetada na rede. O número de cópias de cada mensagem é vinculado ao gasto de transmissão e de ocupação do *buffer* dos nós componentes, além de muito relacionado à taxa de entrega, efeito direto do seu funcionamento.

Como não existe garantia que um enlace, uma vez estabelecido, permaneça disponível por um período longo de tempo, uma ideia intuitiva é tentar aproveitar a oportunidade de contato e transmitir o máximo de mensagens possíveis entre os pares de comunicação. Este procedimento é explorado por uma classe de protocolos de roteamentos denominada replicadores[2]. Pois é esperado que, com o aumento do número de cópias de uma determinada mensagem, aumentem-se as chances de alguma alcançar seu nó de destino. Entretanto, tal procedimento reflete-se em um grande uso de recursos, resultando em um rápido preenchimento do *buffer*, principalmente dos nós intermediários, e um elevado gasto energético dos nós. Como consequência, podemos ter uma queda de desempenho da rede, pois o tempo de vida dos dispositivos tende a se extinguir mais rapidamente.

Grande parte das aplicações em RTAIs são voltadas para equipamentos móveis onde recursos como energia e espaço de armazenamento são relevantes, como por exemplo aplicações em redes de sensores. Com isso, observa-se que a abordagem, de replicar um grande volume de mensagens em ambientes com escassez de recursos não é o mais indicado. Devido a isso, uma categoria de protocolos chamados encaminhadores foram se desenvolvendo. Nesta categoria, o protocolo de roteamento aproveita-se da coleta de alguma informação que lhe proporcione um melhor encaminhamento ou uma decisão mais acurada e inteligente de qual mensagem a se encaminhar. O ideal é que na oportunidade de contato o nó transmissor estude a vi-

[1]Considera-se *overhead* neste trabalho o número de cópias geradas para cada mensagem.

[2]Neste trabalho a expressão "roteamento replicador" é às vezes empregada em referência aos "roteamentos baseados em replicação", o mesmo com "roteamento encaminhador" ao invés de "roteamentos baseados em encaminhamento".

abilidade de enviar mensagens ao seu par de comunicação, além de selecionar dentre as mensagens, qual transmitir primeiro. Entretanto este tipo de procedimento não é simples de ser aplicado, decorrência do tipo de movimentação dos nós, limitação do espaço de armazenamento, reserva energética, tempo de contato, etc..

Desta forma, para que o desempenho dessas redes seja satisfatório, os recursos dos nós (energia e armazenamento) devem ser bem aproveitados ou administrados.

1.2 Objetivos

Assim, este trabalho discute a eficiência de mecanismos, que *a priori* não apresentam dependência com relação ao cenário ao qual são empregados, e que podem trazer ganhos na economia de recursos, seja número de transmissões e/ou gasto de energia. Grande parte destes mecanismos têm como objetivo tentar manter a rede operacional por um tempo maior, contudo garantindo um desempenho satisfatório.

Com esta finalidade, propomos gerenciamentos e políticas que tentam diminuir a quantidade de mensagens transmitidas entres os nós para que apenas mensagens relevantes sejam repassadas, e propomos modos de gerenciamento de energia que realizam uma economia de energia pelo desligamento da interface sem fio em determinados períodos sem comunicação. Por meio de simulações foi possível verificar que modos simples de gerenciamentos de energia podem aumentar o tempo operacional da rede.

Propomos também um encaminhamento seletivo que ajusta o número de mensagens repassadas ao nó de contato. Este ajuste utiliza o histórico de encontros do nó. A partir deste histórico é avaliado o interesse em enviar mensagens ao nó receptor e, em caso afirmativo, quais mensagens transmitir. O objetivo é penalizar nós que pouco interagem com outros nós da rede e a eles repassar uma menor quantidade de mensagens. Através de simulações pôde-se concluir que o encaminhamento seletivo proposto diminuiu o número de mensagens transmitidas entre os nós e manteve um desempenho satisfatório.

1.3 Organização da Dissertação

Segue uma breve descrição do conteúdo dos capítulos:

- Capítulo 2 – apresenta as Redes Tolerantes a Atrasos e Interrupções, suas características, arquitetura e trabalhos relacionados.

- Capítulo 3 – explica o funcionamento do simulador, que foi desenvolvido com a finalidade de avaliar os mecanismos e o encaminhamento propostos.

- Capítulo 4 – apresenta os mecanismos de gerenciamentos de recursos propostos para estas redes, em conjunto com resultados de simulações.

- Capítulo 5 – é proposto o encaminhamento seletivo e apresentado seu funcionamento conjuntamente com resultados de simulação.

- Capítulo 6 – por fim, este capítulo apresenta as conclusões e trabalhos futuros.

Capítulo 2

Redes Tolerantes a Atrasos e Interrupções

Este capítulo apresenta uma breve explicação sobre a arquitetura de redes RTAI, assim como, as principais formas de funcionamento de seus protocolos de roteamento, suas classificações e características. Além de apresentar alguns trabalhos relacionados que preocupam-se com a escassez de recursos nestes tipos de rede.

2.1 Descrição

Redes do tipo RTAI ou no original DTN (*Delay/Disruption/Disconnect Tolerant Networks*) são redes caracterizadas pela intermitente conectividade entre seus nós, isto devido à esparsa densidade e/ou mobilidade de seus componentes. Em tais redes, não existe garantia que um caminho entre uma fonte e um destino exista a todo instante [1] e, além de apresentar frequentes desconexões, os atrasos na transmissão podem ser maiores que os suportados por aplicações de redes tradicionais.

Muitos esforços foram dedicados a fim de se desenvolver uma arquitetura de rede robusta a estas características. E a solução encontrada, para contornar a dificuldade das desconexões na entrega de mensagens, foi o armazenamento persistente em memória (*buffer*) dessas mensagens pelos nós intermediários, aliado ao encaminhamento oportunista das mesmas. Assim, explora-se a mobilidade dos próprios nós na tentativa de entregar alguma mensagem ao nó de destino [1]. Desta forma, quando deseja-se enviar uma mensagem, esta é armazenada em *buffer* e encaminhada

nó a nó desde a origem até o destino aproveitando oportunamente a disponibilidade de tais enlaces. Logo, seu correto funcionamento depende da capacidade de armazenamento dos nós, da capacidade desses nós em transferir dados durante o tempo de contato entre eles, além da necessidade de cooperação entre os componentes da rede [8, 9].

O tipo de solução empregada em RTAIs pode ser aplicado a várias outras redes com características similares, assim as RTAIs possuem aplicações em diversos campos de pesquisa, onde as mais comuns são em projetos de monitoramento, comunicação submarina, comunicação espacial, aplicações militares, etc..

Por exemplo, o projeto SeNDT *(Sensor Networking with Delay Tolerance)* [10], desenvolvido pelo *Trinity College Dublin*, tem por objetivo auxiliar autoridades públicas e empresas preocupadas com o meio ambiente. Este projeto utiliza redes de sensores a fim de possibilitar o monitoramento da água de lagos, e também realiza um acompanhamento do nível de poluição sonora em rodovias.

Com relação ao monitoramento da água de lagos, um problema enfrentado pelo projeto foi a grande extensão geográfica do lago escolhido para a análise. Caso fosse utilizada uma rede de sensores tradicional o custo de implantação, para cobrir toda a área de monitoramento, seria elevado devido a necessidade de uma rede de sensores altamente densa. Para contornar este problema foram utilizadas soluções de RTAIs. Os sensores foram organizados em regiões e, balsas de dados[1] *(message ferries* ou apenas *ferries*) trafegavam entre elas, garantindo a coleta de informações e conectividade. Desta forma pôde-se diminuir os gastos com a quantidade de sensores empregados na aquisição de informações [11].

A mesma solução também foi empregada em [12], onde pedestres transportavam dispositivos sem fio que coletavam informações vindas de outros nós fixos distribuídos em uma área. Pelo movimento dos próprios pedestres as informações eram repassadas aos demais nós.

Outra aplicação do mesmo projeto foi o monitoramento da poluição sonora em rodovias. Para isso foram instaladas unidades de sensoriamento que enviam as informações previamente coletadas, mediante consulta do operador que pôde permane-

[1]Muitas vezes ao invés de se utilizar o termo "balsa de dados" *(message ferries)* utiliza-se "mula de dados" (*Data MULE - Mobile Ubiquitous LAN Extension)*

cer em um veículo durante a operação. Esta solução mostrou-se bastante vantajosa devido ao baixo custo, robustez e tolerância a atrasos [11].

Em DakNet [13], os *Data MULEs* eram os próprios meios de transportes públicos, a meta era prover conectividade em regiões rurais com carência de infraestrutura. Em geral soluções deste tipo são implementadas em regiões distantes dos grandes centros urbanos, onde soluções convencionais de redes não podem ser implantadas devido falta de infraestrutura física e/ou alto custo. Alguns exemplos de projetos que compartilham do mesmo objetivo são: o projeto TIER *(Technology and Infrastructure for Emerging Regions)* [14], *Wizzy Digital Courier* [15], SNC *(Sámi Network Connectivity)* [16], entre outros.

Em [17, 18, 19], nós especiais chamados *ferries*, o mesmo que *Data MULEs*, proativamente movem-se para entregar mensagens entre os nós móveis ou estacionários. Os *ferries* são alertados pelos outros nós através de *beacons* de longo alcance. Tendo recebido um alerta, os *Data MULEs* se deslocam para o local onde a troca de dados pode ocorrer. Presumivelmente, o *Data MULE* deve manter um rastro da posição dos nós, pois eles podem ser tanto originadores como destinos das mensagens. Isto implica que tal esquema de roteamento depende do *Data MULE* ter conhecimento da localização dos nós e, para algumas aplicações pode ser questionável que tal característica esteja ou seja disponível para o esquema de roteamento [20].

Outro exemplo, é o projeto ZebraNet [21, 22] que fez o monitoramento da vida selvagem de zebras com o auxílio de sensores em forma de colares. Estes sensores coletavam informações de movimentação das zebras, o objetivo era que as informações coletadas pelos sensores fossem entregues a uma estação coletora (estação-base), para que mais tarde fossem analisadas. Porém, nem todos os colares (zebras) permaneciam, a todo instante, dentro do alcance da estação-base para que seus dados fossem enviados diretamente. Para solucionar esse problema, os colares também trocavam informações coletadas entre si, de forma que a maior quantidade possível de informação chegasse até a estação-base [21]. Cada sensor também computava quantas vezes ele já esteve ao alcance da estação coletora, assim pôde-se montar uma hierarquia entre os nós relativa ao número de contatos com a estação. Cada vez que um sensor estava na área de alcance de outro, esta contagem do número de contatos era informada antes da troca efetiva de mensagens. Um nó **a** ao se encon-

trar com outro que possuía um nível de hierarquia maior, por exemplo um nó $\underline{\mathbf{b}}$, entregava seus dados coletados a este nó $\underline{\mathbf{b}}$, teoricamente melhor habilitado. Assim pela troca de dados entres os nós, as informações coletadas conseguiam alcançar a estação coletora, novamente a mobilidade dos próprios nós permitiu a entrega das informações.

Outro exemplo de monitoramento da vida selvagem pode ser encontrado em [23], onde estuda-se o comportamento de coiotes nas montanhas de Santa Cruz na Califórnia.

Existem também alguns projetos que implementam redes submarinas de comunicação de dados. A comunicação submarina não utiliza radio frequência devido ao seu pequeno alcance na água; a fim de comparação, o alcance de uma tecnologia *Wi-Fi* neste meio seria medido em centímetros caso utilizado. Em geral para viabilizar uma comunicação debaixo d'água são utilizadas redes acústicas, compostas por sonares com taxas de transmissão entre 1 e 5 *kbps* aproximadamente. Este meio de comunicação pode ser considerado hostil uma vez que apresenta problemas como: desvanecimento, múltiplos caminhos, reflexões no fundo e superfície do mar, etc.. Assim, a taxa de erro neste ambiente é considerável mesmo com o uso de pacotes com códigos corretores de erros. Pertubações, como o barulho de uma embarcação trafegando próximo destas redes, normalmente inviabilizam qualquer transmissão por vários minutos. Além disso, a velocidade do som na água é de cerca de 1500 m/s, muito menor que a velocidade da luz no vácuo. Essa velocidade de propagação faz com que, para uma mesma distância, o atraso de propagação seja muito maior, o que obriga uma atenção especial às aplicações sensíveis a atraso [11]. A situação se agrava quando se utiliza redes de múltiplos saltos para cobrir grandes áreas [24].

As interrupções, a alta taxa de erro e o longo atraso sugerem o uso de RTAIs nas comunicações submarinas. O projeto *Seaweb* [25] da marinha americana provê funcionalidades de alcance, localização e navegação, utilizando redes formadas por boias, veículos submarinos autônomos e nós repetidores para a comunicação [11].

Devido à diversidade de RTAIs uma grande dificuldade é prover uma arquitetura que contemple interoperabilidade entre seus vários tipos e protocolos de roteamento. Uma ideia foi apresentada em [26], onde Fall propôs uma arquitetura generalizada baseada em mensagens assíncronas. Para tratar de características únicas em RTAIs,

esta arquitetura obedece a vários componentes, a seguir serão apresentados alguns, embora muitas destas características ainda estejam sendo estudadas e sujeitas a modificações.

- ***Gateway***: Necessários para conseguir interoperabilidade entre RTAIs, assim *gateways* RTAIs devem se localizar nos pontos de interconexão entres as diferentes redes. Os *gateways* são encarregados da translação de protocolos entre as RTAIs [27]. Além disso, podem ser responsáveis pelo armazenamento não volátil de mensagens, caso uma entrega confiável seja requerida pelo nó emissor.

- **Vinculação tardia** (***Late binding***): Para rotear mensagens entre diferentes RTAIs, uma tupla de nomes é usada para os endereços de localização. A primeira parte é única (global), hierarquicamente estruturada por regiões (*Region name*). A segunda parte identifica um nome (*Entity name*) resolvido para uma rede específica e local. Quando uma mensagem é roteada para uma região diferente da atual, somente a primeira parte é usada até se alcançar o limite de cobertura entre as regiões, ou seja, até encontrar um nó *gateway*. Atingida a região onde se encontra o destino da mensagem, a segunda parte é então resolvida como um endereço local. A forma de atribuição de nomes globais ainda está em estudo. O nome de vinculação tardia vem em contrapartida ao termo vinculação antecipada (*Early binding*) onde antes do encaminhamento da mensagem já se sabe o local que o destinatário se encontra.

- **Transferência de custódia** (***Custody transfer***): A transferência de custódia tem como objetivo transferir a responsabilidade da entrega de uma mensagem a outro nó [26, 28]. A arquitetura RTAI não exige que todos os nós RTAIs aceitem a transferência de custódia [29]. Por exemplo, pode ser possível que um nó tenha capacidade de armazenamento suficiente para agir como custódio (nó que aprova o pedido de custódia), mas opte por não aceitar a transferência de custódia quando sua capacidade de bateria estiver abaixo de um determinado limiar. Os nós também podem tomar decisões individuais sobre a aceitação da custódia, baseados por exemplo no roteamento, em políticas de segurança; no tamanho, na prioridade ou no tempo máximo de

vida da mensagem [11]. A transferência de custódia é um conceito generalizado para confiabilidade fim-a-fim. Enquanto que em muitas redes as mensagens são simplesmente descartadas quando a memória se esgota, o mesmo não pode ser feito em RTAIs se a custódia da mensagem tiver sido aceita. Um custódio só pode apagar um agregado (*bundle*)[2] em duas situações: primeira, transferindo-se o agregado para um outro custódio; segunda, se o tempo de vida do agregado expirar. Algumas estratégias na escolha de qual mensagem se solicitar a transferência de custódia, podem ser encontradas em [30]. Em RTAIs, um dos recursos mais disputados é o acesso ao armazenamento em cada nó.

- **Classe de serviços** (***Class of service***): Como as RTAIs podem ter recursos limitados e aplicações diversas, a classe de serviços tenta classificar o grau de urgência na entrega de alguma mensagem entre: baixo, normal ou alto. Analogamente aos serviços postais são por exemplo: os telegramas ou as postagens simples. A classe de serviços pode usar conjuntamente a transferência de custódia, para garantir uma entrega confiável. Isto representa para cada nó RTAI alocar espaço em *buffer*, capacidade de banda, tempo de processamento e energia. Entretanto como reservar estes recursos são questões que permanecem em aberto [27].

Enquanto mecanismos detalhados de cada componente requerem significativa investigação, esta arquitetura generalizada enfatiza diversas decisões notáveis de considerações [27].

Como característica importante, a arquitetura RTAI prevê a utilização de uma técnica de comutação de mensagens aliada a um armazenamento persistente de dados. Esta arquitetura define uma nova camada, abaixo da camada de aplicação, conforme Figura 2.1, chamada camada de agregação (*bundle layer*). Como as aplicações em RTAIs podem definir mensagens de tamanhos diferentes, denominadas Unidades de Dados da Aplicação (ADUs - *Application Data Units*), uma das tarefas da camada de agregação é transformar estas ADUs em uma ou mais Unidades de Dados

[2]Termo geralmente empregado ao se referir a uma mensagem em RTAIs, pois devido a dificuldade em prever oportunidades futuras, o nó tenta enviar o máximo de informação de uma vez, daí o nome de agregado.

de Protocolo (PDUs - *Protocol Data Units*). Assim estas PDUs são armazenadas e encaminhadas pelos nós. A interoperabilidade dos dispositivos e redes é realizada pela camada de agregação, que isola as camadas inferiores da camada de aplicação. Desta maneira, as informações só alcançam a camada de aplicação do nó de destino, entre os nós intermediários as informações trafegam de uma camada de agregação para outra [31, 32].

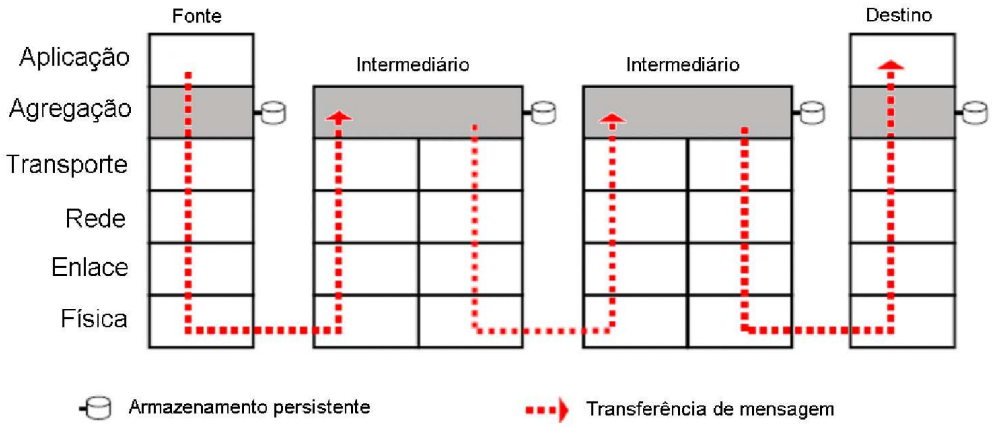


Figura 2.1: Camada de agregação.

Em RTAIs, pressupõe-se que os nós não são alcançáveis ou contactados a qualquer instante. Esta característica contrasta fortemente com o funcionamento de redes tradicionais [20]. Por isso, um conceito importante que deve ser considerado em sua arquitetura é o contato. Um contato corresponde a uma ocasião favorável para a troca de dados ou informações entre os nós. Da abordagem tradicional de teoria de grafos, podemos ter uma rede como um grafo $G(V, E)$, onde V são os vértices (nós) e E as arestas, representando os enlaces. Todavia para RTAIs, caso desejemos semelhante representação, teríamos a seguinte forma $G(V, E(t))$, ou seja, temos uma dependência a mais nos enlaces que é a questão temporal [11].

Segundo a arquitetura destas redes, os contatos classificam-se em: persistentes, sob demanda, programados, oportunistas e previsíveis. Os cinco tipos de contatos serão brevemente apresentados a seguir, suas classificações e explicações em grande parte foram retiradas de [11, 20, 33].

- **Contatos Persistentes**: São aqueles que sempre estão disponíveis, porém não significa que serão sempre alcançáveis.

- **Contatos sob Demanda**: Aqueles que necessitam de alguma ação para que sejam instanciados, porém uma vez acionados permanecem como contatos persistentes até o fim de sua ação. Por exemplo, uma rede de sensores onde os nós alternam seu estado entre "dormir/acordar", com isso seu funcionamento requer uma mensagem específica para o chaveamento entre os estados dos nós.

- **Contatos Programados**: Em algumas RTAIs, uma agenda de contato pode ser preestabelecida entre dois ou mais nós antes da troca de mensagens, estabelecendo o horário e a duração de cada contato. Por isso, esse tipo de contato é denominado contato programado. Uma característica de redes com contatos programados é a exigência de uma sincronização temporal da rede, a fim de que a troca de informações possa ser realizada com sucesso. Aplicações em redes espaciais são um exemplo de rede com contatos programados.

- **Contatos Previsíveis**: Os contatos previsíveis são aqueles onde é possível recorrer a previsões sobre o horário e a duração dos contatos com base em históricos previamente realizados. Logo, ao contrário dos contatos programados, os contatos previsíveis possuem certo grau de incerteza. No contato previsível o nó tem informação de um intervalo de tempo que o contato pode ocorrer e não o instante exato deste contato, como ocorre no programado. Um exemplo de ambiente com este tipo de contato é o cenário onde veículos são utilizados como mensageiros móveis sendo responsáveis pelo armazenamento, transporte e entrega de dados entre regiões distintas. Como a visita dos mensageiros móveis pode ser variável, devido falhas mecânicas, problemas de estrada interditada, entre outras causas, pode-se estimar o horário e a duração das próximas visitas baseando-se no histórico das últimas, porém o horário pode ser impreciso.

- **Contatos Oportunistas**: Ocorrem diante de encontros não previamente programados entre os nós. Esse tipo de contato tem como objetivo obter vantagens de contatos realizados totalmente ao acaso para realizar a comunicação com qualquer nó que esteja fora do alcance de um nó fonte. O conceito de contato oportunista permite comunicação entre nós no qual em nenhum momento exista um caminho inteiramente conectado entre eles. Geralmente, os nós des-

conhecem qualquer informação acerca do estado, da localização ou padrões de mobilidade dos outros nós. Um exemplo de contato oportunista pode ser encontrado no experimento realizado por [34], denominado PSN *(Pocket Switched Networks)* onde usuários portavam dispositivos (*iMotes*) que transferiam dados entre si quando possível. Estes *iMotes* foram entregues a várias pessoas que participaram de uma conferência e desta forma, ao seu término, pôde-se recolher estes dispositivos e estudar padrões quanto a distribuição dos tempos entre contatos. Outro exemplo pode ser encontrado em [35], que consistiu na propagação de um jornal eletrônico entre alunos de uma universidade.

Assim em decorrência do tipo de contato ou tipos de informações a que se pode recorrer, diferentes propostas de roteamento foram e podem ser desenvolvidas.

2.2 Principais Grupos de Protocolos de Roteamento em RTAIs

As RTAIs apresentam muitos desafios que não estão presentes em redes tradicionais. Grande parte destes desafios são provenientes da necessidade de lidar com desconexões e atrasos extremamente longos, o que impacta diretamente o roteamento. Além disso, existem problemas secundários que os protocolos de roteamento precisam enfrentar, por exemplo, lidar com uma limitação de recursos nos nós [36]. Entretanto, na literatura não existem muitos trabalhos que tratam ou preocupam-se com estes últimos [37].

Vários protocolos de roteamento para RTAIs já foram concebidos [1, 38, 39, 40, 41, 42], entre outros. Haja vista que os protocolos de roteamento convencionais não foram projetados para redes com frequentes desconexões, e por isso não apresentam um bom desempenho neste tipo de ambiente [41].

Embora a possibilidade de desconexão devido à mobilidade, em redes *ad hoc*, também seja estudada há muito tempo, existe uma diferença na abordagem de tratamento, pois para resolver o roteamento, em redes *ad hoc*, assume-se a possibilidade de estabelecer um caminho válido entre a origem e o destino com uma maior frequência, o que não se verifica comumente em RTAIs. Desta maneira, diversas variações de protocolos de roteamento tradicionais foram desenvolvidas para redes

RTAIs [43, 44, 45]. Além de maneiras de funcionamento híbrido, com parte do roteamento utilizando protocolos para redes tradicionais e outra com roteamentos para RTAIs, como em [46].

Por existirem vários tipos de RTAIs devido diferentes tipos de cenário ou tipos de contato, diferentes soluções de roteamento foram encontradas. Segundo [47], estas soluções são classificadas de acordo com o grau de informação disponível sobre a topologia da rede. Assim as RTAIs são divididas conforme o seu cenário (topologia), que são ditos determinísticos ou estocásticos.

Nos cenários classificados como determinísticos, assume-se que a duração dos contatos é conhecida. No entanto, pode-se considerar também informações adicionais, como por exemplo, a capacidade de armazenamento dos nós da rede. Assim, dependendo da quantidade de informação disponível, diferentes algoritmos podem ser projetados. Em [48] podemos encontrar diferentes tipos e quantidades de informação acerca da rede que são classificadas e modeladas por "oráculos". Um oráculo é simplesmente uma abstração que é capaz de responder a quaisquer perguntas sobre determinado assunto [11]. Algoritmos que exploram diferentes oráculos foram projetados justamente para serem utilizados como comparadores de melhor caso, além de se verificar qual tipo de informação tem maior relevância para o desempenho da rede.

Ao contrário do cenário determinístico, no cenário estocástico o comportamento da rede não é conhecido. Em algumas RTAIs, os nós não possuem nenhuma informação sobre o estado da rede, o que dificulta na idealização de protocolos de roteamento.

Logo, o cenário é referido como determinístico quando sabe-se a informação dos instantes de contato entre os nós e a duração dos mesmos. Caso contrário, o cenário é classificado como estocástico.

Assim como as RTAIs apresentam uma classificação, os protocolos de roteamento também podem ser agrupados. A classificação mais usual divide, os protocolos de roteamento nestas redes, em dois principais grupos: os baseados em replicação (*replication-based*) e os baseados em encaminhamento (*forwarding-based*).

A principal diferença entre estas duas classificações é o fato do nó ao transmitir uma mensagem permanecer com esta armazenada, no caso do roteamento baseado

em replicação ou apagá-la de seu *buffer* no caso do roteamento baseado em encaminhamento. Uma observação importante é que neste trabalho o uso da palavra "encaminhamento" referir-se a transmitir alguma mensagem e apagá-la do *buffer* só tem essa conotação quando aplicada a classificação entre os tipos de roteamento. Desta forma todas as vezes que utilizarmos o termo "encaminhamento" estamos nos referindo a olhar a tabela de roteamento e decidir qual mensagem transmitir, contudo mantendo a mesma armazenada no *buffer* do emissor, salvo casos do receptor ser o próprio destino.

Além disso, os protocolos podem ser organizados em classes referentes a quantidade de informação que necessitam para funcionar, assim podemos estabelecer três classes, que são: sem conhecimento nenhum da rede (*Zero knowledge*), conhecimento parcial (*Partial knowledge*) e conhecimento completo da rede (*Complete knowledge*), conforme apresentado na Tabela 2.1. Ressaltando que a classe sem conhecimento é relativa a um estado inicial da rede. O protocolo pode principiar sem conhecimento prévio, e durante um período inicial realizar uma coleta de informações que realimente sua forma de funcionamento. Podemos ter protocolos replicadores ou encaminhadores presentes nas três classes.

