Pós-Graduação em Ciência da Computação

"Um Sistema de Disseminação de *Pieces* para a Melhoria do Desempenho de Aplicações BitTorrent sobre MANETs"

Por

Nivia Cruz Quental

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, SETEMBRO/2009

Universidade Federal de Pernambuco

CENTRO DE INFORMÁTICA

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Nivia Cruz Quental

"Um Sistema de Disseminação de Pieces para a Melhoria do Desempenho de Aplicações BitTorrent sobre MANETs"

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR(A): ***Prof. Paulo André da Silva Gonçalves***

RECIFE, SETEMBRO/2009

Quental, Nivia Cruz

Um sistema de disseminação de pieces para a melhoria do desempenho de aplicações bittorrent sobre manets / Nivia Cruz Quental. - Recife: O Autor, 2009.

xvii, 79 folhas : il., fig.

Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2009.

Inclui bibliografia, glossário e apêndice.

1. Redes de computadores. 2. Redes sem fio. I. Título.

004.6 CDD (22. ed.) MEI2009- 119

Dissertação de Mestrado apresentada por **Nivia Cruz Quental** Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"Um Sistema de Disseminação de Pieces para a Melhoria do Desempenho de Aplicações Bittorrent sobre Manets"**, orientado pelo **Prof. Paulo André da Silva Gonçalves** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Carlos André Guimarães Ferraz
Centro de Informática / UFPE

Prof. Artur Ziviani
Laboratório Nacional de Computação Científica

Prof. Paulo André da Silva Gonçalves
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 08 de setembro de 2009.

Prof. Nelson Souto Rosa
Vice-Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Esta dissertação é dedicada a todos aqueles que, apesar dos entraves pessoais e burocráticos, persistem no empenho em produzir pesquisa realmente relevante para o progresso da ciência, tecnologia, e , consequentemente, da humanidade.

AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus, pela vida, por estar sempre guardando meus caminhos, acalentando-me nas horas de aflição e soprando em meu coração qual o melhor caminho a seguir nos momentos decisivos.

À minha família, meus pais Roberto e Ozita e meus irmãos Roberto Junior e Jamily, que sempre acreditaram em mim dando-me todo o apoio e carinho que uma pessoa precisa para realizar os seus sonhos.

Ao meu orientador, o professor *Docteur* Paulo André da Silva Gonçalves pela confiança no meu potencial dando-me autonomia para tomar decisões e por sempre se mostrar presente em todas as etapas deste trabalho. Agradeço também pelas preciosas orientações com respeito à minha postura de pesquisadora.

Ao Centro de Informática da Universidade Federal de Pernambuco, pela infraestrutura e por ser um Centro de Excelência, garantindo a conclusão de um mestrado de alta qualidade.

Aos colegas de jornada Átila, Bruno Almeida, Bruno Gentilini, Bruno de Jesus, Hermano e Rafael pelo companheirismo, pelas críticas construtivas e pela atenção ao presenciar minhas apresentações ao longo do curso. Agradeço também a Bruno de Jesus e seus colegas do LabCom-CTG/UFPE por permitir que eu usasse recursos do laboratório para realização de parte dos experimentos.

À Érica Teixeira, pela sua amizade, alegria e entusiasmo, sempre ouvindo os meus desabafos e apresentando uma visão otimista da vida. Uma amizade conquistada nos tempos difíceis do mestrado que pretendo levar sempre comigo.

A todos os companheiros do projeto CIn/Samsung, entre eles Adeline, Allan, Antônio, Fernando Buononato, Flávia, Mozart, Nancy, Patrícia, Thiago e Valquíria, pela torcida

e energia positiva para a conclusão de mais esta etapa da minha vida.

Aos amigos da graduação na UPE Adélia, Cleyton, Fernando, Heron, Marcelo e Milena pela amizade e por terem sido fundamentais na minha formação como pesquisadora. Aos amigos Alessandra, Aurora, Edson, Ewerson, George, Flávio, Juliana, Mércia, Patrícia, Paula, Sylvia e Vanessa pelos momentos de descontração e compreensão diante das limitações advindas de uma vida acadêmica.

"... ouvi em minha cabeça meu pai dizendo:

Levante-se quando cair

Mantenha-se orgulhoso quando for rebaixado

Todas as lições que você aprender

Não serão as que você planeja

Cada passo naquela montanha valerá a pena

Mas quando você caminhar pelo vale

Talvez você ande como um homem

E quando você prosseguir sem mim

Ande como um homem."

—BON JOVI (Walk Like a Man - tradução livre)

RESUMO

As redes *ad hoc* sem fio ou MANETs são conhecidas pela sua flexibilidade e independência de uma infraestrutura para coordenação de nós. A quantidade de aplicações apoiadas nesta arquitetura tem aumentado vertiginosamente nos últimos anos dado que operações de resgate, aplicações de conferências e de compartilhamento de arquivos encontraram nestas redes um ambiente favorável ao seu desenvolvimento.

Estas aplicações emergentes executadas no topo das MANETs funcionam em uma lógica *peer-to-peer*, uma vez que os *peers* da rede executam a mesma função e repassam informações seguindo algoritmos de roteamento no nível aplicação. Além disso, uma MANET compartilha vários pontos em comum com uma rede P2P, entre eles a descentralização e a flexibilidade mediante a entrada e saída de nós. Desta forma, entende-se que as aplicações desenvolvidas para MANETs podem ser inspiradas nos protocolos P2P utilizados com sucesso na Internet. Entretanto, as MANETs possuem uma natureza bastante diversa da Internet, uma vez que oferecem novos desafios, como a mobilidade dos nós e as limitações na largura de banda. Estes fatores levam ao fraco desempenho das aplicações P2P nesta arquitetura, em comparação ao desempenho das mesmas na Internet. Assim, melhorias destes protocolos destinadas ao ambiente sem fio se fazem necessárias.

Esta dissertação propõe um Sistema de Disseminação para otimização do protocolo BitTorrent no ambiente de MANETs. O sistema proposto consiste em estratégias específicas para disseminação de conteúdo e em uma interface que permite que algumas mensagens de `PIECE` do BitTorrent sejam entregues no modo *broadcast*, tirando proveito da natureza difusora do meio de comunicação sem fio.

O desempenho do BitTorrent tradicional e de sua versão melhorada com o Sistema

de Disseminação proposto foram avaliados através de simulações com o auxílio do NS-2. Os resultados demonstram que o uso do Sistema de Disseminação proposto permite uma redução de até 30% no tempo de *download* e uma redução de até 50% no *overhead* de segmentos de acordo com os cenários estudados.

Palavras-chave: BitTorrent, MANETs, Sistema de Disseminação, P2P

ABSTRACT

Ad hoc networks or MANETs are flexible and independent networks wich does not require an infrastructure for coordination of nodes. The number of applications using this architecture has increased in the last years. The develoment of applications for rescue operations, conferences and file sharing fits well in the wireless environment. These applications work in a P2P-like logic, since peers play the same role and forward information through routing algorithms in application level.

MANETs and P2P have many points in common, like descentralization and flexibility in topology changes. However, MANETs have a very different nature from the Internet, because the wireless medium introduces new chalenges, like nodes mobility and bandwidth limitations. These factors lead the P2P application to poor performances in this architecture, when compared to P2P protocols over the Internet.

In this work we propose a Dissemination System to optimize BitTorrent performance over MANETs. This system stablishes a set of enhancements for content dissemination. Also, it provides an interface to send some BitTorrent's `PIECE` messages in broadcast mode in order to take advantages from the broadcast nature of MANETs.

We evaluate the performance of the traditional BitTorrent and its modified version with our Dissemination System using the NS-2 simulator. Results points out that download time has decreased up to 30% and the segments overhead is reduced up to 50% against the approach which does not use our proposed system.

Keywords: BitTorrent, MANETs, Dissemination System, P2P

SUMÁRIO

LISTA DE FIGURAS

GLOSSÁRIO

ABR *Associativity-Based Routing*. 10

ACK *Acknowledgment*, ou reconhecimento de pacote recebido. 8

AODV *Ad-hoc On-Demand Distance Vector Routing*. 11

BMI *BitTorrent Mobile Interface*, interface de difusão de mensagens pertencente ao Sistema de Disseminação proposto. 4, 38, 40

CAN *Content Addressable Network*. 16

CGSR *Clusterhead Gateway Switch Routing*. 9

CSMA/CA *Carrier Sense Multiple Access/Collision Avoidance*, ou Acesso Múltiplo com Detecção de Portadora/Prevenção de Colisão. 7

CTS *Clear to Send*. 8

DHT *Distributed Hash Table*. 15

DIFS *Distributed Interframe Space*. 8

DSDV *Destination Sequenced Distance Vector*. 9

DSR *Dynamic Source Routing*. 11

DSSS *Direct Sequence Spread Spectrum*. 7

GSD *Group-based Service Discovery Protocol*. 32

HTTP *Hypertext Transfer Protocol*. 22

IEEE *Institute of Electrical and Electronics Engineers*. 7

IP *Internet Protocol*. 21

LAN *Local Area Network* ou rede local. 7

MAC *Medium Access Control*, ou Camada de Controle de Acesso ao Meio. 7

MANET *Mobile ad hoc Network* ou Rede Móvel *ad hoc*. 1, 6

MIMO *Multiple Input, Multiple Output*. 7

MPP *Mobile Peer to Peer*. 33

MPR *Multipoint Relay*. 10

NAV *Network Allocator Vector*. 8

nodeID Identificador de um nó em uma rede. 15

NS-2 *Network Simulator* versão 2. 45

OFDM *Orthogonal Frequency Division Multiplex*. 7

OLSR *Optimized Link State Routing*. 10

ORION *Optimized Routing Independent Overlay Network*. 33

P2P *Peer to Peer* ou par a par. 2, 14

PHY Camada física. 7

RFID *Radio-Frequency IDentification* ou identificação por rádio frequência. 1

RTS *Request to Send*. 8

SIFS *Short Interframe Space*. 8

SSR *Signal Stability Routing*. 10

TCP *Transmission Control Protocol*. 4, 13, 38

TORA *Temporary Ordered Routing Algorithm*. 10

UDP *User Datagram Protocol*. 4, 38

VANET *Vehicular ad hoc Network*. 13

VoIP *Voice over IP*. 14

WLAN *Wireless Local Area Network* ou rede local sem fio. 1, 7

WRP *Wireless Routing Protocol*. 10

ZRP *Zone Routing Protocol*. 12

CAPÍTULO 1

INTRODUÇÃO

O crescimento vertiginoso na variedade de aplicações móveis tem estimulado pesquisas para o desenvolvimento de protocolos otimizados para as arquiteturas de rede onde elas estão inseridas. As redes *ad hoc* sem fio, também conhecidas como MANETs (*Mobile ad hoc Networks*), constituem uma tecnologia adequada para ambientes onde não é viável ou conveniente a existência de uma infraestrutura centralizadora. As MANETs podem atender às necessidades de sistemas que vão desde aplicações de *groupware* à operações de resgate. Em geral, procura-se adaptar arquiteturas P2P já utilizadas na Internet para redes *ad hoc*, uma vez que as mesmas já possuem uma natureza *Peer to Peer*. Observa-se entretanto, que há a necessidade de se propor soluções otimizadas para redes *ad hoc*, uma vez que sobrepor aplicações P2P em MANETs sem qualquer otimização provoca impactos negativos no desempenho.

1.1 MOTIVAÇÃO

O século XX foi marcado pelo advento das primeiras tecnologias de comunicação sem fio, apoiadas pela descoberta da radiotelegrafia no fim do século XIX [1]. A evolução das redes de computadores e da Internet em paralelo ao progresso das telecomunicações ao longo dos anos teve como consequência a convergência dessas tecnologias. Atualmente, é possível contar com redes de dispositivos que trocam dados em um ambiente sem fio, como as redes de sensores, celulares, RFID, *bluetooth* e as WLANs (*Wireless Local Networks*). Dentro do universo das WLANs, destacam-se as redes *ad hoc* sem fio ou MANETs. Estas redes independem de uma infraestrutura física para o controle das comunicações entre os nós presentes, como um ponto de acesso para a rede externa [2]. A flexibilidade caracterizada

pela rápida reconfiguração e pela sua autonomia tem permitido o uso das MANETs em sistemas de resgate, conferências, aplicações de *groupware*, aplicações veiculares, entre outros.

Apesar das vantagens, a operação de uma MANET apresenta desafios advindos de sua simplicidade e dos meios físicos de propagação. A mobilidade dos nós, as interferências no canal de comunicação e as limitações de energia e memória têm impacto significativo no desempenho da rede em comparação com uma rede tradicional.

O nicho de aplicações para MANETs compreende cenários mais reduzidos que os das aplicações destinadas à Internet. Isto significa que estas aplicações podem sofrer queda de performance quando sobrepostas diretamente em MANETs. Entretanto, é possível criar soluções para estas redes inspirando-se em modelos de comunicação já estabelecidos com sucesso na Internet, como é o caso das redes de compartilhamento P2P, ou *Peer to Peer*.

Entre os protocolos propostos para P2P na Internet, o BitTorrent ganhou destaque no meio acadêmico nos últimos anos devido à sua eficiência no *download* de conteúdos extensos e por procurar promover justiça no gerenciamento das conexões. Seu uso em MANETs tem sido objeto de estudo de diversas pesquisas, uma vez que vislumbra-se o compartilhamento de arquivos grandes em aplicações que envolvem as redes *ad hoc*. Por exemplo, em cenários de resgate e até mesmo em conferências geralmente deseja-se compartilhar vídeos (filmagens no primeiro caso e anúncios no segundo), que, mesmo de curta duração, podem requerer centenas de *megabytes* para armazenamento.

As redes P2P e as MANETs possuem em comum propriedades de descentralização e de auto-organização, além do dinamismo da topologia e a realização de tarefas de roteamento. As semelhanças entre MANETs e redes P2P têm culminado em pesquisas que buscam sinergias entre estas duas tecnologias objetivando melhorar o desempenho de aplicações P2P onde a adoção de redes sem fio se faz necessária [3].

A forma como a aplicação P2P é adaptada à MANET influencia significativamente no desempenho geral do sistema. Assim, o sucesso da adaptação de protocolos P2P já existentes na Internet para as redes *ad hoc* sem fio dependem desta decisão arquitetural.

1.2 OBJETIVOS

Esta dissertação tem como objetivo geral propor um Sistema de Disseminação para obter um compartilhamento mais eficiente de arquivos no BitTorrent em MANETs. Neste âmbito, entende-se como eficiência a melhoria nos tempos de *download* e um menor *overhead* de segmentos, ou seja, menos pacotes no nível de transporte, em comparação à abordagem tradicional. Considera-se a abordagem tradicional como aquela sem qualquer tipo de alteração para as redes *ad hoc*.

Para alcançar este objetivo geral, os seguintes objetivos específicos são definidos:

1. Destacar as limitações do uso tradicional do BitTorrent em MANETs, através de estudos de trabalhos relacionados e de um estudo de simulação;

2. Identificar possíveis pontos de melhoria de desempenho e propor um Sistema de Disseminação simples e eficiente para otimizar o *download* no BitTorrent em um ambiente de MANETs, indicando os principais pontos de otimização;

3. Realizar experimentos de simulação e comprovar a eficiência da proposta através da análise dos resultados em comparação à abordagem tradicional.

1.3 O SISTEMA DE DISSEMINAÇÃO PROPOSTO

O presente trabalho propõe um Sistema de Disseminação, que visa tirar vantagem da natureza difusora das MANETs com a escolha de um *peer* para disseminar algumas mensagens de `PIECE` do BitTorrent em modo *broadcast*. Nesta proposta, um *peer* pode assumir dois papéis:

- *Peer* Disseminador - escolhe algumas mensagens de `PIECE` do BitTorrent para serem distribuídas via *broadcast* periodicamente;

- *Peer* comum - não realiza distribuições e recebe o conteúdo vindo do disseminador e aquele obtido por suas requisições.

O Sistema de Disseminação conta com um conjunto de estratégias para escolher quais mensagens de `PIECE` do BitTorrent devem ser transmitidas via *broadcast* de modo a tornar mais eficiente o processo de obtenção do arquivo. Este sistema ainda propõe uma interface denominada BMI (*BitTorrent Mobile Interface*), sendo uma sub-camada de aplicação responsável pela disseminação de conteúdo via *broadcast*. A BMI permite que algumas mensagens de `PIECE` do BitTorrent possam ser disseminadas pelo transporte não-confiável do UDP, ao invés de usar o transporte confiável do TCP, que é um protocolo exclusivamente ponto-a-ponto. Este último ainda é utilizado no transporte das demais mensagens.

O objetivo da disseminação é a redução do envio de requisições por conteúdo durante o *download* de um arquivo. Isto é possível graças a disseminação de algumas mensagens de `PIECE`, fazendo com que não haja a necessidade de requisição por estas mensagens por parte dos destinatários. Assim, espera-se obter uma redução no tempo de *download* em comparação ao desempenho observado sem o Sistema de Disseminação proposto.

Esta proposta enfoca a otimização do processo de *download* em si, de modo que a escolha e localização de nós não fazem parte do escopo deste trabalho, bem como mecanismos de segurança e reputação.

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

O Capítulo 2 descreve o funcionamento das redes *ad hoc* sem fio destacando padrões, protocolos de roteamento, questões relacionadas à camada de transporte e principais àreas de aplicação. As redes P2P são apresentadas de forma geral no Capítulo 3 com destaque ao protocolo BitTorrent. O Capítulo 4 discute as sinergias entre protocolos P2P e as MANETs, bem como as formas de modificar protocolos P2P para aplicações em MANETs encontradas na literatura e em um estudo preliminar do comportamento do BitTorrent sobre MANETs. O Capítulo 5 propõe um Sistema de Disseminação como solução simples e eficiente para melhorar o desempenho do BitTorrent em uma rede *ad hoc*. O Capítulo 6 traz uma discussão acerca dos resultados obtidos na aplicação do Sistema de Disseminação em comparação com a abordagem tradicional, além de descrever

como os experimentos foram conduzidos. Finalmente, o Capítulo 7 faz reflexões sobre os resultados apresentados e aponta os trabalhos futuros advindos do presente estudo.

CAPÍTULO 2

REDES AD HOC MÓVEIS (MANETS)

A arquitetura de uma rede sem fio pode ser infraestruturada ou *ad hoc*. No primeiro caso, a rede conta com uma infraestrutura que possui funcionalidade de ponto de acesso, ou seja, uma entidade centralizadora que coordena as comunicações no nível de enlace. No segundo caso, a rede se forma espontaneamente, sem auxílio de um ponto de acesso e são denominadas MANETs (*Mobile ad hoc Networks*). Graças à sua simplicidade e versatilidade, as MANETs ganham cada vez mais espaço na implantação de sistemas ubíquos. Contudo, apresentam problemas que podem limitar a sua aplicabilidade para alguns cenários. Este capítulo apresenta as características gerais relacionadas às MANETs, bem como os principais protocolos de roteamento utilizados.

2.1 CARACTERÍSTICAS DAS MANETS

As MANETs podem ser definidas como sistemas autônomos de nós móveis [2]. Estas redes não necessitam de uma infraestrutura prévia para seu funcionamento e são formadas apenas para um propósito específico e em caráter tipicamente temporário. A topologia da rede pode ser fixa ou dinâmica de acordo com a mobilidade do nós e de outros fatores, como o balanceamento de energia e de carga. O propósito de uma MANET é prover funcionalidades de roteamento para nós em uma rede sem fio. Estas redes são caracterizadas por uma topologia de múltiplos saltos (ou *multi-hop*), na qual nós podem se comunicar com outros que estejam fora de seu alcance através de nós intermediários alcançáveis.

Cada nó em uma MANET é um dispositivo composto de transceptores sem fio com antenas omnidirecionais ou direcionais. Os protocolos projetados para MANETs precisam levar em consideração alguns aspectos físicos da rede, como o nível de energia e de

interferência entre canais, que são características determinantes no comportamento das mesmas. Em alguns cenários, como em certas redes de sensores, não é possível trocar ou recarregar a bateria dos dispositivos, sendo desejável que o consumo de energia seja limitado. Adicionalmente, a questão das colisões originadas pelas tentativas de acesso simultâneo ao meio por parte dos nós contribui decisivamente para uma maior perda de pacotes em relação às redes cabeadas.

2.2 O PADRÃO IEEE 802.11

O IEEE 802.11 é um padrão que define a camada física (PHY) e o controle de acesso ao meio (MAC) em WLANs. Este padrão prevê o suporte tanto a redes com infraestrutura quanto a redes *ad hoc*. São definidos três padrões para a camada física:

- 802.11a - Opera na faixa de 5GHz e determina taxas de transmissão de até 54 Mbps utilizando o OFDM (*Orthogonal Frequency Division Multiplex*) como técnica de modulação;
- 802.11b - Tornou-se mais popular que este primeiro por ter sido finalizado anteriormente e garante taxas de até 11Mbps com a técnica de modulaçao DSSS (*Direct Sequence Spread Spectrum*) na faixa de 2.4GHz;
- 802.11g - Compatível com o 802.11b, esta padronização consegue atingir taxas de até 54Mbps na faixa de 2.4GHz com o esquema de modulação OFDM.

Especificações adicionais foram definidas pelo IEEE a fim de definir segurança, qualidade de serviço e suporte à tecnologia MIMO (*Multiple Input Multiple Output*) com os padrões 802.11i, 802.11e e 802.11n, respectivamente.

Na camada enlace das redes sem fio as colisões precisam ser tratadas, uma vez que a transmissão simultânea de quadros em uma mesma região de alcance causa prejuízos à recepção. O protocolo MAC é de acesso múltiplo com detecção de portadora e prevenção de colisões (CSMA/CA - *Carrier Sense Multiple Access/ Collision Avoidance*). Isto significa que o padrão IEEE 802.11 não dá suporte à detecção de colisões na camada

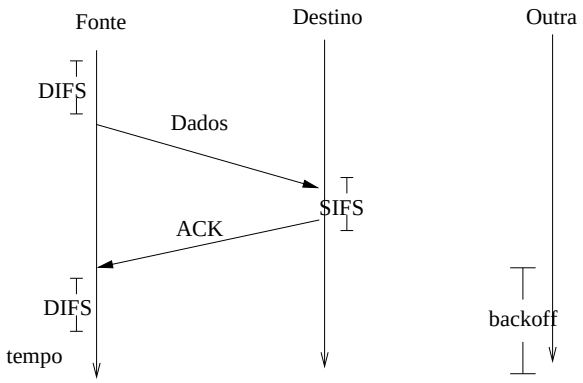


Figura 2.1 Transmissão de quadros no CSMA/CA

enlace, mas possui mecanismos para minimizar a probabilidade das mesmas ocorrerem [4].