Classes	**Característica**
Sem Conhecimento	Nenhum oráculo é utilizado, somente a atual/local informação disponível
Conhecimento Parcial	Explora o comprometimento no uso de algumas informações
Conhecimento Completo	Todos os oráculos são utilizados. Representam os comparadores de melhor caso.

Tabela 2.1: Classes de protocolos

2.2.1 Protocolos de Roteamento Baseados em Replicação

A categoria baseada em replicação tem a vantagem da multiplicidade onde, várias cópias da mesma mensagem podem ser disseminadas entre vários nós. A finalidade é de aumentar a probabilidade de que alguma cópia alcance o nó de destino [49], além

da tentativa de reduzir o atraso na entrega. Dentro da categoria dos replicadores ainda encontramos os do tipo *flooding* e os com quota de replicação ou replicação controlada. Os roteamentos do tipo *flooding* enviam réplicas das mensagens para tantos nós quanto possível. Neste tipo de funcionamento, o número de réplicas das mensagens é proporcional ao número de nós da rede.

Um dos primeiros esquemas de roteamento do tipo *flooding* foi o protocolo Epidêmico [50]. No protocolo Epidêmico quando dois nós encontram-se são trocadas informações de quais mensagens cada um possui em *buffer*. Esta informação é passada na forma de um vetor (*Summary Vector*), conforme Figura 2.2 que ilustra a interação entre dois nós **a** e **b**. Após esta permuta de vetores os nós tomam conhecimento de quais mensagens seu par de comunicação possui. De posse disso, cada nó sabe qual mensagem pode tentar replicar ao seu contato. Assim as mensagens são disseminadas como um "vírus", e no mínimo espera-se que uma das cópias alcance o destinatário. Vale ressaltar que este tipo de roteamento não necessita de informações da rede para funcionar, e atinge altas taxas de entrega. Porém seu maior problema é o alto custo em número de transmissões e espaço de armazenamento utilizado, visto que uma grande quantidade de transmissões pode ser realizada.

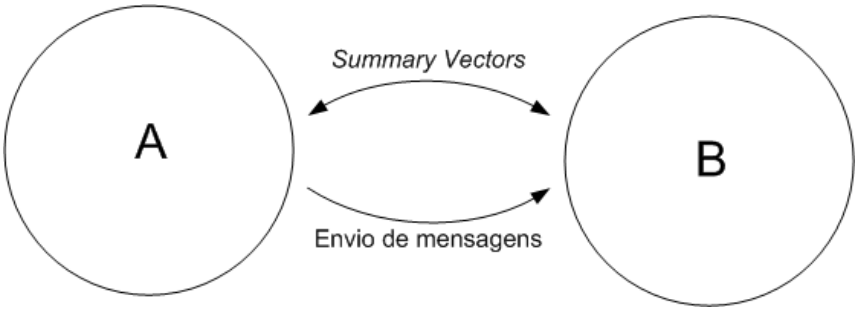


Figura 2.2: Funcionamento do protocolo Epidêmico.

Os roteamentos com quota de replicação surgiram motivados pela problemática do alto uso de recursos, por isso esta classe limita o número de cópias de determinada mensagem. Por analogia é uma epidemia onde nem todos os nós expostos são "contaminados". Em geral o objetivo é gerar um pequeno número de cópias a fim de assegurar que o número de transmissões seja menor e possa ser controlado.

Basicamente os roteamentos de replicação controlada podem ser do tipo: replicação probabilística ou *gossiping*, replicação determinística e replicação baseada em utilidade.

Os roteamentos do tipo *gossiping* espalham as mensagens com uma probabilidade $p <= 1$, esta probabilidade pode ser indicada no início do seu funcionamento. Quando o valor da probabilidade p for igual a um, o protocolo torna-se Epidêmico, se $p = 0$ apenas ocorre a transmissão por contato direto.

Cabe ao grupo de replicação determinística informar quantas cópias de cada mensagem podem existir na rede. Normalmente, estes protocolos são referenciados como *k-copies*, onde o k representa o número máximo de cópias para cada mensagem. Esta grupo também é conhecido como roteamento Baseado em Árvore de Inundação (*Tree-Based Flooding*).

2.2.2 Protocolos de Roteamento Baseados em Utilidade

Em geral quando tratamos de RTAIs, devido sua robustez, diferentes nós podem estar presentes, como:

- Nós com diferentes capacidades de:
  - Armazenamento
  - Processamento
    - Ex.: PDAs, sensores, *laptops*, etc..
- Nós que se movem diferentemente → veículos e pedestres
- Nós com relacionamentos sociais
  - Mesma vizinhança, mesmo andar ou ambiente de trabalho
  - Amigos em comum

Motivados pela ideia que estas características podem ser levadas em consideração surgiram os roteamentos baseados em utilidade. Os roteamentos baseados em utilidade tanto podem ser encaminhadores como replicadores.

Estes roteamentos utilizam informações contextuais para decidir sobre a réplica ou o encaminhamento de alguma mensagem, de acordo com o funcionamento do protocolo. Estas informações são expressas em funções, chamadas funções utilidade. As funções utilidade demonstram o quão adequado representa o nó para transportar alguma mensagem. Dependendo da relação entre os nós e tipos de nós, a função

utilidade pode abranger diferentes informações e pesos atribuídos a cada uma. Alguns exemplos de informações que podem ser utilizadas: tempo de último encontro com o destino, a quantidade de vezes que encontrou com o destino, a capacidade de armazenamento atual do nó, seu nível de energia, etc.. Assim temos basicamente dois tipos de funções utilidade:

- Dependentes do destino ou DD (*Destination-Dependent*)
  - Ex.: tempo que encontrou com o destino da mensagem
- Independentes do destino ou DI (*Destination-Independent*)
  - Ex.: nível de energia do nó

Um exemplo de protocolo que toma decisões baseado em utilidades é o RAPID [42].

2.2.3 Protocolos de Roteamento Baseados em Encaminhamento

O outro grande grupo de classificação abrange os roteamentos encaminhadores. Os roteamentos encaminhadores são os tipos que mais tentam economizar no uso de recursos dos nós. O esquema mais econômico é o denominado transmissão direta onde a entrega só pode ser feita pelo contato direto do nó fonte com o destino.

Geralmente, os roteamentos baseados em encaminhamento fazem uso de informações ou estimativas que os nós obtêm sobre a rede, por exemplo, tempo médio de contato. Baseado nessa estimativa ou informação, um nó é capaz de decidir para qual nó encaminhar uma mensagem e o melhor instante de fazê-lo. Alguns protocolos deste tipo utilizam somente a informação do próximo salto para tomar decisões de roteamento, enquanto outros baseiam-se em métricas fim-a-fim para tal resolução.

Um exemplo de roteamento que considera a informação de próximo salto é o PROPHET (*Probabilistic Routing Protocol using History of Encounters and transitivity*) [1]. Lindgren apresenta um roteamento baseado em probabilidades, utilizando para tal uma métrica associada ao histórico de encontros do nó. Assim como

no roteamento Epidêmico, quando dois nós encontram-se são trocadas listas com informações que identificam as mensagens armazenadas em cada nó. A diferença é que agora existe uma informação adicional à lista. Esta informação corresponde a probabilidade ($P_{(a,b)}$ - *Delivery Predictability*) de cada nó **a** entregar mensagens para um destino conhecido **b**.

O valor desta probabilidade pode aumentar de duas formas: por contato direto ou de forma transitiva. Por contato direto, sempre que **a** e **b** se encontram a probabilidade de entrega de **a** em relação a **b** é aumentada e vice-versa.

Devido a maneira transitiva, caso um nó **a** se encontre frequentemente com um nó **b** e **b** se encontre frequentemente com um nó **c**, o nó **a** torna-se um nó favorável para se encaminhar mensagens destinadas ao nó **c**. Assim, apesar do nó **a** não estar em contato direto com o nó **c** sua probabilidade de entrega com relação ao nó **c** é aumentada devido o contato com o nó **b**.

Quando dois nós deixam de se encontrar, a probabilidade de entrega de cada um em relação ao outro decai. Este decaimento é controlado por uma constante k, chamada de constante de envelhecimento. Este protocolo atinge taxas de entrega satisfatórias com um atraso na entrega de mensagens um pouco maior que o Epidêmico. Contudo reduz muito o número de mensagens transmitidas na rede. Uma desvantagem é seu grau de dependência com o padrão de movimentação dos nós componentes da rede. As etapas de seu funcionamento podem ser vistas na Figura 2.3, a seguir.

O PROPHET pode ser classificado como um roteamento de replicação/encaminhamento baseado em função utilidade, bastando apenas alterar sua política na administração do *buffer* dos nós. Neste trabalho, o PROPHET utilizou uma política de administração do tipo replicação.

Ao contrário do PROPHET, Jones [51] apresenta um protocolo chamado MEED (*Minimum Estimated Expected Delay*) que toma decisões levando em consideração o desempenho fim-a-fim. O MEED tem como base o roteamento em redes tradicionais, com algumas adaptações às singularidades das RTAIs. A métrica adotada por este protocolo é o menor atraso fim-a-fim. Na verdade, trata-se de uma extensão do protocolo MED (*Minimum Expected Delay*) apresentado por [48]. Ao contrário do MED o MEED calcula a métrica utilizando históricos dos contatos observados pelos

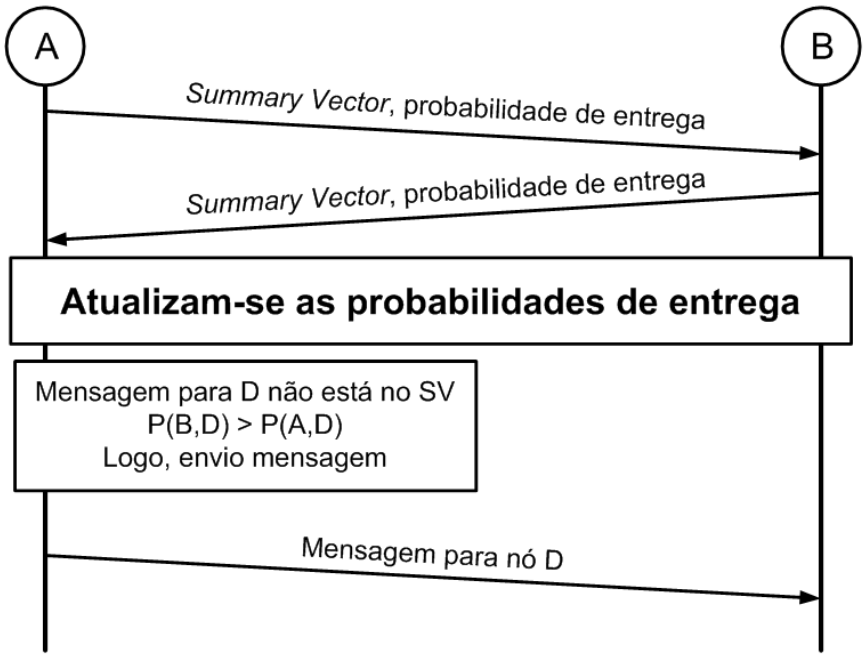


Figura 2.3: Etapas do funcionamento do PROPHET.

nós da rede. Os nós gravam as conexões e as desconexões de cada contato de acordo com uma janela deslizante de históricos (*sliding history window*), onde o tamanho da janela é um parâmetro ajustado independente para cada nó e determina o intervalo de tempo entre as gravações. Quanto maior o tempo da janela mais lentamente a métrica é modificada e vice-versa. Uma desvantagem é o grande uso de recursos de processamento, dado que as rotas precisam ser recalculadas toda vez que chega um contato [11].

2.2.4 Protocolos de Roteamento Híbridos

Existem também os protocolos que são classificados como híbridos. Um exemplo simples de protocolo deste tipo são os denominados protocolos de Dois Saltos (*Two-Hops*). Onde cada nó ao seu primeiro contato replica uma mensagem e posteriormente o nó receptor somente pode encaminhar esta mensagem recebida ao próprio destino.

Um outro exemplo é o protocolo chamado *Spray and Wait* [52]. Neste protocolo um limite superior ao número de permissões de réplicas (*tokens*) é fixado no instante da criação de mensagens. Seu funcionamento ocorre em duas fases. A fase *spray*

onde as mensagens são disseminadas e a fase *wait* onde os nós com uma cópia da mensagem aguardam pelo contato direto com o destinatário. Ou seja, na fase *spray* o nó de origem da mensagem faz cópias desta para os L primeiros nós que encontra. Depois, na fase *wait* cada nó que recebeu uma cópia não a repassa a mais nenhum outro contato, somente se o nó de contato for o próprio destinatário. Um parâmetro importante para este roteamento é a quantidade de cópias (permissões) que podem ser geradas para cada mensagem criada. Outra característica é a existência de diferentes formas de se "espalhar" as mensagens. Por exemplo, o nó fonte pode passar apenas uma cópia para cada novo nó encontrado, ou múltiplas permissões de cópia para que os nós receptores também possam desempenhar a difusão das mensagens, ou seja, tornem-se nós replicadores (*relay nodes*) [53].

Um outro esquema adotado posteriormente por [54] chama-se *spraying* binário. No *spraying* binário o nó de origem da mensagem inicia com L permissões de cópia para cada mensagem e, a cada encontro com um nó que não possua alguma de suas mensagens o nó transmissor a repassa em conjunto com $n/2$ permissões de cópia. Onde n representa a quantidade atual de permissões de cópia que o nó transmissor possui para aquela mensagem. Ou seja, a cada transmissão o nó transmissor repassa metade das permissões que possui ao nó receptor. Este procedimento repete-se até que qualquer nó que possua $n > 1$ permissões permaneça com apenas uma única permissão, nesta situação interrompe-se a replicação desta mensagem e passa-se ao encaminhamento exclusivo ao destinatário da mensagem.

O *spraying* binário atinge altas taxas de entrega aliada a uma economia de recursos, porém pode enfrentar problemas quando aplicado a uma rede com nós de diferentes velocidades de mobilidade, por exemplo uma rede que contenha pedestres e veículos. Repassar uma mensagem a um veículo normalmente torna-se mais proveitoso, pois é um nó que possivelmente terá uma maior abrangência de contatos pela rede. Porém o *sraying* binário não faz distinção entre os nós, deste modo pode ocorrer de todos os contatos na fase *spray* serem pedestres e com isso, caso posteriormente o nó venha a ter contato com algum veículo, fique impedido de replicar alguma mensagem. Este procedimento pode resultar na não entrega de mensagens, caso o destino esteja em uma região distante que só seria possível contato por veículos, ou a um atraso maior na entrega.

A Tabela 2.2 apresenta alguns protocolos, dividindo-os entre os dois principais grupos (replicadores ou encaminhadores), além de classificá-los em relação a vazão durante as oportunidades de transferência, a capacidade de armazenamento dos nós, ambos como finita(o) ou ilimitada(o). Semelhante abordagem é feita em [42], contudo a informação entre parênteses refere-se ao tipo de política de descarte aplicada pelo protocolo, e não ao tipo de modelo de movimentação empregado nas simulações. As políticas de descarte indicam qual mensagem deve ser descartada quando o *buffer* estiver sem espaço e, uma nova mensagem é recebida pelo nó.

<table>
<tr><th>Tipo</th><th>Armazenamento</th><th>Vazão</th><th>Roteamento (descarte)</th></tr>
<tr><td rowspan="2">Replicador</td><td rowspan="2">Finito</td><td>Finita</td><td>RAPID (menor utilidade)</td></tr>
<tr><td rowspan="3">Ilimitada</td><td>PROPHET (FIFO), MV (FIFO), SWIM (msg já entregue)</td></tr>
<tr><td rowspan="2">Ilimitado</td><td>Epidêmico(-),Spray and Wait(-)</td></tr>
<tr><td rowspan="2">Encaminhador</td><td>Alg. Dijkstra Modificado Jain et al (overflow), MobySpace (-)</td></tr>
<tr><td>Finito</td><td>Finita</td><td>MEED (FIFO), MaxProp (maior número de saltos, menor prob.)</td></tr>
</table>

Tabela 2.2: Classificação de protocolos de roteamento

Em [36] podemos encontrar uma outra forma de classificação dos protocolos de roteamento em RTAIs. Nesta classificação temos uma organização aproximada de algumas estratégias de roteamento, usando as classificações de quantidade de cópias das mensagens (replicação) e quantidade de informação (conhecimento) necessária para o funcionamento do algoritmo.

Conforme Figura 2.4, retirada de [36], os dois eixos tanto de replicação e/ou conhecimento mostram-se bem explorados, porém a região central não. Esta região central representa protocolos que agregam vantagens de ambos tipos de estratégias, de modo a prover um aumento da taxa de entrega e uma diminuição do atraso na entrega. Em particular a área na parte de baixo à esquerda, representa esquemas que utilizam menor quantidade de informação e menor quantidade de número de

cópias. Nesta região encontram-se as estratégias que podem ser aplicadas ao mundo real, pois não requerem informações precisas, como de horários de contatos (*schedules*), ou um substancial número de réplicas. Assim, protocolos nesta região não irão consumir uma grande quantidade de recursos pela inundação de mensagens duplicadas. Variantes do protocolo Epidêmico que trabalham a partir da aquisição de alguma informação da topologia mostram-se como um bom passo para esta direção [36].

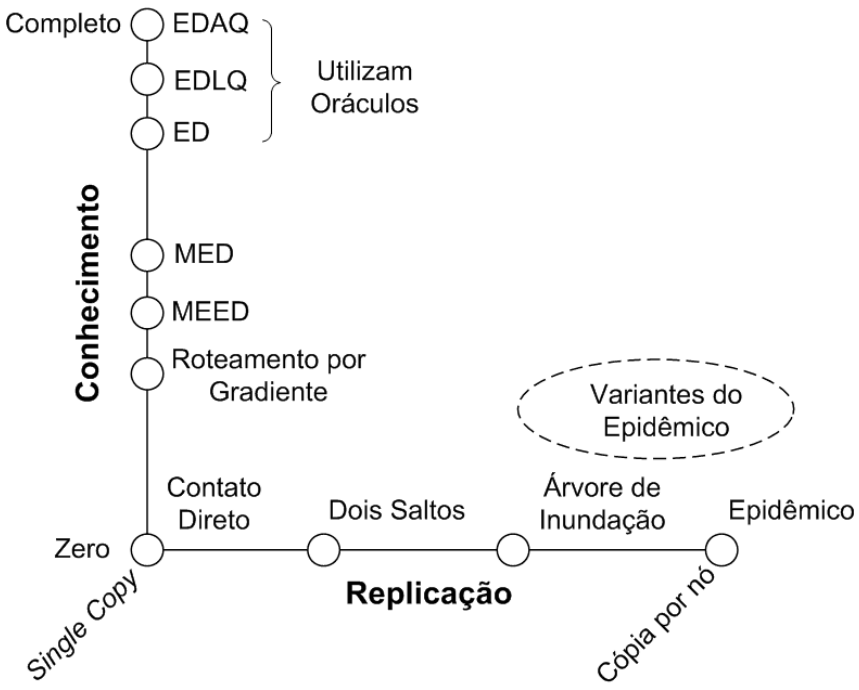


Figura 2.4: Outra classificação de roteamentos em RTAI.

Na classificação de Roteamento por Gradiente (*Gradient Routing*), um peso é atribuído a cada nó, que representa o quão adequado é o nó para uma dada entrega. Este peso pode ser uma determinada métrica própria do protocolo de roteamento. A mensagem tentará sempre seguir um caminho em que esta métrica possua valor mais elevado. Um tipo de roteamento que segue princípio semelhante pode ser encontrado em [40]. Em [55], podemos encontrar uma análise teórica para este tipo de roteamento.

Na parte superior à esquerda encontramos os protocolos que necessitam de uma maior quantidade de informação para funcionar. Esta região representa os protocolos que utilizam diferentes tipos de informação conseguidas através dos oráculos do conhecimento, apresentados em [48]. Existem vários oráculos, onde cada um,

disponibiliza diferentes informações, a saber:

- **Oráculo de Resumo dos Contatos** - Informa o tempo médio até que um novo contato seja realizado entre dois nós, porém não informa o instante e a duração exata dos contatos [11].

- **Oráculo de Contatos** - Informa o instante de início e a duração de todos os contatos entre dois nós quaisquer.

- **Oráculo de Ocupação** - Informa sobre a ocupação do *buffer* de transmissão de qualquer nó da rede. Pode repassar apenas informações sobre os enlaces locais ou sobre todos os nós.

- **Oráculo de Demanda de Tráfego** - Informa a demanda de tráfego de todos os nós, para isto necessita conhecer todas as mensagens que todos os nós da rede desejam enviar.

A partir do uso das informações destes oráculos foram propostos alguns tipos de algoritmos, cujo alguns exemplos são: ED, EDLQ e o EDAQ. No algoritmo ED (*Earliest Delivery*) os atrasos de propagação e de transmissão são precisamente conhecidos e o caminho escolhido é aquele em que a mensagem alcançará o destino no menor tempo. Este algoritmo utiliza a informação do **Oráculo de Contatos**.

O EDLQ (*Earliest Delivery with Local Queuing*) utiliza a ocupação do *buffer* de cada nó para adicionar uma estimativa do atraso na fila de transmissão. Utiliza os **Oráculos de Contatos e de Ocupação**, este último com informações apenas de enlaces locais.

O algoritmo EDAQ (*Earliest Delivery with All Queues*) usa a informação sobre a demanda de tráfico de todos os nós a fim de computar exatamente a demora na fila. Assim, necessita dos **Oráculos de Contatos e Ocupação**, porém este último com informação referente a todos os nós.

A partir do trabalho de [48] pôde-se concluir que os algoritmos que trabalharam com a maior quantidade de informação conseguiram altas taxas de entrega e menores atrasos de entrega. Entretanto, para alguns cenários a diferença entres os algoritmos foi pequena indicando que mesmo com uma quantidade grande de informação a melhora no desempenho não foi significativa. Além disso, as técnicas apresentadas

assumem que uma informação acurada sobre a tabela completa de horários de encontros dos nós é conhecida. Isto pode ser possível em cenários onde a conectividade é extremamente previsível e confiável. Porém, na realidade os horários de encontros entre os nós, em grande parte dos casos, podem ser imprecisos ou completamente imprevisíveis [36].

Em geral, roteamentos do tipo replicação consomem mais recursos (armazenamento e energia), pois toda oportunidade de contato é aproveitada. A cada nova ocasião favorável, o nó tenta repassar mensagens armazenadas em *buffer* que o seu nó par de comunicação não possua. Com isso, roteamentos deste tipo atingem melhores taxas de entrega e menores atrasos na entrega de mensagens, ao custo de maior uso de recursos. Ao contrário dos roteamentos baseados em encaminhamento, que utilizam menos recursos a cada contato. Nestes tipos de roteamento, avalia-se a viabilidade de envio de alguma mensagem. Com isso um menor número de transmissões é realizado e a taxa de entrega pode ser próxima de um roteamento do tipo replicação, dependendo do cenário onde é empregado.

Existe um comprometimento entre a utilização de recursos e a taxa de entrega. Devido a isso temos dois extremos: alocações/usos de recursos e desempenho, conforme Figura 2.5. Tem-se o menor uso de recursos quando trabalha-se com poucas cópias da mensagem, porém pode-se perder em desempenho, tanto em taxa de entrega quanto atraso no recebimento, pois são menores as chances de entrega.

Conforme buscamos um melhor desempenho da rede, o número de cópias das mensagens tende a aumentar a fim de favorecer uma quantidade maior de oportunidades de entrega, porém se o número de cópias for excessivo melhoramos o desempenho ao custo de um maior uso de recursos.

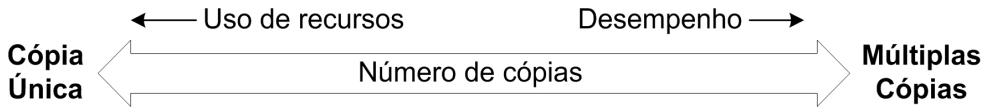


Figura 2.5: Comprometimento entre utilização de recursos e desempenho.

Logo, mecanismos de otimização foram e são projetados. Muitos desses mecanismos não são exclusivos do tipo de roteamento e portanto podem se adequar a diversos protocolos, alguns mecanismos encontrados na literatura serão apresentados na Seção 2.3 a seguir.

2.3 Alguns Mecanismos de Gerenciamento de Recursos

Além da própria contribuição dos protocolos de roteamento para a redução de mensagens transmitidas, o que representa uma economia de energia e menor uso de *buffer*, existem mecanismos ou procedimentos que cooperam para a mesma finalidade. A seguir serão descritos alguns mecanismos que denominamos de mecanismos de gerenciamentos de recursos.

2.3.1 Gerenciamentos de Recursos Relacionados à *Buffer*

Conforme as características das RTAIs, o espaço de armazenamento dos nós é importante para garantir seu funcionamento. Principalmente devido ao fato de que os nós podem permanecer com a mensagem armazenada por um período grande de tempo. Com isso, caso o nó venha a receber alguma mensagem e este esteja sem espaço em *buffer*, é provável que tenha que descartar alguma. Porém pode acontecer desta mensagem que o nó acabou de receber ser uma mensagem que já alcançou o destino, e desta forma não precisaria mais ser transmitida nem mantida em *buffer*. Além disso, algum nó que estivesse com esta mensagem armazenada poderia descartá-la sem prejudicar a taxa de entrega.

Devido este problema existem protocolos que aplicam uma "limpeza" nos *buffers*, removendo mensagens que já atingiram o destinatário. Alguns protocolos que utilizam este método são: o protocolo SWIM (*Shared Wireless Infostation Model*) [38] e o *MaxProp* [40]. O protocolo SWIM usa mecanismos, denominados VACCINE e IMMUNE_TX. O funcionamento destes mecanismos é semelhante a uma lista de *acks*. Antes da troca de mensagens, os dois nós trocam esta lista de *acks*, que contém os identificadores das mensagens que já foram entregues ao nó de destino. Logo, são mensagens que podem ser descartadas. A diferença entre os dois protocolos está na forma de composição da lista. No *MaxProp*, assim que um nó encaminha uma mensagem para o destino esta mensagem é adicionada a lista de *acks*. No protocolo SWIM, a mensagem é colocada na lista de *acks* quando entregue a um nó intermediário denominado coletor (*collection station*). Em [56], encontramos um mecanismo denominado *Passive Cure* que apaga do *buffer* dos nós mensagens já

entregues. Para esta finalidade, uma difusão de *acks* é realizada, porém o reconhecimento é feito somente pelo próprio nó que fez a entrega.

A lista de *acks* auxilia numa melhor administração do *bufffer* pois elimina mensagens que já foram entregues, deixando que o espaço de armazenamento seja reservado para mensagens que ainda não alcançaram o destino. Porém, o número de transmissões de mensagens entre os nós continua grande. É também necessário que as entradas na lista de *acks* expirem quando um tempo suficiente seja alcançado, pois a lista tende a crescer com o número de mensagens, e por fim acaba-se tendo uma lista de tamanho relevante a se trocar. Para amenizar isso, pode-se ao invés de transmitir a lista transmitir um *hash* da lista de *acks*.

2.3.2 Políticas com relação ao descarte de mensagens

Mesmo com o uso da lista de *acks*, o descarte de alguma mensagem pode ser inevitável. Nestes casos é necessário que os nós utilizem políticas de descarte. Isto para garantir que mensagens mais importantes sejam preservadas. Em [57], algumas políticas de descarte são enumeradas: descarte da mensagem mais antiga (FIFO), descarte aleatório (ALEAT) e descarte da mensagem cujo recebimento tenha sido recente. Em [58], algumas políticas foram propostas e avaliadas, como: mensagem mais encaminhada (MOFO) (*Evict most forwarded first*), mensagem com o menor tempo de vida, mensagem com o menor valor de probabilidade de entrega, dentre outras.

2.3.3 Gerenciamentos Relativos à Economia de Energia

Existem trabalhos que consideram o gasto energético que cada nó possui decorrente da interface sem fio, e partir disso modelam o funcionamento dos seus nós ou administram o uso da interface em alguns períodos. Em [27, 59, 60] podemos encontrar um tipo de gerenciamento que utiliza dois rádios a fim de estimar os intervalos em que o nó pode "dormir" (*wakeup intervals*) e assim perca poucas oportunidades de contato. Quanto aos rádios, um opera em baixa potência e outro funciona em alta potência. O rádio com baixa potência é utilizado para descoberta de oportunidades de comunicação enquanto, o rádio com alta potência é encarregado da transmissão de dados. O rádio de alta potência só é posto em funcionamento caso o rádio de

baixa potência confirme a presença de algum nó vizinho. Também é mostrado que o conhecimento da carga de tráfego (*traffic load*) pode ser usado como controle do quanto que o nó pode "dormir/acordar" para prover uma maior conservação de energia, enquanto suficientes oportunidades de comunicação são mantidas. O tipo de mecanismo também contempla a descoberta de mais de um contato por vez, e os resultados de economia são relevantes, porém é necessário o uso de dois rádios por nó.

Uma ideia semelhante pode ser encontrada em [61] onde existe nós que os autores denominaram de *throwboxes*, que são na verdade nós fixos com uma maior capacidade de armazenamento e processamento. Estes nós foram acrescentados à rede para aumentar o número de oportunidades de encontro, porém além disso estes também realizam um tipo de gerenciamento de energia. Conforme mostrado em [62] mesmo um número reduzido de *throwboxes* já foi suficiente para um aumento na taxa de entrega, porém o gasto elevado de energia fez com que estes nós apresentassem um tempo de vida curto. Assim para que fossem mais efetivos, cada *throwbox* deveria realizar algum gerenciamento que não prejudicasse o desempenho da rede. Desta forma, cada *throwbox* passou a operar com dois rádios, onde o primeiro chamado $Tier$-0 monitorava o recebimento de *beacons* da balsa de dados, neste caso um ônibus, e o outro ($Tier$-1) era encarregado da transmissão de dados. Quando o $Tier$-0 recebia um *beacon* da balsa de dados o nó fixo estimava, através de um algoritmo de predição, quando surgiria uma oportunidade de transmissão da própria *throwbox* com algum veículo. Calculada esta predição o segundo rádio ($Tier$-1), que até então estava "dormindo", era acordado em um instante próximo desta oportunidade de transmissão. Assim o rádio $Tier$-0 era encarregado da descoberta de oportunidades de contato enquanto o outro $Tier$-1 realizava a transmissão de dados. Vale ressaltar que a trajetória da balsa de dados era conhecida.

Em [63] os autores modelam o comportamento dos nós entre acordar e dormir baseados no valor de números primos. Na verdade a aplicação decide qual deve ser o *duty cycle* do nó, daí verifica-se qual número primo n mais se aproxima do resultado da expressão $1/n \approx$ *duty cycle*. Este valor de n passa a ser calculado para todos os nós, corretamente relacionado ao valor do *duty cycle* instruído pela aplicação. Por exemplo, um nó **a** deve ter *duty cycle* de 20%, logo, segundo a expressão $1/n \approx 0.20$

o valor de n deve ser 5; a aplicação de um segundo nó **b** elege um *duty cycle* de 30%, para este caso o valor que mais se aproxima é $n = 3$. Assim o nó **a** funcionará em intervalos de 5 em 5, $c_a = 5$, e o nó **b** funcionará em intervalos de 3 em 3 unidades de tempo, $c_b = 3$. Além disso, cada nó também possui um instante em que ele próprio usará como referência para o início dos próprios intervalos de *duty cycle*, que são: a_i referente ao primeiro nó e b_i com relação ao segundo nó. Na Tabela 2.3 abaixo, foi plotado alguns instantes de funcionamento dos nós, os valores marcados foram justamente os intervalos de funcionamento de cada nó. A primeira linha da tabela indica uma referência de tempo global. Assim pela tabela, as colunas onde ambos os valores de c_a e c_b estão marcados representam a sobreposição dos instantes de *slots* de tempo, indicando que caso os dois nós estejam na área de alcance um do outro eles poderão se comunicar, pois estarão com a interface sem fio ativada.

t	0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16
c_a	-	**0**	1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**	13	14	**15**
c_b	-	-	**0**	1	2	3	4	**5**	6	7	8	9	**10**	11	12	13	14

Tabela 2.3: Tabela de intervalos de funcionamento dos nós **a** e **b**

Pela tabela, no instante 7 s os nós poderiam se comunicar. Desta forma diminuiu-se o gasto de energia referente aos períodos de escuta. Porém, neste mecanismo, se os nós escolherem os mesmos números primos e tiverem valores de referência diferentes, eles nunca estarão com sobreposição de *slots* de tempo, logo não poderão se comunicar. Pode também ocorrer do valor de sobreposição dos nós ser elevado e que demore muito a ocorrer. Os autores amenizam estes problemas escolhendo ao invés de um número primo dois números primos, cujo a soma do inverso deles resulte aproximadamente no valor de *duty cycle* desejado, $1/n_1 + 1/n_2 \approx$ *duty cycle*, e utilizam um algoritmo próprio para escolha deste par de números de modo que tente diminuir o tempo necessário para a primeira sobreposição. Porém mesmo assim, valores de *duty cycle* com poucas opções de pares de números primos ainda são prejudicados.

2.4 Conclusões do Capítulo

Neste capítulo, apresentamos de forma breve a arquitetura de redes RTAIs suas classificações, características e, abordamos o funcionamento de alguns protocolos de roteamento para estas redes, mostrando que o próprio protocolo pode contribuir para um menor número de transmissões de mensagens. Ao fim, apresentamos alguns mecanismos ou gerenciamentos que encontramos na literatura que contribuem para uma melhor utilização de recursos. No próximo capítulo será detalhado o ambiente de simulação utilizado para avaliar os mecanismos que serão apresentados e propostos no Capítulo 4.

Capítulo 3

Ambiente de Simulação

A fim de avaliar o encaminhamento seletivo proposto e apresentado no Capítulo 5, assim como os mecanismos de gerenciamento de energia que serão apresentados no Capítulo 4, foi desenvolvido um ambiente de simulação. Este ambiente teve por objetivo modelar a escassez de recursos numa RTAI em alto nível, de forma que características externas ao problema fossem abstraídas.

3.1 Simulador

Para avaliar o problema da escassez de recursos em RTAIs, foi desenvolvido um ambiente de simulação que utiliza recursos básicos do NS-2, um simulador a eventos discretos. Aproveitando-se desta característica do NS-2, o simulador desenvolvido evolui em intervalos fixos de tempo que denominamos de *snapshots*. A cada *snapshot*, todos os contatos existentes entre os nós são determinados e, baseado nisso, todas as trocas de mensagens possíveis entre os nós são realizadas. Após essa troca, a simulação avança para o próximo *snapshot*. A lista de contatos de um determinado nó, construída a cada *snapshot*, é composta pelos nós que se encontram a uma distância igual ou inferior ao alcance de transmissão estabelecido.

Conforme Figura 3.1, podemos acompanhar as etapas desde a descoberta de vizinhança até a troca de mensagens no intervalo de um *snapshot*.

Em cada *snapshot*, a troca de mensagens é realizada de maneira sequencial do seguinte modo. No início do *snapshot*, todos os nós da rede podem trocar mensagens, ou seja, fazem parte de uma lista de nós habilitados para a comunicação. Então,

um nó, escolhido aleatoriamente desta lista, verifica seus contatos e, dentre estes, escolhe um como par de comunicação. A cada *snapshot* apenas uma mensagem pode ser trocada entre esses dois nós e tanto os nós vizinhos do nó fonte quanto os vizinhos do nó receptor ficam impedidos de trocar mensagens naquele mesmo *snapshot*. Com isso, simulamos a existência de um protocolo simples de controle de acesso ao meio que evita colisões e que permite transmissões concorrentes no mesmo *snapshot* apenas quando todos os nós envolvidos estão fora dos alcances de interferência. Por exemplo, um nó **a** escolhido aleatoriamente em um *snapshot*, verifica inicialmente quais são seus vizinhos de um salto. Dentre estes vizinhos, o nó **a** escolhe um nó **b** para ser seu par de comunicação. Depois de realizado o envio de uma das mensagens a serem trocadas entre os nós **a** e **b**, os vizinhos de **a** e de **b** ficam impedidos de se comunicar com qualquer outro nó naquele mesmo *snapshot*. Em seguida a essa troca de mensagem, todos os nós "envolvidos" nessa comunicação (nós **a**, **b** e seus vizinhos) são retirados da lista de nós habilitados a realizar uma comunicação ainda nesse *snapshot*. Então, um novo nó é escolhido aleatoriamente dessa lista e assim sucessivamente até que a lista se esvazie por completo. Quando isso ocorre, a simulação avança para o próximo *snapshot*. Em *snapshots* sucessivos, os nós **a** e **b** continuarão trocando mensagens até que: todas as mensagens escolhidas tenham sido enviadas ou até quando **a** e **b** permanecerem no alcance um do outro.

A escolha do intervalo de tempo entre *snapshots* pode ser vista como uma maneira de modelar a restrição de recursos da rede, pois o intervalo determina quanto tempo é necessário para a transmissão de uma única mensagem. Por exemplo, se utilizarmos mensagens de 10 KBytes e o intervalo de tempo do *snapshot* for de 0.1 segundo, isto equivale a uma taxa de transmissão de 800 Kbps. Se aumentarmos para um intervalo de tempo de 0.5 segundo, a taxa de transmissão passa a ser de 160 Kbps. Vale ressaltar que estes são os valores da taxa de transmissão da interface de rede considerando-se a sobrecarga envolvida (cabeçalhos, reconhecimento, intervalo entre quadros, etc..).

Cada mensagem possui um identificador único na rede. Além disso, todos os nós mantêm informações atualizadas de todas as mensagens presentes em seus *buffers*, sobre quantos saltos ela já fez até chegar a este nó, a quantidade de vezes que foi encaminhada pelo nó em questão, o instante em que foi gerada, etc.. Na imple-

mentação, cada nó possuiu dois *buffers*, um para as mensagens próprias, ou seja, as geradas pelo próprio nó (*source buffer*) que é ilimitado e outro para as mensagens sujeitas a encaminhamento (*relay buffer*), este finito. A utilização de um *source buffer* ilimitado é uma escolha de implementação, justificada pelo fato de que o nó que gerou a mensagem não tem interesse em descartá-la antes que ela seja repassada. No instante da troca de mensagens o nó só pode enviar uma única mensagem por *snapshot*.

Devido os modos de gerenciamentos propostos no Capítulo 5, tornou-se necessário acrescentarmos o gasto de energia na interface sem fio dos nós em decorrência de seu modo de funcionamento. Assim, foi computado os gastos inerentes à transmissão, recepção e por *listening* de transmissões de outros nós. Desta forma, todos os nós possuíam um valor inicial de energia E e a cada *snapshot*, de acordo com o modo de operação do nó, este valor de energia era decrementado. Assim, todos os nós que no *snapshot* desempenharam a ação de transmissores decrementavam sua energia inicial de En_{Tx}, todos os nós que foram receptores decrementavam seu valor de energia inicial de En_{Rx} e, os demais nós que não foram nem receptores nem transmissores diminuiam sua energia inicial de En_{Li}. Os valores dos gastos da interface utilizados estão na Tabela 3.1 a seguir, conforme retirados de [60]. Estes gastos variam conforme o rádio utilizado, porém em geral em ordem de consumo de energia temos o maior gasto transmitindo, recebendo e ouvindo, como podemos encontrar em [61] e [64].

Modo de Operação	Gasto em mW
En_{Tx}	1.3272
En_{Rx}	0.9670
En_{Li}	0.8437

Tabela 3.1: Gastos na interface sem fio

No *snapshot* em que o nó transmitiu ou recebeu apenas este gasto foi computado, ou seja, não foram somados gastos de ouvir e transmitir ou ouvir e receber no mesmo *snapshot* para um mesmo nó. Quando o valor de energia inicial de qualquer nó chegou a zero este nó foi então classificado como "morto" e não mais foi incluído

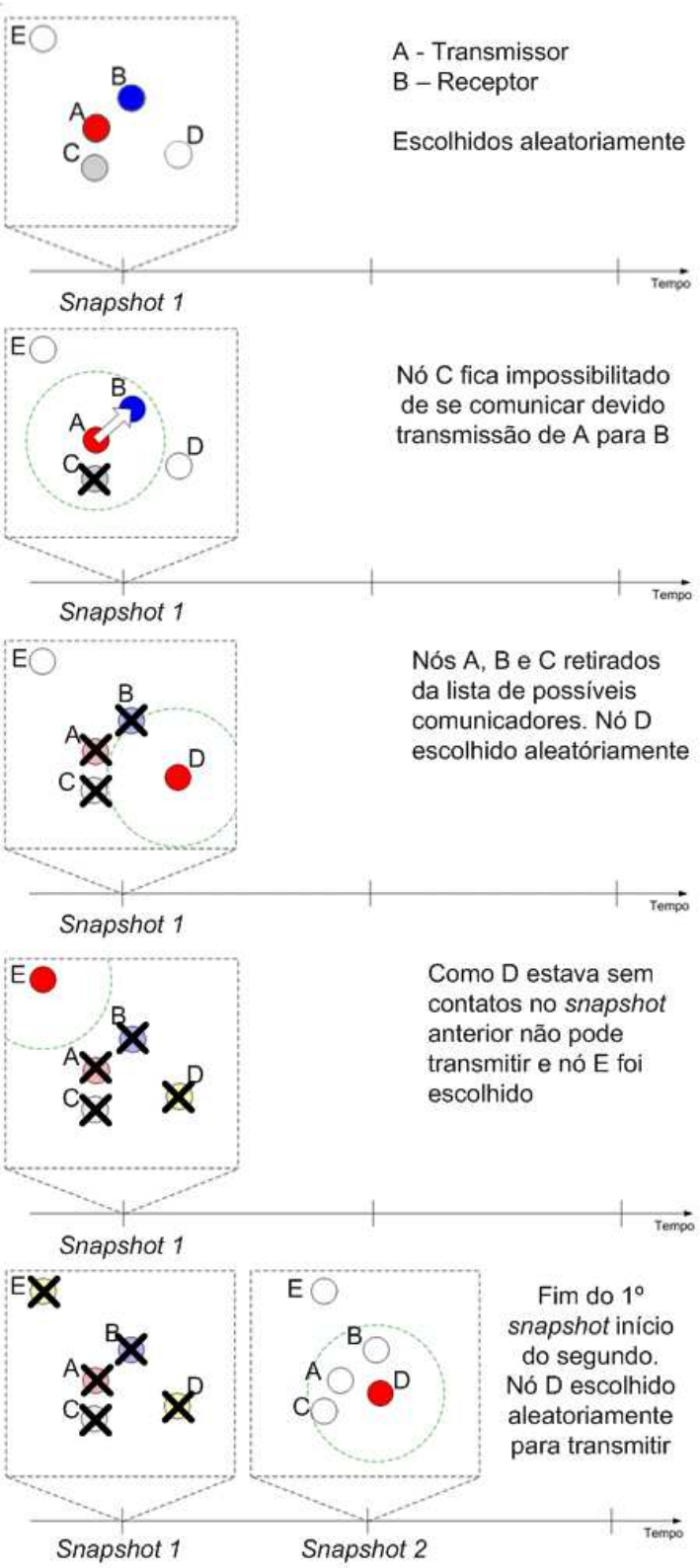


Figura 3.1: Etapas do *snapshot*.

na lista de nós habilitados à comunicação. Vale ressaltar que devido a forma de implementação os gastos de energia são computados ao fim do *snapshot* e, desta

forma pode ocorrer do nó ter uma energia mínima, para não ser classificado como "morto", mas que seria insuficiente para a transmissão ou recepção e este acabe realizando tais ações.

Sucessivos testes foram realizados com cenários mais simples cujos resultados eram intuitivos com o objetivo de validar o simulador. Assim, foram realizados seguidos refinamentos a fim de ajustá-lo ao funcionamento esperado. Neste simulador, implementamos os roteamentos Epidêmico e PROPHET, porém com algumas modificações apresentadas a seguir.

Considerações

No roteamento Epidêmico original todas as mensagens presentes no vetor de sumários são repassadas, visto que, os nós possuem *buffers* e taxa de transmissão ilimitados. Desta forma também não há necessidade de se preocupar com a ordem de envio das mensagens, pois independente da ordem todas serão repassadas ao nó receptor. No nosso caso, pretendíamos justamente avaliar o impacto da falta de recursos neste tipo de roteamento, logo algumas modificações foram implementadas para permitir este funcionamento.

Além de um problema de decisão de qual mensagem transmitir e/ou descartar, passou a existir um caso proveniente da restrição de armazenamento do *buffer* atual dos dois nós envolvidos na troca de mensagens. Neste nosso caso, como o *buffer* dos nós é limitado, seu conteúdo pode ficar em constante mudança, como consequência, seu vetor de sumários também. Por exemplo, conforme Figura 3.2, um nó **a** pode estar sem espaço para acomodar uma nova mensagem vinda de um nó **b**, logo o nó **a** acaba excluindo de seu *buffer* atual alguma mensagem a fim de receber esta nova. Com isso seu *buffer* pode se tornar diferente do *buffer* do nó **b** e um novo vetor de sumários (SV) é montado e informado. Este novo vetor informa a diferença entre os dois *buffers* e uma nova transmissão pode acabar sendo realizada. Devido este problema, os nós envolvidos na transmissão e recepção podem permanecer numa troca de mensagens constante e, a situação dos dois *buffers* tornarem-se espelhados, que originalmente ocorria mais facilmente, pode demorar ou até mesmo não ser atingida.

Desta forma, foi implementado o que denominamos de *blacklist*, que é uma lista

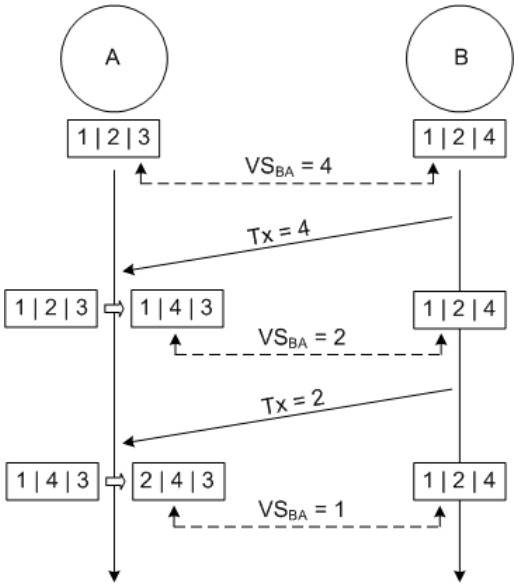


Figura 3.2: Problema devido ao *buffer* limitado no roteamento Epidêmico.

que todos os nós possuem e que é preenchida com o nó receptor que o nó transmissor acabou de se comunicar. Assim que as mensagens acordadas no início da negociação sejam transmitidas, mesmo que este nó permaneça em contato ele não é selecionado no próximo *snapshot* como um possível receptor. Somente quando o tempo de permanência na *blacklist*, que denominamos quarentena, se esgotar é que este nó pode ser selecionado como um possível nó receptor para este nó transmissor. Assim se evitou o fato de dois nós permanecerem em constantes transmissões ou retransmissões da mesma mensagem. Foram feitos testes de simulação com diferentes valores de quarentena, e para estes diferentes valores testados (5, 10, 15 e 20 segundos) a variação da taxa de entrega foi pequena e permaneceu dentro do intervalo de confiança estipulado (95%). Outra consideração foi no algoritmo PROPHET. No PROPHET existe um limiar mínimo para o envio de mensagens, onde mesmo que o receptor tenha uma probabilidade de encontro maior que o emissor, esta probabilidade deve ser maior que este limiar para ser repassada. Aqui não adotamos este limiar, basta o nó receptor ter uma probabilidade maior para o transmissor repassar a mensagem ao seu atual par de comunicação, pois no artigo de referência [58] não fica claro qual valor foi utilizado.

3.2 Conclusões do Capítulo

Neste capítulo, apresentamos o ambiente de simulação desenvolvido para levar em conta as restrições de recursos em RTAIs, assim como as considerações realizadas na implementação dos protocolos Epidêmico e PROPHET.

Capítulo 4

Mecanismos de Gerenciamento de Recursos em RTAIs

O mau uso de recursos disponíveis em RTAIs pode afetar seu funcionamento e desempenho. Por exemplo, uma má administração do *buffer* dos nós pode resultar em transmissões e armazenamentos indevidos, como o envio e a estocagem de mensagens que já alcançaram o destinatário. Como consequência, um número grande de transmissões desnecessárias acabam sendo realizadas, refletindo-se em um maior gasto energético. Indo mais além, o problema mencionado pode resultar na sobrecarga dos *buffers* dos nós da rede, e ocasionar descartes inapropriados, ou seja, descartes de mensagens que ainda não haviam sido entregues. Tratando-se de uma rede oportunista, esta perda de mensagens pode gerar uma queda em seu desempenho. Por esta razão, existe uma série de mecanismos que de alguma forma otimizam ou administram o uso de recursos dos nós, pois o problema agrava-se quando trabalhamos com uma rede onde recursos como armazenamento e energia são escassos.

Neste capítulo serão apresentados alguns mecanismos que podem otimizar o uso de recursos dos nós componentes, seguidos de resultados obtidos por simulação. Os mecanismos foram divididos em grupos onde as soluções indicavam maior impacto. Embora alguns mecanismos sejam relacionados, a divisão foi feita da seguinte forma: soluções relacionadas à *buffer*, políticas para baixas taxas de transmissão e modos de gerenciamento de energia.

4.1 Soluções Relacionadas à *Buffer*

Uma das formas de melhorar a utilização dos recursos é otimizar ou administrar o espaço de armazenamento. Como as mensagens podem levar muito tempo para serem entregues, torna-se necessário sua permanência nos nós intermediários. Com isso, se um nó possui pouco espaço em memória para este armazenamento, é possível que tenha que recorrer ao descarte caso venha a receber uma nova mensagem. Se a mensagem descartada ainda não havia chegado ao destinatário, o desempenho da rede pode diminuir, devido ao aumento da perda de mensagens. O pior caso ocorre quando a mensagem que desencadeia o descarte é uma mensagem que já foi entregue ao nó de destino. Ou seja, uma mensagem que não teria mais necessidade de transitar pela rede e gerar transmissões e alocações em *buffer*.

Devido este problema existem protocolos que aplicam uma "limpeza" nos *buffers*, removendo mensagens que já atingiram o destinatário, conforme apresentado na Seção 2.3 do Capítulo 2. Para esta finalidade é montada uma lista que contém as mensagens já entregues (lista de *acks*) e, antes do envio efetivo das mensagens esta lista é trocada pelos nós envolvidos na comunicação.

Há uma outra abordagem, que não vimos referência na literatura, e que evita o descarte de mensagens. Nesta outra abordagem, que denominamos encaminhamento proativo, são enviadas apenas o número de mensagens que o receptor pode armazenar. Isto evita que mensagens sejam descartadas e que muitas transmissões sejam realizadas.

Na implementação deste tipo de mecanismo, os nós em contato podem trocar informações no início da negociação da troca de mensagens. Por exemplo, no caso do roteamento Epidêmico, a informação adicional sobre o tamanho do espaço disponível em *buffer* do nó receptor, pode ir junto do *Summary Vector*. Assim, além de saber as mensagens que precisam ser enviadas para o outro nó, o nó transmissor saberá também quantas mensagens seu par ainda pode comportar.

Apesar de trivial e simples, este mecanismo proposto possui grande relação com a economia de recursos em RTAIs e pode trabalhar como um complemento para a lista de *acks*. Quando o *buffer* de um nó estiver cheio, o mesmo só terá chance de retirar mensagens dele quando encontrar com o destino de alguma das mensagens ou quando encontrar com outro nó que possua o *ack* de alguma das mensagens (no caso

do uso de lista de *acks*). Isso fará com que menos transmissões sejam realizadas, economizando a energia gasta nesta tarefa. Além disso, fará com que nenhuma mensagem seja descartada, evitando um possível desperdício de alguma informação que ainda não foi entregue.

Por outro lado, o encaminhamento proativo pode possuir o efeito indesejado de impedir que as mensagens sejam disseminadas rapidamente pela rede. O que pode aumentar o atraso na entrega da informação e reduzir a taxa de entrega. Portanto para este mecanismo pode existir um compromisso entre uma possível redução na taxa de entrega e a redução do gasto de energia. Este compromisso tem maior impacto quando a capacidade de armazenamento do nó é menor. Além disso, vale ressaltar que quando utilizado somente o encaminhamento proativo, sem a lista de *acks*, as mensagens só serão descartadas apenas pelo seu tempo de expiração atribuído pela aplicação.

Estes mecanismos, lista de *acks* e encaminhamento proativo, foram implementados no roteamento Epidêmico e no PROPHET. Para os experimentos realizados foram utilizados 50 nós, com alcance de 100 m, que se movimentavam em uma área quadrada de 1000 m de lado durante 1000 segundos. Além disso, foram criados 30 cenários distintos para cada configuração de mobilidade. Com isso, todos os resultados apresentados são médias dos resultados dos 30 cenários com intervalos de confiança de 95%. Para a movimentação dos nós dois tipos de cenários foram gerados, com baixa e alta mobilidade, ambos utilizando o modelo de mobilidade *Random Waypoint*. Para os cenários com baixa mobilidade, os nós se movimentavam com velocidade máxima de 3 m/s e pausa de 10 s. No cenário de alta mobilidade, a velocidade máxima era de 10 m/s e pausa de 2 s. Foram utilizadas 250 mensagens, cada nó gerou 5 mensagens para destinos distintos no início da simulação.

Os resultados da simulação referentes ao protocolo Epidêmico, como taxa de entrega, número de mensagens enviadas, número médio de saltos das mensagens entregues, número de mensagens descartadas e número de mensagens confirmadas na origem, podem ser vistos nas Figuras 4.1 a 4.10.

Embora não apresentado os resultados do PROPHET, já era esperado que a lista de *acks* e o encaminhamento proativo apresentassem poucas melhorias, pois o número de mensagens tanto transmitidas como as descartadas é muito reduzido,

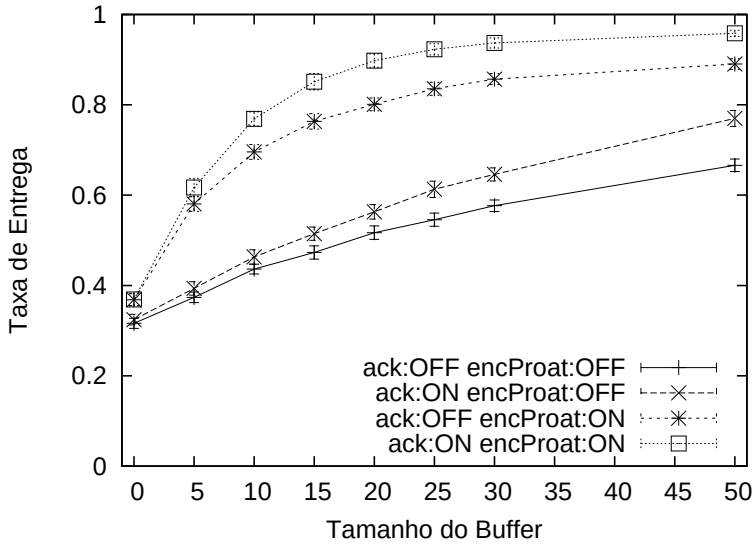


Figura 4.1: Taxa de Entrega - Baixa velocidade.