No CSMA/CA, se o canal estiver ocioso por um período maior ou igual a DIFS (*Distributed Interframe Space*) é permitido que a estação transmita. Se nenhuma estação interferir, o quadro é recebido com sucesso. Ao receber o quadro, o receptor aguarda um tempo SIFS (*Short Interframe Space*) e envia um quadro de ACK para reconhecimento. Se o transmissor percebesse o canal ocupado, adiaria seu acesso até encontrar o canal disponível pelo tempo DIFS. Além disso, calcularia um tempo adicional aleatório, conhecido como *backoff*, para finalmente realizar a transmissão. Um exemplo pode ser observado na Figura 2.1.

O IEEE 802.11 pode usar quadros de controle, como o RTS (*Request to Send*) e o CTS (*Clear to Send*) para reservar o canal para o transmissor, diminuindo a probabilidade de colisões. Quando deseja enviar um quadro, a estação manda um RTS indicando o tamanho dos dados e do quadro de ACK. O receptor responde com um CTS, autorizando o envio. Ao perceberem os quadros de RTS e CTS, os demais evitam fazer transmissões e aguardam o fim da transmissão com auxílio de um vetor de alocação de rede (NAV - *Network Allocator Vector*) que guarda a informação de tamanho dos dados. A Figura 2.2 ilustra este fluxo.

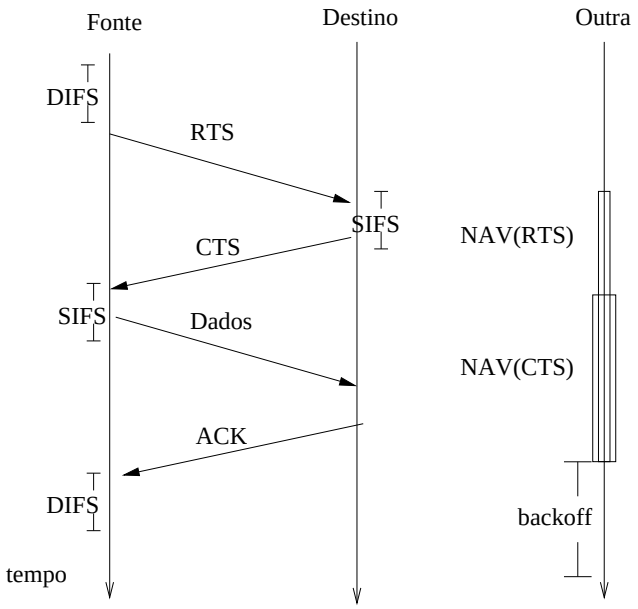


Figura 2.2 Transmissão de quadros no CSMA/CA com reserva de canal

2.3 ROTEAMENTO EM MANETS

O roteamento em uma MANET pode ser pró-ativo, reativo ou híbrido. Os protocolos de roteamento pró-ativos mantêm tabelas de roteamento atualizadas por meio de distribuição periódica de informações de rotas através da rede. Os protocolos reativos encontram rotas sob demanda inundando a rede com requisições. Os protocolos híbridos procuram combinar as características dos pró-ativos e reativos. Uma taxonomia pode ser encontrada nos trabalhos de Royer *et al.* [5] e Requena *et al.* [6]. A seguir, são apresentados alguns destes protocolos.

Protocolos Pró-ativos

- DSDV (*Destination Sequenced Distance Vector Routing*) - Baseia-se no algoritmo Bellman-Ford e procura solucionar problemas de formação de *loops* na rede através do uso de números de seqüência para distinguir rotas [7];

- CGSR (*Clusterhead Gateway Switch Routing*) - Sua arquitetura se baseia em *clusters*, cada qual com seu *clusterhead*. Ele tem acesso a todos os nós do *cluster*. O

acesso aos outros *clusters* é garantido por um *gateway* [8];

- WRP (*Wireless Routing Protocol*) - Este protocolo é similar ao DSDV porém cada nó utiliza quatro tabelas para manter informações da rede: tabela de distância, tabela de roteamento, tabela de custo e tabela de retransmissão [9];

- OLSR (*Optimized Link State Routing*) - É uma otimização de um protocolo de estado de enlace puro e herda suas características de estabilidade [10]. As mensagens de manutenção do roteamento são reduzidas graças ao mecanismo de *Multipoint Relays* (MPRs). Um conjunto de MPRs de um nó é formado pelos vizinhos cuja distância para este nó é de apenas um salto e que possuem *links* bidirecionais para todos aqueles que estiverem a uma distância de dois saltos daquele nó [11]. Os vizinhos com distância de um salto, mas que não participam do conjunto de MPRs podem ler e processar os pacotes que receberem, porém não podem retransmiti-los. Assim, os MPRs são os únicos responsáveis pela criação das rotas. O conjunto de MPRs é recalculado quando mudanças nas vizinhanças são detectadas, como falha em *links* ou a entrada de um novo vizinho apto a pertencer ao conjunto. Esta atualização também pode ocorrer por quebra de *links* na vizinhança a dois saltos de distância. A Figura 2.3(a) ilustra uma rede sem MPRs. A Figura 2.3(b) mostra a redução da inundação em uma rede com MPRs.

Protocolos Reativos

- TORA (*Temporary Ordered Routing Algorithm*) - Estabelece um grafo acíclico, onde cada nó mantém informação sobre seus vizinhos, podendo criar, manter e deletar rotas [12];

- ABR (*Associativity-Based Routing*) - A seleção de rotas neste protocolo é feita através de uma métrica chamada *grau de estabilidade*. Quanto maior o grau, menor a mobilidade. Nós anunciam sua presença por meio de sinais de *beacon* [13];

- SSR (*Signal Stability Routing*) - Divide-se em em dois sub-protocolos: estático e dinâmico. O estático objetiva receber e encaminhar pacotes. O dinâmico lida com

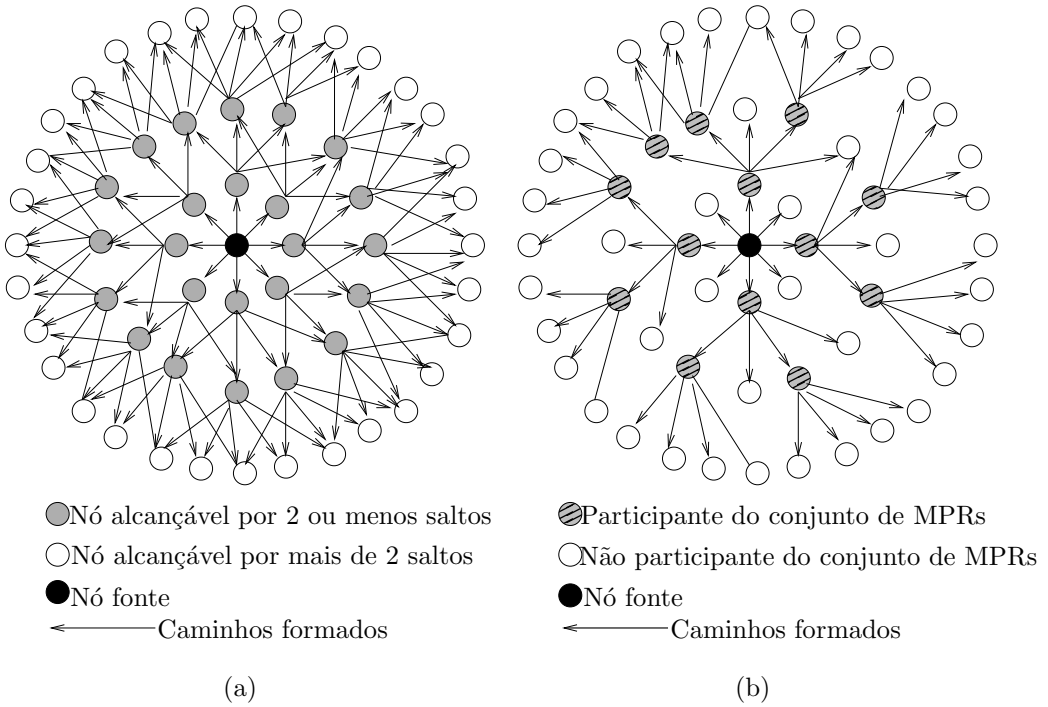


Figura 2.3 (a) Rede sem MPRs (b) Rede com MPRs

estabilidade da rede e tabelas de roteamento [14];

- DSR (*Dynamic Source Routing*) - As rotas são descobertas através do envio de mensagens de requisição as quais são repassadas ao longo da rede até encontrar o seu destino. O protocolo mantém as rotas descobertas em *cache* [15];

- AODV (*Ad-hoc On-Demand Distance Vector Routing*) - É um protocolo de roteamento reativo que procura minimizar o número de *broadcasts* necessários para criação rotas sem manter uma lista completa de roteamento como nos protocolos proativos [16]. A descoberta de caminhos é feita sob demanda. Pacotes `RREQ` (*Route Request*) são usados para o envio de requisições de rota para o nó de destino ou um intermediário que conheça uma rota atualizada de encaminhamento. Um pacote `RREQ` é enviado por meio de inundação, conforme ilustra a Figura 2.4(a). Um pacote `RREP` (*Route Response*) é usado para dar uma resposta e é enviado em *unicast* ao remetente pelo caminho reverso do `RREQ`, como ilustra a Figura 2.4(b). Desta forma, é fundamental que os *links* sejam simétricos. Caso isto não seja possível, uma

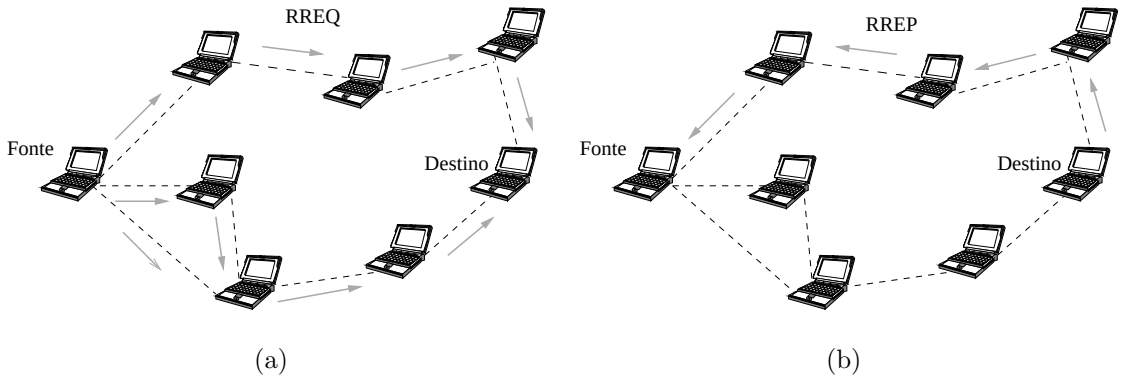


Figura 2.4 Descoberta de rotas com o AODV. (a) Remetente procura por destinatário através de *flooding* (b) Resposta do destino segue o caminho reverso do RREQ recebido

mensagem adicional `RREP-ACK` pode ser enviada em resposta a `RREP` para detectar possíveis problemas de assimetria. Quando um nó detecta uma rota "quebrada", ele envia mensagens de erro `RERR` para informar seus vizinhos. Um número de sequência atribuído a cada nó permite a atualização de caminhos na tabela de roteamento. Cada nó deve manter na sua tabela de roteamento o número de seqüência dos nós de destino e atualizá-lo sempre que receber mensagens de resposta com número de seqüência maior que o anterior. Assim, a formação de *loops* é evitada. Outra grande vantagem do AODV é a inexistência de entidades centrais e a redução do tráfego de mensagens, uma vez que as requisições são feitas apenas sob demanda.

Protocolos Híbridos

- ZRP (*Zone Routing Protocol*)- Este protocolo procura conciliar as vantagens de ambas abordagens, sugerindo uma solução híbrida, sendo pró-ativo para nós na vizinhança e reativo para comunicação entre vizinhanças. Trata-se na verdade de um *framework* que permite utilizar os protocolos reativos e pró-ativos anteriormente citados [17].

2.4 TRANSPORTE E APLICAÇÕES EM MANETS

Acima dos protocolos de rede, as MANETs podem contar com o protocolo de transporte confiável TCP [18]. Sabe-se, porém, que, em algumas situações, a implementação tradi-

cional do TCP não tem bom comportamento mediante ambientes sem fio por usar perdas de pacotes como indicadores de congestionamento [19]. Desta forma, o mecanismo de controle de congestionamento TCP pode ser invocado em situações onde há perdas causadas pela mobilidade dos nós ou por interferências, e não apenas por congestionamento, podendo resultar em uma desnecessária degradação do *throughput*. Existem trabalhos que procuram resolver este problema como em [20], onde propõe-se um mecanismo de controle de congestionamento TCP baseado em informações de roteamento. O protocolo UDP, por outro lado, não apresenta este problema, uma vez que não prevê controle de congestionamento em sua especificação original. Contudo, é um protocolo de transporte não-confiável, não dando garantias de entrega dos pacotes. Dependendo do tipo de solução proposta para a MANET, é possível que ambos TCP e UDP convivam em uma mesma aplicação, aproveitando-se as vantagens de cada um.

O nicho de aplicações que funcionam em MANETs inclui troca de arquivos, jogos em rede, sensoriamento de ambientes e trabalhos colaborativos em redes sociais, entre outras. Uma área emergente é a distribuição e o compartilhamento de conteúdo em redes veiculares, conhecidas como VANETs [21]. Aplicações militares e de resgate também impulsionam o desenvolvimento de novas soluções para as redes *ad hoc* [22].

2.5 RESUMO

As MANETs são redes que não requerem um ponto de acesso para coordenação dos nós da rede. A sua topologia flexível permite vislumbrar aplicações em lugares remotos ou onde não há infraestrutura de rede pré-concebida, como em operações de resgate. Também pode-se imaginar o uso de MANETs onde a comunicação ocorre em caráter temporário, como em conferências. O amadurecimento das pesquisas em torno de protocolos de enlace e roteamento para estas redes tem levado ao surgimento de adaptações dos protocolos de transporte e de aplicação para MANETs.

CAPÍTULO 3

REDES PEER TO PEER (P2P)

Na camada aplicação, os protocolos colaborativos, como aqueles desenvolvidos para arquitetura P2P, são os mais adequados para gerenciar comunicação de alto nível em MANETs. Entre os protocolos P2P destaca-se o BitTorrent, que ganhou destaque no meio acadêmico nos últimos anos devido à sua eficiência no *download* de conteúdos extensos e procurar promover justiça no gerenciamento das conexões. Este capítulo apresenta os tipos de redes P2P e os protocolos mais relevantes, com destaque para o protocolo BitTorrent.

3.1 GERAÇÕES DE REDES P2P

Assim como uma MANET, uma rede P2P (*Peer to Peer*) [23] é um sistema distribuído, onde seus nós se comunicam em uma rede de topologia dinâmica e por um tempo transiente. Na Internet, estas redes são conhecidas como redes *overlays*, uma vez que seus protocolos permitem que as funcionalidades de encaminhamento sejam implementadas na camada aplicação ou acima desta, formando uma espécie de sub-rede virtual. Isto permite que detalhes da topologia física da rede sejam transparentes aos usuários de P2P. Estas redes vem sendo intensamente utilizadas desde o fim dos anos 90 e isto ocorreu graças ao seu grande sucesso entre usuários de aplicações de compartilhamento de arquivos e de Voz sobre IP (VoIP).

As redes P2P podem ser divididas em três gerações ao longo da história da Internet, de acordo com suas arquiteturas. A primeira é caracterizada pelo uso de indexadores de arquivos centralizados. O Napster é o exemplo mais conhecido desta geração. No contexto do Napster, *peers* precisam contactar um servidor que indique quais *peers* dispõem do

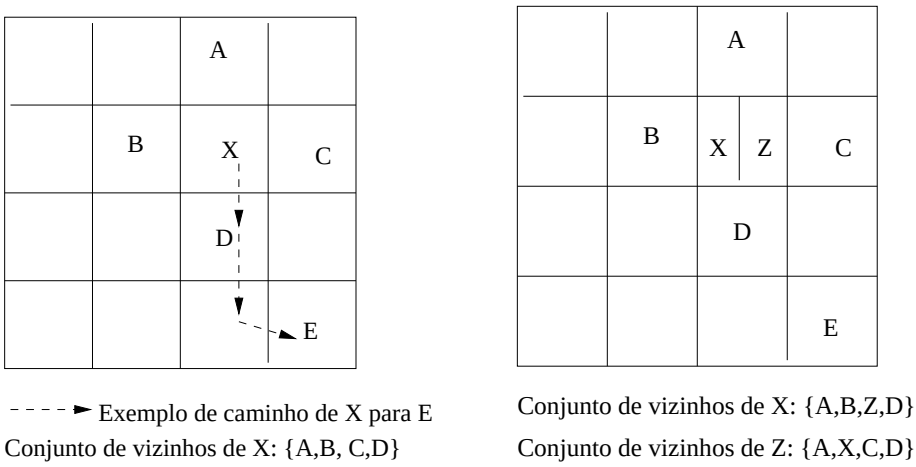
arquivo desejado. Na verdade, ainda não há um consenso a respeito do enquadramento das características da primeira geração no conceito de *Peer to Peer*, uma vez que, nesta geração, a arquitetura não é livre de servidores. Contudo, é indiscutível a importância do Napster no desenvolvimento das outras tecnologias que surgiriam anos mais tarde [24].

A segunda geração é marcada por arquiteturas totalmente ou parcialmente descentralizadas, onde a busca é feita através da inundação (*flooding*) de consultas na rede. O conceito de *super-peer* surge neste momento como fator de melhoria do desempenho. Um *super-peer* consiste em um *peer* que se torna responsável por encaminhar aos seus subordinados apenas as requisições que lhe dizem respeito, formando hierarquias e assim, minimizando tráfego desnecessário na rede [23].

A terceira geração traz uma nova forma de organizar os *peers* na rede P2P: *as Redes P2P Estruturadas*. O roteamento destas redes é feito através de Tabelas Hash Distribuídas ou DHTs (*Distributed Hash Table*). A cada nó desta estrutura é atribuído um identificador único (um *nodeID*), assim como cada item armazenado possui uma chave correspondente [3]. Desta forma, as redes da geração anterior podem ser consideradas *Redes P2P Não-Estruturadas*, por não possuírem estas características.

3.2 REDES P2P ESTRUTURADAS

As DHTs são geralmente empregadas em sistemas onde o principal objetivo é fazer descoberta de serviços ou localização de objetos em sistemas distribuídos. Objetos neste contexto podem ser vistos como nomes em um serviço de DNS, ou arquivos em um serviço de compartilhamento, por exemplo. Os valores destes objetos são identificados com o auxílio de chaves, que são mapeadas aos nós da rede por meio de uma função de *hash*, cujo resultado aponta para o endereço (ou identificador) do nó onde a chave será armazenada. Assim, em uma rede estruturada, cada nó pode possuir vários pares <chave, valor>. Este tipo de rede é denominada estruturada porque é logicamente organizada em um grafo que mapeia as chaves para nós. A seguir são apresentadas algumas redes P2P estruturadas encontradas na literatura.



(a) (b)

Figura 3.1 Exemplo de rede CAN. (a) Nós dispostos em um plano cartesiano (b) Alteração na rede com a chegada de um novo nó

3.2.1 CAN (Content Addressable Network)

Provê uma arquitetura baseada em um espaço cartesiano multi-dimensional virtual [25]. Cada nó dispõe de uma zona neste espaço, que pode ser subdividida, caso um novo nó entre na rede. Sua tabela de roteamento contém o endereço IP e as coordenadas correspondentes a cada um dos seus vizinhos. A chave é mapeada às coordenadas do nó ao qual deve pertencer por uma função de *hash*. Na Figura 3.1(a) tem-se que os vizinhos do nó $\boldsymbol{X}$ são os nós $\boldsymbol{A}$,$\boldsymbol{B}$,$\boldsymbol{C}$ e $\boldsymbol{D}$, onde $\boldsymbol{D}$ é usado como uma rota possível para $\boldsymbol{X}$ alcançar o nó $\boldsymbol{E}$. A Figura 3.1(b) ilustra a chegada de um novo nó $\boldsymbol{Z}$, que randomicamente escolhe o nó $\boldsymbol{X}$ para que este compartilhe o seu espaço. A consequência é que $\boldsymbol{Z}$ passa a ser vizinho de $\boldsymbol{X}$ no lugar de $\boldsymbol{C}$. Sistemas de armazenamento em larga escala, como o OceanStore, Farsite e Publius já usaram o CAN para rápida inserção e recuperação de conteúdo [23].

3.2.2 Chord

O Chord é uma DHT que utiliza função de *hash* consistente para mapear chaves de itens para nós. O *hashing* consistente procura fazer com que os nós recebam a mesma

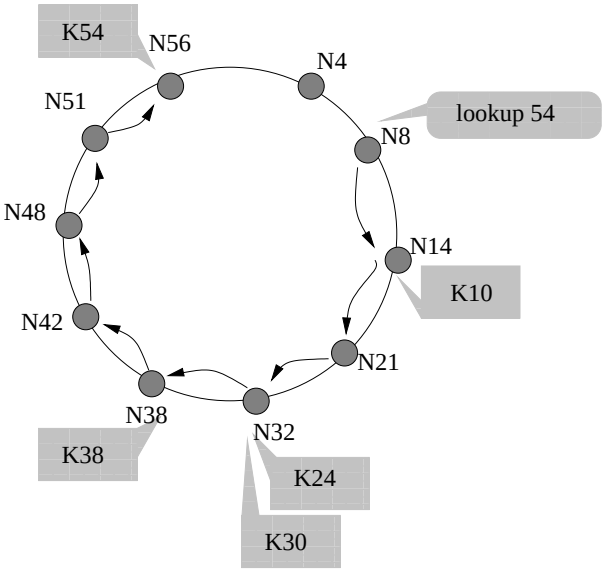


Figura 3.2 Exemplo de mapeamento de chaves a nós em uma rede Chord

quantidade de chaves. Este método garante balanceamento de carga em meio a frequente entrada e saída de nós [26].