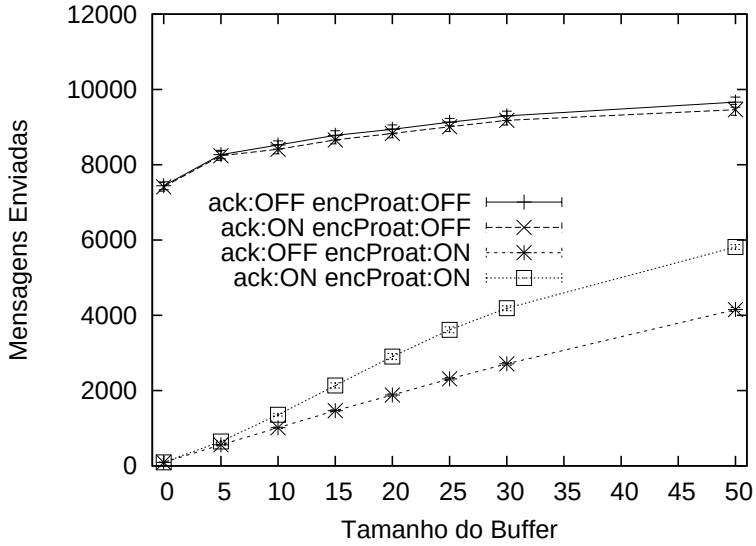


Figura 4.2: Mensagens enviadas - Baixa velocidade.

decorrência da natureza do protocolo.

Além disso os mecanismos não influenciam no aumento da taxa de entrega pois pela maneira de funcionamento do PROPHET a quantidade de descarte deste roteamento é baixa e, a escolha de quando enviar alguma mensagem ou qual mensagem transmitir no instante de contato não é muito influenciada. Assim, o resultado do uso destes mecanismos surtiu mais efeito no protocolo Epidêmico, que por suas características consome mais recursos da rede.

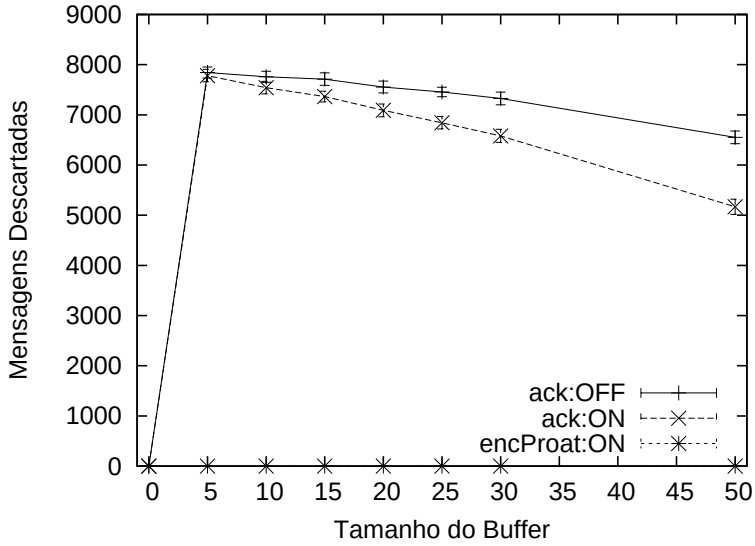


Figura 4.3: Mensagens descartadas - Baixa velocidade.

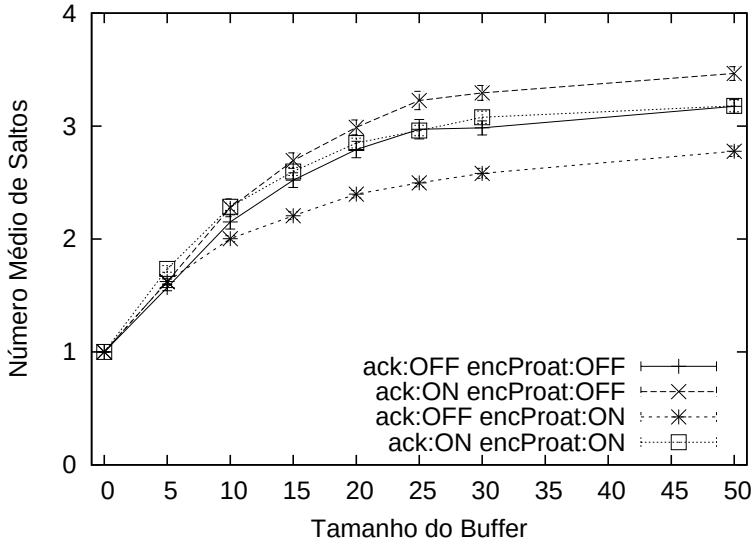


Figura 4.4: Número médio de saltos - Baixa velocidade.

O melhor resultado obtido na taxa de entrega, Figuras 4.1 e 4.6, foi com a combinação dos dois mecanismos, lista de *acks* e encaminhamento proativo. Pois um mecanismo complementa o outro, enquanto um realiza uma "limpeza" nos *buffers*, retirando de circulação mensagens entregues, o outro evita descartes desnecessários.

A lista de *acks* sozinha apresenta maior impacto com valores de *buffer* maiores, pois a probabilidade do nó encontrar outro que possua uma mensagem já entregue é maior. A lista demonstra melhora na taxa de entrega, pois a "limpeza" dos

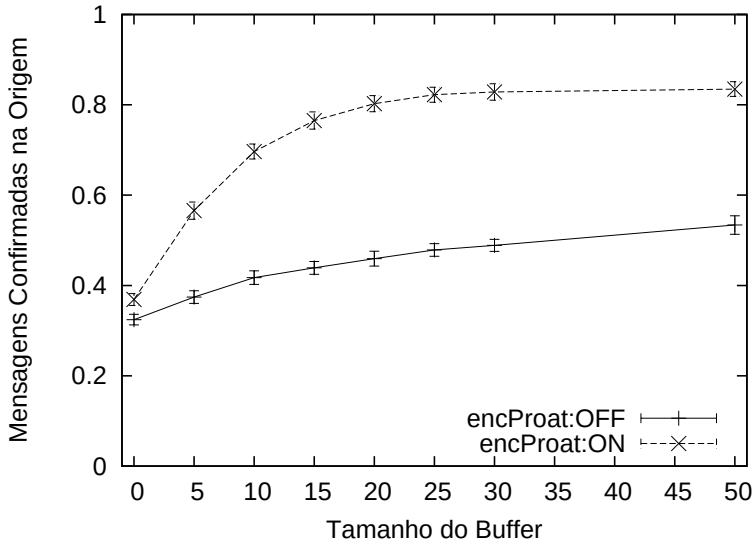


Figura 4.5: Mensagens Confirmadas na origem - Baixa velocidade.

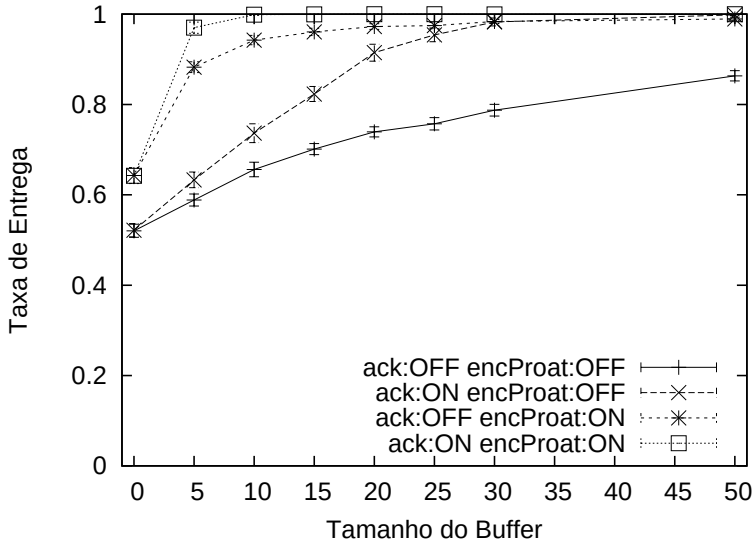


Figura 4.6: Taxa de Entrega - Alta velocidade.

buffers favorece a troca somente de mensagens importantes, porém quando utilizada sozinha ainda apresenta um número grande de mensagens transferidas entre os nós, Figuras 4.2 e 4.7.

O encaminhamento proativo obteve melhor resultado na taxa de entrega que a lista de *acks*, pois impediu que o nó permanecesse muito tempo com um único contato, perdendo oportunidades de troca de mensagens com outros vizinhos. Como cada nó só se comunica com um único nó por vez e, o encaminhamento proativo

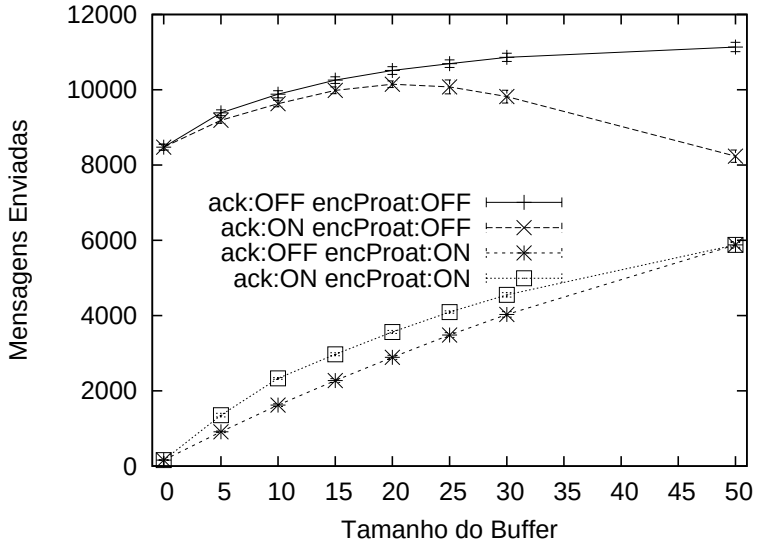


Figura 4.7: Mensagens enviadas - Alta velocidade.

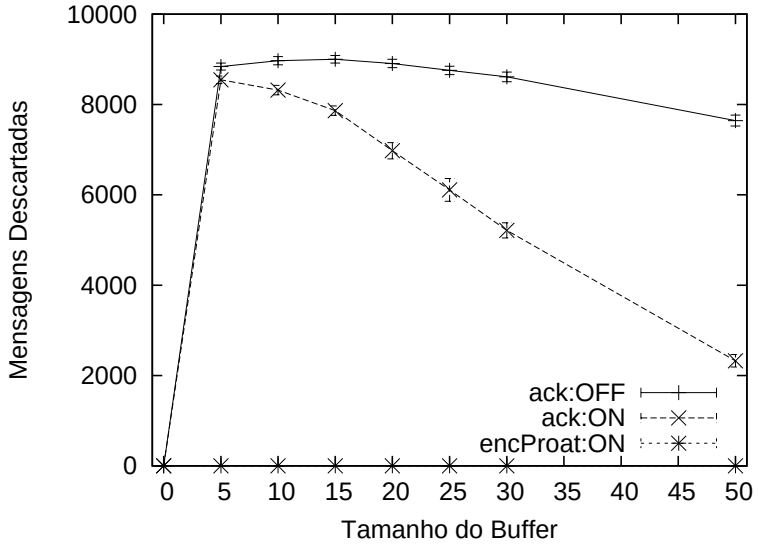


Figura 4.8: Mensagens descartadas - Alta velocidade.

evita que o nó emissor envie mensagens a um outro nó que esteja com o *buffer* cheio, este mecanismo proporciona que o nó transmissor possa ter mais oportunidades de transmissão de mensagens a nós que realmente possam armazenar novas mensagens. Porém, como dito anteriormente, ele pode afetar a disseminação rápida de mensagens pela rede.

A partir das Figuras 4.2 e 4.7 ,vemos que o número de mensagens enviadas com o uso da lista de *acks* e o encaminhamento proativo diminui bastante. Isto

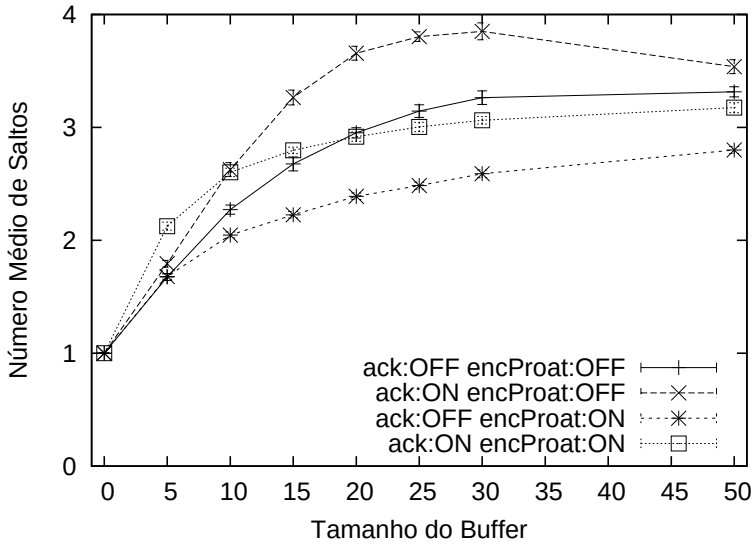


Figura 4.9: Número médio de saltos - Alta velocidade.

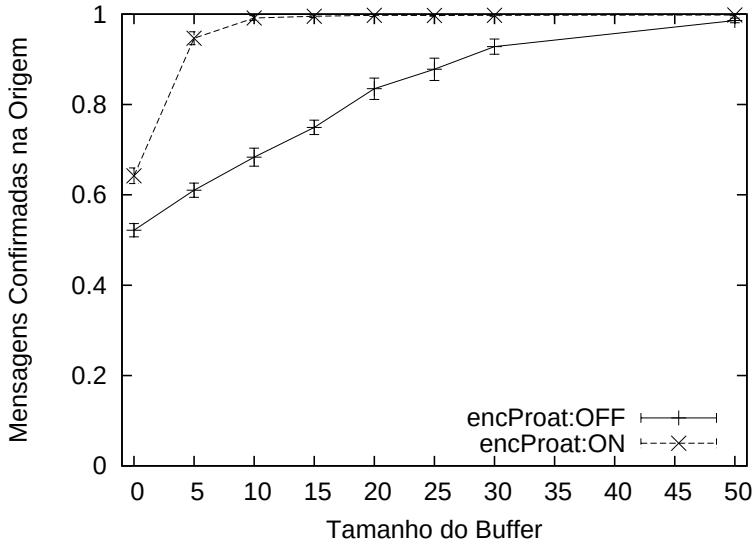


Figura 4.10: Mensagens Confirmadas na origem - Alta velocidade.

ocorre, pois o uso de *acks* indica quais mensagens podem ser apagadas de maneira segura dos nós intermediários, o que diminui a quantidade de mensagens em *buffer* e evita transmissões desnecessárias. Além disso, o encaminhamento proativo permite que um nó só envie o número de mensagens que o seu par suporta, diminuindo a quantidade de mensagens enviadas. Com as políticas desligadas, quando o tamanho do *buffer* aumenta, ocorre um aumento no número de transmissões, pois o espaço possível de armazenamento é maior e o vetor de sumários (*Summary Vector*) pode

conter mais mensagens. Quando utilizamos somente a lista de *acks*, com o aumento da capacidade de armazenamento, mais facilmente os *buffers* tendem a se tornarem "espelhados" e por isso o número de transmissões decai.

O número médio de saltos das mensagens entregues ao destinatário pode ser visto no gráfico das Figuras 4.4 e 4.9. Com o uso apenas do encaminhamento proativo evita-se transmissões quando o *buffer* não compotar mais nenhuma mensagem, logo um menor número de encaminhamentos, ou saltos, das mensagens pode ser verificado. Quando utilizamos a lista de *acks*, favorecemos a uma maior quantidade de transmissões, logo um maior número de saltos nas mensagens entregues pode ser notado. Vale ressaltar que com a lista de *acks* temos uma taxa de entrega maior e o gráfico plota o número médio de saltos das mensagens entregues. Com os dois mecanismos ativos, evitamos transmissões com o *buffer* cheio, porém somente depois de verificar se alguma mensagem já foi entregue, logo na média, o número de saltos é maior que o caso onde apenas o encaminhamento proativo está acionado.

Os gráficos das Figuras 4.3 e 4.8 apresentam os resultados quanto a quantidade de mensagens descartadas. O número de descartes diminui com o uso da lista de *acks* pois como ocorre uma "limpeza" sobre as mensagens que já foram entregues, aproveita-se mais do espaço de armazenamento para as mensagens que chegam ao nó. Com o aumento do tamanho da capacidade de armazenamento dos nós, temos a tendência do *buffer* dos nós tornarem-se "espelhados", com isso menos transmissões são realizadas e consequentemente menos descartes ocorrem. Quando o tamanho do *buffer* é restrito, menos mensagens são entregues para o mesmo tempo de simulação, logo menos mensagens são inclusas na lista de *acks*, daí sua menor contribuição em termos de redução no número de descartes. A curva com o encaminhamento proativo acionado foi utilizada apenas para comprovar que não ocorriam descartes. As formas de descarte FIFO ou MOFO apresentaram praticamente os mesmos resultados, com sobreposição dos intervalos de confiança. Todos os gráficos das Figuras 4.1 a 4.10 foram obtidos com a forma de descarte MOFO. O número de descartes para o caso de *buffer* com tamanho zero foi nulo, pois uma vez que o nó não possui espaço para armazenar nenhuma mensagem, senão a própria mensagem gerada por ele, não ocorrerem descartes. Vale explicar que, da forma implementada, os nós possuíam dois *buffers*, um para as mensagens geradas por ele próprio (*source buffer*), que

era ilimitado, e outro que guardava as mensagens repassadas por outros nós (*relay buffer*), este limitado. Neste trabalho, todas as vezes que mencionarmos apenas a palavra *buffer* estamos nos referindo ao *relay buffer*. Nos gráficos, o eixo **x** representa o tamanho do *relay buffer*. O objetivo do valor de *buffer* igual a zero, testado na simulação, foi verificar a taxa de entrega apenas pelo contato direto do nó receptor com o nó de destino.

A partir dos gráficos de mensagens confirmadas na origem, Figuras 4.5 e 4.10, podemos constatar que o uso da lista de *acks* em conjunto com o encaminhamento proativo proporcionou que mais confirmações alcançassem os nós fontes.

4.2 Políticas para Baixas Taxas de Transmissão

Outra grande dificuldade encontrada pelos protocolos de roteamento é qual mensagem priorizar no instante de envio. Quando dois nós estão no alcance de contato e em condições de transmitir ou receber, o desejável é que cada um deles obtenha mensagens ainda não vistas. No entanto, o tempo desse contato pode ser curto em relação ao tempo despendido na transmissão de cada mensagem, tornando-se impossível a transmissão de todas as mensagens desejadas. Desta forma, a difusão das mensagens pela rede fica prejudicada e aumenta-se o tempo necessário para sua entrega. Uma série de fatores influenciam no envio das mensagens, entre eles: o padrão de mobilidade dos nós da rede [53], a capacidade de transmissão dos nós e o tamanho das mensagens.

Para contornar o problema, utilizam-se políticas de prioridade de encaminhamento, que indicam ao nó qual mensagem deve ser prioritária no instante de envio. Isto faz com que haja uma garantia de que as mensagens com maior importância, como por exemplo, as destinadas aquele nó receptor ou as menos enviadas, sejam transmitidas primeiro aumentando a possibilidade de serem entregues durante aquele contato [65].

Alguns exemplos de política de prioridade de encaminhamento são: a política LEFO (*LEss FOrwarded*) e a LHOC (*Less HOp Count*). Existem outros diversos tipos de políticas, como as que levam em conta o tempo de permanência da mensagem nos nós intermediários, o tempo de expiração da mensagem, o tempo em que a

mensagem foi criada, etc.. Neste trabalho trataremos apenas das políticas LEFO e LHOC, apresentadas a seguir.

Ao utilizarmos a política LEFO enviamos as mensagens por ordem decrescente do número de vezes que o nó transmitiu determinada mensagem. O intuito é de aumentar a propagação das mensagens pela rede, pois o número de encaminhamentos de cada mensagem é uma característica local de cada nó, assim o nó ter um número menor de encaminhamentos para determinada mensagem indica que o próprio nó contribuiu pouco para sua aparição pela rede.

A política LHOC procede de forma similar à LEFO, porém a prioridade de envio está relacionada à mensagem com menor número de saltos, o que trata-se de uma informação própria de cada mensagem. Enquanto que a política LEFO leva em conta uma contagem local de cada nó, a política LHOC trabalha com uma característica global. Caso o nó emissor possua alguma mensagem destinada ao nó receptor esta mensagem é a primeira a ser enviada independente da política de prioridade. Tanto a LEFO como a LHOC são normalmente aplicadas a roteamentos baseados em replicação.

Implementamos estas duas políticas em etapas, a saber: maior prioridade às mensagens destinadas ao próprio nó receptor; depois aplicamos as particularidades de cada política, ocorrendo empate de várias mensagens com o mesmo número de encaminhamentos ou mesmo número de saltos, LEFO e LHOC, respectivamente, priorizamos dentre estas à(s) originada(s) pelo próprio nó emissor, caso o empate ainda persista, escolhe-se a mensagem mais antiga do *buffer* (FIFO). Por questões de clareza enumeramos a ordem de prioridades utilizada, abaixo:

Ordem de prioridades

1. Destinadas ao nó receptor
2. Mensagens menos transmitidas ou com menor número de saltos
3. Caso de empate → Mensagem originada pelo próprio nó transmissor
4. Mensagem mais antiga (FIFO)

Para a simulação escolhemos dois cenários. No primeiro, os nós tinham velocidade de 10 m/s e 2 s de pausa, no outro cenário os nós movimentavam-se com

velocidade 3 m/s e pausa de 10 s. Nos dois ambientes, cada nó gerou, no início da simulação, uma mensagem para outro nó distinto, totalizando 50 mensagens na rede. A ideia de dois cenários com velocidades médias diferentes era de verificar a influência do tempo de contato entre os pares transmissores e receptores no uso das duas políticas.

Nestes dois cenários simulamos um ambiente sem descarte, ou seja valor de *buffer* igual a 50 e, outro ambiente com valor de *buffer* igual a 10 (com descarte). No caso do *buffer* igual a 50 a restrição de memória não influenciou o resultado. Com o *buffer* limitado para 10 mensagens alguns nós tiveram que deletar mensagens, o intuito era tornar mais importante qual mensagem priorizar no instante de contato.

O valores do eixo **x** designados por valores de *snapshots*, indicam variações no valor das taxas de transmissão. Estes valores são próprios da modelagem empregada e conforme apresentado no Capítulo 3, quanto maior os valores no eixo **x** menores são as taxas de transmissão empregadas. As políticas de encaminhamento foram testadas no roteamento Epidêmico apenas, uma vez que implementadas no PROPHET alterariam seu funcionamento. Os resultados podem ser vistos a seguir, Figuras 4.11 a 4.13.

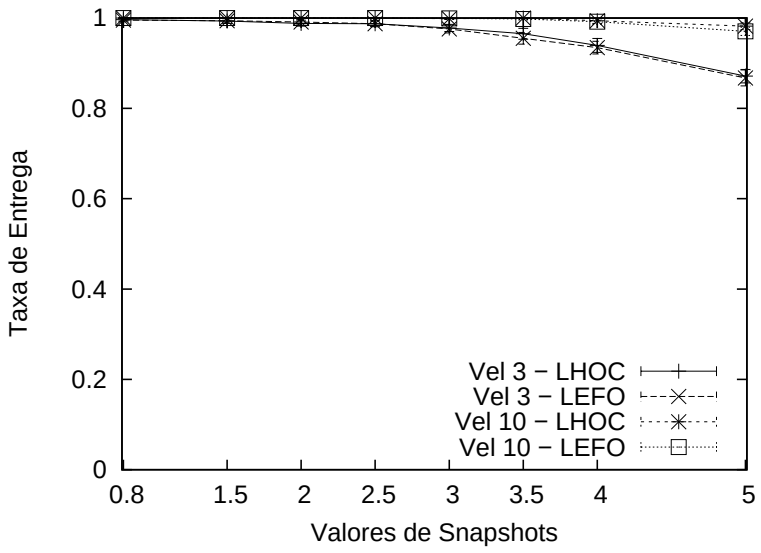


Figura 4.11: Taxa de entrega - Valor de *buffer* = 50.

Em todos os casos as duas políticas tiveram praticamente as mesmas taxas de entrega, Figuras 4.11 e 4.13. Nestes dois cenários a restrição de *buffer* teve um impacto maior do que as baixas taxas de transmissão.

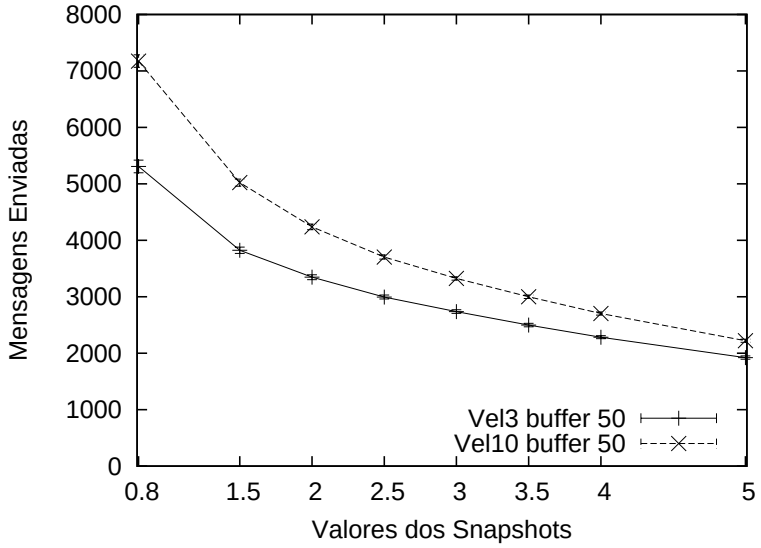


Figura 4.12: Número de mensagens enviadas - Valor de *buffer* = 50.

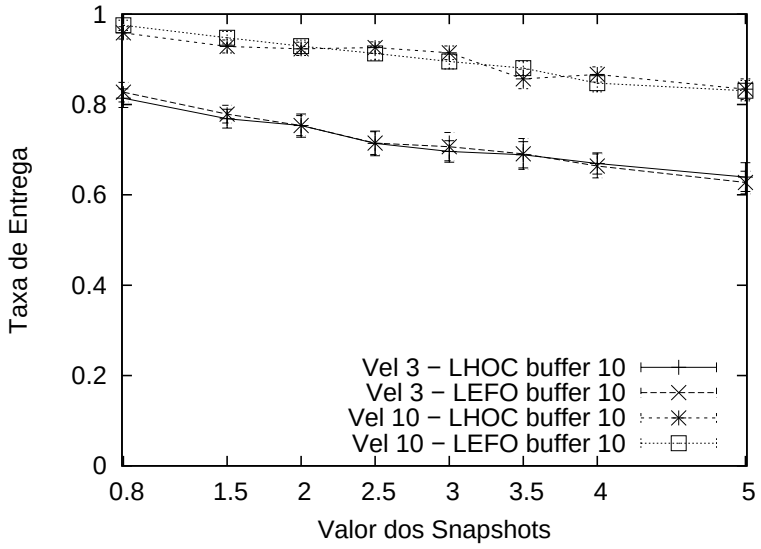


Figura 4.13: Taxa de entrega - Valor de *buffer* = 10.