Tanto os identificadores dos nós, também conhecidos como nodeIDs, quanto as chaves são identificados por valores de m bits. Os nodeIDs são ordenados logicamente em um anel, onde as chaves são distribuídas com base na função *sucessor*. Isto significa que, dada uma chave, a mesma é atribuída ao nó cujo identificador é igual ou tem maior valor. O anel possui um tamanho limite de 2xm.

A Figura 3.2 apresenta um exemplo de anel chord onde m=6 e o número de nós é 10. Neste anel duas posições lógicas ficam vagas. O nó com nodeID igual a 8 está procurando pela chave 54 e o faz procurando sucessores ao longo do anel. Otimizações desta busca são feitas através de tabelas de roteamento, onde cada nó armazena os identificadores de uma quantidade determinada de nós logicamente mais próximos. Serviços de DNS e sistemas de arquivos cooperativos são exemplos de aplicações que podem adotar o Chord em suas arquiteturas.

3.2.3 Pastry e Tapestry

Assim como no Chord, a DHT Pastry [27] também utiliza uma função de *hash* consistente para atribuir chaves a nós. Porém, Pastry considera que cada nó tem:

- um *Leaf Set*, no qual encontram-se os nodeIDs numericamente próximos;
- uma tabela de roteamento, onde estão armazenados os nós cujo nodeID tem prefixo em comum com o seu nodeID, cujos valores possuem uma base 2^b, sendo b um parâmetro da rede; a tabela possui $2^b - 1$ colunas e $\log 2^b N$ linhas, sendo N o número de nós na rede;
- um conjunto de vizinhos, em que são armazenados os nodeIDs dos nós mais próximos de acordo com alguma métrica de proximidade adotada.

Durante uma busca, primeiro o nó consulta o seu *Leaf Set*. Caso não seja bem sucedido, faz uma consulta a sua tabela de roteamento. Finalmente, se esta consulta não obtiver sucesso, o nó mais próximo possível é escolhido para receber a mensagem.

A Figura 3.3 apresenta um exemplo de tabela de roteamento em Pastry. Neste exemplo, b é igual a 2, ou seja, os identificadores estão na base 4. Na linha n (considerando índice inicial 0) o que está antes do primeiro hífen são os n primeiros dígitos em comum com o nó em questão. Em seguida, o $n+1$ ésimo dígito terá o valor da coluna onde está (por exemplo, na coluna dois, o valor é dois). A seguir, são os dígitos que não são comuns ao identificador do nó. Se não houver nós cujos identificadores se enquadrem em certa entrada, ela permanece vazia.

De acordo com Lua [23], a diferença entre Pastry e Tapestry está no tratamento da questão da localidade e da replicação de dados. Tapestry, por exemplo, utiliza múltiplas fontes para dados no objetivo de prover um serviço mais tolerante a falhas. Alguns serviços *multicast* e sistemas de *backup* adotam estas tecnologias.

NodeId = 10233102 (b=2)				
0 dígitos em comum	−0−2212102	1	−2−2301203	−3−1203203
1 dígito em comum	0	1−1−301233	1−2−230203	1−3−021022
2 dígitos em comum	10−0−31203	10−1−32102	2	10−3−23302
3 dígitos em comum	102−0−0230	102−1−1302	102−2−2302	3
4 dígitos em comum	1023−0322	1023−1−000	1023−2−121	3
5 dígitos em comum	10233−0−01	1	10233−2−32	
6 dígitos em comum	0		102331−2−0	
7 dígitos em comum			2	

Figura 3.3 Exemplo de tabela de roteamento em Pastry

3.2.4 Viceroy

Este protocolo foi projetado para lidar com descoberta de dados e recursos baseada na arquitetura *butterfly* [23]. Assim como no Chord, usa o conceito de sucessor para mapear chaves a nós.

3.2.5 Kademlia

Nesta DHT, as chaves são associadas aos nodeIDs se estes forem numericamente próximos. Para localizar um par <chave, valor correspondente>, Kademlia utiliza a noção de "distância" entre dois nós através da operação lógica XOR, ao invés de distância geográfica
[28]. O uso de XOR facilita o processo de análise formal, devido à sua aritmética simples. Um exemplo de rede Kademlia pode ser observado na Figura 3.4. Neste exemplo, a distância entre os nós identificados por "110" e "100" é 2 ("10" em binário), pois equivale ao resultado de uma operação de XOR entre eles. Assim, dada uma chave, é possível procurar na tabela de roteamento qual o nó que a armazena, verificando qual o nó que está a uma menor "distância" desta chave. Cada nó mantem uma tabela de roteamento

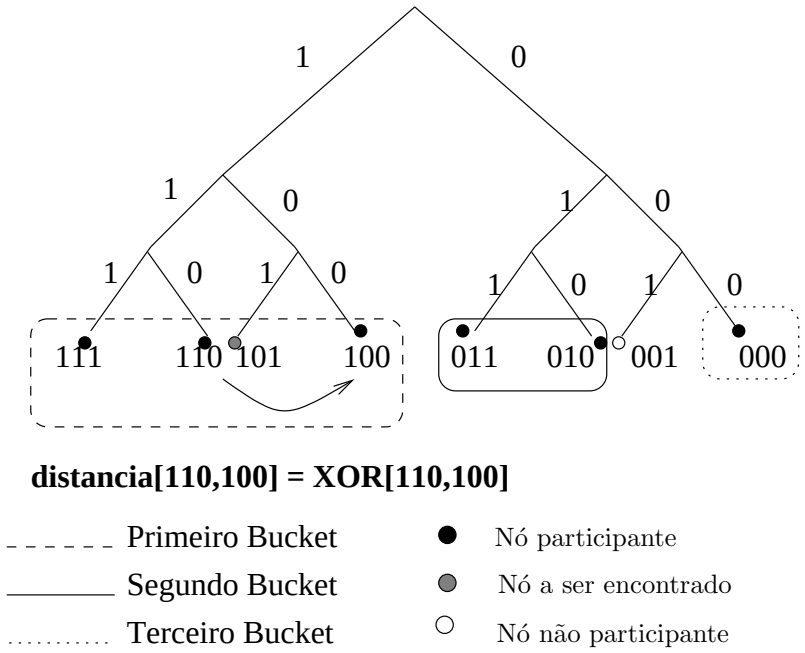


Figura 3.4 Exemplo de Rede Kademlia com espaço de chaves de 3 bits

na forma de um conjunto de *buckets*. O número de *buckets* corresponde ao número de bits escolhido para representar as chaves. Na Figura 3.4 tem-se três *buckets* porque neste exemplo os nodeIDs são representados por três bits. Cada entrada num *bucket* é uma tupla <endereço IP, porta UDP, nodeID>. Nós são adicionados aos *buckets* à medida que entram na rede. Um nó é inserido no *bucket* i de outro nó, se a distância entre eles estiver entre 2^i e 2^{i+1}, onde $0 \leq i < \#buckets$.

3.3 REDES P2P NÃO-ESTRUTURADAS

Estas redes são geralmente empregadas em sistemas onde o principal objetivo é compartilhar conteúdo na Internet. A seguir, algumas das redes P2P não estruturadas mais relevantes:

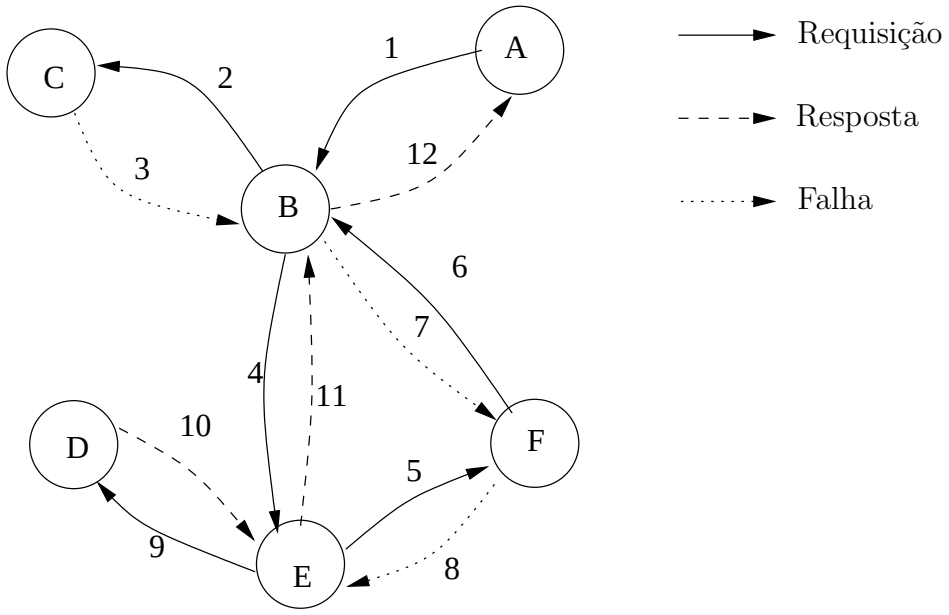


Figura 3.5 Exemplo de Rede Freenet

3.3.1 Freenet

É uma rede adaptativa onde seus pares fazem consultas através de chaves independentes de localização [29]. De modo geral, a Freenet é um sistema de *cache* distribuído. O usuário publica seu arquivo e o sistema se encarrega de mantê-lo disponível ou até replicá-lo, caso seja popular, mesmo que o usuário que o publicou esteja *offline*. Adicionalmente, provê serviços de segurança contra pares maliciosos e permite compartilhamento de espaço em disco não utilizado. Pode também ser considerada uma rede P2P fracamente estruturada, uma vez que usa um esquema de chaves para identificar os objetos que serão compartilhados, com o auxílio da função SHA-1.

A Figura 3.5 apresenta a tentativa do *peer* ***A*** em obter um conteúdo que está no *peer* ***D***. Nota-se que cada *peer* é consultado por vez até que se consiga chegar ao destino desejado.

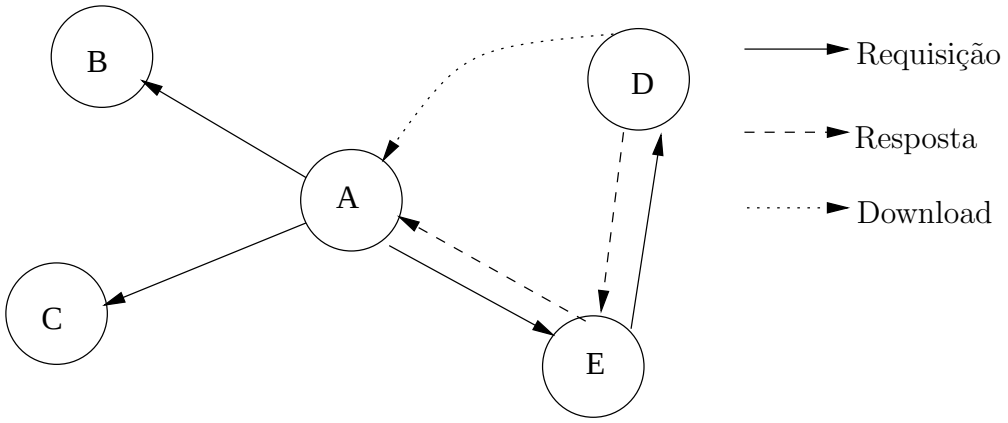


Figura 3.6 Exemplo de Rede Gnutella

3.3.2 Gnutella

Neste modelo, as consultas são feitas por *flooding*. Isto significa que, ao consultar a rede, o *peer* envia mensagens de busca para todos os vizinhos simultaneamente, que, por sua vez, repassam aos seus vizinhos, e assim por diante. Essa inundação é controlada por meio de um *flag* que tem função similar ao *time-to-live* do IP. Os *peers* procuram por novos *peers* disponíveis enviando periodicamente mensagens de "ping". Na Figura 3.6 o *peer* ***A*** encontra o conteúdo desejado, neste exemplo, no *peer* ***D***, em dois passos, graças ao *flooding*. Por outro lado, esta técnica pode impactar criticamente no desempenho da rede, quando a mesma não é bem dimensionada.

3.3.3 FastTrack

Esta rede é composta por *peers* e *super-peers* [30]. Os *super-peers* possuem a função de hierarquizar a rede, diminuindo o impacto do *flooding*. Quando um *peer* deseja encontrar algum conteúdo, este consulta o seu *super-peer*, que por sua vez, faz a busca entre os *peers* os quais atende e por meio de consultas aos demais *super-peers*. Caso encontre o conteúdo, o *super-peer* informa qual *peer* deve ser conectado para a realização do *download*. A Figura 3.7 mostra um esboço de arquitetura P2P com *super-peers*. A rede FastTrack é um exemplo de rede que usa esta técnica. Para a troca de mensagens es-

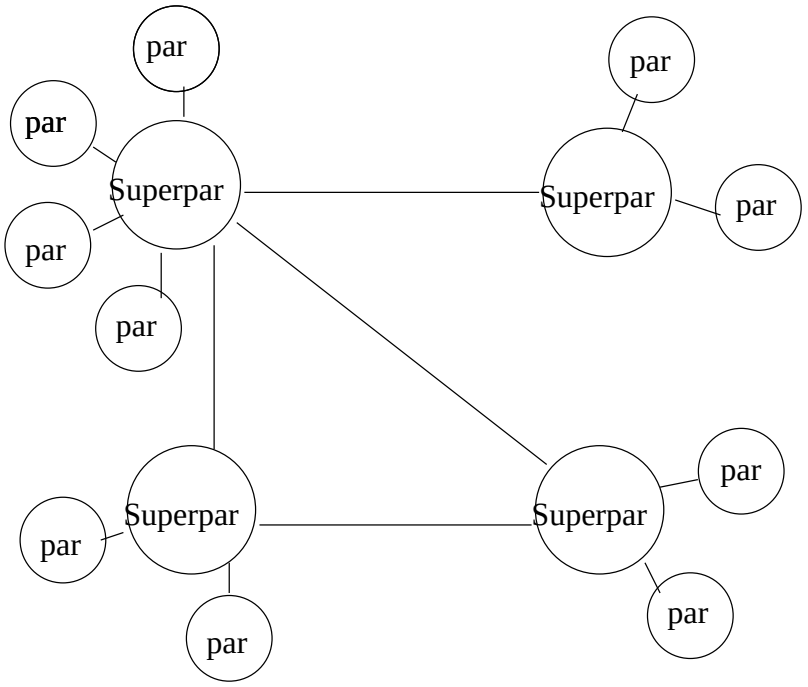


Figura 3.7 Uma rede com *super-peers*

pecíficas do FastTrack, inclui cabeçalhos diferenciados em mensagens HTTP. Programas que adotaram este protocolo incluem Kazaa, Morpheus e iMesh, entre outros. As versões mais recentes do Gnutella também utilizam o conceito de *super-peers*.

3.3.4 Emule

Esta rede foi concebida a partir da rede *eDonkey* Esta rede permite *download* a partir de múltiplas fontes, detecção e correção de arquivos corrompidos,métodos mais expressivos de consulta, como expressões *booleanas*, sistema de créditos, entre outros recursos.

3.4 BITTORRENT

A rede BitTorrent [31] foi concebida por Bram Cohem em 2003. Esta rede atraiu a atenção da comunidade interessada em P2P, graças à sua capacidade de permitir o compartilhamento de arquivos grandes e de favorecer *peers* que possuem melhor desempenho. As próximas seções apresentam os detalhes de funcionamento deste protocolo.

3.4.1 Aspectos Gerais

O BitTorrent é uma rede não-estruturada que permite o *download* a partir de múltiplas fontes, flexibilizando este processo, assim como o Emule. Isto é possível graças à divisão lógica do arquivo em pedaços denominados *pieces*, que por sua vez, são divididos em blocos de tamanho fixo, cujo valor pode variar de acordo com a implementação. Assim, *peers* podem sair e entrar na rede com o conteúdo parcialmente baixado e continuar o *download* posteriormente.

O equilíbrio das cooperações é garantido pela política *tit-for-tat* pela qual *peers* agem com o mesmo grau de altruísmo com que foram tratados por seus colaboradores em ações recentes. Os *peers* podem bloquear algumas de suas conexões temporariamente para limitar a quantidade de *uploads* simultâneos em um processo conhecido como *choking*.

A rede faz uso de uma entidade centralizada para o registro de *peers* chamada *Tracker*. O *Tracker* informa via HTTP quais são os *peers* que estão na rede e compartilham o arquivo desejado. O contato inicial com o *Tracker* é feito através de um metarquivo, `.torrent`, que contém informações do arquivo a ser buscado, como tamanho, valor *hash*, url do *Tracker*, entre outras.

Em um *download*, um *peer* que possui um arquivo completo é denominado *seed* ou semente. Caso possua apenas parte dele, o *peer* é chamado de *leecher*. A obtenção de um arquivo ocorre por meio do *download* de blocos, em uma mensagem que traz a informação sobre a qual *piece* ele pertence e o *offset* ao qual corresponde. Ao fim do *download*, os *pieces* são ordenados, concatenados e tem sua integridade verificada.

3.4.2 O Processo de download

O processo de *download* de um arquivo π por um cliente C, que utiliza um metarquivo τ_π para consultar um *Tracker* Γ_π ocorre nos seguintes passos [32]:

1. C faz uma consulta a Γ_π usando τ_π como parâmetro. Este arquivo `.torrent` foi criado e disseminado por uma *seed* inicial σ_0^π;

2. Γ_π responde com uma lista de *leechers* ou *seeds* $P = p_1, p_2, \ldots, p_n$ em processo de

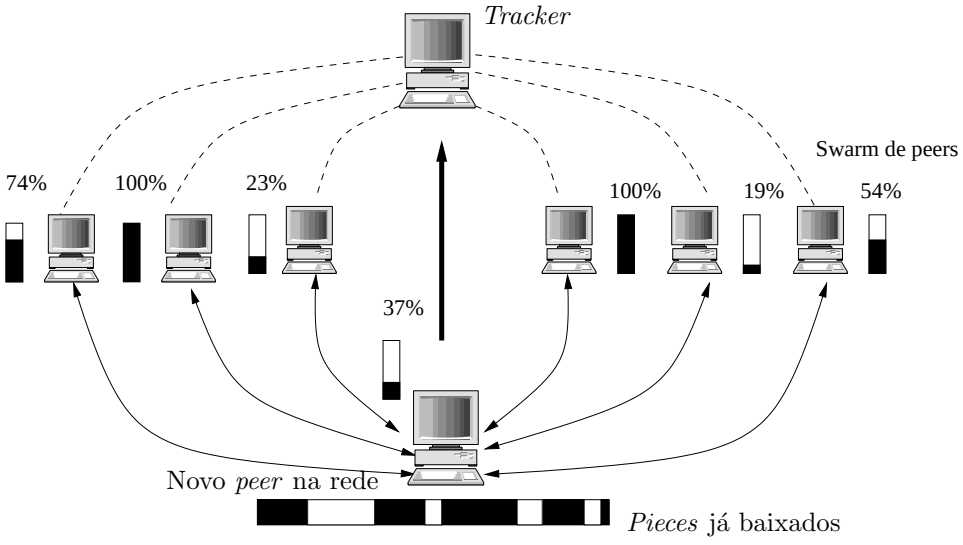


Figura 3.8 Exemplo de rede BitTorrent

download/upload de π;

3. C se torna parte da π-*swarm*, o conjunto de *peers* baixando π a partir dos elementos de P;

4. Durante a transferência, os *peers* podem trocar *bitfields*. O *bitfield* de um arquivo π em um nó particular indica os *pieces* que ainda não foram baixados. Cada bit neste campo está relacionado a um *piece* ;

5. Ao fim do *download*, C pode decidir se tornar um *seed* deste arquivo. Se sim, precisará informar isto ao *Tracker*.

A Figura 3.8 ilustra um exemplo de rede BitTorrent.

3.4.3 Mensagens de Protocolo

O BitTorrent possui um conjunto de mensagens de protocolo para estabelecer conexões no nível de aplicação e trocar informações do *download*. Estas mensagens são descritas a seguir:

- `HANDSHAKE` - usada no primeiro contato;

- `BITFIELD` - carrega informação sobre *pieces* já baixados;
- `INTERESTED` - informa que um *peer* está interessado nos *pieces* de outro *peer*;
- `REQUEST` - usada para requisitar um *piece*, especificando o *offset* do bloco desejado;
- `PIECE` - traz consigo um bloco de um *piece*, ou seja, dados do arquivo, informando o índice do *piece* e o *offset* do bloco. Entende-se índice como um identificador da posição que o *piece* ocupa no arquivo. O *offset* serve para identificar o bloco, indicando uma posição dentro de um *piece* em termos de *bytes*;
- `HAVE` - informa que um *peer* acabou de completar um *piece*;
- `CHOKE` - informa a um *peer* que este está impedido de requisitar *pieces* de um outro *peer*;
- `UNCHOKE` - informa a um *peer* que este pode requisitar *pieces* de um outro *peer*.

A Figura 3.9 mostra um exemplo de fluxo de execução. Inicialmente ***peer 1*** envia um `HANDSHAKE` a ***peer 2*** para informar que entrou na rede e ***peer 2*** responde com a mesma ação. Assim, ***peer 1*** envia a mensagem de `BITFIELD` para informar ***peer 2*** sobre os *pieces* que possui e vice-versa. Caso um deles tenha interesse em baixar *pieces* do outro deve enviar a mensagem de `INTERESTED`. Neste exemplo, ***peer 1*** está interessado em ***peer 2***. Para que isto seja possível, ***peer 2*** precisa verificar se pode desbloquear o *upload* para ***peer 1*** para poder enviar a mensagem de `UNCHOKE`. Uma vez desbloqueado, ***peer 1*** envia um `REQUEST` especificando o *piece* e o *offset* do bloco desejado. Finalmente, ***peer 2*** envia a mensagem de `PIECE` , fornecendo o *piece* pedido por ***peer 1*** no *offset* indicado na requisição. Ao completar um *piece*, ***peer 1*** envia `HAVE` para os demais *peers* que se encontram na rede, inclusive ***peer 2***.