Embora o tempo de contato tenha diminuído, devido ao aumento da velocidade do nó, a taxa de entrega no cenário onde os nós se movimentavam mais, foi maior. Isto porque o número de interações entre os nós foi maior. Este fato pôde ser constatado pelo número de encaminhamentos feitos pelos nós no cenário de alta velocidade, Figura 4.12. A partir da quantidade de encaminhamentos percebemos que os nós tiveram um maior número de encontros no cenário de alta mobilidade do que no de menor mobilidade. Vale ressaltar que em tal simulação o espaço em *buffer*

era suficiente para que não ocorressem descartes, ou seja, neste caso o número de encaminhamentos representa o número mínimo de vezes que os nós se encontraram, pois os encontros em que não geraram transmissões (os dois nós com as mesmas mensagens) não foram contabilizados.

Assim, para este cenário de testes e para o tipo de implementação feito, o número maior de encontros mostrou-se mais relevante que a ordem no envio das mensagens.

4.3 Modos de Gerenciamento de Energia

Nos dispositivos móveis um dos maiores consumidores de energia é a interface sem fio. Além do gasto inerente à comunicação (Tx e Rx), a interface sem fio também consome razoável energia em períodos de "escuta" (*listening*) por transmissões de outros nós.

Como redes RTAIs têm um forte potencial de uso em dispositivos móveis, onde normalmente a fonte de energia é limitada, torna-se relevante a procura por poupar energia em períodos onde não haja possibilidade de comunicação. Pois mesmo que existam nós com grande quantidade de energia, em uma rede heterogênea, podemos encontrar dispositivos cuja reserva energética seja ou esteja menor que de outros [60].

Cada nó pode operar em quatro modos: transmitindo, recebendo, ouvindo ou "dormindo". Que correspondem a quatro tipos de gastos de energia distintos [66]. Em geral, os tipos de gerenciamento atuam sobre os modos de operação dos dispositivos, chaveando os nós para o estado "dormindo" caso não haja nenhum vizinho habilitado ou disponível para comunicação. Deste modo, eficientes mecanismos de gerenciamento de energia podem permitir que os nós componentes permaneçam operacionais por um maior período de tempo.

Ordenando os modos de operação por gasto energético temos: o maior gasto transmitindo, depois recebendo, ouvindo e por fim "dormindo". Embora para comunicações sem fio de curto alcance, como em redes de sensores, o gasto na interface, por ouvir e receber seja bem próximo [66]. Sobre o consumo no estado "dormindo", este é relacionado à manutenção ou funcionamento do circuito do dispositivo.

Combinado com a tendência natural dos nós de uma RTAI apresentarem-se dispersamente conectados, uma quantidade relevante de energia pode ser poupada de-

sabilitando os rádios/interface dos nós em períodos sem comunicação. Porém por ser uma rede oportunista o maior desafio é balancear o período do nó permanecer ativo com o instante de contato, pois perder uma oportunidade de comunicação em tais redes pode prejudicar severamente seu desempenho.

Em ambientes onde o número de contatos do nó é alto, o mecanismo de gerenciamento deve ser robusto para permitir que o nó não perca muitas oportunidades de contato. De modo geral, em cenários determinísticos, mecanismos ou estratégias de gerenciamento de energia são menos complexos de serem aplicados, pois pode-se recorrer a uma maior quantidade de informação que indique o melhor instante para o nó desabilitar sua interface, sem prejudicar o desempenho da rede.

A taxa de entrega, com o uso de gerenciamentos, possui forte relação com o tempo operacional do nó, pois é de se esperar que um nó que poupe mais energia permaneça em funcionamento por um período maior de tempo e, eventuais oportunidades de entrega possam surgir posteriormente. Como consequência, o atraso na entrega de mensagens pode ser maior.

Alguns modos de gerenciamento foram testados e, um breve resumo deles será apresentado na Tabela 4.1 a seguir, maiores detalhes de cada um serão apresentados em conjunto com seus respectivos resultados.

Os modos de gerenciamento implementados têm por objetivo não dependerem do tipo de movimentação dos nós nem de informações coletadas previamente. Desta forma foi feita uma avaliação de tipos simples de gerenciamento que a princípio podem ser aplicados a quaisquer ambientes. O tipo de modelo de mobilidade dos nós foi o *Random Waypoint* onde cada nó possuía velocidade máxima de 3 m/s e pausa de 10 s, utilizamos 50 nós, com 50 mensagens diferentes trafegando pela rede. Todos os nós iniciaram a simulação com a mesma quantidade de energia, e todas as mensagens foram geradas no início das simulações. Não computamos gasto com funcionamento do circuito, indicado pelo gasto no estado "dormindo", levamos em consideração apenas o gasto com relação a interface de comunicação. O gasto com chaveamento de ligar e desligar à interface sem fio também não foi computado.

As simulações foram realizadas em duas partes. Na primeira, comparamos cada tipo de gerenciamento apenas com o caso sem gerenciamento, em situações de duas quantidades de energia inicial distintas. Na segunda parte, realizamos a comparação

entre alguns tipos de gerenciamento selecionados e, adicionamos simulações complementares, por exemplo: com diferentes valores de alcances dos nós e quatro quantidades de energia inicial, conforme explicado a seguir.

Um dos parâmetros das simulações foi a quantidade de energia inicial E em cada nó, onde especificamos quatro valores iniciais. Na primeira situação, a energia inicial era suficiente para que todos os nós terminassem a simulação com alguma reserva de energia, ou seja, nenhum nó "morria" no decorrer da simulação. Em uma situação posterior, a energia inicial foi tal que o tempo de vida da rede[1] fosse de aproximadamente 50%, isto sem o uso de qualquer gerenciamento. Assim denominamos essa quantidade de energia inicial de $E/2$. A energia denominada de $E/4$ refere-se a quantidade $E/2$ dividida por 2, o mesmo aplica-se a $E/8$ com relação a $E/4$. Na primeira parte das simulações utilizamos apenas as quantidades E e $E/2$, as demais foram utilizadas na segunda parte dos testes.

Gerenciamentos	**Funcionamento**
G1	Nó sem contato "dorme" por um tempo T, porém o tempo no estado "dormindo" é crescente
G2	Nó emissor coleta informações do receptor e calcula o tempo de dormir
G3	Nó "dorme" depois que percebe o meio ocupado por n vezes consecutivas
G4	Existe uma probabilidade que indica a possibilidade do nó ir "dormir" ou não
G5	Associa-se uma porcentagem do quanto "dormir", calculada pelo número de mensagens do nó e seu espaço em *buffer*

Tabela 4.1: Modos de Gerenciamento de Energia

Conforme o modo de gerenciamento G1, toda vez que o nó está sem vizinhos passa-se ao modo "dormindo". Porém este tempo de dormir T é aumentado de uma unidade a cada vez que a varredura por contatos for malsucedida. Assim que o nó consegue transmitir ou receber dados o tempo de dormir, acumulado ou não, retorna

[1]Definimos como tempo de vida da rede o instante em que o primeiro nó da rede "morre".

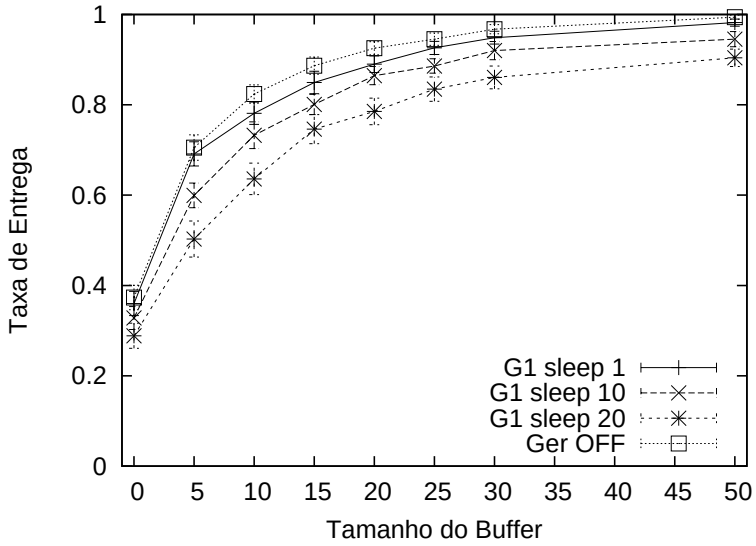


Figura 4.14: Taxa de entrega - Energia E - G1.

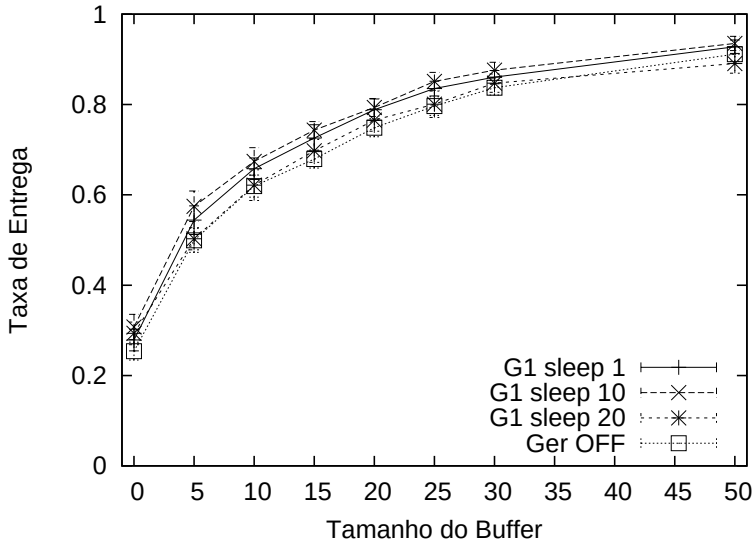


Figura 4.15: Taxa de entrega - Energia $E/2$ - G1.

ao valor inicial. Por exemplo, o nó **a** está habilitado a se comunicar porém na fase de descoberta de vizinhança o nó verifica que não possui vizinhos, logo passa para o modo "dormindo", no qual permanecerá por T segundos como informado no início da simulação. Após o término dos T segundos o nó "acorda", e em devida oportunidade realiza a varredura pela descoberta de vizinhos. Caso novamente o nó não possua vizinhos ele passa ao estado "dormindo", porém o tempo no qual permanecerá neste estado será agora de $T + 1$ segundos. Se posteriormente ocorrer outra tentativa sem

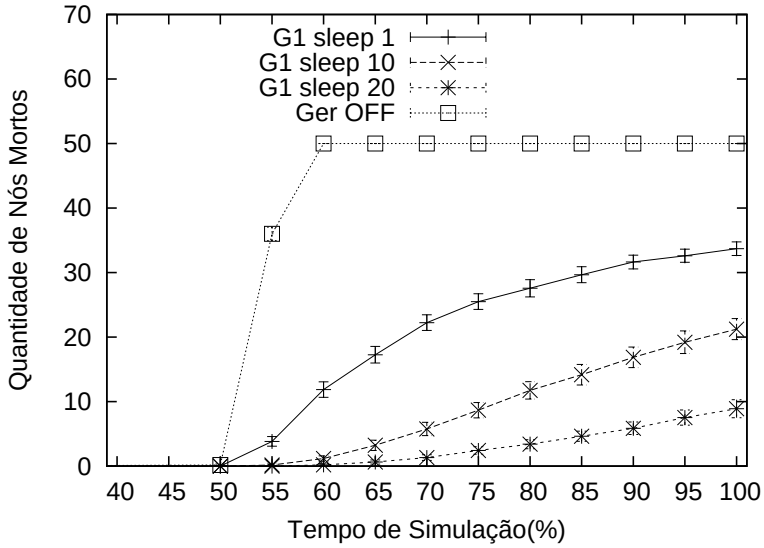


Figura 4.16: Núm. de nós mortos - Energia $E/2$ - G1.

êxito na descoberta de vizinhança, o tempo passará para $T + 2$ segundos e assim sucessivamente. Caso este nó consiga, em algum instante, receber ou transmitir alguma mensagem o valor da duração no estado "dormindo" retorna ao valor T dado inicialmente.

Este gerenciamento foi testado com três valores fixos de T, a saber: T pôde assumir os valores 1, 10 ou 20 segundos. A partir dos resultados podemos constatar que o modo de gerenciamento G1, Figura 4.14 situação de energia inicial suficiente E, obteve melhor resultado apenas com tempo de dormir de 1 segundo, pois para este tipo de gerenciamento o aumento do tempo de dormir, não favoreceu a taxa de entrega. Ou seja, para esta estratégia se o tempo de dormir acumulado for grande, acontece com maior frequência o caso do nó perder oportunidades de contato, o que se reflete em uma menor taxa de entrega. Na situação em que os nós podiam "morrer" ($E/2$), na Figura 4.15, os resultados de taxa de entrega foram próximos do gerenciamento desligado, obstante que foi preservada uma maior quantidade de nós ativos ao fim, Figura 4.16. Na situação de $E/2$, a taxa de entrega tornou-se satisfatória pois como menos nós "morriam", as chances de entrega ainda se tornavam possíveis mesmo que os nós permanecessem por um longo período com sua interface desligada.

O modo de gerenciamento G2 tenta coletar alguma informação para calcular o tempo em que o nó irá "dormir" quando estiver sem vizinhança. No instante de

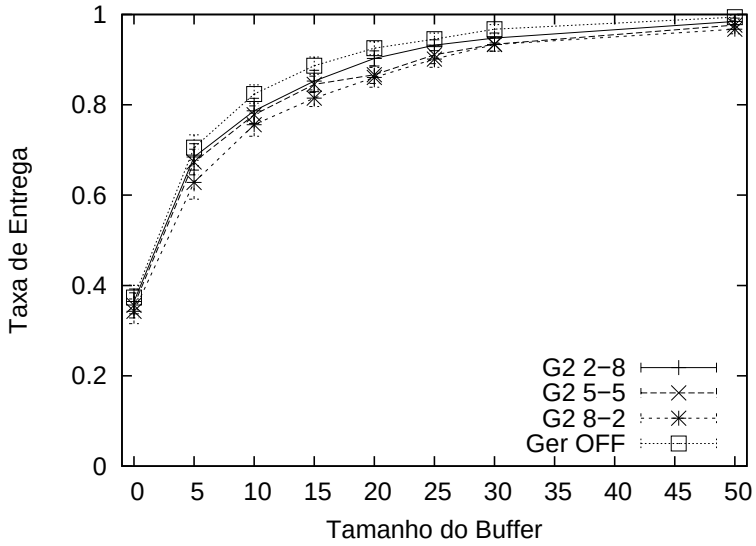


Figura 4.17: Taxa de entrega - Energia E - G2.

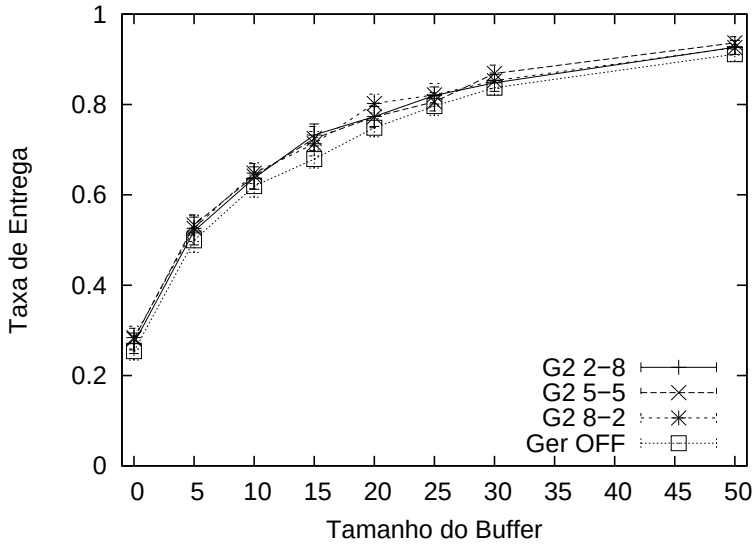


Figura 4.18: Taxa de entrega - Energia $E/2$ - G2.

contato, os nós transmissor e receptor, trocam duas informações: a quantidade de vizinhos da última vez que realizaram a descoberta de vizinhança, e o instante em que coletou-se tal informação. A partir disto, o nó calcula o valor de T conforme a Equação 4.1.

$$T = ceil((now - instant) \times P1 - NumViz \times P2) \tag{4.1}$$

Da Equação 4.1, *now* é o instante em que o nó percebe que está sem vizi-

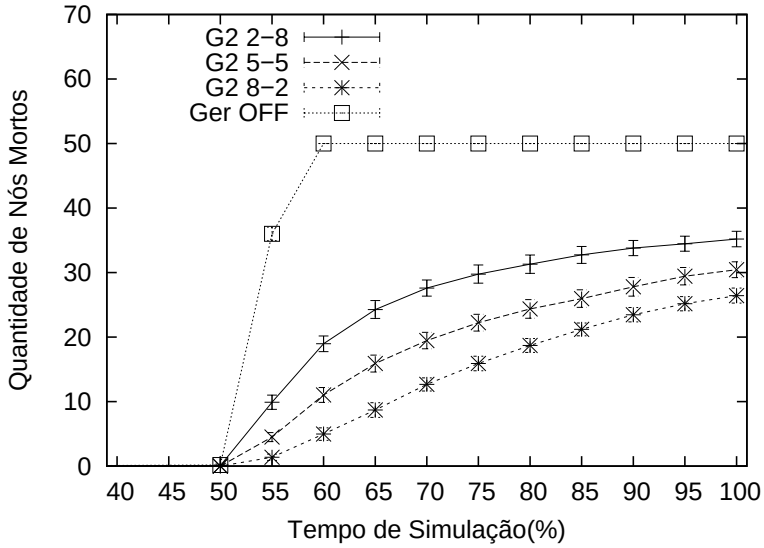


Figura 4.19: Núm. de nós mortos - Energia $E/2$ - G2.

nhos, *instant* indica quando a informação de quantidade de vizinhos foi coletada e $NumViz$ informa a quantidade de vizinhos que o nó par de comunicação obteve na última varredura. Os valores $P1$ e $P2$ são pesos dados a cada uma das informações. Caso o nó encontre um par de comunicação que não tenha nenhuma informação do número de contatos anteriores, e este nó transmissor venha a ficar sem vizinhos, seu funcionamento torna-se igual ao gerenciamento G1, caso idêntico quando o nó estiver sem nenhum contato. Caso o valor calculado de T seja negativo, o nó não deve "dormir", visto que a informação ou é recente ou a quantidade de vizinhos de seu último contato foi alta. Além disso foi fixado um valor máximo (L) para o tempo do nó permanecer no estado "dormindo". Seu funcionamento foi resumido abaixo no Algoritmo 1.

A ideia do cálculo de tempo deste gerenciamento era levar em consideração a atualidade da informação e, o número de vizinhos que o nó par de comunicação (receptor) teve na última varredura por contatos. Quanto mais recente e maior a quantidade de vizinhos, menor deve ser o tempo do nó (transmissor) "dormir", pois há grande chance dos vizinhos de seu último nó receptor ainda estarem próximos. No gerenciamento G2, para $P1$ e $P2$ foram utilizados os valores: (0.2, 0.8); (0.5, 0.5) e (0.8, 0.2). Estes pares de pesos estão indicados nas figuras como 2-8 referente

ao par (0.2 e 0.8); 5-5 representando (0.5 e 0.5) e (0.8 e 0.2) indicado por 8-2.

```
if Número de vizinhos == 0 then
    Ativo gerenciamento G2;
    Calculo Tempo de Dormir conforme Equação 4.1
    if Tempo de Dormir < 0 then
        Nó não deve "dormir" → Não ativo gerenciamento
    else if Tempo de Dormir > L then
        Tempo de Dormir = L
    else
        Aplico Tempo de Dormir calculado
else
    Não ativo gerenciamento
```

Algoritmo 1: Algoritmo do gerenciamento G2

Pelos resultados de G2, Figura 4.17, temos taxas de entrega satisfatórias nas duas situações de energia inicial, aliado a uma redução do número de nós "mortos" ao fim da simulação. O número de transmissões também apresentou uma redução, porém os nós permaneciam um tempo grande no estado ouvindo. A informação que mais influenciou no cálculo de T foi a do instante de coleta da informação, a parametrização de G2 que aplicava um peso maior a esta informação (2-8) o número de vezes que os nós "dormiram" foi maior. Um diferencial interessante é que neste modo de gerenciamento, quando o nó consegue alguma informação de seu nó par de comunicação, o tempo de dormir é calculado individualmente para cada nó, o que o torna mais flexível. O valor para L utilizado nas simulações foi de 20 segundos. Foram feitas algumas simulações retirando a parte de funcionamento G2, ou seja, utilizando exclusivamente o funcionamento de G2. Estas simulações apresentaram resultados semelhantes, com exceção do número de nós mortos maior ao fim da simulação para o caso de uso exclusivo de G2.

Os modos de gerenciamento apresentados, em grande maioria, os nós ainda apresentavam um elevado número de vezes no estado ouvindo. Desta forma segundo G3, quando nó verifica n vezes consecutivas que o meio está ocupado passa-se ao estado "dormindo". Para os períodos de desligamento da interface foram testados os valores 3, 4 e 5 segundos. Valores menores que os dos outros modos de gerenciamento que necessitam de igual parâmetro de entrada, pois se o meio está ocupado o nó

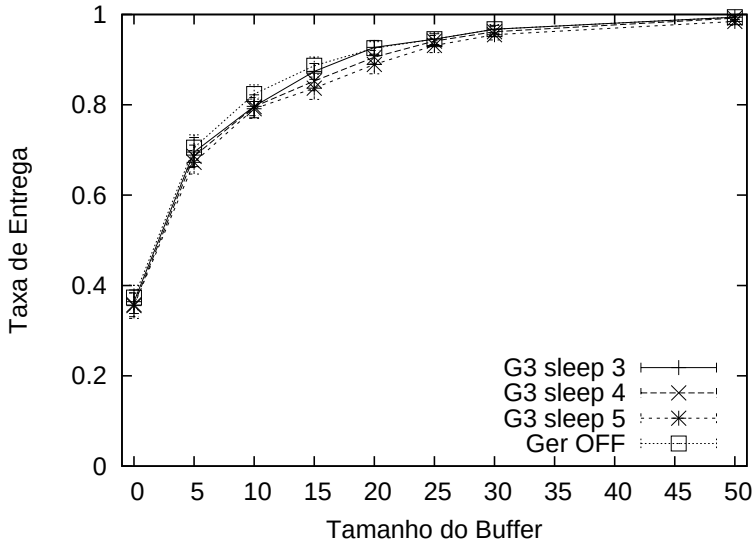


Figura 4.20: Taxa de entrega - Energia E - G3.

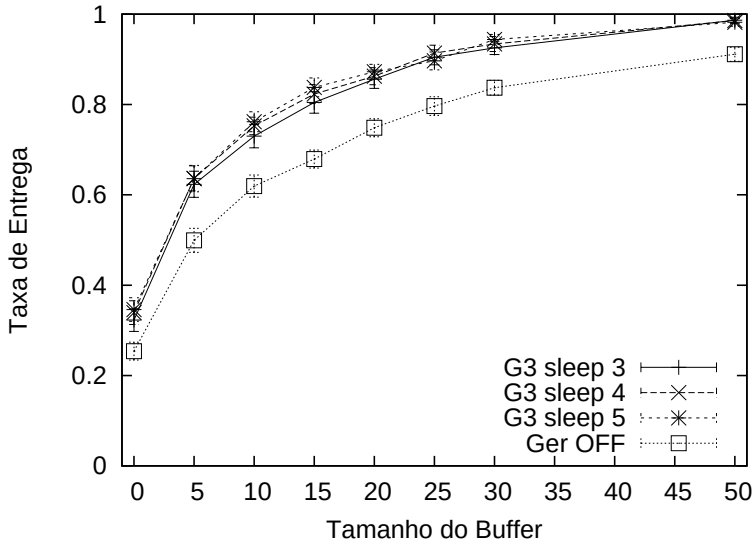


Figura 4.21: Taxa de entrega - Energia $E/2$ - G3.

deve "dormir" pouco.

Segundo os gráficos da Figura 4.20, podemos verificar que o gerenciamento G3 apresentou taxas de entrega satisfatórias nas duas situações de energia inicial. Isto devido aos gerenciamentos anteriores, em algumas situações apesar de reduzirem os números de transmissões, o número de vezes que o nó permaneceu no estado ouvindo ainda era expressivo. O gerenciamento G3 conseguiu reduzir o número de transmissões e o número de vezes que o nó permaneceu ouvindo, assim o tempo de

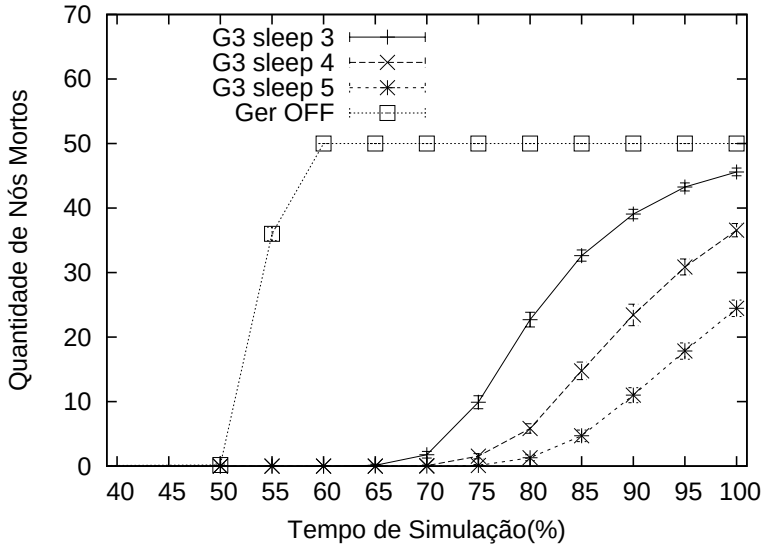


Figura 4.22: Núm. de nós mortos - Energia $E/2$ - G3.

vida da rede foi maior, por isso uma maior taxa de entrega na situação de energia inicial reduzida ($E/2$). Com um tempo de vida da rede maior o número de oportunidades de entrega obteve um aumento. O número de vezes que o nó teve de ouvir consecutivamente para passar para o estado "dormindo", ou seja parâmetro n, foi fixado em 3. Este parâmetro n também foi testado com valor 2, porém deste modo os nós passavam para o estado "dormindo" um maior número de vezes e a taxa de entrega apresentou uma queda comparada ao caso de $n = 3$.

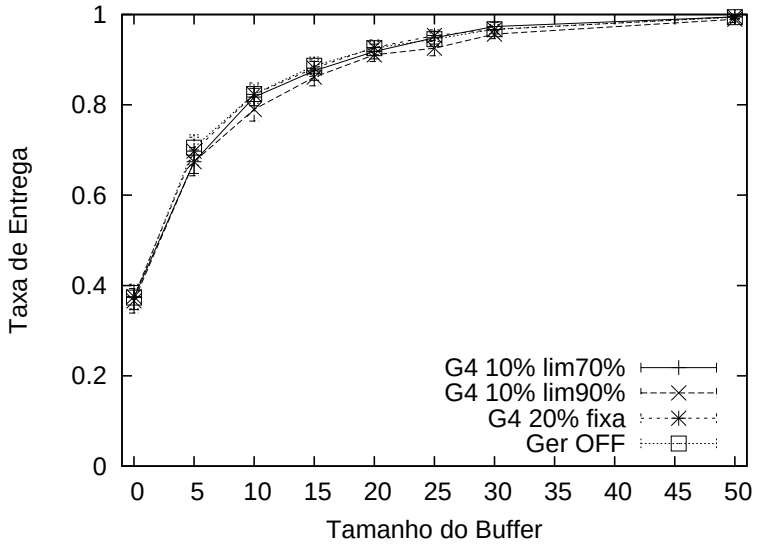


Figura 4.23: Taxa de entrega - Energia E - G4.