3.4.4 BitTorrent e DHTs

Implementações sem *Tracker* usam protocolos baseados em DHTs para armazenar a localização dos *peers*. O protocolo DHT mais comumente utilizado para substituir o

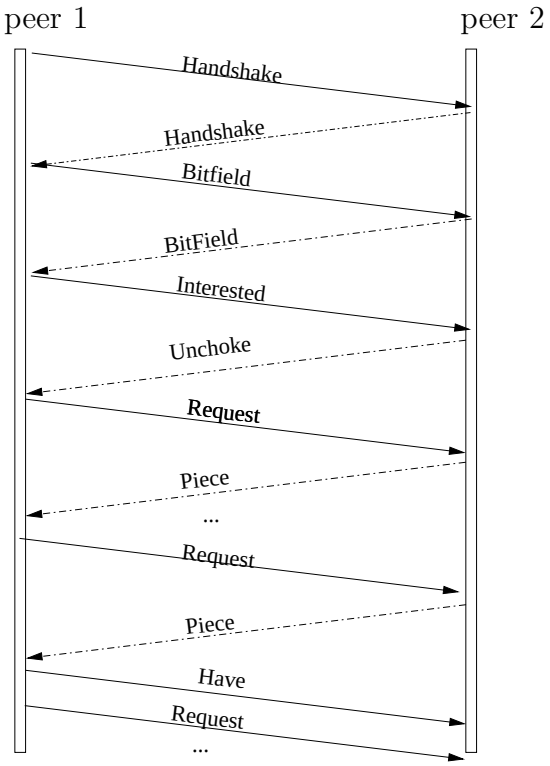


Figura 3.9 Exemplo de fluxo de mensagens no BitTorrent

Tracker é o Kademlia. Neste caso, *peers* BitTorrent incluem um nó Kademlia para contactar outros nós no intuito de obter informações de localização de *peers* disponíveis para *download* usando o protocolo BitTorrent tradicional. O metarquivo `.torrent` passa a ser, então, representado por uma chave. Em outras palavras, cada *peer* torna-se um *Tracker* [33].

3.5 RESUMO

As redes P2P valorizam o caráter colaborativo da Internet sendo oposto ao modelo cliente-servidor, que concentra serviços em poucas máquinas de grande capacidade, o que introduz pontos de falha [24]. As redes P2P podem ser estruturadas ou não-estruturadas. As primeiras são encontradas em aplicações onde deseja-se localizar recursos, enquanto as últimas são amplamente utilizadas na Internet para compartilhamento de arquivos. A popularidade das redes como o Napster e Gnutella chamou a atenção da comunidade

acadêmica interessada em P2P. O BitTorrent foi um protocolo desenvolvido procurando unir a simplicidade da centralização do Napster com a flexibilidade e tolerância a falhas do Gnutella e demais redes não-estruturadas. O sucesso do BitTorrent na Internet tem levado a estudos sobre seu uso em outras arquiteturas de rede, como nas MANETs.

CAPÍTULO 4

P2P SOBRE MANETS

No contexto das redes sem fio, aplicações P2P podem ser úteis em cenários de vídeo-conferência, de resgate e em aplicações *groupware*. O uso de modelos P2P em MANETs é mais interessante que a adoção de modelos cliente-servidor, pois este último não garante tolerância a falhas e pode tornar o uso de uma aplicação inviável. Este capítulo apresenta aspectos que devem ser levados em consideração para a adaptação de protocolos P2P em MANETs, bem como os trabalhos relacionados mais relevantes.

4.1 PRINCIPAIS DESAFIOS

Apesar das redes *Peer to Peer* se assemelharem às MANETs quanto à descentralização e à flexibilidade, a adaptação de uma aplicação P2P ao ambiente sem fio não é uma tarefa trivial. Na Internet, as redes P2P podem contar com com uma maior estabilidade de rotas e enlaces, o que não é possível no meio *wireless* e isto traz conseqüências para o desempenho das aplicações. Outros desafios advindos da arquitetura *ad hoc* são enumerados a seguir [3]:

- **Acesso Múltiplo** - a probabilidade de colisões e atrasos aumenta por causa da ausência de um ponto central de coordenação;
- **Mobilidade dos Nós** - a mobilidade dos nós da MANET pode implicar mudanças frequentes de topologia;
- ***Churn*** - entrada e saída frequente de nós na rede causa alterações na topologia;
- **Energia Limitada** - protocolos P2P que requerem a transmissão periódica de

mensagens de roteamento podem levar à perda excessiva de energia por parte dos nós;

- **Compensação Estado vs. Eficiência** - o nicho de aplicações de MANETs compreende cenários mais reduzidos que os das tradicionais redes P2P, que em geral funcionam na escala da Internet. Para manter a escalabilidade, as redes P2P tendem a armazenar menos estado, aumentando o volume de tráfego na rede.

Os protocolos projetados para MANETs precisam levar em consideração alguns aspectos físicos da rede, como o nível de energia e de interferência entre canais, que são características determinantes no comportamento dos mesmos. Em alguns cenários, como em algumas redes de sensores, não é possível trocar ou recarregar a bateria dos dispositivos, sendo desejável que o consumo de energia seja limitado.

4.2 ABORDAGENS DE IMPLEMENTAÇÃO

Uma abstração P2P pode ser implementada em uma MANET na camada de rede ou acima dela. O termo *abstração P2P* denota a utilização de conceitos de P2P, sem a obrigatoriedade de ser um *overlay*, que poderia dar idéia de sub-rede, um conceito que não é comumente aplicável às MANETs [3]. Para Hu *et al.* [3], uma abstração P2P pode ser implementada em uma MANET seguindo dois tipos de abordagem: *Layered Design* e *Integrated Design*. Na primeira abordagem, o protocolo P2P é implementado de forma independente do protocolo de roteamento da MANET. Na segunda abordagem, o protocolo P2P está integrado ao protocolo de roteamento da MANET. O *Layered Design* tem o benefício de uma maior simplicidade de implementação e manutenção. Isto, porém, tem o custo de uma redução de desempenho em comparação com o *Integrated Design*, que prima por uma maior eficiência uma vez que duas ou mais camadas interagem diretamente entre si. Neste contexto, é possível que a integração seja completa, onde todas as funcionalidades do protocolo P2P exploram ao máximo os recursos da camada de roteamento fazendo com que as duas camadas integradas formem uma estrutura única. Por outro lado, a integração pode ser parcial, otimizando apenas algumas funcionalidades do protocolo P2P

com chamadas a operações pertencentes às camadas inferiores. A implementação de um *Integrated Design* é possível com a aplicação de técnicas de *cross-layering*, que possui um conceito mais amplo.

4.3 CROSS-LAYERING

Para Srivastava *et al.* [34], o *cross-layering* é caracterizado por uma violação da arquitetura de referência em camadas, de modo a permitir comunicação direta entre protocolos de camadas não adjacentes ou compartilhamento de variáveis entre camadas.

Srivastava *et al.*, crêem que há três motivações que encorajam violações da arquitetura de referência em camadas na presença de um ambiente sem fio:

- Os problemas inerentes aos canais sem fio e sua convivência difícil com protocolos de camadas superiores, como o TCP;
- A possibilidade de se ajustar parâmetros de alto nível com base no estado atual da rede de forma dinâmica;
- Oportunidade de se oferecer novas modalidades de comunicação que as arquiteturas atuais não suportam, como por exemplo, a natureza *broadcast* do canal sem fio.

Srivastava *et al.* ainda em seu trabalho apresentam as propostas de *cross-layering* comumente adotadas em implementações:

- Criação de interfaces - Novas interfaces podem ser acrescentadas entre camadas, redefinindo as fronteiras entre as mesmas. O fluxo da informação pode ocorrer da camada inferior para a superior (Figura 4.1(a)), da superior para a inferior (Figura 4.1(b))ou o fluxo pode ser interativo (Figura 4.1(c)). Um exemplo de implementação seria uma interface que comunicasse à camada de transporte o motivo de atrasos na entrega de segmentos TCP, evitando o problema do mau emprego do mecanismo de controle de congestionamento.
- Fusão de camadas adjacentes - Consiste na criação de uma “super camada” que

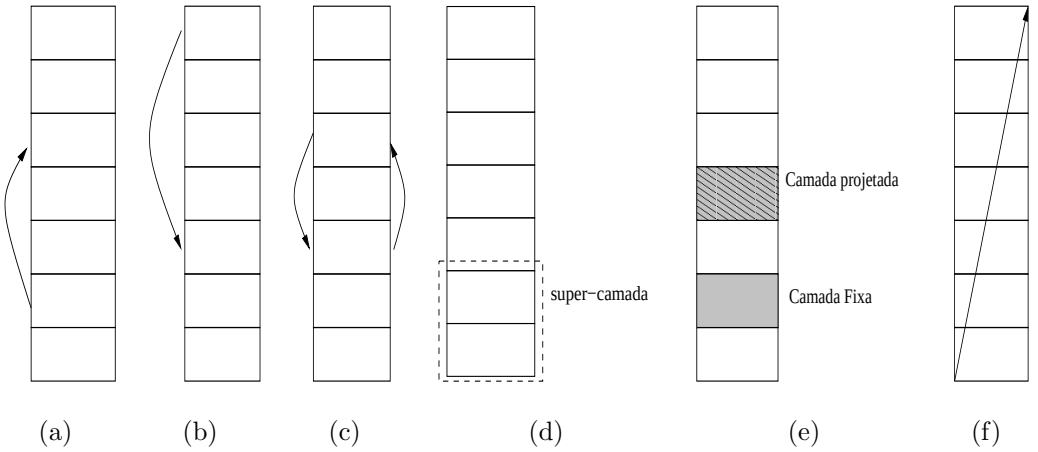


Figura 4.1 Métodos de *cross-layering*. (a) Interface da camada inferior para superior (b) Interface da camada superior para inferior (c) Interface de fluxo interativo (d) Fusão de camadas adjacentes (e) Acoplamento sem interfaces (f) Calibramento vertical entre camadas

englobe os serviços de duas camadas adjacentes. Um exemplo é uma possível associação entre as camadas MAC e PHY. A Figura 4.1(d) ilustra esta técnica.

- Acoplamento sem interfaces - neste caso nenhuma interface é definida e existe dependência direta entre as camadas envolvidas, conforme mostra a Figura 4.1(e).

- Calibramento vertical entre camadas - Consiste no ajuste de parâmetros, cuja mudança de valor ajusta os parâmetros das outras camadas, em cascata. A Figura 4.1(f) apresenta este método. O calibramento pode ser feito em tempo de projeto ou em tempo de execução.

Entretanto, o *cross-layering* é uma metodologia que deve ser aplicada com cautela, uma vez que violações de arquitetura podem resultar no detrimento da longevidade do sistema ou até mesmo, descaracterizar completamente a arquitetura original.

4.4 TRABALHOS RELACIONADOS

A seguir, são apresentados os trabalhos relacionados à interação entre protocolos P2P e MANETs, separando-os em trabalhos que usam a abordagem tradicional de *Layered Desing* e trabalhos que usam alguma técnica de *cross-layering*. A partir deste estudo,

os pontos de melhoria utilizados por estas propostas são destacados na Seção 4.5. Comparações entre os trabalhos relacionados a estes pontos e o Sistema de Disseminação proposto são apresentados na Seção 5.1.4.

4.4.1 Trabalhos relacionados baseados no Layered Desing

O trabalho de Oliveira *et al.* [35] foca na comparação do desempenho entre protocolos de roteamento mediante uma aplicação P2P Gnutella. Os resultados mostram que o desempenho dos protocolos é diferente do que é observado em outros tipos de aplicação unicast. Desta forma, compreende-se que não há um consenso sobre qual o melhor protocolo de roteamento a ser adotado juntamente a aplicações P2P. Já Franciscani *et al.* [36] propõem diferentes algoritmos de reconfiguração de redes sem fio para uma aplicação baseada no protocolo P2P Gnutella e os compara entre si. Os resultados indicam que cada algoritmo proposto possui melhor ou pior desempenho dependendo da homogeneidade da MANET, ou seja, da semelhança entre os dispositivos que constituem a rede.

Cramer e Fuhrmann [37] avaliam o desempenho do protocolo Chord sobre MANETs. Os resultados apontam ineficiências deste protocolo neste *design* para lidar com as constantes mudanças na topologia da rede *ad hoc*.

Da Hora *et al.* [38] propõem técnicas para melhorar o desempenho de aplicações P2P em MANETs. Para protocolos P2P estruturados, como Chord, propõe-se o acréscimo de redundâncias nas mensagens de pesquisa para reduzir o consumo de energia e o tempo de resposta. Para protocolos P2P não-estruturados, como o Gnutella, propõe-se um repasse seletivo de mensagens, que permite dobrar o percentual de requisições atendidas.

Em [39] é proposta a adição do método de descoberta de serviços GSD (*Group-based Service Discovery Protocol*) ao BitTorrent quando o mesmo é utilizado em MANETs. Os resultados demonstram uma melhoria no tempo de *download* de arquivos.

4.4.2 Trabalhos relacionados baseados em Cross-layering

Bisignano *et al.* [40] propõem o *middleware* Expeerience baseado no *framework* JXTA com o objetivo de prover suporte de alto nível para o desenvolvimento de aplicações P2P em MANETs. Entre os serviços, está o suporte a múltiplas interfaces e descoberta de serviços, entre outros.

Conti *et al.* [41] apresentam uma arquitetura de pilha de protocolos para redes *ad hoc* com ênfase em aplicações P2P Pastry. Os autores concluem que há vantagens ao integrar camada de aplicação e roteamento e encorajam o uso desta abordagem. O substrato DHT para MANETs denominado Ekta é proposto por Pucha *et al.* [42]. Nesta abordagem, a DHT Pastry é integrada ao protocolo reativo de roteamento DSR com o objetivo de implementar DHTs eficientemente em MANETs. A proposta se mostrou mais eficiente que o simples uso do *broadcast* da camada física para a descoberta de recursos nos cenários estudados. O desempenho desta integração foi melhorado por meio do reuso de rotas em uma extensão denominada Ekta+ [43]. Este tipo de integração pode também ver vista no MADPastry [44], porém, com o protocolo AODV. Diferentemente de Ekta, o MADPastry usa também informações de localização. Resultados apontam desempenho semelhante e, em alguns casos, melhor que uma implementação de *broadcast* e uma implementação de Pastry convencional.

O protocolo FastTrack foi integrado ao AODV por Tang *et al.* em [45] e avaliado em comparação ao seu equivalente no *Layered Design*. O trabalho mostra melhorias no atraso médio de pacotes, redução no *overhead* de mensagens e aumento da taxa de entrega de pacotes com o uso da técnica de *cross-layering*.

Park *et al.* propõem em [46], uma versão do Gnutella para operar com eficiência em MANETs usando informações de roteamento, nível de energia e conectividade. Os resultados indicam desempenho superior à abordagem tradicional em termos de tempo de resposta e taxa de acertos. O XL-Gnutella [47] é fruto de um trabalho semelhante, porém, tirando proveito da pró-atividade do protocolo de roteamento OLSR para obter informações da topologia. Neste estudo, observou-se que a taxa de sucesso em consultas é superior a uma implementação tradicional. Klemm *et al.* [48] propõem a abordagem

ORION (*Optimized Routing Independent Overlay Network*), que consiste na otimização da descoberta de rotas com um protocolo de aplicação similar ao Gnutella com auxílio da camada de roteamento. Esta abordagem também inclui um protocolo de transporte otimizado para MANETs. Os resultados mostram um desempenho superior quando comparado com implementações *overlays* sobre TCP. A pilha de protocolos MPP (*Mobile Peer to Peer*) proposta por Gruber *et al.* [49] implementa o *cross-layering* desde a camada física até a camada aplicação, representado por um protocolo similar ao Gnutella. Comparações com ORION foram feitas obtendo-se menor overhead e maio taxa de sucesso em buscas.

Em [32] é avaliado o desempenho do protocolo BitTorrent sobre uma MANET que emprega o protocolo de roteamento ANSI. Avaliações são feitas considerando-se uma abordagem *cross-layer* e uma abordagem tradicional de organização em camadas. Os resultados apontam um melhor desempenho quando o *cross-layering* é utilizado. Krifa *et al.* em [50] apresentam o BitHoc, uma solução baseada em BitTorrent para redes *ad hoc*. A solução se divide em um componente de gerenciamento de pares no papel de Tracker e um componente de compartilhamento de conteúdo, adaptando a seleção de peças de acordo com a topologia. Souza *et al.* [51] apresentam uma modificação do protocolo BitTorrent para lidar com localidade espaço-temporal, uma propriedade presente em redes *ad hoc* críticas, na qual nós próximos tendem a baixar o mesmo conteúdo. A proposta apresentada consiste em dividir a rede em *clusters* cada um com seu líder, responsável por fazer o *download* e repassar o conteúdo baixado em *multicast* para os pares do seu *cluster*. Estes pares são desconectados da rede BitTorrent e aguardam passivamente pelos conteúdos do líder. Este trabalho conclui que a implementação proposta permitiu o aumento da taxa de transferência e a redução do tráfego na rede, porém sem tratar a questão da mobilidade. Este trabalho também se destaca por enfatizar a importância de se aproveitar os recursos do meio sem fio para otimizar o *download*.

4.5 PONTOS DE MELHORIA DO DESEMPENHO DE APLICAÇÕES P2P EM MANETS

Os estudos do Estado da Arte indicam que para o sucesso do uso de um protocolo P2P em uma MANET, modificações precisam ser feitas para que a aplicação se adapte a realidade de um ambiente de rede *ad hoc* sem fio. Também observou-se que a escolha de quais protocolos abaixo da aplicação melhor influenciam no desempenho do sistema é uma questão em aberto. Sabe-se, entretanto, que a camada transporte, de roteamento, bem como as camadas enlace e física podem fornecer informações essenciais à otimização destas aplicações. Estes estudos permitem levantar pontos de melhoria para a proposição de novas soluções para o uso de protocolos P2P em MANETs. Estes pontos são listados a seguir:

- Utilizar protocolos de descoberta de serviços como auxiliar para localização de pares na rede;
- Usar *cross-layering* para integrar completamente a aplicação à camada de roteamento;
- Usar *cross-layering* para que a aplicação tenha acesso a informações de outras camadas não-adjacentes da pilha de protocolos;
- Aproveitar a natureza *broadcast* do meio sem fio.

Considerando-se o objetivo de propor um sistema de otimização simples e eficiente do BitTorrent para MANETs, algumas considerações devem ser feitas sobre os pontos levantados. A localização de pares e a descoberta de serviços não faz parte do foco deste trabalho. A proposta desta dissertação se concentra em melhorar o processo do *download* em si. O uso de *cross-layering* deve ser evitado para garantir a simplicidade de uma eventual implementação da proposta, pois exigiria alterações no *kernel* do sistema operacional do dispositivo móvel. Entretanto, um ponto podem ser aplicado em uma proposta de otimização do BitTorrent para MANETs de modo a atingir os objetivos deste trabalho: o melhor aproveitamento do meio sem fio nas transmissões. Este ponto está

relacionado ao uso do *broadcast* como facilitador da difusão de mensagens. O *broadcasting* é um método de comunicação que permite enviar uma unidade de dados de protocolo para todos os destinos simultaneamente [52]. No contexto de redes sem fio, o *broadcast* abrange os nós que se encontram dentro do alcance daquele nó que envia a mensagem. Uma MANET pode realizar disseminação de informações por *broadcast* com facilidade devido à natureza do meio de transmissão sem fio. A camada física detecta o sinal eletromagnético e a camada MAC é responsável por identificar se o quadro recebido em sua interface está endereçado ou não ao nó que o recebeu. Desta forma, se parte das mensagens em uma transmissão pudesse ser enviada em modo *broadcast*, a largura de banda poderia ser economizada, já que de qualquer forma os quadros referentes a esta mensagem chegariam ao receptor que estivesse ao alcance do emissor. Entretanto, o *broadcast* deve ser controlado, uma vez que a possível aceitação de dados redundantes levam a atrasos devido ao processamento dessa informação nas camadas superiores e a gerar respostas que apenas contribuem para o congestionamento da rede.

4.6 BITTORRENT EM MANETS: UM ESTUDO PRELIMINAR

Além dos pontos levantados pelos trabalhos relacionados, é importante realizar um estudo simulado sobre o funcionamento de uma aplicação BitTorrent em uma MANET sem qualquer otimização. Este estudo permite comprovar a necessidade de se propor um sistema que promova melhorias no desempenho. Resultados parciais deste estudo podem ser encontrados em [53]. Assim, o protocolo clássico do BitTorrent foi sobreposto a uma MANET, ou seja, a aplicação foi vinculada à rede *ad hoc* sem qualquer adaptação. Para tal, foi utilizado o simulador NS-2, com o acréscimo do suporte aos protocolos BitTorrent [54] e OLSR [55].

Para o estudo de caso considera-se o seguinte cenário:

- Os nós da MANET estão dispostos uniformemente em forma de grade distantes meio metro entre nós vizinhos adjacentes;
- O número de pares simulados são 4(2x2), 9(3x3), 16(4x4) e 25 (5x5);

- Os nós compartilham um arquivo de 100MB a partir de uma única semente;
- O tamanho de um *piece* foi fixado em 512KB;
- Não há movimentação, tampouco entrada e saída de nós.

Este cenário foi definido para reproduzir uma situação de compartilhamento de um vídeo de curta duração em uma sala de reunião, uma situação factível para MANETs. Os protocolos AODV e OLSR foram escolhidos para a camada de roteamento a título de comparação dos resultados. Assim é possível verificar o impacto no desempenho geral do sistema causado pela escolha de um protocolo de roteamento reativo e proativo, respectivamente.