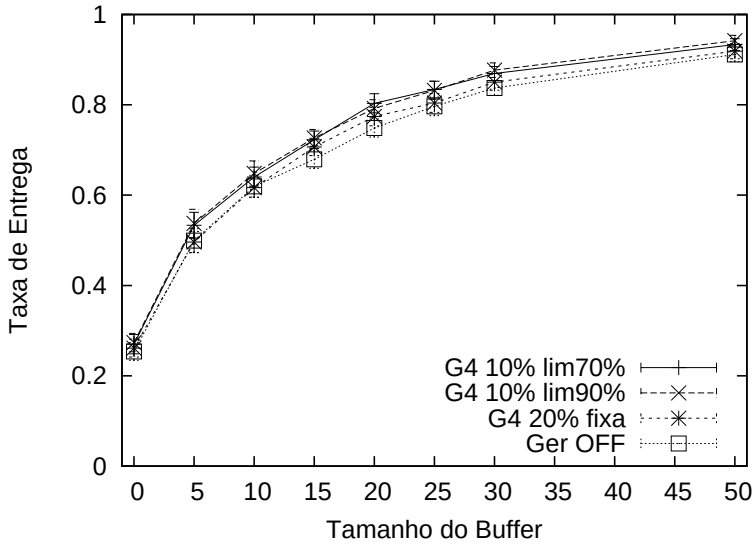


Figura 4.24: Taxa de entrega - Energia $E/2$ - G4.

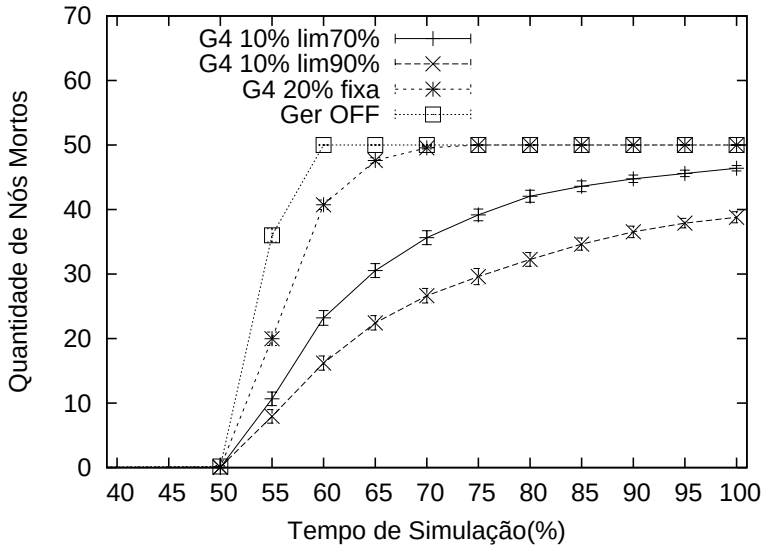


Figura 4.25: Núm. de nós mortos - Energia $E/2$ - G4.

A partir do modo de gerenciamento G4, quando o nó está sem vizinhos lança-se uma probabilidade p afim de decidir se o nó passa ao estado "dormindo" ou não. Uma vez "dormindo", de tempos em tempos, o nó testa novamente esta probabilidade com o intuito de verificar se continua neste estado ou acorda. Caso o nó acorde e não possua vizinhos, sua probabilidade p é então aumentada de δ e mais uma vez lança-se esta nova probabilidade para determinar a escolha de seu estado. Se o nó acorda e possui vizinhança o valor de p não se altera e sua interface sem fio perma-

nece ativa. Quando em algum instante o nó recebe ou transmite alguma mensagem, sua probabilidade p acumulada ou não, retorna ao valor inicial. Devido o fato do nó poder atingir um estado em que a probabilidade de "dormir" seja máxima, e desta forma permaneça "dormindo" até o fim da simulação, um limite superior L ao valor da probabilidade foi fixado no início da simulação. Este limite foi variado assim como o valor do quanto de incremento à probabilidade (δ). Este gerenciamento teve por iniciativa não depender do parâmetro de entrada T.

A saber, δ pôde assumir os valores: 10%, 20%, 30%, 40% e os limites testados foram de 70%, 80% e 90%. As combinações apresentaram valores da taxa de entrega, em situações de energia E e $E/2$, próximos do caso sem gerenciamento inclusive com sobreposição de intervalos de confiança. Na Figura 4.23, apresentamos apenas os resultados para incremento de 10% com limites de 70% e 90% e um outro caso com p fixo em 20%. A diferença maior foi na quantidade de nós mortos ao fim da simulação, conforme aumentávamos o valor de δ menos nós morriam ao fim, porém a quantidade praticamente não se alterou com o aumento de $\delta = 30\%$ para 40%. O limite superior tem influência sobre o retorno do nó ao estado acordado, assim conforme aumenta-se os valores limites mais vezes os nós permanecem "dormindo". O valor que mais influenciou foi justamente o limite superior pois o valor de δ representa o quão rápido pôde o nó atingir o limite superior de probabilidade.

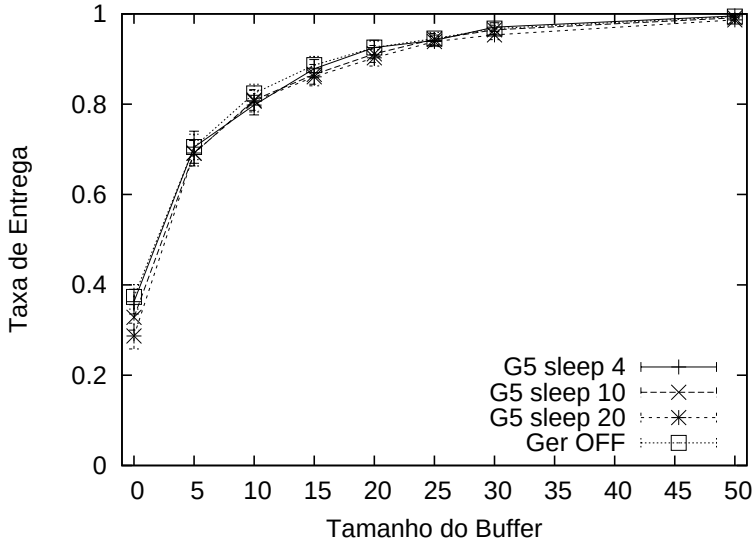


Figura 4.26: Taxa de entrega - Energia E - G5.

Conforme modo de gerenciamento G5, calcula-se uma porcentagem do valor de T

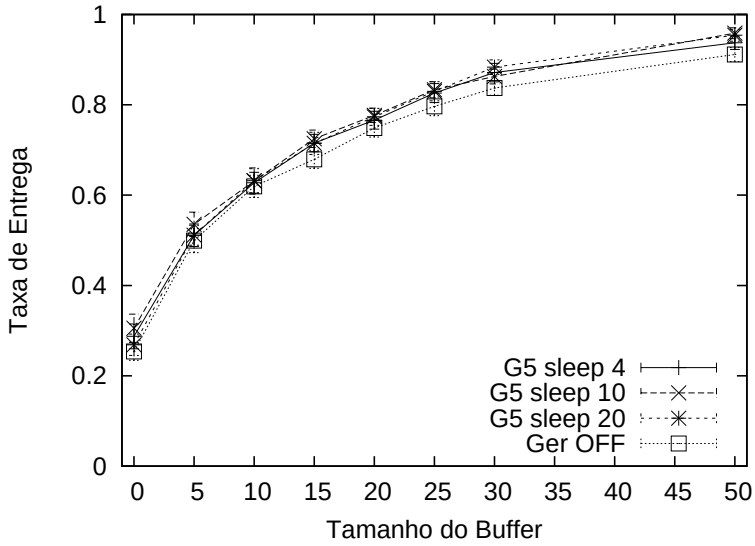


Figura 4.27: Taxa de entrega - Energia $E/2$ - G5.

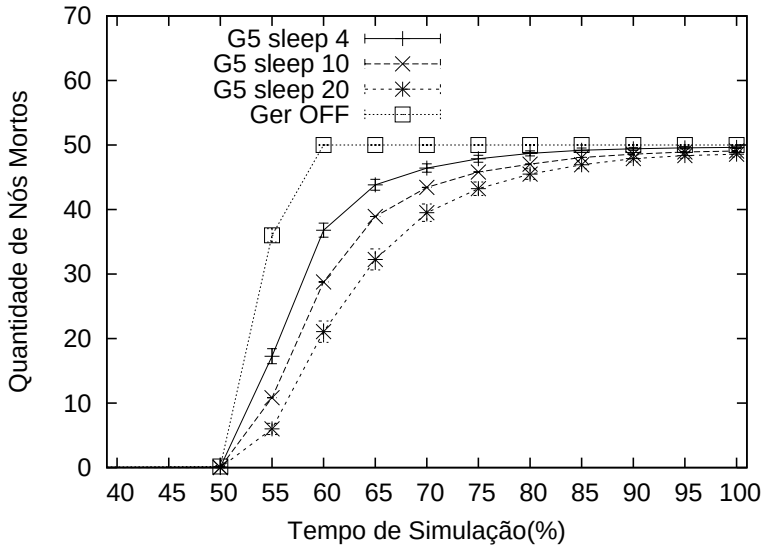


Figura 4.28: Núm. de nós mortos - Energia $E/2$ - G5.

obtida a partir do número de mensagens que o nó possui e o tamanho de seu *buffer*. Esta porcentagem é calculada quando o nó está sem vizinhos, segundo a Equação 4.2. Para o caso de *buffer* zero, o valor desta porcentagem é integralmente aplicado. O intuito era levar em consideração que se o nó possui poucas mensagens armazenadas ele pode permanecer por mais tempo com sua interface sem fio desativada, pois menores são os números de nós destinatários que ele procura. Em contrapartida, caso o próprio nó seja destino de alguma mensagem, pode-se favorecer um atraso no

recebimento.

$$\text{Porcentagem de } T = round\left(T \times \left(1 - \frac{num.msgs}{Tam.buffer}\right)\right) \tag{4.2}$$

Segundo Figuras 4.26 e 4.27, todas as parametrizações de G5 mantiveram taxas de entrega alta nas duas situações de energia inicial. Com o aumento do valor de *buffer* mais mensagens podem ser armazenadas e, o tempo de dormir T obtido pela Equação 4.2 tende a ser nulo ou baixo, devido a isto, já era esperado que para o caso de E as taxas de entrega para valores maiores de *buffer* se mantivessem próximas do gerenciamento desligado. Para o caso de $E/2$, temos um deslocamento da curva de taxa de entrega pela mesma situação exposta anteriormente, pois no início da simulação o nó passa a ter um valor de T alto, e economiza mais em energia, com o decorrer da simulação este valor tende a cair devido recebimento de mensagens, assim este deslocamento deve-se a esta economia inicial. Podemos perceber que a subida da curva de nós mortos é mais acentuada com o aumento do tempo de simulação, o que constata que temos uma economia maior quando o *buffer* do nó ainda não foi preenchido por completo.

A fim de comparação, selecionamos alguns gerenciamentos, e realizamos novas simulações variando o alcance de comunicação dos nós e os valores de energia inicial. Uma dificuldade encontrada foi uma parametrização única em cada gerenciamento que atendesse aos cenários testados. Assim nem sempre as melhores combinações de tempos T ou valores de probabilidades dos testes realizados na primeira parte, foram utilizadas nesta segunda etapa. Contudo, os casos selecionados para esta segunda parte da simulação possuem resultados presentes também na primeira parte. Segue as informações de quais modos de gerenciamento foram selecionados e suas parametrizações na Tabela 4.2.

Para o caso de alcance 100 m, Figura 4.30, todos os modos de gerenciamento apresentaram praticamente a mesma taxa de entrega, ou seja, mesmo o nós desativando suas interfaces sem fio por alguns períodos, a taxa de entrega foi pouco influenciada. Com o alcance reduzido para 50 m, Figura 4.29, os maiores prejudicados foram os gerenciamentos G2 e G5. O modo de gerenciamento G5 demonstrou piores resultados com o aumento do *buffer*. Sua taxa de entrega caiu principalmente com *buffer* = 50 pois com o alcance menor, menores são os números de contatos

Gerenciamentos	**Parametrização**
G2	Combinação de pesos P1 = 0.2 e P2 = 0.8 e parte complementar G1 com T fixo igual a 1 s
G3	Com valor de $T = 4$ s e $n = 3$
G4	Com δ de 10% e limite superior de 70%
G5	Com $T = 4$ s

Tabela 4.2: Gerenciamentos de Energia Selecionados

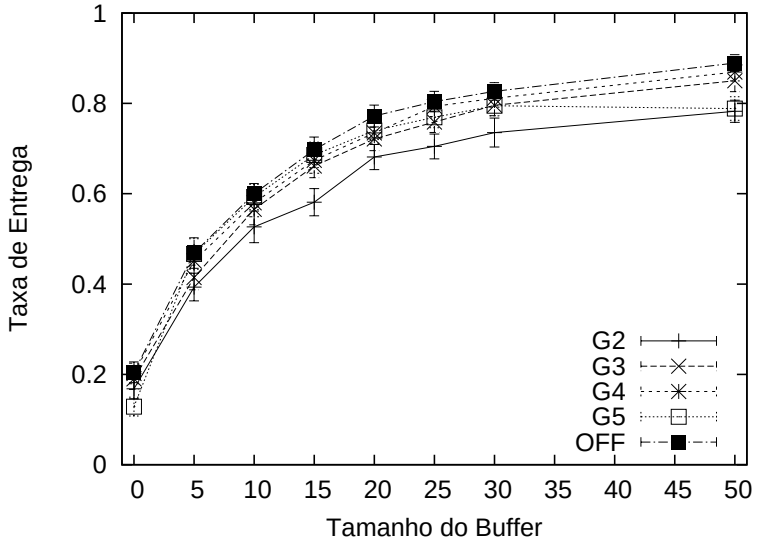


Figura 4.29: Alcance 50m e Energia E.

encontrados, logo é provável que os nós possuam menos mensagens armazenadas e assim o cálculo da porcentagem de T atinja valores altos, desta forma, o nó acaba deixando de aproveitar um número maior de oportunidades de comunicação. No caso de G2, com o alcance reduzido, mais vezes a parte G1 entrou em funcionamento. Para constatar tal afirmação realizamos simulações, para o mesmo cenário, porém com uso exclusivo de G2. Neste caso, a taxa de entrega foi maior e se aproximou do gerenciamento desligado, porém sua economia foi reduzida. Os gerenciamentos G3 e G4 não foram muito prejudicados. Para G3, diminuir o alcance fez com que menos vezes os nós passassem ao estado "dormindo", pois menores foram as chances dos nós ouvirem transmissões de outros componentes. Para G4, a redução do alcance fez com que mais rápido o nó atingisse o limite superior de δ. Para o mesmo cenário,

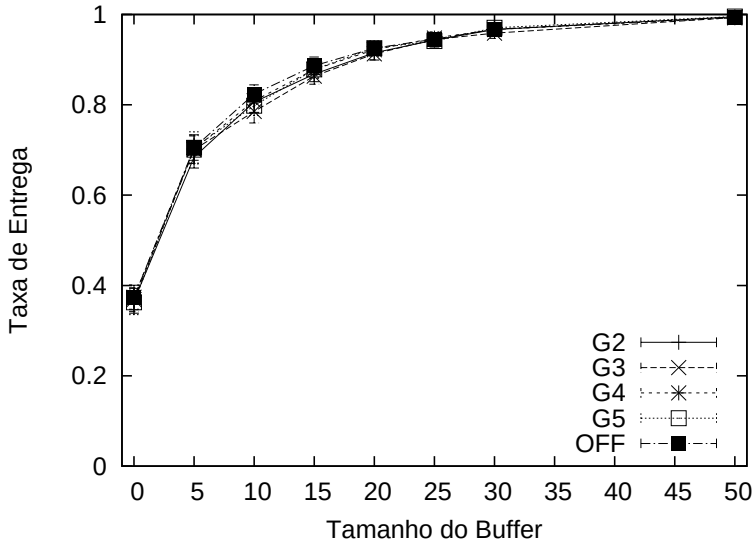


Figura 4.30: Alcance 100m e Energia E.

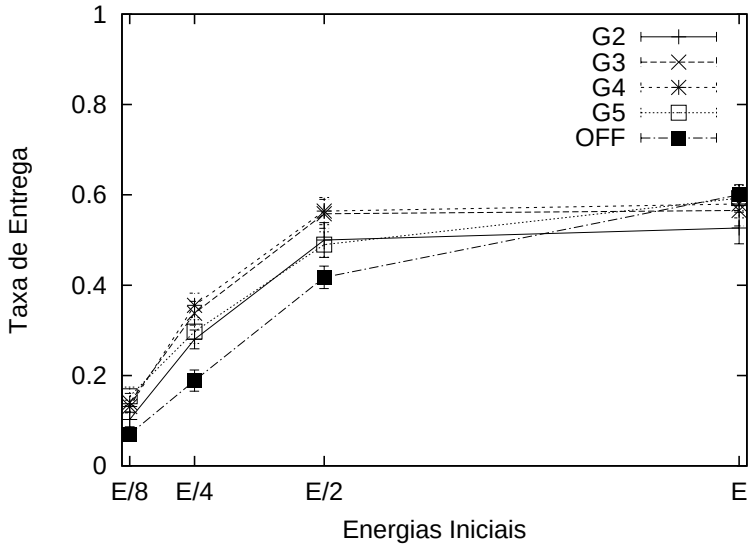


Figura 4.31: Alcance 50m e *buffer* 10.

realizamos testes com G4 tendo $\delta = 10\%$ e limite de 90%. Estes testes apresentaram taxas de entrega mais baixas, pois mais rápido o limite superior de probabilidade foi atingido e menores foram as chances de retorno para o estado acordado.

Segundo Figuras 4.31 e 4.32, quando diminuímos a quantidade de energia inicial favorecemos o aparecimento mais rápido de nós mortos na rede, logo para o caso de gerenciamento desligado a taxa de entrega cai devido a perda de nós fontes e/ou destinos. Porém quando passamos para o caso de energia E, todos os nós terminam

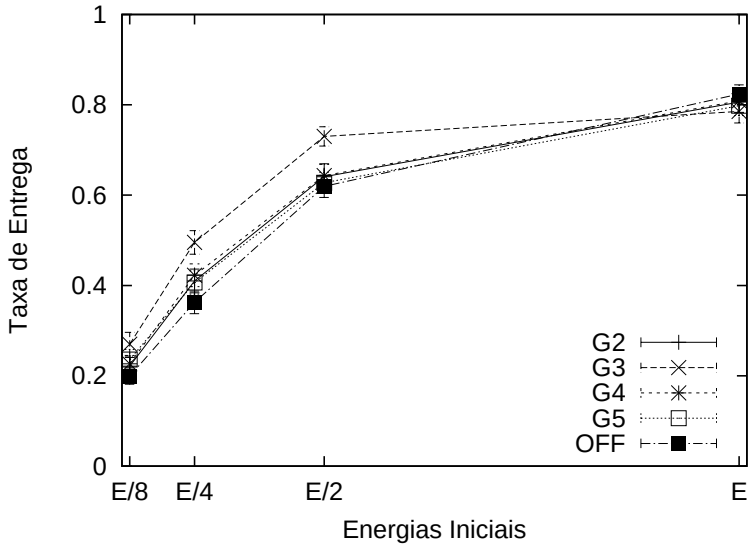


Figura 4.32: Alcance 100m e *buffer* 10.

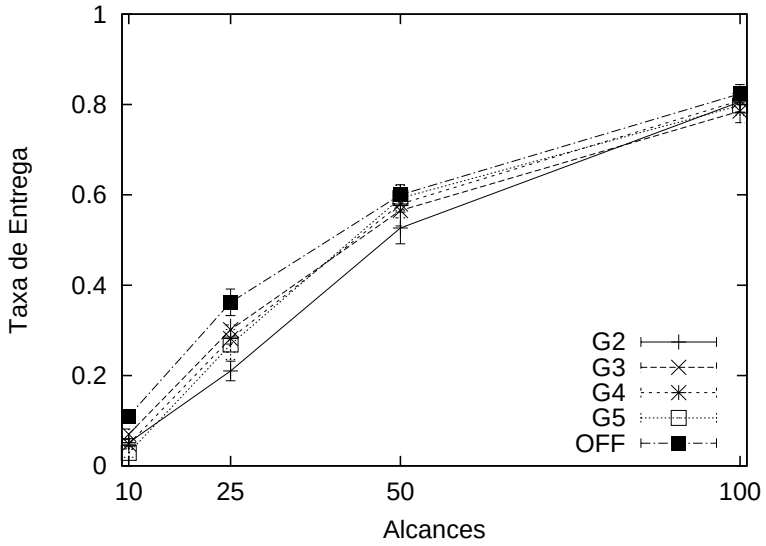


Figura 4.33: Energia E e *buffer* 10.

a simulação cooperando com a rede e, neste caso a taxa de entrega do gerenciamento desligado é mais alta porque mesmo com alcance reduzido, as oportunidades de contato são todas aproveitadas. Assim o gerenciamento desligado representa a maior taxa de entrega possível, visto que qualquer gerenciamento implica em ocasionais perdas de contato. Para G5 o número de nós mortos tende a ser mais alto que nos demais gerenciamentos, pois com o preenchimento de seu *buffer* o nó dorme menos, por isso um aumento na taxa de entrega quando passamos de $E/2$ para E.

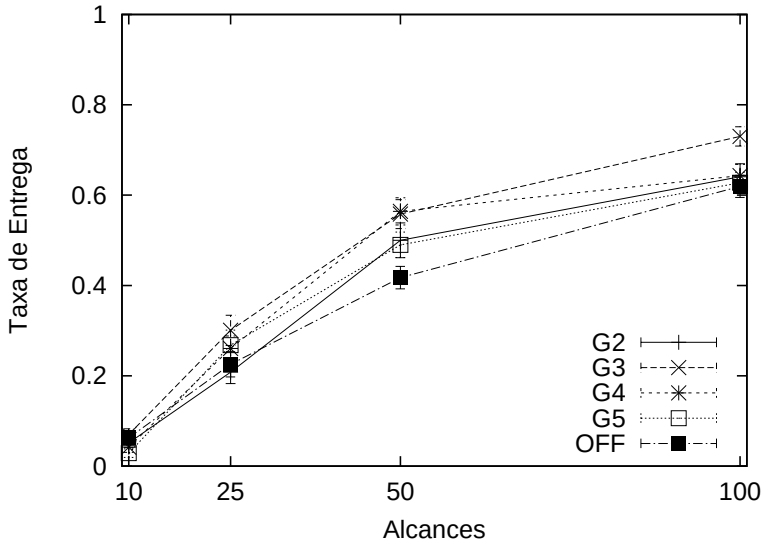


Figura 4.34: Energia $E/2$ e *buffer* 10.

Na situação de E os nós dormem menos porém não deixam de contribuir, logo G5 praticamente não perdeu oportunidades de contato e se aproximou do caso sem gerenciamento. Em G2 na variação de $E/2$ para E a taxa de entrega se manteve, com o alcance reduzido o número de vizinhos utilizado no cálculo de T foi menor, logo T foi menor, porém a ação da parte G1 foi mais frequente e, a taxa de entrega foi mais influenciada pela perda de contatos do que pela perda de nós. Os gerenciamentos G3 e G4 apresentaram resultados semelhantes em taxa de entrega, porém as causas são diferenciadas. O gerenciamento G3 ataca diretamente o caso do nó ouvir consecutivamente enquanto G4 trata do caso do nó sem vizinhança, os valores de taxa de entrega foram próximos porém depois de testadas diversas parametrizações.

Em outros testes com as mesmas variações nos eixos x e y, porém com diferentes valores de T a taxa de entrega em G3 foi influenciada conjuntamente pelo valor de energia inicial e o valor de tempo de dormir. Para G3 o aumento de T entre os valores de energia $E/8$ até $E/4$, favoreceu aos nós permanecerem um maior tempo em atividade, logo valores maiores de T garantiram uma taxa de entrega maior. Enquanto que da faixa de energia $E/2$ até E este comportamento se inverteu, maiores valores de T incorreram em menores taxas de entrega, pois eram maiores o número de perdas de contato. Dentre os valores de T simulados o que conseguiu melhor proveito, em média, foi justamente o parâmetro $T = 4$ segundos. Para G4 a influência maior surgiu pela maior rapidez em atingir o limite de probabilidade. O

mesmo fato dito anteriormente aconteceu para o G4 com $\delta = 10\%$ e limite de 90%.

Conforme os gráficos com variações de alcance, Figuras 4.33 e 4.34, para um menor alcance qualquer desligamento da interface é mais problemático. Pois intuitivamente o nó já possui um número de contatos menor, logo qualquer período de desligamento provoca uma maior perda de contatos e, quando os nós não "morrem" esta perda de contato nem sempre pode ser recuperada no avanço do tempo de simulação. Quando a energia inicial é reduzida, torna-se mais interessante esta economia, pois passamos a ter uma perda no número de nós contribuintes da rede, assim a economia favorece uma possibilidade de um encontro mesmo que tardia, o que só não se aplica ao caso sem gerenciamento.

4.4 Conclusões do Capítulo

Podemos constatar neste capítulo, que o encaminhamento proativo mostrou-se eficiente para a diminuição do número de transmissões de mensagens, assim como um bom mecanismo complementar à lista de *acks* pois aumentou a taxa de entrega, diminuiu o número de transmissões e aumentou o número de mensagens confirmadas nos nós de origem.

A partir do uso dos modos de gerenciamento de energia, podemos verificar que eles favoreceram a um acréscimo no tempo operacional da rede e desta forma, possíveis mensagens que talvez não fossem entregues aos seus destinos, devido o nó destino não estar mais em atividade, possam atingir seus destinatários mesmo que posteriormente.

Com relação a comparação entre os modos de gerenciamento, os modos de gerenciamento que apresentaram melhores resultados foram os modos G3 e G4. O fato da taxa de entrega menor em situações de alcance reduzido não inviabiliza o uso de modos de gerenciamento pois, possíveis entregas podem surgir posteriormente, devido ao menor número de nós "mortos". Além disso, buscamos nesta comparação utilizar uma parametrização única para cada modo de gerenciamento nos diferentes ambientes testados, assim uma parametrização específica para um certo ambiente pode permitir melhores resultados.

A economia de recursos pode ser proveniente do tipo de roteamento utilizado,

como apresentado no Capítulo 2, assim no Capítulo 5 apresentaremos um tipo de encaminhamento de tenta diminuir o número de transmissões realizadas.