No experimento foi medido o tempo de *download* em função do número de pares. A Figura 4.2 mostra o resultado obtido. Não houve nenhuma diferença significativa no desempenho do *download* ao se comparar o OLSR com o AODV. Adicionalmente, observa-se que este tempo aumenta exponencialmente com o aumento do número de pares presentes na rede. No caso de 25 pares, por exemplo, os participantes teriam que esperar cerca de uma hora para obter o conteúdo completo. Este fato poderia inviabilizar a aplicação, dependendo da necessidade de obter este conteúdo em tempo mínimo. Assim, conclui-se que quando sobreposto em uma MANET sem qualquer modificação, a aplicação BitTorrent tem o desempenho bastante comprometido no cenário estudado. Isto ocorre devido às limitações inerentes à rede *ad hoc*. Assim, compreende-se a necessidade de se adotar estratégias para otimizar o BitTorrent para este tipo de ambiente com o objetivo de melhorar o seu desempenho.

4.7 RESUMO

As MANETs já possuem em si uma natureza P2P, uma vez que não dependem de infraestrutura centralizadora e seus nós podem agir como roteadores. Estudos no Estado da Arte apontam diversos esforços no intuito de adaptar protocolos P2P bem sucedidos na Internet ao ambiente de MANETs. As propostas incluem desde adaptações sem alterações na comunicação da pilha de protocolos até integração total entre a camada de

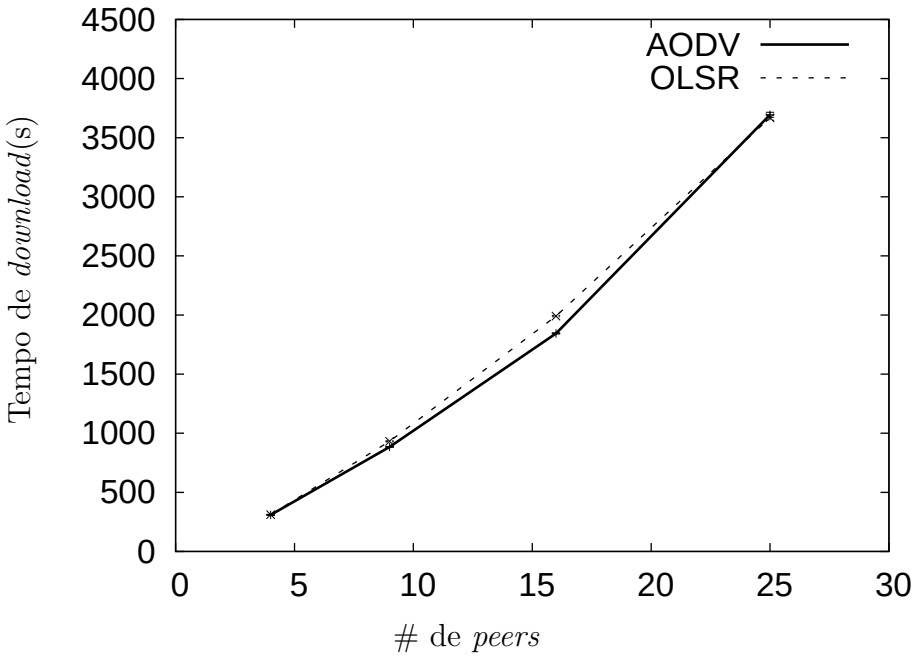


Figura 4.2 Tempo de *download* em função do número de *peers*

aplicação e a camada de roteamento. Este estudo motivou a realização de um experimento preliminar explorando a fragilidade de usar o BitTorrent em uma MANET sem qualquer adaptação. Os resultados deste experimento confirmaram a necessidade de se propor uma abordagem que melhore o desempenho do BitTorrent em MANETs.

CAPÍTULO 5

O SISTEMA DE DISSEMINAÇÃO PROPOSTO

Estudos do Estado da Arte mostram que a utilização do BitTorrent em uma MANET requer modificações no processo de *download* para melhorar o seu desempenho. Experimentos preliminares indicam que a aplicação BitTorrent em meio a um ambiente sem fio sem qualquer modificação apresenta uma queda bastante acentuada no desempenho com o aumento do número de pares na rede. Também observou-se que um dos fatores decisivos no desempenho deste tipo de aplicação é a forma como o meio de transmissão é aproveitado.

Este capítulo apresenta o Sistema de Disseminação proposto e a sua arquitetura como uma forma de melhorar o desempenho da aplicação BitTorrent em uma MANET.

5.1 UM SISTEMA DE DISSEMINAÇÃO DE PIECES PARA O BITTORRENT

Conforme observado no Capítulo 3, o BitTorrent é um protocolo baseado em mensagens ponto-a-ponto, ou seja, as conexões entre os pares são estabelecidas individualmente. De acordo com os estudos de trabalhos relacionados, o meio sem fio pode ser melhor aproveitado pela aplicação BitTorrent. O Sistema de Disseminação é um conjunto de estratégias propostas para otimizar o desempenho do BitTorrent tanto no tempo de *download* quanto em termos de *overhead* de segmentos e roteamento. Estas otimizações são feitas com o *Layered Design*, ou seja, não há interações *cross-layers* entre camadas da pilha de protocolos. O sistema procura tirar proveito dos recursos da MANET difundindo parte das mensagens de aplicação em modo *broadcast*, melhorando o desempenho do sistema como um todo. Não se trata de uma substituição ao protocolo do BitTorrent, e sim, de melhorá-lo mediante um ambiente sem fio, conservando as suas características de descentralização

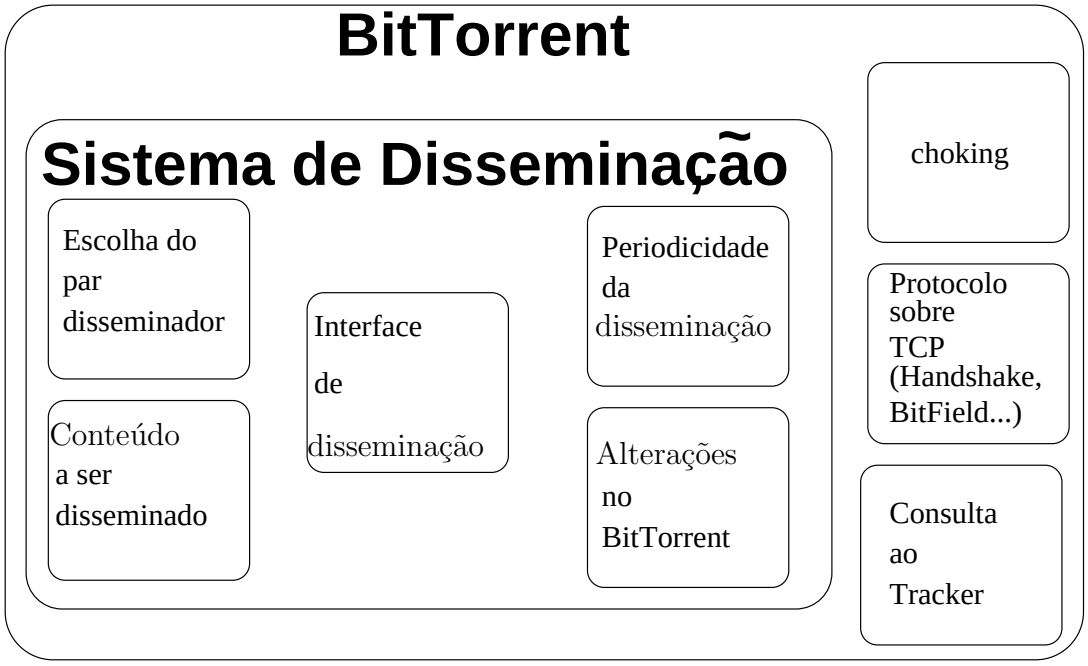


Figura 5.1 Arquitetura do BitTorrent com o Sistema de Disseminação

do *download*. A Figura 5.1 mostra a arquitetura do BitTorrent com o Sistema de Disseminação proposto. Os elementos que constituem o Sistema de Disseminação são descritos resumidamente a seguir. Maiores detalhes serão apresentados nas próximas seções.

Interface de disseminação - O protocolo original do BitTorrent se baseia na troca de mensagens *unicast*. Para permitir que algumas delas sejam enviadas em modo *broadcast*, uma nova interface é proposta. A esta interface é dado o nome de *BitTorrent Mobile Interface* (BMI), uma subcamada da camada de aplicação. A interface proposta adota o protocolo de transporte não-confiável UDP, para que estas mensagens não sejam entregues pelo TCP, que é um protocolo de transporte exclusivamente ponto-a-ponto. A aplicação deve utilizar ambos os mecanismos *unicast*, por meio do TCP e *broadcast*, por meio da interface BMI/UDP. A decisão de quando cada mecanismo será usado para transportar os dados depende das informações vindas dos pares interessados no *download*.

Conteúdo a ser disseminado - A maior parte das mensagens enviadas durante o

download são as mensagens de `PIECE`, que contem a informação do índice do *piece* e o *offset* do bloco correspondente. Detalhes sobre o conceito de de índice e *offset* podem ser encontrados na Seção 3.4.3. Assim, pode-se vislumbrar a melhoria do desempenho geral do compartilhamento com a seleção de algumas mensagens de `PIECE` para serem distribuídas em modo *broadcast*. Para que se obtenha melhores resultados, estas mensagens não devem ter a informação de índice e *offset* iguais àquelas já requisitados em *unicast*, pois isso geraria o tráfego de informações redundantes. Assim, são estabelecidos diferentes critérios de seleção de *pieces* e blocos para ambos os casos, detalhados na Seção 5.1.3.

Escolha do *peer* disseminador - Nesta proposta se estabelece que um *peer* disseminador é a semente responsável por decidir o que deve ser difundido para todos os outros pares, de acordo com as necessidades destes. Convenciona-se também que um único *peer* é o disseminador, para evitar a difusão de mensagens que já foram enviadas por terceiros. Caso fossem admitidos múltiplos disseminadores, haveria a necessidade de troca de mensagens entre os mesmos sempre que houvesse escolha de *pieces* e blocos para disseminação, o que geraria um *ovehead* extra de mensagens. Além disso, o uso de apenas um nó como disseminador não introduz um ponto de falha na rede, uma vez que os nós fora do seu alcance ainda podem receber blocos e *pieces* tradicionalmente via *unicast*. No Sistema de Disseminação proposto, um *peer* pode ser uma semente disseminadora ou um *peer* comum. Um *peer* comum neste trabalho é considerado aquele que faz requisições por mensagens de `PIECE` e que eventualmente recebe este tipo de mensagem sem a necessidade de requisitá-la. A Figura 5.2 ilustra um *peer* disseminador e diversos pares comuns compartilhando um conteúdo via *unicast* e *broadcast*.

Periodicidade da disseminação - A disseminação não deve interferir negativamente no andamento da parte do *download* feita em *unicast*. A periodicidade pode ser modelada como uma variável aleatória com distribuição uniforme no intervalo $[0,M]$, sendo M um parâmetro de projeto.

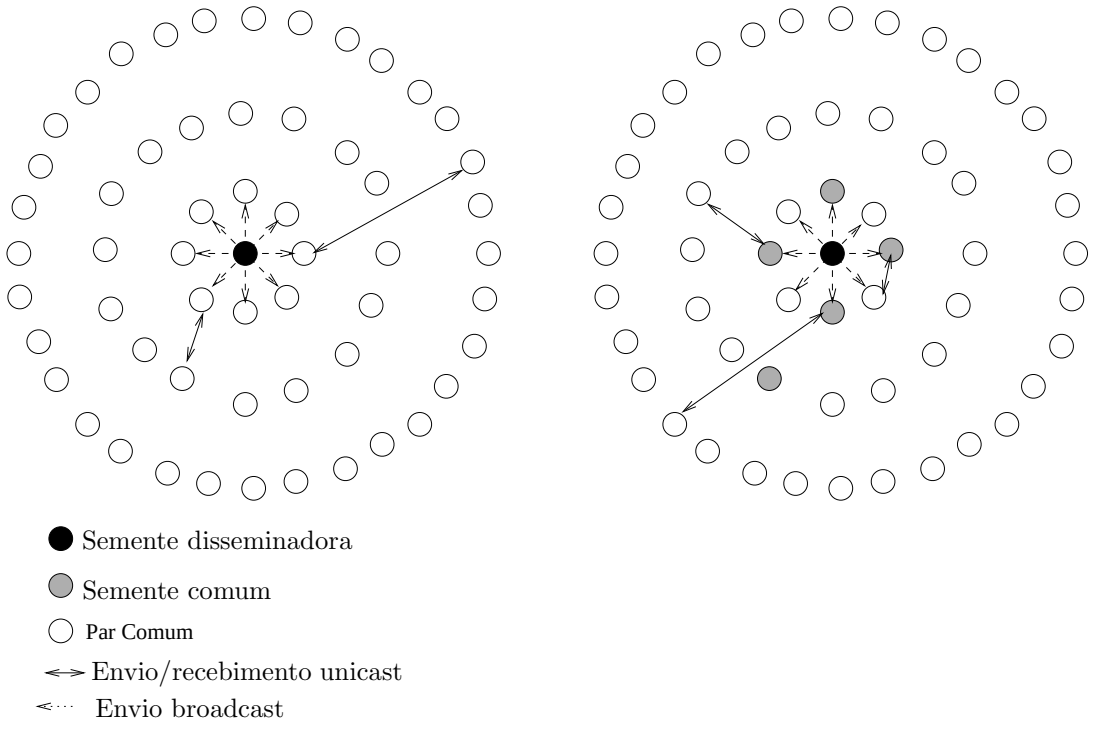


Figura 5.2 Aplicação BitTorrent em uma MANET com o Sistema de Disseminação

Alterações na implementação do BitTorrent - A implementação do BitTorrent deve ser alterada no tratamento de mensagens `BITFIELD`, `HAVE` e novas funcionalidades, como a escolha do *peer* disseminador e a seleção de *pieces* e blocos a serem disseminados devem ser acrescentadas.

5.1.1 BitTorrent Mobile Interface (BMI)

O BitTorrent Mobile Interface (BMI) é a interface criada para permitir que a camada aplicação possa solicitar o envio de mensagens BitTorrent em *broadcast*. O BMI é uma subcamada de aplicação que liga o BitTorrent ao protoloco de transporte UDP introduzindo novas funcionalidades. O BMI consiste em encapsular um identificador de interesse do *peer* nas mensagens do BitTorrent a serem difundidas. Este interesse identifica qual arquivo ele deseja baixar. De acordo com esta informação, a interface BMI pode saber se a mensagem recebida pertence ao arquivo desejado e deve, portanto, ser entregue ao

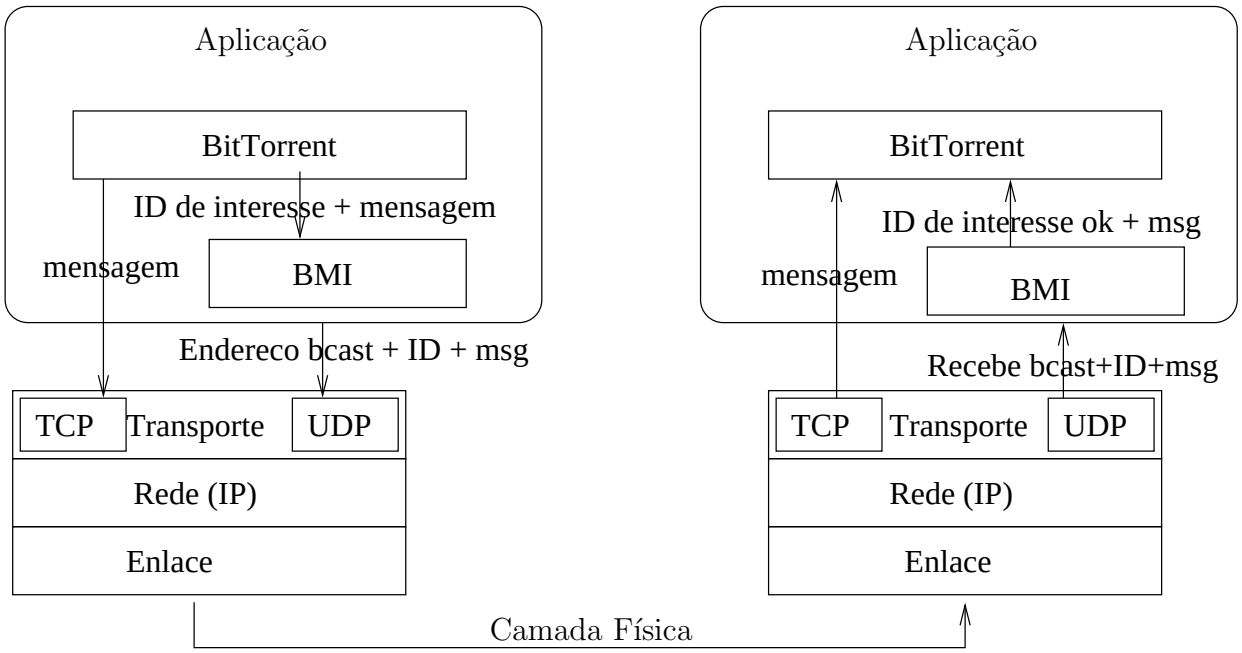


Figura 5.3 Pilha de protocolos com a interface BMI

BitTorrent. Outra função do BMI é garantir que o pacote seja direcionado ao endereço *broadcast* da rede *ad hoc*. A Figura 5.3 ilustra a comunicação entre as camadas aplicação, transporte e a interface BMI.

5.1.2 Algoritmo de escolha do peer disseminador

A escolha do único *peer* disseminador é feita pelo critério da primeira semente a entrar na rede. Cada *peer* executa um algoritmo distribuído que verifica o *status* dos demais aproveitando-se da mensagem `BITFIELD`, que já é recebida na implementação tradicional do BitTorrent. Este critério confere simplicidade ao algoritmo por não exigir comunicação extra entre os pares.

Com a recepção da primeira mensagem de `BITFIELD` é possível verificar se o emissor da mensagem também é uma semente. A cada recepção do `BITFIELD`, um contator é incrementado para verificar se todos os pares pertencentes ao *swarm* sugerido pelo *Tracker* já foram verificados. Se esta condição for atendida e nenhum desses pares for semente, o próprio *peer* se declara disseminador. Este algoritmo distribuído é descrito no

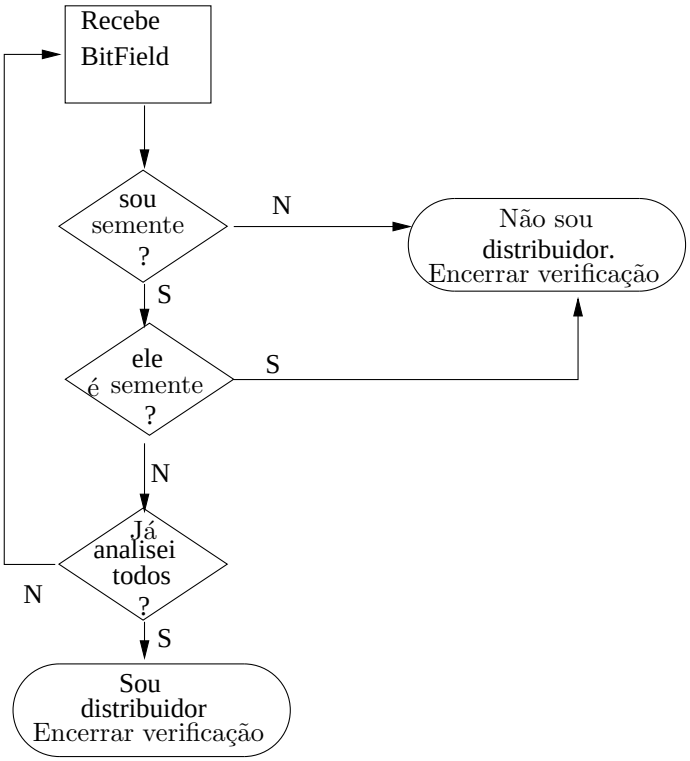


Figura 5.4 Fluxograma da escolha do disseminador

fluxograma da Figura 5.4.

5.1.3 Lógica de Seleção de Pieces e Blocos para Disseminação

A seleção de *pieces* e blocos para o envio de requisições na maioria das implementações tradicionais do BitTorrent segue o critério da raridade, ou seja, pela política de *rarest-first*. Os *pieces* mais raros são requisitados antes dos demais. Já o bloco, é escolhido aleatoriamente [31].

Para o Sistema de Disseminação proposto, o critério de seleção de *pieces* a serem enviados pelo disseminador deve ser diferente do critério utilizado para seleção de *pieces* requisitadas por um *peer* comum, para evitar mensagens redundantes. No entanto, se o critério de escolha para requisição de *pieces* em *unicast* for o da raridade, o critério de escolha de *pieces* para *broadcast* por parte do disseminador não deve ser o do menos raro. Esta combinação causaria uma distribuição preferencial dos *pieces* que a maioria

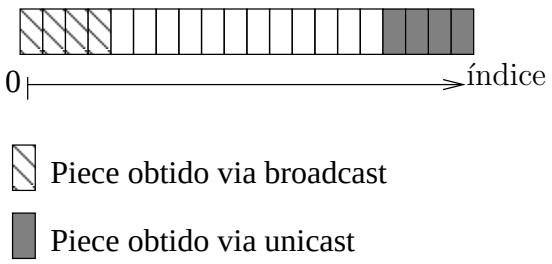


Figura 5.5 Blocos baixados via *broadcast* e *unicast*

dos pares já possuem, desperdiçando recursos. Assim, o critério do disseminador pode ser também o da escolha entre os *pieces* mais raros, mas com o compromisso de evitar escolher o mesmo *piece* que seria requisitado em *unicast*.