Capítulo 5

Encaminhamento Baseado em Histórico de Encontros

Os mecanismos apresentados anteriormente, no Capítulo 4, buscam o aperfeiçoamento no uso de recursos. Porém, por natureza o protocolo de roteamento também pode contribuir para este aperfeiçoamento, como é o caso de roteamentos baseados em encaminhamento. Estes roteamentos aproveitam-se da coleta de alguma informação que lhe proporcione um melhor encaminhamento ou uma decisão mais acurada e inteligente de qual mensagem trocar no instante de contato. Como consequência, um menor número de transmissões é realizado e economiza-se no gasto de energia além de se administrar melhor o espaço de armazenamento. A maior dificuldade encontrada por esta classe de roteamento é definir quando é o melhor instante de troca de mensagens.

Neste capítulo, será apresentado um tipo de encaminhamento que avalia o interesse de se enviar alguma mensagem ao nó receptor, e em caso positivo, qual mensagem enviar no instante de contato dos nós, utilizando para isso uma métrica a mais: o histórico de encontros do nó.

Em geral, os trabalhos voltados para roteamento em RTAIs almejam a resolução de dois problemas: a baixa taxa de entrega e o grande atraso na entrega de mensagens. Porém, como grande parte das aplicações são voltadas para equipamentos onde recursos como energia e espaço de armazenamento são relevantes, estas soluções não devem consumir recursos em demasiado.

Diversos algoritmos de roteamento buscam a diminuição de custos, principal-

mente de transmissão, pelo encaminhamento de mensagens a apenas alguns nós, que em decorrência de alguma métrica de avaliação própria do protocolo, sejam ou estejam mais capacitados a receber uma mensagem do que outros. Estes nós são assim procurados devido a sua melhor disposição em entregar alguma mensagem.

De forma intuitiva, podemos verificar que se um nó **a** tem uma taxa de encontros alta com outros nós da rede, este nó **a** é um nó que torna-se um candidato interessante para se encaminhar alguma mensagem, pois a chance de entrega pode ser mais alta que comparada a de outro que pouco interage com os demais integrantes. Daí a ideia de se recorrer a um histórico de encontros, conforme encontramos em [67, 68, 45], porém com a finalidade de classificar o nó como um bom ou mau retransmissor ou "portador" de mensagens. Esta ideia pode ser vista como a tentativa de classificar o nó em relação ao quão "social" ele se apresenta. Porém, a maior dificuldade é no instante de contato classificar o nó como um bom retransmissor ou não. Além de priorizar qual mensagem transmitir primeiro em situações de contatos de curta duração, e selecionar qual mensagem descartar quando o *buffer* tornar-se insuficiente [37].

A partir desta problemática, apresentamos um tipo de encaminhamento seletivo que consegue uma taxa de entrega satisfatória, enquanto diminui o número de transmissões de mensagens. Seu funcionamento vale-se da tentativa de classificar o nó receptor como um mau retransmissor de mensagens e desta maneira, transmitir a ele uma quantidade reduzida de mensagens ou, às vezes, nem transmitir. Esta classificação é feita comparando o histórico de encontros dos nós em contato, e a partir desta avaliação, opta-se por um comportamento proveniente de um roteamento do tipo replicador ou outro com uma replicação controlada. Assim, quando a escolha for para um tipo de roteamento do tipo replicador, o nó transmissor, caso possua ao menos uma mensagem que o nó receptor não tenha, tentará transmitir esta mensagem. Caso contrário, mesmo que o nó transmissor possua alguma mensagem ainda não vista pelo nó receptor, nem sempre a transmitirá. Ou seja, na escolha por um roteamento do tipo replicador toda oportunidade de contato será aproveitada, caso haja alguma mensagem a se transmitir, senão recorreremos a uma métrica que decidirá o interesse em repassar alguma mensagem. Desta forma o encaminhamento seletivo pode ser visto como um meta protocolo, que a partir de uma classificação

própria escolhe qual tipo de protocolo de roteamento deve ser o mais adequado em certa ocasião.

Este chaveamento do tipo de roteamento ocorre segundo um algoritmo de seleção/classificação. Este algoritmo será apresentado a seguir na Seção 5.2. Porém antes cabe uma explicação de como o histórico de encontros funciona.

5.1 Histórico de Encontros dos Nós

Como o encaminhamento seletivo tem por base de funcionamento o histórico de encontros dos nós, antes de apresentarmos o algoritmo de seleção/classificação, vale explicar como foi implementado este histórico de encontros.

O histórico de encontros contabiliza todos os contatos de um nó em um período de tempo passado, semelhante como encontrado em [67]. Cada nó $\underline{\mathbf{a}}$ assim que se encontra com outro nó $\underline{\mathbf{b}}$ o acrescenta ao seu histórico de encontros e vice-versa. Assim que um encontro é computado, um contador é acionado referente a esta entrada de informação. Este contador é incrementado com o tempo e, ao atingir T segundos o nó é retirado do histórico de encontros tanto de $\underline{\mathbf{a}}$ quanto de $\underline{\mathbf{b}}$. Caso um nó $\underline{\mathbf{b}}$ ainda esteja no histórico de encontros do nó $\underline{\mathbf{a}}$ e o nó $\underline{\mathbf{a}}$ volte a se encontrar com o nó $\underline{\mathbf{b}}$, o tempo T referente a esta entrada é então zerado. Logo que o nó $\underline{\mathbf{b}}$ fique fora do alcance do $\underline{\mathbf{a}}$, o contador volta a ser incrementado. Ou seja, o tempo T representa um tempo de validade das entradas no histórico de encontros. Deste modo, quando um nó permanece mais que T segundos no histórico de encontros de outro nó, este é retirado.

Como cada nó possui um identificador único, mesmo que um nó encontre com outro frequentemente, isto indicará apenas uma única entrada no histórico de encontros do nó com referência a este encontro. A única alteração nesta situação será a atualização do contador no instante de encontro.

Vale ressaltar que um nó pode se movimentar pouco e ter uma taxa de encontros alta, basta para isso estar localizado em um lugar estratégico como um ponto de encontro ou ponto de passagem.

5.2 Algoritmo de Seleção/Classificação

A falta de garantia que um enlace, uma vez estabelecido, permaneça por um longo período de tempo, faz com que nós em contato tentem transmitir o máximo de mensagens possível, e assim aumentem as chances de que alguma mensagem alcance o destino. Porém, este procedimento reverte-se em um alto uso de recursos da rede, seja energético ou de armazenamento.

Uma forma de diminuir o gasto de recursos é reduzindo o número de transmissões feitas. Logo, o ideal é que na oportunidade de contato, o nó transmissor estude a viabilidade de se enviar mensagens ao seu par de comunicação, além de selecionar dentre as mensagens qual transmitir primeiro. Porém este tipo de procedimento não é simples de ser aplicado, devido ao tipo de movimentação dos nós, limitação do espaço de armazenamento, reserva energética, tempo de contato, etc.. Além de ser notável que o número de cópias de uma determinada mensagem impacta diretamente na taxa de entrega do protocolo de roteamento em uso. Assim, temos um compromisso entre utilização de recursos e taxa de entrega.

De maneira intuitiva podemos verificar que se um nó tem uma taxa de encontro alta com outros nós, ele é um nó que merece receber alguma mensagem no instante de contato, pois a chance dele se encontrar com o nó de destino da mensagem é maior que a de um outro nó que pouco se relaciona ou interage com os demais integrantes da rede.

A fim de comprovarmos esta heurística contabilizamos algumas informações em um cenário de testes. Este cenário continha dois tipos de nós, cujo diferença entre eles dava-se na velocidade de movimentação. Um grupo de nós possuía velocidade média mais alta que o outro. Neste ambiente, contabilizamos o número de contatos de cada nó e, verificamos que a média de encontros dos nós com alta velocidade foi maior que a média de encontros dos nós de baixa mobilidade. Além disso, parametrizamos os nós com um tamanho de *buffer* que fosse suficiente para não ocorrer descarte. Assim, caso um nó já possuísse uma mensagem, esta não seria enviada novamente pelo nó transmissor no instante de contato. Desta forma, foi possível comparar o número de mensagens enviadas pelos nós. Os nós com alta velocidade apresentaram um maior número de transmissões que os nós com baixa velocidade, confirmando seu maior número de oportunidades de contatos.

Deste modo, a partir do histórico de encontros do nó podemos classificá-lo como um bom ou mau "portador" de mensagens. Caso o nó tenha um número grande de encontros passados, ele é um nó propenso a ser classificado como bom "portador" de mensagens, pois é de se esperar que ele consiga dar maior fluidez a troca de mensagens, além de maior agilidade em possíveis entregas. Por outro lado, se o nó teve um número reduzido de encontros em um período de tempo passado, ele é visto como um mau "portador", pois são menores as chances dele efetuar uma entrega.

Assim, o balanceamento do número de mensagens vale-se desta observação, número de encontros passados, e funciona em etapas, conforme o Algoritmo 2. O primeiro teste compara a quantidade de contatos passados dos nós envolvidos. Esta informação é obtida pelo histórico de encontros de cada nó. Caso o nó receptor possua um número de encontros passados maior que o do nó emissor, este nó receptor é visto como um bom retransmissor perante este nó transmissor. Logo, é interessante repassar a este nó receptor uma quantidade grande de mensagens, assim a política de encaminhamento selecionada é a LEFO (política procedente de um roteamento do tipo replicador). Mais especificamente utilizamos a política LEFO empregada em etapas conforme apresentado no Capítulo 4.

Caso o nó receptor tenha uma quantidade menor de contatos, recaímos no segundo teste. Este segundo teste verifica se dentre os contatos do receptor existe algum nó que apenas o próprio nó receptor encontrou, ou seja, algum nó que o nó transmissor não tenha registro de encontro. O nó transmissor pode até já ter se encontrado com este nó porém, este econtro pode ter sido em um tempo superior ao período de validade do histórico de contatos. Em resumo, este teste verifica se o complemento dos históricos de contatos entre o nó receptor e o transmissor é diferente de vazio ($C_{Rx}Tx \neq \emptyset$). Caso afirmativo, apesar do nó receptor ter uma quantidade menor de contatos passados, com algum destes, o nó transmissor não se encontrou ou este encontro ocorreu em um período de tempo superior ao tempo de validação do histórico de encontros. Logo, é viável enviar mensagens destinadas somente para estes contatos diferentes, caso o transmissor possua alguma.

Por último, temos o caso do nó receptor, além de ter um número de contatos menor não se encontrou com nenhum contato diferente, dentro do tempo de validação do histórico, e assim trabalha conforme o protocolo PROPHET. No decorrer do texto

utilizaremos a sigla PROP quando nos referenciarmos a forma de encaminhamento do protocolo PROPHET, maiores detalhes sobre o encaminhamento PROP serão apresentados a seguir.

Um detalhe é que esta classificação entre bom ou mau retransmissor é feita localmente, e o referencial são os próprios nós envolvidos. Ou seja, um nó pode ser um bom ou um mau "portador", dependendo do seu par de comunicação.

Logo, o encaminhamento proposto tem por base a estratégia de funcionamento dos protocolos de roteamento Epidêmico ou PROPHET, assim utiliza a política de envio de mensagens ou do tipo LEFO ou a política PROP, respectivamente.

Além disso como característica, no instante de seleção da política de envio de mensagens, o encaminhamento apresentado utiliza um parâmetro do tipo DI e caso recaia na última opção de funcionamento utiliza um parâmetro DD para a seleção de qual mensagem enviar.

```
if #contatos ($receptor) + δ > #contatos ($emissor) then
    EPIDÊMICO
else
    if $receptor teve contatos diferentes do $emissor then
        Envio de mensagens apenas para
            estes nós diferentes
    else
        PROPHET
```

Algoritmo 2: Algoritmo de Seleção/Classificação

O parâmetro δ, presente no primeiro teste do algoritmo, é um ajuste quanto a classificação do nó entre bom ou mau retransmissor ser mais ou menos persistente. Este parâmetro obriga que a classificação de um nó em relação a outro demore mais tempo para se alterar, pois a classificação vem da diferença do número de contatos passados adicionada do valor do parâmetro δ. Quando feita de forma direta, sem o uso de δ, a classificação entre os nós torna-se muito oscilante.

Tipo de encaminhamento do protocolo PROPHET

Devido utilizarmos um encaminhamento seletivo que chaveia entre as políticas de transmissão LEFO e PROP, uma breve apresentação das equações que regem o

funcionamento da política PROP será mostrada a seguir.

Conforme o funcionamento do PROPHET apresentado no Capítulo 2, sabemos que trata-se de um roteamento que utiliza a probabilidade de encontro do nó para decidir se o nó receptor deve ou não receber alguma mensagem, além de decidir qual. A probabilidade de encontro pode ser calculada ou atualizada de três modos, apresentados abaixo:

- **Contato Direto**

  Por contato direto, quando o nó se encontra com outro o valor da *delivery predictability* ($P_{(a,b)}$) referente a este nó é modificado, conforme a Equação 5.1. Onde P_{init} é uma constante de inicialização com valor entre 0 e 1.

$$P_{(a,b)} = P_{(a,b)_{old}} + (1 - P_{(a,b)_{old}}) \times P_{init} \tag{5.1}$$

- **Envelhecimento**

  A segunda maneira é por envelhecimento. A medida que um par de nós deixa de se encontrar, esta métrica diminui conforme a Equação 5.2. O valor de γ é a constante de envelhecimento, com valor entre 0 e 1. E k é o número de unidades de tempo transcorridas desde a última vez que a métrica foi atualizada. Quanto mais tempo os nós deixam de se encontrar mais o valor de $P_{(a,b)}$ decai.

$$P_{(a,b)} = P_{(a,b)_{old}} \times \gamma^k \tag{5.2}$$

- **Propriedade Transitiva**

  O último modo origina-se da propriedade transitiva (Equação 5.3), onde se um nó **a** encontra-se frequentemente com um nó **b**, e **b** encontra-se frequentemente com **c**, logo, o nó **c** provavelmente é um bom nó para se encaminhar mensagens destinadas ao nó **a**. O valor de β decide quanto de impacto a propriedade transitiva tem sobre a *delivery predictability*.

$$P_{(a,c)} = P_{(a,c)_{old}} + (1 - P_{(a,c)_{old}}) \times P_{(a,b)} \times P_{(b,c)} \times \beta \tag{5.3}$$

5.3 Ambiente de Simulação

O encaminhamento seletivo teve por base o protocolo Epidêmico, porém adaptado para coletar informações do histórico de contatos e calcular a probabilidade de entrega ($P_{(a,b)}$).

Para os experimentos realizados foram utilizados 50 nós, com alcance de 100 m, que se movimentavam em uma área quadrada de 1000 m de lado durante 1000 segundos, com valor de *snapshot* de 1 segundo. Para os testes com o encaminhamento seletivo e o PROP foram reservados um período no tempo de simulação referente ao tempo de *warm up*, necessário para o cálculo da probabilidade de entrega. Para a movimentação dos nós, quatro tipos de cenários foram gerados, todos utilizando o modelo de mobilidade *Random Waypoint*, que cria um tipo de movimentação aleatória simples e não tendenciosa.

A diferença entre os três primeiros cenários foi a quantidade de nós com baixa velocidade, que variou de 20%, 50% a 80% . No quarto cenário todos os nós possuíam a mesma velocidade. Isto devido ao fato que o nó que se movimenta pouco tende a ter um número de contatos menor que outro que seja mais ágil, salvo situações onde o nó permanece em um ponto de encontro, por exemplo. E justamente estes nós menos interativos que devem ser menos favorecidos no instante da troca de mensagens.

Estes mesmos parâmetros foram utilizados nas simulações apresentadas no Capítulo 4, com exceção dos cenários.

Os nós com baixa mobilidade possuíam velocidade máxima de 1 m/s e tempo de pausa de 10 s, simulando um grupo de pessoas portando algum tipo de equipamento portátil. Já para os nós com alta mobilidade, a velocidade mínima era de 3 m/s e velocidade máxima de 10 m/s com tempo de pausa de 10 s, simulando veículos em ruas fechadas de uma cidade. Com exceção do cenário onde todos os nós tinham a mesma velocidade, com valor de 3 m/s e pausa de 10 s que tratou-se do cenário 4.

Cada mensagem gerada tinha um tamanho fixo de 10 KBytes, e cada nó gerou apenas uma mensagem destinada a outro nó aleatório. A geração das mensagens ocorreu no início de cada simulação. Além disso, foram criados 30 cenários distintos para cada configuração. Com isso, todos os resultados apresentados são médias dos resultados dos 30 cenários com intervalos de confiança de 95%.

A política de descarte utilizada foi a MOFO (*Evict most forwarded first*), apresentada em [58], onde descarta-se a mensagem mais encaminhada pelo nó. O valor do tempo limite de validação do nó no histórico de contatos (T) foi de 30 segundos. Todos os valores necessários para o cálculo da probabilidade de entrega, P_{init}, k, γ e β, foram mantidos conforme o artigo do PROPHET [1].

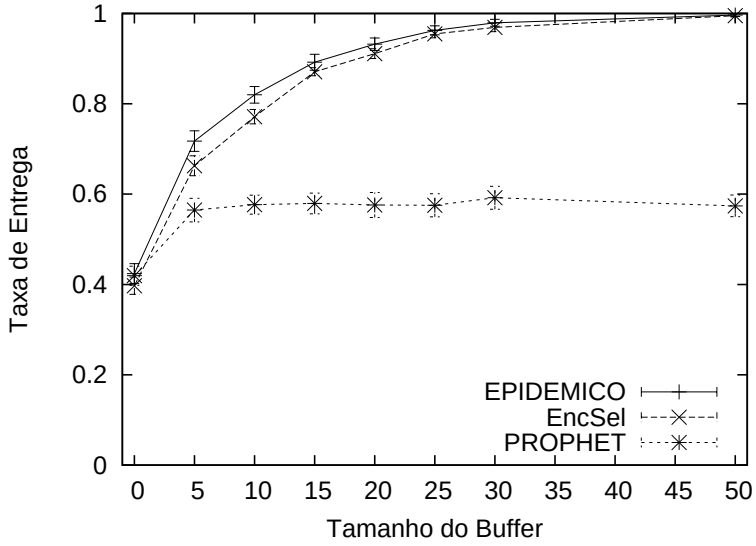
O encaminhamento foi comparado com dois outros tipos de roteamentos: Epidêmico e PROPHET. Ou seja, o primeiro que utiliza uma política de encaminhamento (LEFO) mais agressiva em número de réplicas de mensagens, o segundo com uma política que faz uma análise sobre o envio. Cabe ao esquema proposto classificar o nó como um mau retransmissor e, assim realizar a transmissão das mensagens ou não.

Embora um cenário com movimentação aleatória não seja propício para classificar os nós conforme seu comportamento futuro, o intervalo de contatos passado pode indicar se o nó interagiu pouco ou não, além disso o parâmetro δ como ajustado, $\delta = 6$, garante que a classificação de bom ou mau retransmissor entre os pares de comunicação demore a se alterar. Utilizamos esta movimentação pois representa um cenário totalmente genérico, e gostaríamos de uma solução de amplo alcance.

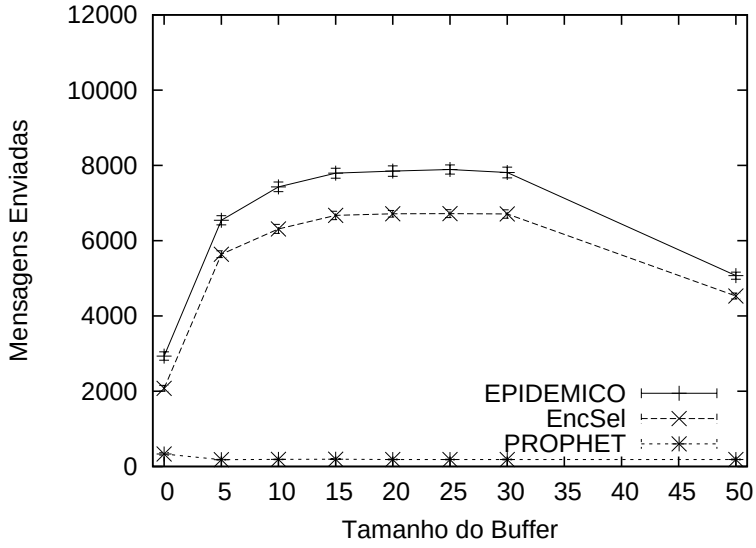
5.4 Resultados da Simulação

Com o intuito de verificar a influência do nível de interação entre os nós no desempenho, tanto do encaminhamento seletivo proposto, no LEFO e no PROP, foram gerados quatro cenários. Onde em três destes, a razão entre o número de nós com menor e maior velocidade aumentava e o último onde todos os nós tinham a mesma velocidade. Através de simulações foram avaliadas nos quatro cenários, a taxa de entrega, o número de mensagens enviadas, o número de mensagens descartadas e o número médio de saltos das mensagens entregues. Foi variado o espaço de armazenamento dos nós a fim de se analisar a influência do número de mensagens injetadas na rede e seu impacto na taxa de entrega, além do uso de recursos dos nós componentes. Nos gráficos, o encaminhamento LEFO está representado pelo protocolo Epidêmico, enquanto o PROP pelo protocolo PROPHET e o encaminhamento seletivo por EncSel.

O valor de *buffer* igual a zero é referente ao caso da entrega de mensagens apenas por contato direto, enquanto que o *buffer* de tamanho 50 é equivalente a um *buffer* infinito, visto que este é o número máximo de mensagens diferentes que transitam pela rede.



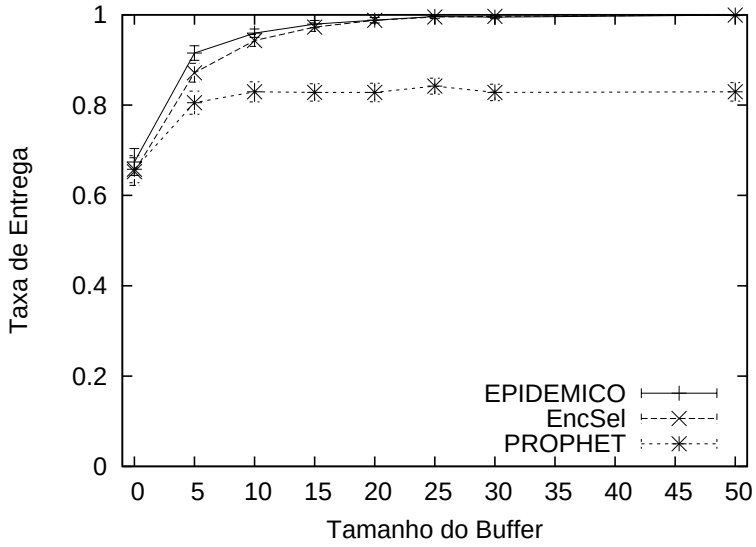
(a) Taxa de entrega.



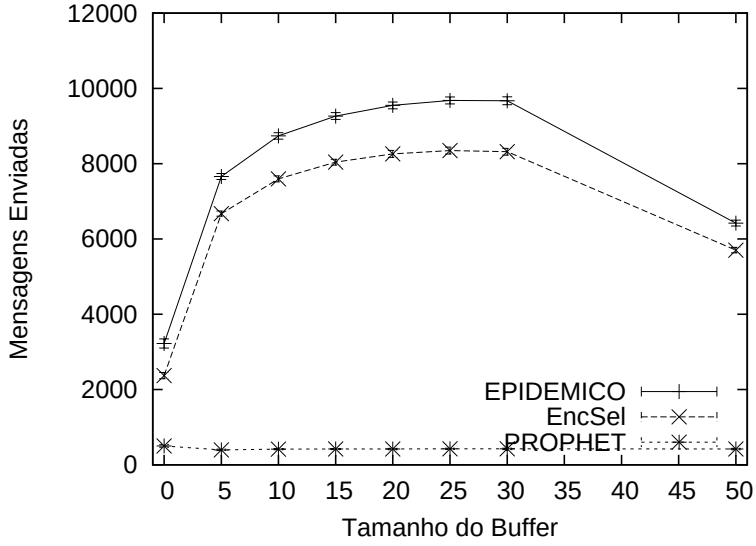
(b) Número de mensagens enviadas.

Figura 5.1: Cenário 1 - 80% dos nós com baixa mobilidade.

A partir dos gráficos de taxa de entrega pôde-se verificar que os três tipos de políticas apresentaram melhores resultados quando o número de nós com alta velocidade cresceu. Ou seja, cenários com um maior número de nós com alta mobilidade



(a) Taxa de entrega.

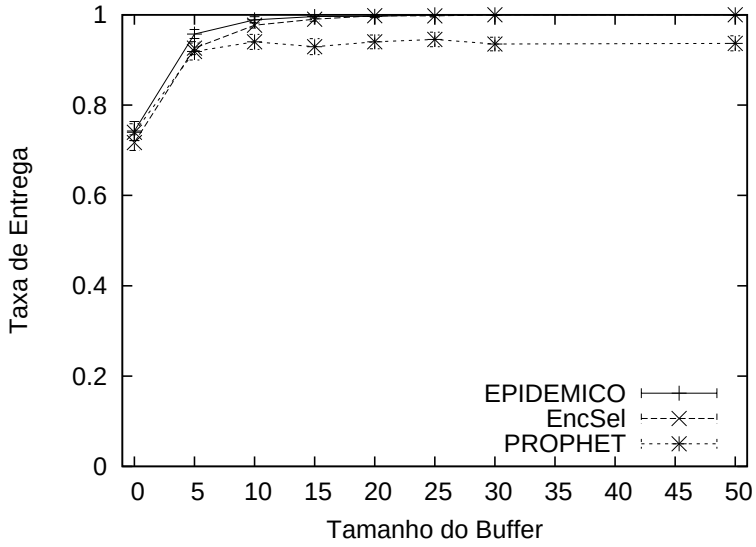


(b) Número de mensagens enviadas.

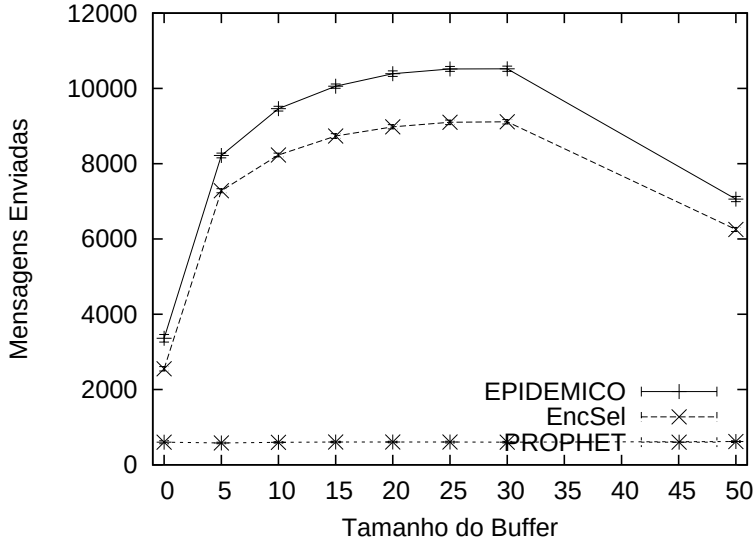
Figura 5.2: Cenário 2 - 50% dos nós com baixa mobilidade.

favorecem uma maior fluidez das mensagens pela rede, portanto a taxa de entrega é melhor.