Para o Sistema de Disseminação, propõe-se que o *piece* a ser requisitado por um *peer* comum, seja o que tiver maior índice entre os mais raros. Para o disseminador, o *piece* com menor índice entre os mais raros é escolhido para ser distribuído. Em ambas as situações, a escolha é feita entre os mais raros, diferenciando-se apenas a escolha do índice. Assim, o preenchimento dos *pieces* ocorre em dois sentidos opostos, conforme ilustra a Figura 5.5. Se um arquivo for logicamente dividido em 20 *pieces*, por exemplo, um *peer* comum irá requisitar o *piece* 19 enquanto o disseminador escolherá o *piece* de índice 0 para enviar em modo *broadcast*. Caso exista apenas um *piece* mais raro, o disseminador escolhe o segundo mais raro e assim por diante.

Além da escolha do índice do *piece*, a escolha de um bloco para requisição/disseminação deve também ser contemplada pelo Sistema de Disseminação. Assim, propõe-se que um *peer* comum os escolha na ordem crescente do *offset* do bloco. Já o disseminador precisa ter conhecimento mínimo do *download* dos demais pares em relação ao blocos que já foram baixados para saber quais deve distribuir. Para isto, a mensagem de `BITFIELD` passa a armazenar a informação também no nível de blocos. Assim, a mensagem de `BITFIELD` agora informa quais blocos de um *piece* foram baixados até então, não só quais *pieces* estão completos, como na implementação tradicional. O disseminador aproveita a chegada de mensagens `HAVE` para atualizar esta informação. Adicionalmente, o disseminador nunca envia mais de uma vez um mesmo bloco, ou seja, ele mantem a informação de

quais blocos ele ainda não difundiu. Esta visão parcial do *download* dos pares é suficiente para escolher o bloco mais raro.

5.1.4 Diferenciais do Sistema de Disseminação

Esta proposta se difere dos trabalhos relacionados por permitir que se aproveite o poder difusor da MANET sem maiores interferências no modo como é feito o *download* pelo transporte confiável e como se estrutura a comunicação na pilha de protocolos.

Enquanto a proposta *cross-layer* de Souza *et al.* [51] requer que grupos sejam formados e que todos os membros do grupo, exceto o líder se desconectem da rede para receber dados via *multicast*, o Sistema de Disseminação proposto mantem a autonomia dos pares que não são a semente disseminadora. A característica do Sistema de Disseminação de independência do disseminador é fundamental para manter a tolerância a falhas. Além disso, nesta proposta, não há necessidade de troca de mensagens entre os pares para definir o *peer* disseminador, economizando recursos da MANET.

O trabalho de Balázsfalvi *et al.* em [39] foca na descoberta de serviços como fator de otimização, enquanto o Sistema de Disseminação proposto se concentra no processo de *download* em si.

No trabalho de Rajagopalan *et al.* [32], a camada de aplicação é totalmente integrada a de roteamento para garantir uma maior eficiência do BitTorrent em MANETs com o *cross-layering*. No Sistema de Disseminação, não se interfere na arquitetura da pilha de protocolos, garantindo a simplicidade em uma eventual implementação, facilitando a sua instalação em um dispositivo móvel.

O BitHoc de Krifa *et al.* [50] conta com informações provenientes do protocolo de roteamento OLSR para atualizar o *Tracker* sobre *peers* disponíveis também com auxílio de *cross-layering*. O Sistema de Disseminação proposto neste trabalho, por outro lado, é independente da escolha protocolo de roteamento utilizado, sendo assim, mais flexível.

5.2 RESUMO

Neste capítulo foi apresentado o Sistema de Disseminação proposto, cujo principal objetivo é oferecer um compartilhamento mais eficiente de arquivos no BitTorrent em MANETs. O sistema consiste em um conjunto de otimizações no protocolo BitTorrent para disseminação de parte do conteúdo do *download*, bem como uma interface para encaminhar este conteúdo para transporte não-confiável via *broadcast*.

CAPÍTULO 6

SIMULAÇÕES E RESULTADOS

A avaliação de desempenho é uma questão fundamental em Sistemas Distribuídos. Com ela, é possível identificar problemas e pontos onde devem ser feitas otimizações no sistema estudado. Por outro lado, a avaliação de desempenho também é fundamental na validação de novas propostas. Este capítulo descreve os experimentos conduzidos para a avaliação do desempenho de uma aplicação BitTorrent em uma MANET com e sem o Sistema de Disseminação proposto. Os resultados obtidos e os cenários avaliados também são discutidos.

6.1 METODOLOGIA DO EXPERIMENTO

O objetivo do experimento é conhecer o desempenho do Sistema de Disseminação proposto perante cenários fixos e móveis avaliando as vantagens e desvantagens da sua adoção. A técnica de avaliação adotada foi a simulação com o Network Simulador (NS-2), assim como no estudo preliminar. Nas seções a seguir detalhes do modelo, cenários e das métricas utilizadas para a realização dos experimentos são apresentados.

6.1.1 Implementação do modelo

Para execução das simulações, o NS-2 precisou ser modificado para dar suporte aos protocolos BitTorrent e OLSR, que foi o protocolo de roteamento escolhido para este experimento. Adotou-se as implementações de Eger *et. al* [54] para o BitTorrent e de Paquerau *et. al* [55] para o OLSR. Na implementação do BitTorrent de Eger *et. al*, o Tracker está modelado como um objeto visível a todos. Decidiu-se que este aspecto seria mantido, uma vez que o foco nesta avaliação é verificar o desempenho durante o *download* e não

durante a descoberta de pares. Para a avaliação do Sistema de Disseminação proposto, foi criada uma versão modificada da implementação de Eger *et. al.* A modificação envolve a adição dos algoritmos de escolha do *peer* disseminador, de determinação da periodicidade e da escolha de peças/blocos para disseminação. Além disso, também foi adicionada a interface BMI para a transmissão de dados em modo *broadcast*.

A periodicidade da distribuição em *broadcast* é definida por uma variável aleatória com distribuição uniforme entre 0 e 1 segundo. Este valor foi obtido através da observação do desempenho com diversos valores, dentre os quais mostrou-se o mais adequado para os cenários estudados. Para modelagem das camadas MAC e física foram adotados os parâmetros da especificação IEEE 802.11g. O range de comunicação dos nós foi fixado em 50 m. O modelo de propagação de sinais utilizado foi o *Two Ray Ground*.

O modelo de mobilidade escolhido foi o *Random Waypoint*. Neste modelo, os nós se movem rumo a um destino aleatório após um tempo de pausa. Ao chegar a este destino, voltam a aguardar o mesmo tempo de pausa para posteriormente escolherem um novo destino e assim por diante.

6.1.2 Cenários de Simulação

Os cenários de simulação descritos neste experimento procuram refletir uma situação semelhante ao compartilhamento de um vídeo promocional de aproximadamente 5 minutos durante uma palestra, utilizando uma rede *ad hoc* criada para este propósito. Cada participante dispõe de um dispositivo móvel, e , por questões de simplicidade, pode-se considerar que a quantidade de participantes não ultrapasse cinquenta. Dependendo da ocasião o vídeo pode ser compartilhado durante a apresentação ou ao seu fim, quando os espectadores começam a deixar o espaço da palestra, se dirigindo a outros lugares.

Para este estudo de caso considera-se o compartilhamento de um arquivo de 100MB com peças de tamanho 512KB e blocos de 16KB. Os nós inicialmente se encontram dispostos uniformemente em uma topologia de grade, distribuídos uniformemente à distância de 1 m entre vizinhos de uma mesma fileira. Cada nó executa uma única instância da aplicação BitTorrent. O *peer* que finaliza um *download* continua na aplicação para

colaborar com os demais. A simulação se inicia sempre com apenas uma semente. A simulação se encerra no momento em que todos os *peers* concluem o *download*. Não há acréscimo, nem retirada de nós durante a simulação. São definidos dois cenários: fixo, com variação no número de *peers* e móvel, com variação na velocidade nos nós. A seguir, um detalhamento destes cenários:

- **Cenário fixo -** os nós permanecem imóveis até o fim do *download*. O número de *peers* simulados são 4 (2x2), 9 (3x3), 16 (4x4), 25 (5x5), 36 (6x6) e 49 (7x7).
- **Cenário móvel -** neste cenário 25 nós se movem de acordo com o modelo *Random Waypoint* com tempo de espera de 10 segundos. Os nós se deslocam em direções aleatórias em uma área de 150m x 150m. As simulações consideram velocidades de 0,5 m/s, 1,0 m/s e 1,5 m/s.

6.1.3 Métricas de Simulação

As métricas a seguir foram avaliadas em ambos os cenários estudados:

- **Tempo médio de *download* do conteúdo -** tempo médio necessário para conclusão do *download*;
- ***Overhead* de roteamento -** representa o número total de pacotes de roteamento transmitidos durante a simulação, incluindo-se transmissões entre saltos;
- ***Overhead* de segmentos -** corresponde ao número total de segmentos enviados no nível de transporte via TCP e UDP, quando utilizado;
- **Número médio de pacotes descartados -** representa a quantidade de pacotes descartados na rede;
- **Atraso médio fim-a-fim -**tempo decorrido entre o envio e recebimento de segmentos no nível de transporte, tanto TCP quanto UDP.

Todos os valores apresentados nos gráficos são médias calculados a partir de 20 execuções com um intervalo de confiança de 99% representado por barras de erro nos

gráficos. Experimentos anteriores demonstraram que esta quantidade de execuções é suficiente para obter intervalos adequadamente pequenos. Em alguns gráficos, observa-se que este intervalo é visualmente imperceptível.

6.2 ANÁLISE DOS RESULTADOS

As seções seguintes descrevem o impacto do uso do Sistema de Disseminação nos cenários fixo e móvel.

6.2.1 Cenário Fixo

A Figura 6.1 apresenta o tempo de *download* em função do número de *peers* na rede. Espera-se que o tempo de *download* cresça com o aumento desta grandeza. Entretanto, observa-se que com o Sistema de Disseminação este crescimento é menos acentuado que na abordagem sem o mesmo. As vantagens do Sistema de Disseminação são percebidas para valores pouco maiores que 9 *peers*, quando uma maior quantidade é contemplada com os dados recebidos via *broadcast* causando uma melhoria no desempenho em comparação com a abordagem tradicional. Para 49 *peers*, por exemplo, o uso do Sistema de Disseminação reduziu em 30% o tempo de *download* com relação à abordagem tradicional. Este resultado é uma consequência da redução do número de mensagens trocadas entre os *peers*, uma vez que algumas mensagens de `PIECE` são enviadas em modo *broadcast*, diminuindo a geração de mensagens de `REQUEST`, o que reduz o tempo para que se receba o arquivo completo.

A Figura 6.2 apresenta o *overhead* de segmentos em função do número de *peers*. Graças ao Sistema de Distribuição, o número de segmentos de transporte é reduzido em relação à abordagem tradicional. Isto ocorre porque menos requisições por peças são enviadas, já que parte delas já chegam pela difusão periódica por parte do disseminador. Isto se reflete em um número menor de segmentos. No caso de 36 *peers*, o número de segmentos com o Sistema de Distribuição é reduzido em 50%.

A Figura 6.3 mostra *overhead* de roteamento em função do número de *peers*. Quando

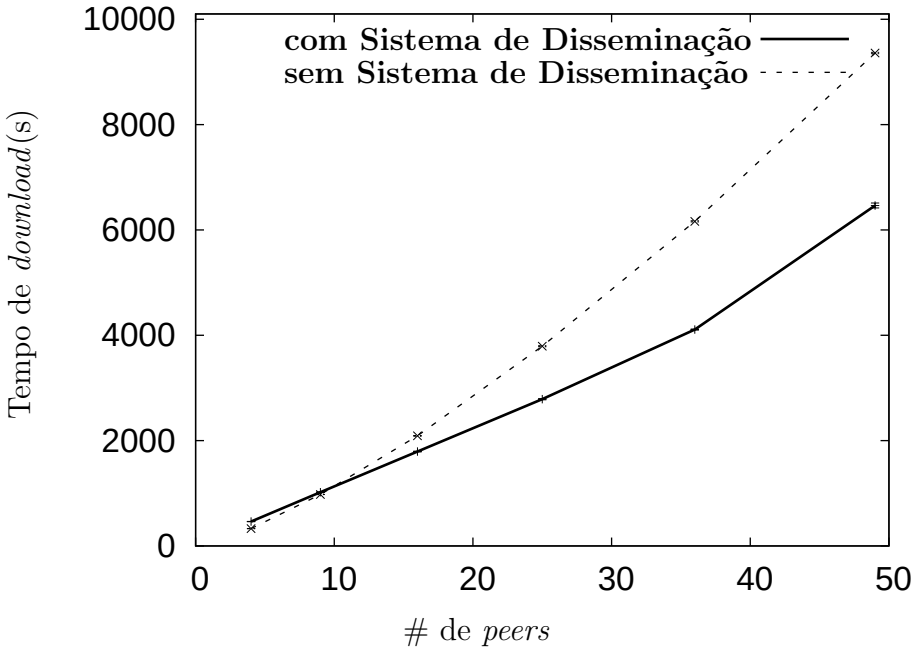


Figura 6.1 Tempo de *download* em função do número de *peers*

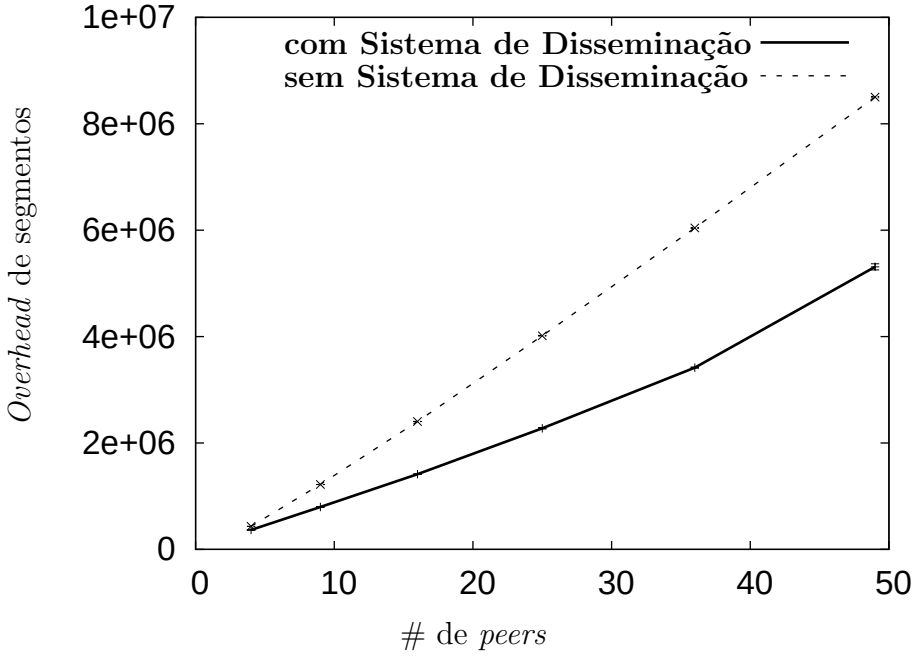


Figura 6.2 *Overhead* de segmentos em função do número de *peers*

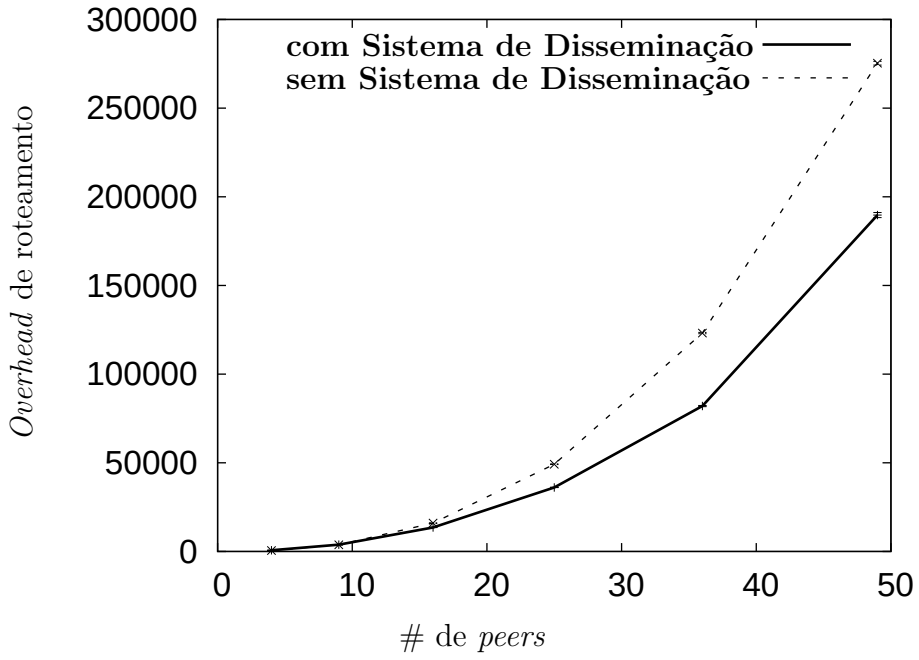


Figura 6.3 *Overhead* de roteamento em função do número de *peers*

o número de *peers* é pequeno, não há grandes diferenças entre as abordagens. Entretanto, à medida que o número de *peers* aumenta, a diferença torna-se mais evidente. Para 49 *peers*, o uso do Sistema de Distribuição fez com que a camada de roteamento gerasse cerca de 20% de pacotes a menos que a abordagem tradicional, como consequência de um número menor de segmentos transportados com o uso do sistema proposto.

A Figura 6.4 ilustra o número de pacotes descartados em função do número de *peers*. Uma vez que número de segmentos enviados diminuiu com o Sistema de Distribuição, isto se refletiu na quantidade de perdas que foi reduzida em até 40%.

A Figura 6.5 mostra o atraso médio fim-a-fim em função do número de *peers*. A diferença entre as duas abordagens é muito pequena. Neste caso, a abordagem com o Sistema de Disseminação mostra um atraso maior a partir de 36 *peers*. Este atraso maior mostra a influência das mensagens geradas pelo disseminador no aumento da fila dos nós receptores, causando atrasos na entrega de outras mensagens, aumentando o atraso médio fim-a-fim como um todo. Este fenômeno é mais perceptível com o aumento do número de nós. Entretanto, observa-se pelas barras de erros que a diferença ainda é pequena. Um

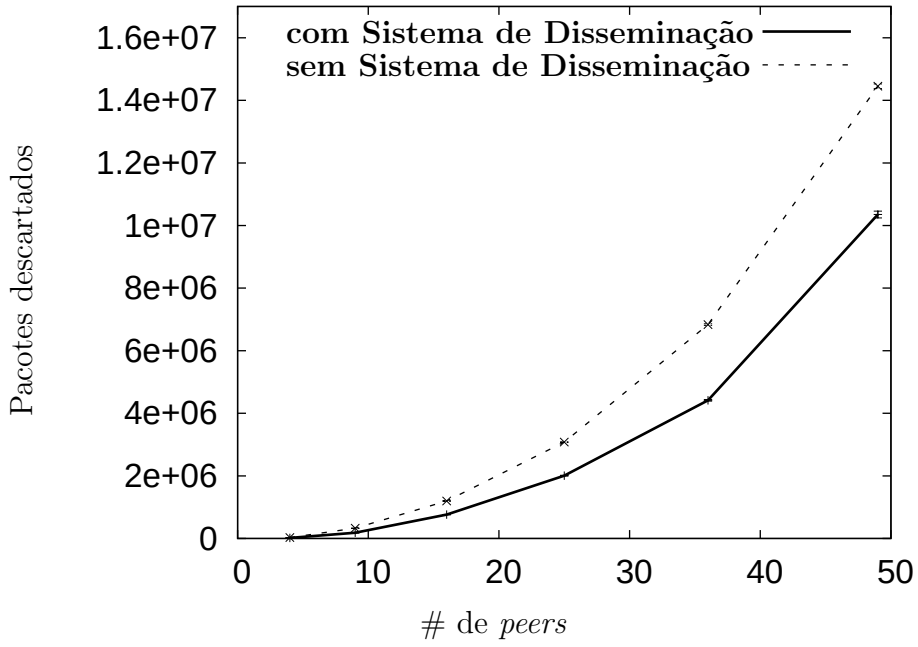


Figura 6.4 Pacotes descartados em função do número de *peers*

fato interessante é que apesar do atraso médio fim a fim ser ligeiramente maior para o Sistema de Disseminação proposto, isto não impacta no desempenho do *download*, já que a quantidade de segmentos gerados é muito menor que na abordagem tradicional. Isto significa que a diferença entre os atrasos com e sem o Sistema de Disseminação proposto tem pouco impacto no tempo de *download*.

6.2.2 Cenário Móvel

A Figura 6.6 mostra a variação do tempo de *download* em função da velocidade dos nós. Neste cenário, observa-se que houve pequena variação do tempo de *download* ao se variar a velocidade para ambos os casos, com e sem o Sistema de Disseminação. Isto se deve à capacidade do protocolo de roteamento OLSR de se adaptar à situações de mobilidade, graças à sua natureza proativa. Desta forma, o tempo de *download* para o cenário móvel permanece semelhante ao caso de 25 *peers* do cenário fixo. Assim como no cenário fixo, percebemos que a abordagem com o Sistema de Disseminação tem desempenho ligeiramente superior à metodologia tradicional.