Conforme esperado, o protocolo PROPHET economiza no número de mensagens transmitidas, porém é o mais influenciado pela mobilidade dos nós. Cenários onde o número de nós com maior velocidade é reduzido, ou onde a velocidade dos nós é relativamente baixa (3 m/s) seu desempenho é prejudicado, Figuras 5.1 e 5.4 respectivamente. Isto acontece pois é menor o número de nós que teoricamente



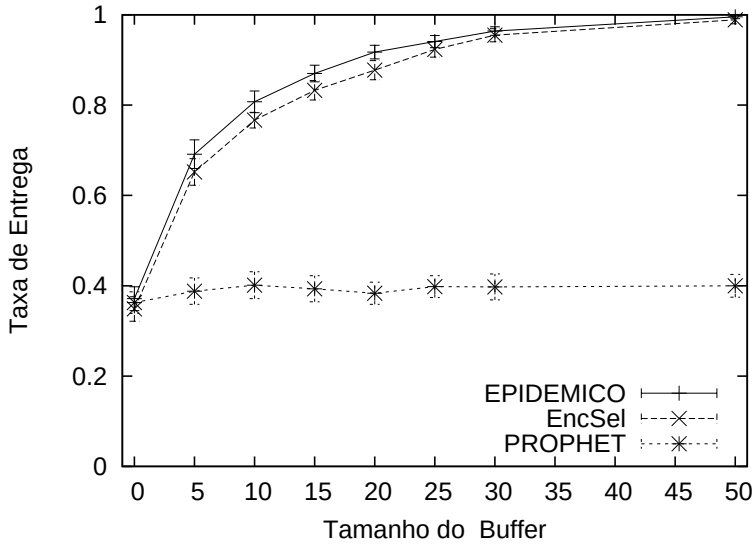
(a) Taxa de entrega.



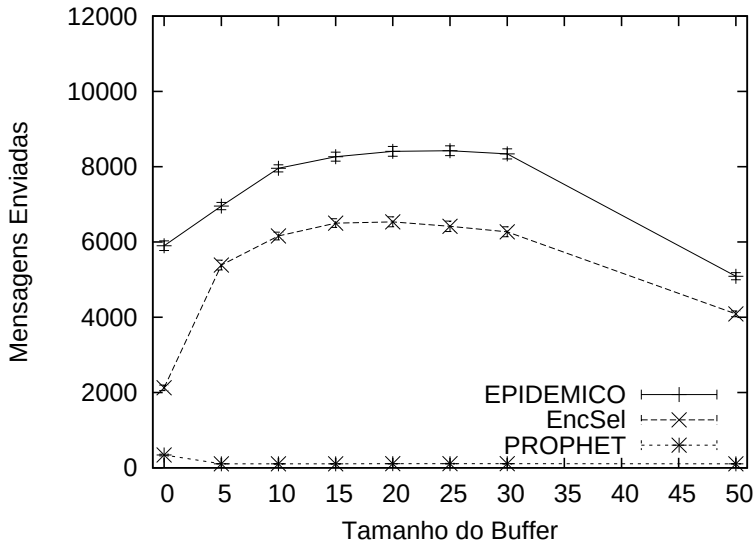
(b) Número de mensagens enviadas.

Figura 5.3: Cenário 3 - 20% dos nós com baixa mobilidade.

possuem uma taxa de encontros alta e, todavia o protocolo tende a tentar transmitir apenas a estes nós. Um outro problema é a diferenciação dos nós, que surge quando os móveis possuem praticamente o mesmo comportamento de variação de velocidade. Ou seja, em redes homogêneas o protocolo apresenta dificuldade de seleção, pois os nós podem apresentar valores próximos de probabilidades de entrega, e definir no instante de contato o envio ou não de uma mensagem acaba sendo mais complicado. Vale ressaltar que se o cenário apresentar nós com comportamento homogêneo porém



(a) Taxa de entrega.

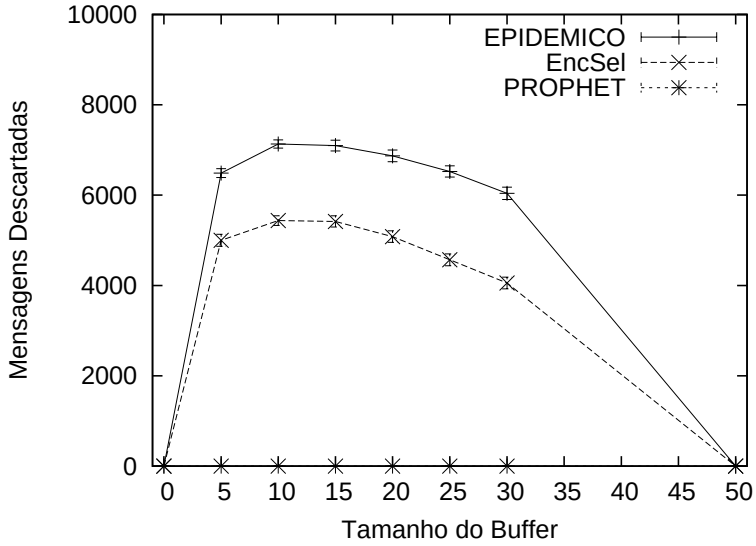


(b) Número de mensagens enviadas.

Figura 5.4: Cenário 4 - Todos o nós com a mesma velocidade de 3 m/s.

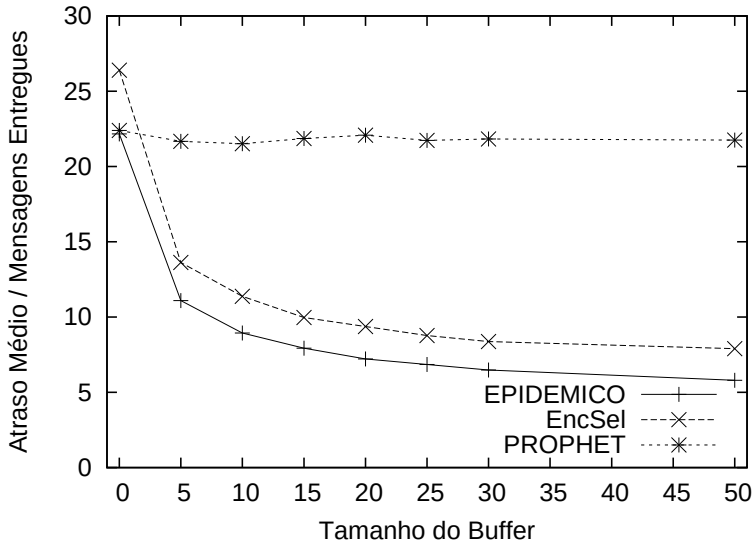
com velocidades médias elevadas, o problema de diferenciação dos nós é amenizado pelo número maior de encontros.

O encaminhamento seletivo apresentou uma taxa de entrega satisfatória comparado ao protocolo Epidêmico, mais precisamente ao encaminhamento LEFO, ao passo que diminuiu o número de mensagens injetadas na rede em todos os cenários. Este número menor de transmissões refletiu-se em uma quantidade menor de mensagens descartadas em comparação ao roteamento Epidêmico, Figura 5.5. Embora



(a) Número médio de mensagens descartadas

Figura 5.5: Cenário 4 - Número médio de mensagens descartadas.



(a) Atraso médio.

Figura 5.6: Cenário 4 - Atraso médio/Mensagens entregues.

apresentado o resultado de descartes em apenas um cenário, nos demais o comportamento foi semelhante. No PROPHET, o número de descartes foi praticamente zero, visto que o número de transmissões é muito baixo. Em grande parte dos pontos, (Figuras 5.1a, 5.2a, 5.3a e 5.4a), ocorreu sobreposição dos intervalos de confiança, na taxa de entrega, entre Epidêmico (LEFO) e o EncSel. Mesmo em cenários com baixa velocidade média ou com poucos nós mais ágeis, o encaminhamento seletivo

manteve um bom desempenho, principalmente nos cenários 1 e 4.

Em decorrência do valor fixo atribuído para δ no algoritmo do EncSel seu funcionamento foi mais próximo do comportamento Epidêmico. Onde em alguns casos não foi a melhor escolha, pela Figura 5.3 (a) podemos ver que a política mais adequada foi a PROP, que manteve uma taxa de entrega alta e diminuiu muito o número de cópias das mensagens.

Porém como pretendíamos que o EncSel atendesse de forma ampla a vários cenários distintos tentamos ajustar o valor de δ para esta finalidade.

Uma ideia futura é a partir do número médio de contatos dinamicamente sintonizar o valor de δ, para que trabalhe mais como o protocolo Epidêmico e menos como PROPHET ou vice-versa em decorrência do cenário.

O número de saltos médio das mensagens entregues foi praticamente o mesmo entre as políticas LEFO e EncSel, em média 4 saltos. Enquanto que no PROPHET foram de 2 saltos. Porém, o atraso médio de entrega no PROPHET tende a ser maior (Figura 5.6), justamente pela relação entre número de transmissões e chances de entrega. Nesta situação, com *buffer* igual a zero o roteamento Epidêmico continua com um número de transmissões elevado devido seu funcionamento, independente se o nó par de comunicação irá armazenar ou não alguma mensagem, com isso os nós permanecem várias rodadas de oportunidades de transmissão (*snapshots*) com o mesmo par de comunicação o que impede uma maior rotatividade de novos contatos, como a taxa de entrega é próxima entre os roteamentos Epidêmico e PROPHET para *buffer* zero, o fato de impedir esta rotatividade se reflete em um maior atraso na entrega. Contudo nas demais situações o roteamento Epidêmico mostra-se melhor, resultado este que melhora com o aumento do *buffer*. O encaminhamento seletivo apresenta comportamento semelhante ao roteamento Epidêmico, mas com um atraso um pouco maior, devido seu menor número de transmissões.

5.5 Conclusões do Capítulo

Neste capítulo, apresentamos o encaminhamento baseado em histórico de encontros que tem por base tentar classificar o nó como um mal retransmissor de mensagens, e desta forma, a este nó repassar uma quantidade reduzida de mensagens. Assim

como, apresentamos a forma de seleção/classificação dos nós e como implementamos o histórico de encontros. Através de simulações, podemos compará-lo com os protocolos Epidêmico e PROPHET e verificamos que o encaminhamento proposto reduziu o número de mensagens transmitidas entre os nós e manteve uma taxa de entrega relevante, embora em alguns cenários a redução do número de mensagens não foi grande, consequência da tentativa de uma parametrização única na forma de classificação dos nós.

Capítulo 6

Conclusões

As RTAIs são um tipo de rede de grande importância por não necessitarem de requisitos de conectividade fim-a-fim, conseguindo aproveitar de maneira oportunista as características de mobilidade dos nós para o seu funcionamento. Entretanto, são redes onde o problema da escassez de recursos pode afetar bastante o seu desempenho.

Neste trabalho, desenvolvemos um ambiente de simulação de RTAIs que leva em conta os problemas gerados pela escassez de recursos intrínseca a essas redes. Isto permitiu a avaliação do impacto de tais restrições no seu desempenho, através da avaliação da influência de alguns mecanismos já existentes na literatura e de uma proposta simples de encaminhamento proativo, até então não empregada ou avaliada em outros trabalhos. Foi analisada a influência no uso de modos de gerenciamento de energia que indicavam possibilidades de períodos de desligamento da interface sem fio do nó. Além disso, verificamos que o próprio tipo de roteamento pode contribuir para uma diminuição do gasto energético em números de transmissões de mensagens.

Com os resultados obtidos pôde-se constatar como o emprego de mecanismos de gerência de recursos em RTAIs, como o uso da lista de *acks* e o encaminhamento proativo proposto, fornecem ganhos significativos. O uso da lista de *acks* e do encaminhamento proativo se mostraram maneiras eficientes de reduzir a quantidade de transmissões realizadas pelos nós e de reduzir o número de saltos na entrega de mensagens mantendo ou melhorando a taxa de entrega. Além do encaminhamento proativo apresentar-se como um bom mecanismo complementar a lista de acks fa-

vorecendo inclusive a um maior número de mensagens entregues reconhecidas pelos nós de origem.

Isto mostra a importância do emprego de tais mecanismos para a preservação dos recursos das RTAIs melhorando o gerenciamento dos mesmos. Assim como constatamos que com o uso de modos de gerenciamento de energia uma quantidade relevante de energia pôde ser poupada e convertida em um aumento do tempo operacional dos nós da rede. Favorecendo que mensagens antes não entregues, devido ao nó de destino não estar mais em funcionamento, possam ser entregues posteriormente.

Podemos também concluir que o encaminhamento seletivo baseado no histórico de encontros passados, que teve por base a tentativa de classificar o nó receptor como um mal retransmissor de mensagens, e desta forma, uma quantidade menor de mensagens forem repassadas, atingiu uma taxa de entrega satisfatória enquanto diminuiu o número de transmissões realizadas. Garantindo que os recursos como espaço e energia fossem melhores aproveitados.

Como trabalhos futuros pretende-se avaliar o impacto de outros tipos de mecanismos de preservação de recursos em RTAIs. Pretende-se também realizar o mesmo estudo de gerenciamento de energia em outros protocolos de roteamento, como por exemplo, o PROPHET. Além de variar o cenário de testes com nós apresentando uma maior mobilidade. E também estudar a eficácia de mesclar o uso dos mecanismos como lista de *acks* e o encaminhamento proativo em conjunto com os gerenciamentos de energia. Como também realizar combinações entre os gerenciamentos que sejam ativados de forma ortogonal, como os ativados quando estão sem vizinhos com o que verifica o meio ocupado. Pode-se explorar o parâmetro δ, que é um dos parâmetros que influenciam na classificação do nó, fazendo com que ele seja atualizado a partir do número médio de contatos do nó, e assim seu valor possa ser ajustado dinamicamente. Além disso, pode-se realizar uma análise da sensibilidade de variação do parâmetro δ.

Referências Bibliográficas

[1] LINDGREN, A., DORIA, A., SCHELN, O., "Probabilistic Routing in Intermittently Connected Networks". In: *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, July 2003.

[2] DTNRG, "The Delay-Tolerant Networking Research Group", `http://www.dtnrg.org`, 2002, [Último acesso: 13/08/2009].

[3] IRTF, "Internet Research Task Force", `http://www.irtf.org`, [Último acesso: 13/08/2009].

[4] RAMANATHAN, R., HANSEN, R., BASU, P., ROSALES-HAIN, R., KRISHNAN, R., "Prioritized Epidemic Routing for Opportunistic Networks". In: *Proceedings of the 1st International MobiSys Workshop on Mobile Opportunistic Networking (MobiOpp 2007)*, pp. 62–66, ACM: New York, NY, USA, 2007.

[5] CHEN, X., MURPHY, A. L., "Enabling Disconnected Transitive Communication in Mobile Ad Hoc Networks". In: *Proceedings of the Workshop on Principles of Mobile Computing (POMC), co-located with PODC*, Aug. 2001.

[6] GHOSH, J., NGO, H. Q., QIAO, C., "Mobility profile based routing within intermittently connected mobile ad hoc networks (ICMAN)". In: *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, pp. 551–556, ACM: New York, NY, USA, 2006.

[7] DE OLIVEIRA, C. T., *Uma Proposta de Roteamento Probabilístico para Redes Tolerantes a Atrasos e Desconexões*, Master's Thesis, Programa de Engenharia Elétrica - COPPE/UFRJ, 2008.

[8] PANAGAKIS, A., VAIOS, A., STAVRAKAKIS, I., "On the Effects of Cooperation in DTNs". In: *Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on*, pp. 1–6, Jan. 2007.

[9] BUTTYÁN, L., DÓRA, L., FÉLEGYHÁZI, M., VAJDA, I., "Barter trade improves message delivery in opportunistic networks", *Ad Hoc Networks*, 2009.

[10] SENDT, P., "Sensor Networking with Delay Tolerance", `http://down.dsg.cs.tcd.ie/sendt`, 2005, [Último acesso: 20/12/2006].

[11] OLIVEIRA, C. T., MOREIRA, M. D. D., RUBINSTEIN, M. G., COSTA, L. H. M. K., DUARTE, O. C. M. B., "Redes Tolerantes a Atrasos e Desconexões". In: *Minicursos do Simpósio Brasileiro de Redes de Computadores (SBRC 2007)*, May 2007.

[12] SHAH, R. C., ROY, S., JAIN, S., BRUNETTE, W., "Data MULEs: Modeling a threetier architecture for sparse sensor networks". In: *IEEE Workshop on Sensor Network Protocols and Applications (SNPA), 2003*, May 2003.

[13] HASSON, A. A., FLETCHER, R., PENTLAND, A., "DakNet: A road to universal broadband connectivity. Wireless Internet UN ICT Conference Case Study", `http://www.medialabasia.org/`, 2003.

[14] TIER, P., "Technology and Infrastructure for Emerging Regions", `http://tier.cs.berkeley.edu/wiki/Home`, 2004, [Último acesso: 11/08/2009].

[15] "Wizzy Digital Courier", `http://www.wizzy.org.za`, 2007, [Último acesso: 06/12/2007].

[16] LINDGREN, A., DORIA, A., "Experiences from Deploying a Real-Life DTN System". In: *Consumer Communications and Networking Conference, 2007. CCNC 2007. 4th IEEE*, pp. 217–221, Jan. 2007.

[17] ZHAO, W., AMMAR, M. H., "Message ferrying: Proactive routing in highly partitioned wireless ad hoc networks". In: *FTDCS*, 2003.

[18] ZHAO, W., AMMAR, M. H., ZEGURA, E., "A message ferrying approach for data delivery in sparse mobile ad hoc networks". In: *MobiHoc*, 2004.

[19] ZHAO, W., AMMAR, M. H., ZEGURA, E., "Controlling the mobility of multiple data transport ferries in a delay-tolerant network". In: *MobiHoc*, March 2005.

[20] FARREL, S., CAHILL, V., (eds), *Delay- and Disruption-Tolerant Networking*. Artech House, 2006.

[21] JUANG, P., OKI, H., WANG, Y., MARTONOSI, M., PEH, L. S., RUBENSTEIN, D., "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet", *SIGPLAN Not.*, v. 37, n. 10, pp. 96–107, 2002.

[22] ZHANG, P., SADLER, C. M., LYON, S. A., MARTONOSI, M., "Hardware design experiences in ZebraNet". In: *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 227–238, ACM: New York, NY, USA, 2004.

[23] CHOU, J., OBRACZKA, K., "CARNIVORES Project at University of California - Santa Cruz", `http://surf-it.soe.ucsc.edu/node/39`, [Último acesso: 21/08/2009].

[24] SOZER, E., STOJANOVIC, M., PROAKIS, J., "Underwater acoustic networks", *IEEE Journal of Oceanic Engineering*, v. 25, pp. 72–83, 2000.

[25] RICE, J., "Seaweb Acoustic Communication and Navigation Networks". In: *International Conference on Underwater Acoustic Measurements: Technologies and Results*, Grécia, 2005.

[26] FALL, K., "A delay-tolerant network architecture for challenged internets", *ACM SIGCOMM*, pp. 27–34, 2003.

[27] JUN, H., *Power Management in Disruption Tolerant Networks*, Master's Thesis, College of Computing Georgia Institute of Technology, 2007.

[28] FALL, K., HONG, W., MADDEN, S., *Custody Transfer for Reliable Delivery in Delay Tolerant Networks*, Tech. rep., Intel Research, 2003.

[29] CERF, V., BURLEIGH, S., HOOKE, A., TORGERSON, L., DURST, R., SCOTT, K., FALL, K., WEISS, H., "Delay-tolerant networking architecture - RFC4838", 2007.

[30] SELIGMAN, M., FALL, K., MUNDUR, P., "Alternative custodians for congestion control in delay tolerant networks". In: *CHANTS '06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pp. 229–236, ACM: New York, NY, USA, 2006.

[31] OLIVEIRA, C. T., TAVEIRA, D. M., BRAGA, R. B., DUARTE, O. C. M. B., "Uma Proposta de Roteamento Probabilístico para Redes Tolerantes a Atrasos e Desconexões". In: *XXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2008)*, Rio de Janeiro, RJ, Brazil, May 2008.

[32] MEDEIROS, M. V. B., SALLES, R. M., *Redes Tolerantes a Atrasos e Desconexões em Sistemas de Comunicações Militares*, Tech. rep., Instituto Militar de Engenharia - IME, Seção de Engenharia de Computação, 2009.

[33] WARTHMAN, F., "Delay-tolerant Networks (DTNs), A Tutorial", `http://www.dtnrg.org/wiki/Docs`, 2003, [Último acesso: 06/04/2009].

[34] HUI, P., CHAINTREAU, A., SCOTT, J., GASS, R., CROWCROFT, J., DIOT, C., "Pocket switched networks and human mobility in conference environments". In: *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 244–251, ACM: New York, NY, USA, 2005.

[35] LEGUAY, J., LINDGREN, A., SCOTT, J., FRIEDMAN, T., CROWCROFT, J., "Opportunistic Content Distribution in an Urban Setting". In: *Procee-*

dings of SIGCOMM Workshop on Challenged Networks (CHANTS 2006), pp. 205–212, ACM: New York, NY, USA, 2006.

[36] JONES, E. P., WARD, P. A., "Routing strategies for delay-tolerant networks", *Computer Communication*, 2006.

[37] ERRAMILLI, V., CROVELLA, M., "Forwarding in opportunistic networks with resource constraints". In: *CHANTS '08: Proceedings of the third ACM workshop on Challenged networks*, pp. 41–48, ACM: New York, NY, USA, 2008.

[38] SMALL, T., HAAS, Z., "Resource and Performance Tradeoffs in Delay-Tolerant Wireless Networks". In: *Proceedings ACM WDTN*, Aug. 2005.

[39] B. BURNS, O. B., LEVINE, B. N., "MV Routing and Capacity Building in Disruption Tolerant Networks". In: *Proceeding of IEEE INFOCOM*, March 2005.

[40] BURGUESS, J., GALLAGHER, B., JENSEN, D., LEVINE, B., "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks". In: *Proceedings of IEEE INFOCOM*, April 2006.

[41] BOICE, J., GARCIA-LUNA-ACEVES, J., OBRACZKA, K., "On-Demand Routing in Disrupted Environments". In: *Proceedings of IFIP Networking 2007*, pp. 155–166, Springer, 2007.

[42] BALASUBRAMANIAN, A., LEVINE, B., VENKATARAMANI, A., "DTN Routing as a Resource Allocation Problem". In: *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 373–384, ACM: New York, NY, USA, 2007.

[43] CHANGLING, L., KAISER, J., *A Survey of Mobile Ad Hoc network Routing Protocols*, Tech. rep., University of Magdeburg, 2005.

[44] ABOLHASAN, M., WYSOCKI, T., DUTKIEWICZ, E., "A review of routing protocols for mobile ad hoc networks", *Ad Hoc Networks*, v. 2, n. 1, pp. 1–22, 2004.

[45] NUNES, C. M., DOTTI, F. L., "Uma Nova Estratégia de Roteamento para Redes Tolerantes a Atrasos". In: *XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2009)*, Recife, PE, Brasil, May 2009.

[46] OTT, J., KUTSCHER, D., DWERTMANN, C., "Integrating DTN and MANET routing". In: *CHANTS '06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pp. 221–228, ACM: New York, NY, USA, 2006.

[47] ZHANG, Z., "Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges", *IEEE Communication Surveys and Tutorials*, 2006.

[48] JAIN, S., FALL, K., PATRA, R., "Routing in a delay tolerant network". In: *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2004)*, 2004.

[49] CONAN, V., LEGUAY, J., FRIEDMAN, T., "Fixed point opportunistic routing in delay tolerant networks", *Selected Areas in Communications, IEEE Journal on*, v. 26, n. 5, pp. 773–782, June 2008.

[50] MITCHNER, W., VADHAT, A., *Epidemic Routing for Partially Connected Ad Hoc Networks*, Tech. rep., Duke University, 2000.

[51] JONES, E. P. C., LI, L., WARD, P. A. S., "Practical Routing in Delay-Tolerant Networks". In: *Proceedings of the ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN 2005)*, aug 2005.

[52] SPYROPOULOS, T., PSOUNIS, K., RAGHAVENDRA, C. S., "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks". In: *Proceedings of ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN 2005)*, pp. 252–259, ACM: New York, NY, USA, 2005.

[53] ABDULLA, M., SIMON, R., "The Impact of the Mobility Model on Delay Tolerant Networking Performance Analysis". In: *Simulation Symposium, 2007. ANSS '07. 40th Annual*, pp. 177–184, March 2007.

[54] SPYROPOULOS, T., PSOUNIS, K., RAGHAVENDRA, C. S., "Efficient Routing in Intermittently Connected Mobile Networks: The Multiple-Copy Case", *IEEE/ACM Transactions on Networking*, v. 16, n. 1, pp. 77–90, 2008.

[55] SPYROPOULOS, T., MEMBER, S., PSOUNIS, K., RAGHAVENDRA, C., "Single-copy routing in intermittently connected mobile networks". In: *In Proceedings of IEEE SECON*, 2004.

[56] HARRAS, K., ALMEROTH, K., BELDING-ROYER, E., "Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding Schemes in Sparse Mobile Networks". In: *International Conferences on Networking (IFIP 2005)*, 2005.

[57] DAVIS, J. A., FAGG, A. H., LEVINE, B. N., "Wearable Computers as Packet Transport Mechanisms in Highly-Partitioned Ad-Hoc Networks". In: *Proceedings of the International Symposium on Wearable Computers (IEEE ISWC)*, Oct. 2001.

[58] LINDGREN, A., PHANSE, K. S., "Evaluation of Queueing Policies and Forwarding Strategies for Routing in Intermittently Connected Networks". In: *Proceedings of the International Conference on Communication System Software and Middleware (Comsware 2006)*, 2006.

[59] JUN, H., AMMAR, M. H., CORNER, M. D., ZEGURA, E. W., "Hierarchical power management in disruption tolerant networks with traffic-aware optimization". In: *CHANTS '06: Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pp. 245–252, ACM: New York, NY, USA, 2006.

[60] JUN, H., AMMAR, M., ZEGURA, E., "Power management in delay tolerant networks: a framework and knowledge-based mechanisms", *Sensor and*

Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on, pp. 418–429, Sept., 2005.

[61] BANERJEE, N., CORNER, M., LEVINE, B., "An Energy-Efficient Architecture for DTN Throwboxes", *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 776–784, may 2007.

[62] ZHAO, W., CHEN, Y., AMMAR, M., CORNER, M. D., LEVINE, B. N., ZEGURA, E., "Capacity Enhancement using Throwboxes in DTNs". In: *Proceedings of IEEE International Conf on Mobile Ad hoc and Sensor Systems (MASS)*, pp. 31–40, October 2006.

[63] DUTTA, P., CULLER, D., "Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications". In: *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pp. 71–84, ACM: New York, NY, USA, 2008.

[64] LU, K., LIN, X., "An Energy-Efficient Scheme of Deploying Nodes in Wireless Sensor Networks". In: *ICNC '07: Proceedings of the Third International Conference on Natural Computation*, pp. 786–790, IEEE Computer Society: Washington, DC, USA, 2007.

[65] GUEDES, R. M., DA SILVA, M. W. R., DE REZENDE, J. F., "Escassez de Recursos em Redes Tolerantes a Atrasos e Interrupções". In: *XXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2008)*, Rio de Janeiro, RJ, Brasil, May 2008.

[66] WANG, Y., WU, H., LIN, F., TZENG, N.-F., "Protocol Design and Optimization for Delay/Fault-Tolerant Mobile Sensor Networks". In: *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, p. 7, IEEE Computer Society: Washington, DC, USA, 2007.

[67] NELSON, S., BAKHT, M., KRAVETS, R., "Encounter-Based Routing in DTNs". In: *IEEE Infocom 2009*, Rio de Janeiro, RJ, Brasil, Apr 2009.

[68] DE OLIVEIRA, E. C. R., DE ALBUQUERQUE, C. V. N., "Análise do Protocolo NECTAR em Cenários com Mobilidade e Frequentes Interrupções". In: *XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2009)*, Recife, PE, Brasil, May 2009.