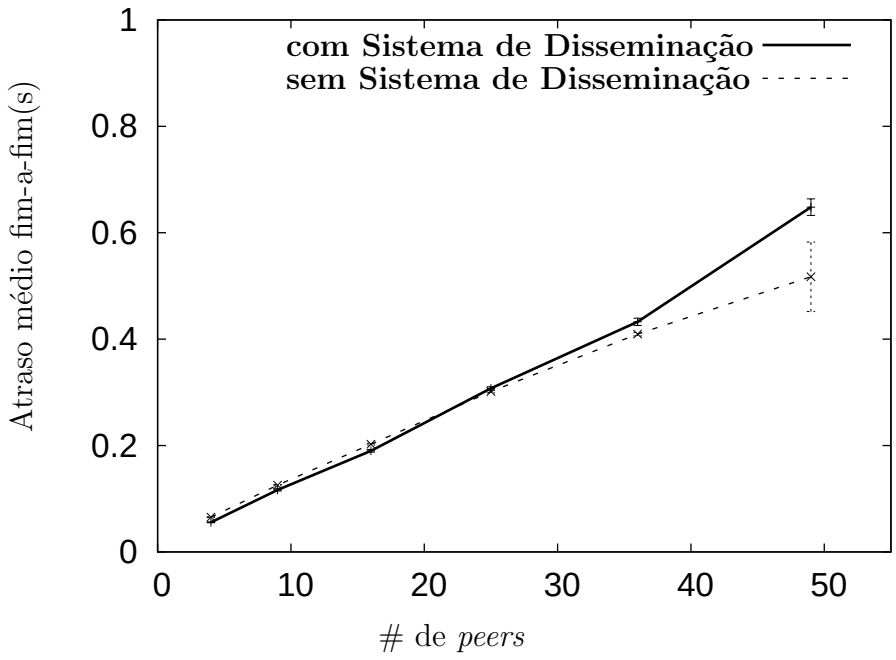


Figura 6.5 Atraso médio fim-a-fim em função do número de *peers*

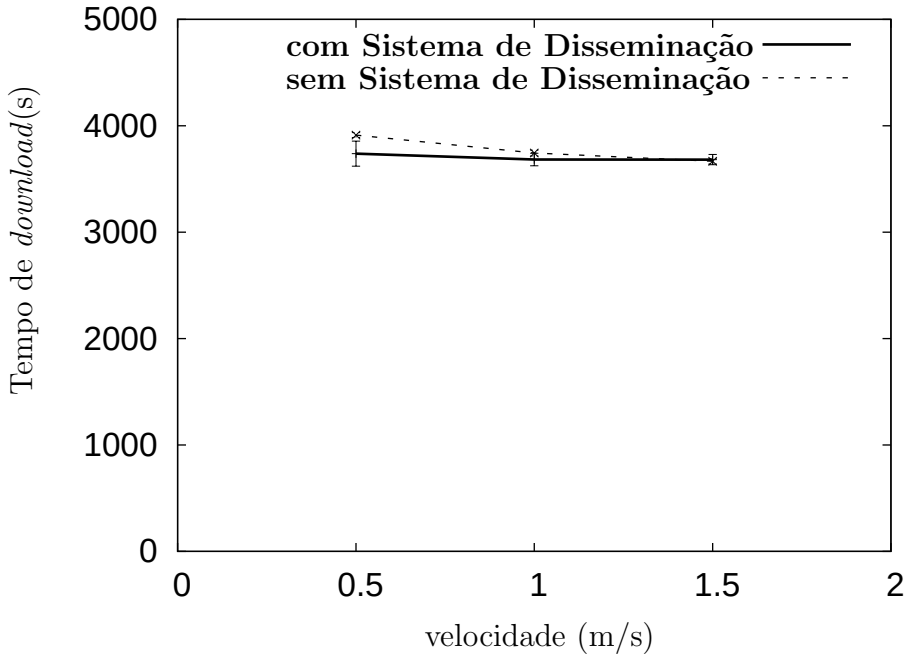


Figura 6.6 Tempo de *download* em função da velocidade

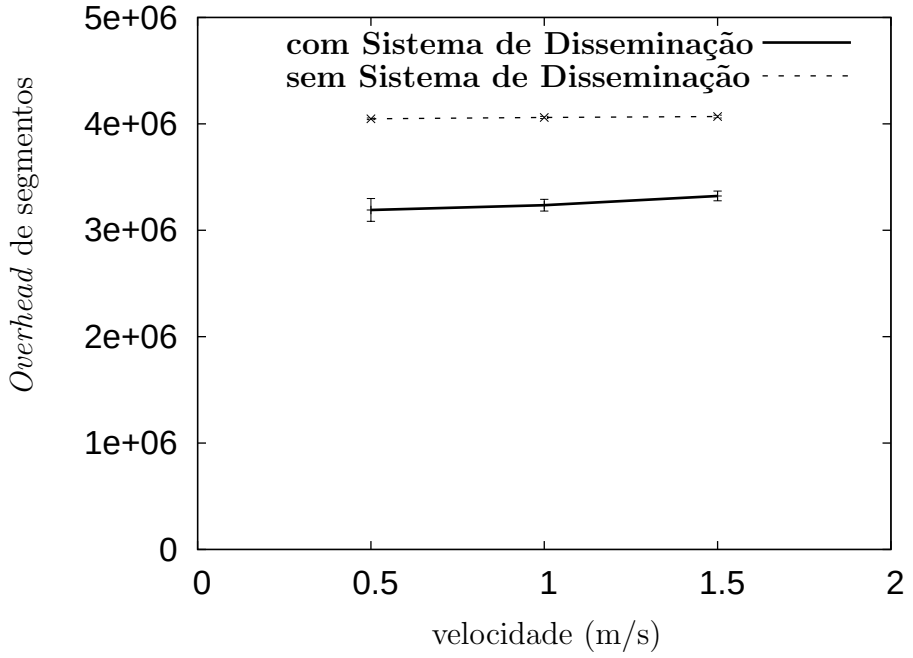


Figura 6.7 *Overhead* de segmentos em função da velocidade

A Figura 6.7 mostra o número de segmentos de transporte em função da variação da velocidade dos nós. Mais uma vez observa-se a superioridade do Sistema de Disseminação, assim como foi observado no cenário fixo, devido à redução no número de mensagens de requisição, graças à distribuição de mensagens de `PIECE` por parte do disseminador. O número de segmentos com o Sistema de Distribuição é aproximadamente 25% menor.

Para o *overhead* de roteamento, ilustrado na Figura 6.8, observa-se que nesta situação também pode-se atribuir ao OLSR o mérito da baixa variação do número de pacotes de roteamento trocados durante a simulação. Nota-se que os gráficos são estatisticamente muito semelhantes.

A Figura 6.9 mostra o número de pacotes descartados em função da velocidade. A diminuição de segmentos transportados na rede com o Sistema de Disseminação proposto impacta no número de pacotes descartados na rede.

A figura 6.10 mostra o atraso fim-a-fim de segmentos em função da velocidade. Neste caso, a abordagem tradicional mostra valores ligeiramente menores, assim como no caso fixo, sem maiores influências no tempo de *download*. Conforme observado anteriormente,

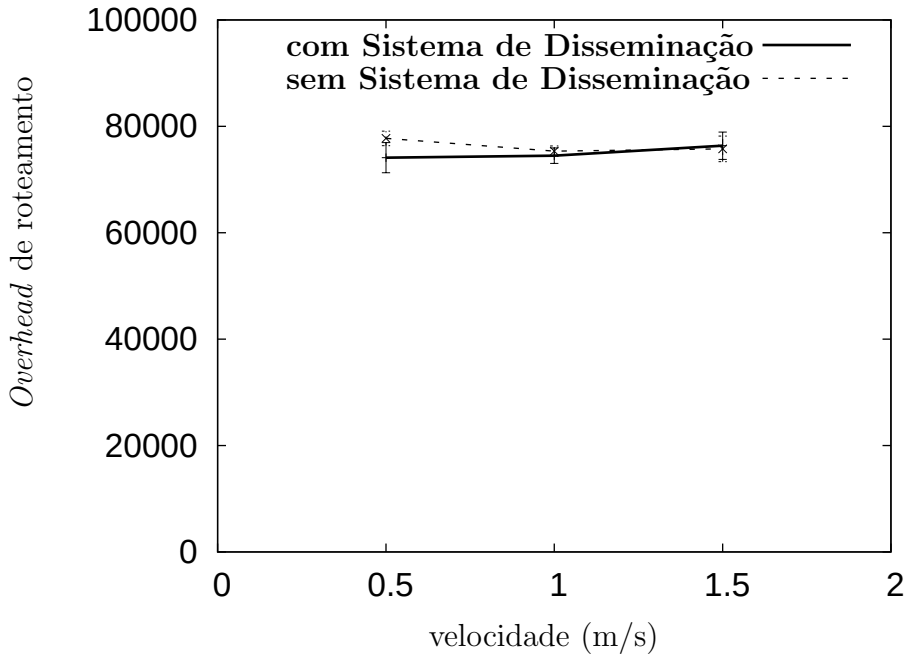


Figura 6.8 *Overhead* de roteamento em função da velocidade

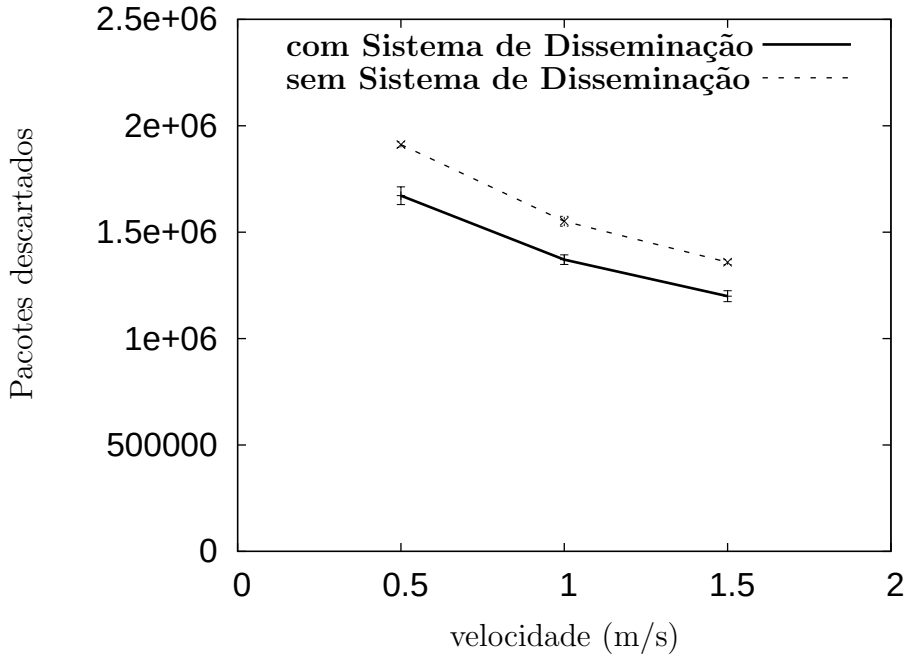


Figura 6.9 Pacotes descartados em função da velocidade

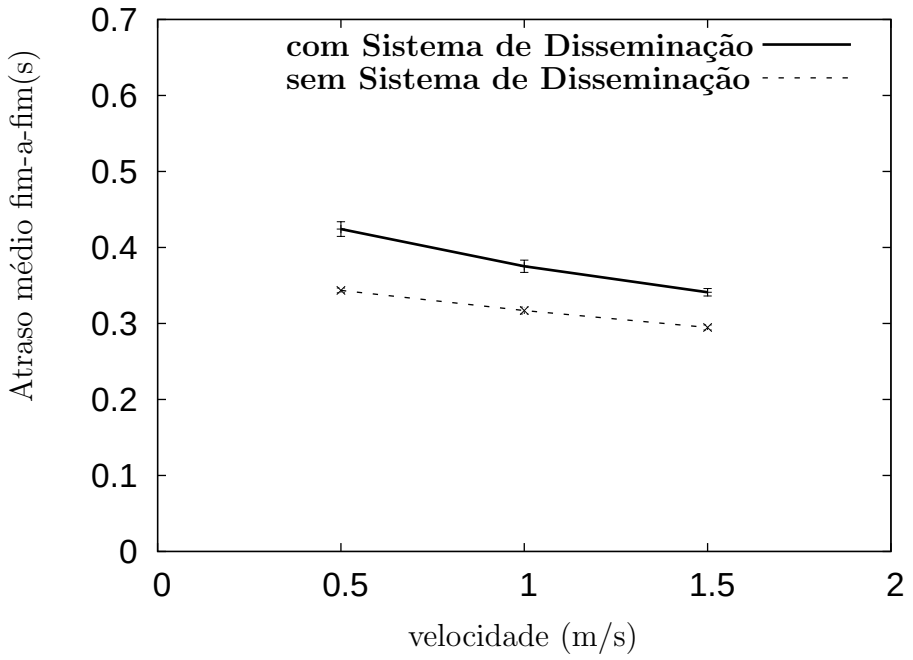


Figura 6.10 Atraso médio fim-a-fim em função da velocidade

este fenômeno se origina do impacto da geração de mensagens por parte do disseminador na fila dos *peers* comuns atrasando o envio das demais mensagens. Assim como no cenário fixo, a quantidade de segmentos gerados é muito menor que na abordagem tradicional, de modo que este resultado tem pouco impacto no desempenho do tempo de *download*.

6.3 RESUMO

Experimentos foram conduzidos com o objetivo de avaliar o desempenho do Sistema de Disseminação proposto em um cenário fixo e um cenário móvel. Os experimentos mostraram que o tempo de *download* pode ser reduzido a até 30% em relação a abordagem sem o sistema proposto no cenário fixo. Este resultado tem impacto expressivo na escalabilidade da aplicação. Neste cenário ainda, pôde-se observar que o *overhead* de segmentos foi reduzido em até 50% e o de roteamento diminuiu em até 25% com o Sistema de Disseminação. Para o cenário móvel confirmou-se a melhor desempenho do Sistema de Disseminação perante a abordagem tradicional em maior parte dos casos observados para este cenário. Desta forma conclui-se que o objetivo de se obter uma forma mais efici-

ente de compartilhar arquivos no BitTorrent em MANETs foi atingido com o Sistema de Disseminação proposto.

CAPÍTULO 7

CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresentou um Sistema de Disseminação como uma proposta para modificação da aplicação BitTorrent em MANETs para melhoria do desempenho diante da abordagem tradicional. Estudos preliminares mostraram que a simples inclusão do BitTorrent nesta arquitetura apresenta problemas de desempenho. Os trabalhos relacionados levantaram possíveis pontos de melhorias em protocolos como o BitTorrent para melhorar seu funcionamento em MANETs.

O Sistema de Disseminação proposto permitiu que o ambiente sem fio fosse melhor aproveitado ao permitir que parte das mensagens de `PIECE` do BitTorrent fossem entregues em modo *broadcast* ao invés de *unicast*, como ocorria na abordagem tradicional. Para tal, o Sistema de Disseminação proposto contou com um conjunto de estratégias para definir os critérios de difusão e com a proposição da *BitTorrent Mobile Interface*, definindo uma interface de disseminação. Além disso, este sistema possui o diferencial da simplicidade para eventual implementação em um dispositivo móvel, uma vez que não é uma abordagem *cross-layer*, de modo que não requer alterações no sistema operacional. Outro diferencial é a não introdução de pontos de falha, já que os *peers* comuns da rede não dependem exclusivamente do nó disseminador para receber conteúdo.

Os experimentos permitiram observar que o desempenho geral do sistema melhorou com o Sistema de Disseminação proposto, em termos de tempo de *download* e *overhead* de segmentos no nível de transporte, em relação à abordagem tradicional. Observou-se que o tempo de *download* se reduziu em até 30% e o *overhead* de segmentos diminuiu em até 50% em comparação à abordagem sem o Sistema de Disseminação proposto. O objetivo de se obter uma forma mais eficiente de compartilhar arquivos no BitTorrent em MANETs foi atingido com o Sistema de Disseminação proposto.

Em trabalhos futuros os pontos de melhorias não abordados na presente dissertação podem ser estudados. Por exemplo, o critério de seleção de *pieces* e blocos se mostrou eficaz, porém pode ser aprimorado. Além da raridade, pode-se considerar a localização da peça desejada e dar prioridade às mais próximas. Adicionalmente, o *Tracker* poderia fazer uma seleção de *peers* levando em consideração a proximidade do nó que lhe fez a requisição pela lista de *peers*. O critério de escolha do disseminador pode ser refinado, dependendo da densidade da rede ao seu redor. Para ambos os casos, o *cross-layering* faz-se necessário, uma vez que a informação de vizinhança é fornecida pelos protocolos de roteamento. Neste caso, a relação custo/benefício no uso de *cross-layering* deve ser cuidadosamente observada.

Novos experimentos podem incluir cenários não contemplados nesta dissertação, como incluir mais sementes no início das simulações e verificar o comportamento da rede quando mais de um arquivo é compartilhado simultaneamente, entre *peers* com interesses diferentes. A escolha do valor máximo M da variável aleatória que descreve a periodicidade da disseminação pode ser feita de acordo com um estudo futuro. Este estudo pode verificar o impacto da porcentagem de mensagens disseminadas com relação ao total de mensagens transmitidas no desempenho geral do sistema, apontando um valor ideal para M.

Em trabalhos futuros também pode-se modelar a descoberta de recursos por parte do *Tracker*. Para tal, considera-se o uso de DHTs para tornar o *Tracker* distribuído. Uma implementação da DHT Kademlia é um passo natural, uma vez que na Internet esta rede já é empregada para a descentralização do *Tracker*. O impacto do aumento de informações de estado nos nós pode ser estudado futuramente, bem como Finalmente, questões de segurança e reputação podem ser abordadas em novas implementações, levando em consideração a prevenção a *peers* maliciosos e cifragem de pacotes.

APÊNDICE A

MODELAGEM DO SISTEMA DE DISSEMINAÇÃO NO NS-2

A.1 O SIMULADOR NETWORK SIMULATOR (NS-2)

O NS-2 [56] é um simulador de redes de propósito geral e de eventos discretos implementado em Tcl/OTcl e C++. A primeira é usada para a criação de *scripts* de simulação, enquanto a última é utilizada para a implementação do simulador e dos protocolos utilizados na simulação. Suporta diversas camadas físicas e protocolos de rede, incluindo aqueles projetados para redes sem fio. É uma das ferramentas de simulação mais respeitadas em avaliações de desempenho de redes e sistemas distribuídos, graças à ampla disponibilidade de recursos e a sua credibilidade no meio científico.

A.2 MODELAGEM DO SISTEMA DE DISSEMINAÇÃO

Para introduzir novos protocolos e alterar implementaçõe de protocolos pré-existentes, é necessário alterar a implementação C++ do NS-2. Para esta proposta, foram acrescentados os módulos OLSR e BitTorrent, sendo este último modificado para dar suporte ao Sistema de Disseminação proposto. A figura A.1 mostra a estrutura de heranças entre as principais classes do NS-2 destacando a introdução dos protocolos BitTorrent e OLSR, bem como a interface BMI, implementada como herdeira da classe `Agent` para o Sistema de Disseminação.

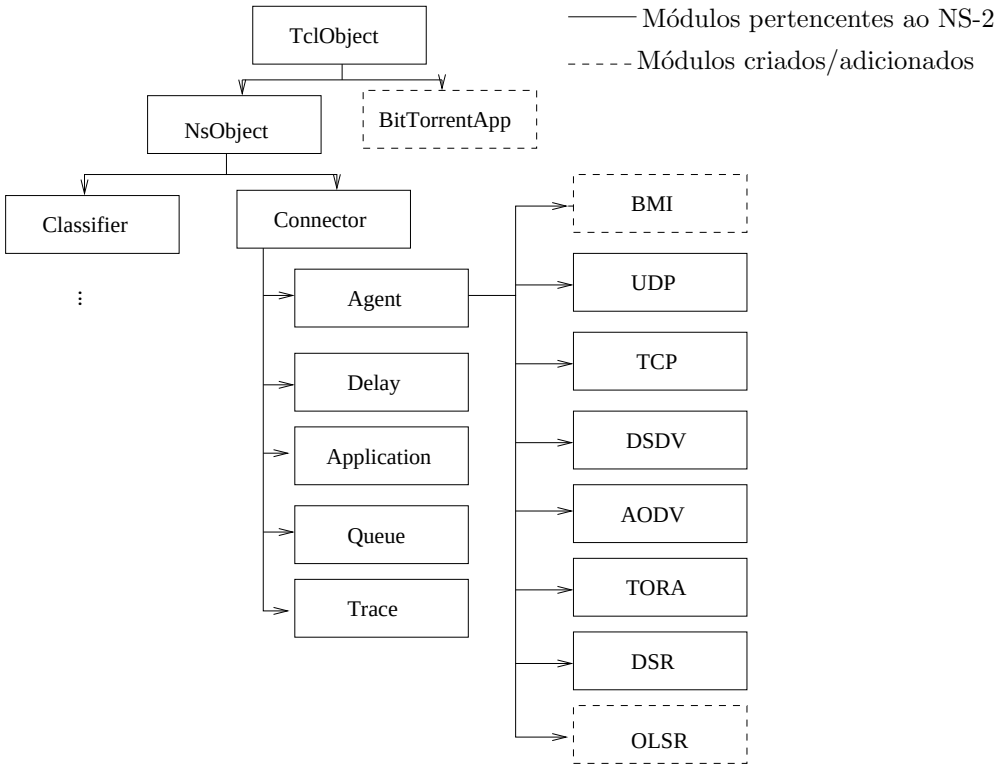


Figura A.1 Estrutura do Network Simulator

A Figura A.2 mostra um diagrama de classes do módulo BitTorrent. A classe `TCPAgent` é pertencente à implementação original do NS-2 e as classes `BMI` e `BcastTimer` pertencem à implementação do Sistema de Disseminação. As demais classes pertencem ao módulo BitTorrent de Eger *et. al* [54]. A seguir, uma breve descrição de cada classe:

- `BitTorrentApp` - Implementação da aplicação BitTorrent. Esta classe foi modificada para interagir com a subcamada BMI e verificar os critérios para seleção de *pieces* e blocos para disseminação, bem como a periodicidade da distribuição.

- `BitTorrentTracker` - Abstração do Tracker. É um objeto estático, visível a todas as instâncias de `BitTorrentApp`. Retorna os *peers* que fazem parte do *swarm* de *peers*.

- `BitTorrentConn` - Representa uma conexão ponto-a-ponto no BitTorrent. Possui um instância do agente `TCPAgent` para estabelecer as conexões. Responsável pelo transporte confiável de mensagens.

- `TCPAgent` - Implementação do protocolo TCP padrão do NS-2.
- `BMI` - Representa a interface BMI, que encaminha as mensagens de `PIECE` repassadas pela aplicação BitTorrent para serem disseminadas via *broadcast*.
- `BcastTimer` - Classe de temporização para determinar a periodicidade da disseminação.
- `BitTorrentData` - Classe que representa as mensagens do BitTorrent a serem transmitidas.
- `BitTorrentDataList`- Classe para enfileirar logicamente as instâncias de `BitTorrentData`, facilitando o controledo envio/recebimento de mensagens na simulação.

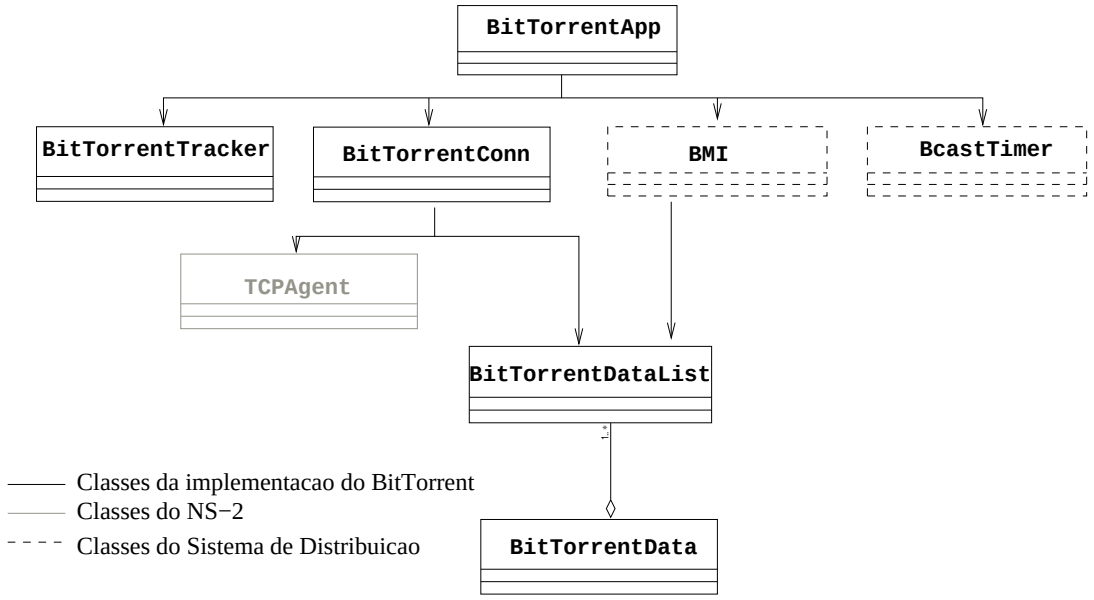


Figura A.2 Diagrama de classes do Módulo BitTorrent

APÊNDICE B

SCRIPT TCL DE SIMULAÇÃO

Um *script* de simulação foi escrito em Tcl para definir os cenários fixo e móvel e iniciar a simulação. O código fonte é apresentado a seguir.

```
#============================================================#
# Informatics Center - Federal University of Pernambuco     #
# Author: Nivia Cruz Quental                                 #
# Implementation of a bittorrent network over a 802.11g mac #
# layer using OLSR routing protocol.                         #
# This script uses the Eger's bittorrent C++ implementation,#
# available at                                               #
# www.tu-harburg.de/et6/research/bittorrentsim/index.html    #
# Parameters: number of peers, file size in MB, time to      #
# start movement in s (-1, for fixed network)                #
# piece size in KB, speed in m/s (any value, for fixed       #
# network), scenario file                                    #
#                                                            #
# Terminology: v<name> for variables created in  this script#
#              p<name> for parameters created in this script#
# February, 2008                                             #
#============================================================#

#---------------------- PROCEDURES -------------------------#
#done is called internally by BitTorrentApp::stop()
proc done {} {
    global vBt_app vFinished_peers pNum_peers vNamTraceFile \
                vTraceFile vNs

    incr vFinished_peers

    if {$vFinished_peers == $pNum_peers} {
        for {set i 0} {$i < $pNum_peers} {incr i} {
            $vBt_app($i) stop
        }
        $vNs flush-trace
        close $vTraceFile
```

```
        puts "meu_bt_wireless.tcl:  [clock seconds]"  ;

    exit 0
    }
}


#------------------INPUT PARAMETERS HANDLING---------------#
global argv


if { $argc > 0 } {
    set i 1
        foreach vArg $argv {

        if {$i==1} {
            set pNum_peers $vArg
        }
        if {$i==2} {
            set pF_size_MB $vArg
        }
        if {$i==3} {
            set pMove_time $vArg
        }
                if {$i==4} {
                        set pChunk_size [expr $vArg *1024]
                }
                if {$i==5} {
                        set pSpeed  $vArg
                }
                if {$i==6} {
                        set scenTclFile $vArg
                }
                if {$i==7} {
                        set topoDim $vArg
                }

    incr i
        }
}

if {$argc != 7} {
    puts "meu_bt_wireless.tcl: Usage-> ns meu_bt_wireless \
       <number of peers> <file size in MB>              \
```

```
       <time to start movement in s> <chunk size in KB> \
       <speed in m/s>  <scenario file>                  \
       <topology dimension in m>"
    exit 0
}


#-BITTORRENT PARAMS (invokes EGER's "bittorrent_default.tcl")-#

source ~/MyNS/ns-2.32_cross/bittorrent/bittorrent_default.tcl

BitTorrentApp set leave_option -1

# number of seeds
set pNum_seeds 1
# queue size at access links (default 50)
set pQ_size_access 25
# delay
set pDelay_min 1
set pDelay_max 50
# file size in bytes
set pF_size [expr $pF_size_MB * 1024.0 *1024]
# number of chunks
set pNum_chunks [format %.0f [expr ceil($pF_size / $pChunk_size)]]

#------------------- CONTROL VARIABLES ---------------------#
set vPeer_count 0
set vFinished_peers 0

#--------SCHEDULER CREATION AND NETWORK BASIC SETUP---------#
#Create a simulator object
set vNs [new Simulator]

remove-all-packet-headers
add-packet-header OLSR TCP Flags

$vNs use-scheduler Heap

# set MSS for all FullTCP connections
Agent/TCP/FullTcp set segsize_ 1460
Queue set limit_ $pQ_size_access

#set global random number generator
global defaultRNG
```

```
$defaultRNG seed [clock seconds]

# set 802.11g configurations (found in mac/mac-802_11.cc)

Mac/802_11 set SlotTime_          0.000020        ;# 20us
Mac/802_11 set SIFS_              0.000010        ;# 10us
Mac/802_11 set dataRate_          54.0e6          ;# 54Mbps
Mac/802_11 set basicRate_         1.0e6           ;# 1Mbps

Mac/802_11 set CWMin_               15
Mac/802_11 set CWMax_               1023
Mac/802_11 set PreambleLength_      96
Mac/802_11 set PLCPHeaderLength_    40
Mac/802_11 set PLCPDataRate_        6.0e6        ;# 6Mbps
Phy/WirelessPhy set RXThresh_ 7.69113e-08 ;# range de 50m


# Wireless setup:
set wless(chan)   Channel/WirelessChannel
set wless(prop)   Propagation/TwoRayGround
set wless(netif)  Phy/WirelessPhy
set wless(mac)    Mac/802_11
set wless(ifq)    Queue/DropTail/PriQueue   ;# for aodv
set wless(ll)     LL
set wless(ant)    Antenna/OmniAntenna
set wless(x)      $topoDim    ;# X dimension of the topography
set wless(y)      $topoDim    ;# Y dimension of the topography
set wless(ifqlen) 25          ;# max packet in ifq
set wless(nn)     $pNum_peers ;# number of nodes
set wless(rp)     OLSR      ;# routing protocol script (dsr or dsdv)
set wless(lm)     "off"     ;# log movement

#Create God to manage wireless ad-hoc network
create-god $wless(nn)
#debug 1;

# set up topography object
set vTopo [new Topography]
$vTopo load_flatgrid $wless(x) $wless(y)

set vChan [new $wless(chan)]
# configure node

$vNs node-config -adhocRouting OLSR \
```

```
 -llType LL \
 -macType Mac/802_11 \
 -propType Propagation/TwoRayGround \
 -ifqType Queue/DropTail/PriQueue \
 -ifqLen 50 \
 -phyType Phy/WirelessPhy \
 -antType Antenna/OmniAntenna \
 -channel $vChan \
 -topoInstance $vTopo \
 -agentTrace ON \
 -routerTrace ON \
 -macTrace ON \
 -movementTrace OFF

 Agent/OLSR set hello_ival_ 3

#--------------------TRACE CONFIGURATION--------------------#
set vTraceFile \
    [open "|awk -v routing_protocol=olsr \
           -f filterMetricas.awk > meu_bt_wireless.tr" w]
$vNs use-newtrace ;#to make a wireless trace
$vNs trace-all $vTraceFile
#----------------P2P TRACE CONFIGURATION--------------------#

# NAME OF TRACE FILE
set vP2P_trace_fName log
append vP2P_trace_fName .NP.
append vP2P_trace_fName $pNum_peers
append vP2P_trace_fName .SF.
append vP2P_trace_fName $pF_size_MB
append vP2P_trace_fName .MT.
append vP2P_trace_fName $pMove_time
append vP2P_trace_fName .CS.
append vP2P_trace_fName [expr round($pChunk_size/1024)]
append vP2P_trace_fName .VL.
append vP2P_trace_fName $pSpeed
append vP2P_trace_fName .clk.
append vP2P_trace_fName [clock seconds]

puts "meu_bt_wireless.tcl:  $vP2P_trace_fName"

set vP2PTraceFile [open $vP2P_trace_fName w]

#------------------- TRACKER CREATION ----------------------#
```

```
# create tracker
set vBt_tracker [new BitTorrentTracker $pF_size $pChunk_size]
$vBt_tracker tracefile $vP2P_trace_fName


# uniform start offset for peers
set vOffset_rng [new RNG]
set vOffset [new RandomVariable/Uniform]
$vOffset set min_ 0
$vOffset set max_ 1 #[BitTorrentApp set choking_interval]
$vOffset use-rng $vOffset_rng

#----------------CREATE NODES(SEEDS AND LEECHES)------------#

# destination coordinates uniformly distributed
set vCoord_rng [new RNG]
$vCoord_rng seed [clock seconds]
set vCoord [new RandomVariable/Uniform]
$vCoord set min_ 10 ;#avoid 0 , because generate bugs
$vCoord set max_ [expr $wless(x) -10]
$vCoord use-rng $vCoord_rng

for {set i 0} {$i < $pNum_peers} {incr i} {

    # make wireless nodes
    set vPeer($i) [$vNs node]

    $vNs initial_node_pos $vPeer($i) 20

    $vPeer($i) set Z_ 0.0


    if {$pMove_time != -1} {

        set x [$vCoord value]
        set y [$vCoord value]

        $vNs at $pMove_time "$vPeer($i) setdest $x $y $pSpeed"

    }

    if {$i < $pNum_seeds} {
        # Create Seeds
```

```
        # 0 for upload capacity, unused in BitTorrentApp
        set vBt_app($vPeer_count) \
            [new BitTorrentApp 1 0 $vBt_tracker $vPeer($i)]

        $vBt_app($vPeer_count) tracefile $vP2P_trace_fName

        # start apps
        $vNs at 0.0 "$vBt_app($vPeer_count) start"

        incr vFinished_peers
    } else {
        # Create Leeches
        # 0 for upload capacity, unused in BitTorrentApp
        set vBt_app($vPeer_count) \
            [new BitTorrentApp 0 0 $vBt_tracker $vPeer($i)]

        $vBt_app($vPeer_count) tracefile  $vP2P_trace_fName

        # start apps
        $vNs at  [$vOffset value] "$vBt_app($vPeer_count) start"
    }


    incr vPeer_count
}
source $scenTclFile

#---------------------- RUN SCHEDULER ----------------------#
$vNs run
```

REFERÊNCIAS BIBLIOGRÁFICAS

[1] G. R. Mateus and A. A. F. Loureiro, *Introdução à Computação Móvel*. Anais da 11a. Escola de Computação, COPPE/Sistemas, NCE/UFRJ, 1998.

[2] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," RFC 2501 (Informational), jan 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2501.txt

[3] Y. C. Hu, S. M. Das, and H. Pucha, "Peer-to-Peer Overlay Abstractions in MANETs," in *Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, 1st ed., J. Wu, Ed. CRC Press, 2005, vol. 1, ch. 47, pp. 857–874.

[4] J. F. Kurose and K. W. Ross, *Redes de Computadores e a Internet - Uma Nova Abordagem*. Addison Wesley, 2004.

[5] E. Royer and C.-K. Toh, "A Review of Current Routing Protocols for ad hoc Mobile Wireless Networks," *IEEE Personal Communications*, vol. 6, no. 2, pp. 46–55, 1999. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=760423

[6] J. Costa-Requena, "Ad-hoc Routing Taxonomy and Resource Discovery," in *Research Seminar on Telecommunications Software*, 2002.

[7] C. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 4, pp. 234–244, 1994.

[8] C. Chiang, H. Wu, W. Liu, and M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel," in *proceedings of IEEE SICON*, vol. 97. Citeseer, 1997, pp. 197–211.

[9] S. Murthy and J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," *Mobile Networks and Applications*, vol. 1, no. 2, pp. 183–197, 1996.

[10] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," RFC 3626, oct 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt

[11] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol," in *Proceedings of the IEEE Multi Topic Conference (INMIC'01)*, December 2001, pp. 62–68. [Online]. Available: http://hipercom.inria.fr/olsr/inmic2001.ps

[12] V. Park and M. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," in *Ieee Infocom*, vol. 3. IEEE, 1997, pp. 1405–1413.

[13] C. Toh, "A Novel Distributed Routing Protocol to Support ad-hoc Mobile Computing," in *Computers and Communications, 1996., Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on*, 1996, pp. 480–486.

[14] R. Dube, C. Rais, K. Wang, and S. Tripathi, "Signal Stability-based Adaptive Routing (SSA) for ad hoc Mobile Networks," *IEEE Personal Communications*, vol. 4, no. 1, pp. 36–45, 1997.

[15] D. Johnson, Y. Hu, and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4," RFC 4728 (Experimental), feb 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4728.txt

[16] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," RFC 3561, jul 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3561.txt

[17] Z. Haas, M. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for ad hoc Networks," IETF Draft.

[18] J. Postel, "Transmission Control Protocol," RFC 793 (Standard), sep 1981, updated by RFC 3168.

[19] I. Chlamtac, M. Conti, and J. J. Liu, "Mobile ad hoc Networking: Imperatives and Challenges," *Ad Hoc Networks*, vol. 1, no. 1, pp. 13–64, July 2003. [Online]. Available: http://dx.doi.org/10.1016/S1570-8705(03)00013-1

[20] Y.-N. Lien and H.-C. Hsiao, "A New TCP Congestion Control Mechanism over Wireless Ad Hoc Networks by Router-Assisted Approach," in *Proceedings of the 27th International Conference on Distributed Computing Systems Workshops(ICDCSW '07)*. Washington, DC, USA: IEEE Computer Society, 2007, p. 84.

[21] W. K. J. Eduardo Cambruzzi, Jean-Marie Farines, "Um Algoritmo Baseado em Peso para Formação e Manutenção de Agrupamentos em Redes Veiculares Ad hoc," in *Anais do 27º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2009.

[22] V. F. S. Mota, T. H. Silva, and J. M. S. Nogueira, "Introduzindo Tolerância a Interrupção em Redes Ad Hoc Móveis para Cenários de Emergência," in *27º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2009, pp. 945–958.

[23] K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes," *IEEEC Communications Surveys & Tutorials*, pp. 72–93, 2005. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1528337

[24] A. Oram, *Peer-to-Peer - Harnessing the Power of Disruptive Technologies*. O'Reilly, 2001.

[25] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A Scalable Content-Addressable Network," in *Proceedings of the 2001 SIGCOMM conference*, vol. 31, no. 4. ACM New York, NY, USA, 2001, pp. 161–172.

[26] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, F. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," *IEEE/ACM Transactions Networking*, vol. 11, no. 1, pp. 17–32, February 2003. [Online]. Available: http://portal.acm.org/citation.cfm?id=638336

[27] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," in *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg (Middleware '01)*. London, UK: Springer-Verlag, 2001, pp. 329–350.

[28] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *Peer-to-Peer Systems*, 2002.

[29] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," in *International workshop on Designing privacy enhancing technologies*. New York, NY, USA: Springer-Verlag New York, Inc., 2001, pp. 46–66.

[30] BerliOS, "The FastTrack Protocol," BerliOS Team, Tech. Rep., 2004.

[31] B. Cohen, "BitTorrent Specification," BitTorrent.org, Tech. Rep., 2006.

[32] S. Rajagopalan and C.-C. Shen, "A Cross-layer Decentralized BitTorrent for Mobile Ad hoc Networks," in *Mobile and Ubiquitous Systems: Networking & Services, 2006 Third Annual International Conference on*, 2006, pp. 1–10. [Online]. Available: http://dx.doi.org/10.1109/MOBIQ.2006.340400

[33] A. Loewenstern, "BitTorrent DHT protocol," Draft, 2006.

[34] V. Srivastava and M. Motani, "Cross-Layer Design: A Survey and the Road Ahead," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 112–119, 2005. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2005.1561928

[35] L. B. Oliveira, I. G. Siqueira, and A. A. F. Loureiro, "Evaluation of Ad Hoc Routing Protocols Under a Peer-to-Peer Application," in *Proceeding of the IEEE Wireless Communications and Networking Conference*, New Orleans, USA, March 2003, pp. 1143–1148.

[36] F. P. Franciscani, M. A. Vasconcelos, R. P. Couto, and A. A. F. Loureiro, "Peer-to-Peer over ad-hoc Networks:(RE) Configuration Algorithms," in *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, 2003, p. 10.

[37] C. Cramer and T. Fuhrmann, "Performance Evaluation of Chord in Mobile ad hoc Networks," in *Proceedings of the 1st MobiShare*. New York, NY, USA: ACM, 2006, pp. 48–53.

[38] D. N. da Hora, D. F. Macedo, J. M. Nogueira, and G. Pujolle, "Otimizando Requisições de Conteúdo em Redes Par-a-Par sobre Redes ad hoc," in *Anais do 24º Simpósio Brasileiro de Redes de Computadores*, 2007.

[39] G. Balazsfalvi and J. Sztrik, "BitTorrent File Sharing in Mobile Ad-hoc Networks," Technical Report, 2006.

[40] M. Bisignano, A. Calvagna, G. D. Modica, and O. Tomarchio, "Expeerience: a JXTA Middleware for Mobile ad-hoc Networks," in *Proceedings of the Third International Conference on Peer-to-Peer Computin*, 2003, pp. 214–215.

[41] M. Conti, E. Gregori, and G. Turi, "Towards Scalable P2P Computing for Mobile Ad Hoc Networks," in *Proceedings of the Second IEEE PERCOMW*, 2004.

[42] H. Pucha, S. M. Das, and Y. C. Hu, "How to Implement DHTs in Mobile Ad Hoc Networks?" in *Proceedings of the 10th ACM Mobicom*, 2004.

[43] ——, "Ekta+: Opportunistic Multiplexing in a Wireless DHT," in *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*, 2006.

[44] T. Zahn and J. Schiller, "MADPastry: A DHT Substrate for Practicably Sized MANETs," in *Proceedings of 5th Workshop on Applications and Services in Wireless Networks (ASWN2005)*, Paris, France, June 2005.

[45] B. Tang, Z. Zhou, A. Kashyap, and T. Chiueh, "An Integrated Approach for P2P File Sharing on Multi-Hop Wireless Networks," in *Proceeding of the 1st IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.(WiMob'2005)*, vol. 3, 2005.

[46] M. Park and W. Kim, "A Gnutella-Based P2P System Using Cross-Layer Design for MANET," *Journal of Electronics, Circuits and Systems (IJECS)*, vol. 1, no. 3, pp. 139–144, 2007.

[47] M. Conti, E. Gregori, and G. Turi, "A Cross-Layer Optimization of Gnutella for Mobile ad hoc Networks," in *Proceedings of the 6th ACM International Symposium on Mobile ad hoc Networking and Computing*. New York, NY, USA: ACM, 2005, pp. 343–354.

[48] A. Klemm, C. Lindemann, and O. Waldhorst, "A Special-Purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks," in *Proceedings of IEEE VTC*, Orlando, FL, USA, Oct. 2003, pp. 2758–2763. [Online]. Available: http://doc.tm.uka.de/2003/waldhorst-vtc2003.pdf

[49] I. Gruber, R. Schollmeier, and W. Kellerer, "Performance Evaluation of the Mobile Peer-to-Peer Protocol," in *Proceedings of the Fourth International Workshop on Global and Peer-to-Peer Computing (GP2PC'2004)*, 2004.

[50] A. Krifa, M. K. Sbai, C. Barakat, and T. Turletti, "BitHoc: A Content Sharing Application for Wireless Ad hoc Networks," in *Proceedings of the IEEE Percom'09*, 2009.

[51] C. Souza and J. M. Nogueira, "Um Estudo do BitTorrent em Redes ad hoc sem Fio Críticas com Localidade Espaço-temporal," in *Anais do 25^{o}Simpósio Brasileiro de Redes de Computadores*, 2008, pp. 329–342.

[52] A. S. Tanenbaum, *Computer networks*. Prentice Hall, 2002.

[53] N. C. Quental and P. A. da S. Gonçalves, "BitTorrent sobre MANETs em Áreas Reduzidas: Mobilidade, Tamanho de Peça e seus Impactos," in *XIV Simpósio Brasileiro de Sistemas Multimídia e Web*, 2008, pp. 101–104.

[54] K. Eger, T. Hossfeld, A. Binzenhofer, and G. Kunzmann, "Efficient Simulation of Large-scale P2P Networks: Packet-level vs. Flow-level Simulations," in *Proceedings of the second workshop on Use of P2P, GRID and agents for the development of content networks*, New York, NY, USA, 2007, pp. 9–16.

[55] L. Paquereau and B. E. Helvik, "A Module-based Wireless Node for NS-2," in *Proceedings of the Workshop on ns-2: the IP network simulator*, New York, NY, USA, 2006, p. 4.

[56] K. Fall and K. Varadhan, "NS Notes and Documentation," The VINT Group, Tech. Rep., feb 2007.