UNIVERSIDADE FEDERAL DA BAHIA
ESCOLA POLITÉCNICA E INSTITUTO DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MECATRÔNICA

VICTOR FRANCO COSTA

CONSENSO FT-CUP EM REDES DESCONHECIDAS:
UM ESTUDO EXPERIMENTAL

Salvador
2009

VICTOR FRANCO COSTA

CONSENSO FT-CUP EM REDES DESCONHECIDAS:
UM ESTUDO EXPERIMENTAL

Dissertação apresentada ao Programa de Pós-graduação em Mecatrônica, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre.

Orientadora: Profa. Dra. Fabíola Gonçalves Pereira Greve

Salvador
2009

Sistema de Bibliotecas - UFBA

Costa, Victor Franco.
Consenso FT-CUP em Redes Desconhecidas: Um Estudo Experimental / Victor Franco Costa. – 2009.
118 f.: il.

Orientadora: Profª. Drª. Fabíola Gonçalves Pereira Greve
Dissertação (mestrado) – Universidade Federal da Bahia, Escola Politécnica e Instituto de Matemática, Salvador, 2009.

1. MANET (Redes móveis auto-organizáveis). 2. Sistemas operacionais distribuídos (Computadores). 3. Tolerância a falha (Computação). 4. Simulação (Computadores). 5. Redes de computação - Protocolos. I. Greve, Fabíola Gonçalves Pereira. II. Universidade Federal da Bahia. Instituto de Matemática. III. Título.

CDD 004.6
CDU 004.72

TERMO DE APROVAÇÃO

Título da Dissertação CONSENSO FT-CUP EM REDES DESCONHECIDAS: UM ESTUDO EXPERIMENTAL

Autor Victor Franco Costa

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre em Mecatrônica, Universidade Federal da Bahia – UFBA, pela seguinte banca examinadora:

Profa. Dra. Fabíola Gonçalves Pereira Greve (Orientadora)
Docteur en Informatique, Université de Rennes I, França
Professora Universidade Federal da Bahia

Prof. Dr. Joni da Silva Fraga (Examinador Externo)
Pós-Doutor em Ciências da Computação, University of California - Irvine, Estados Unidos
Professor Universidade Federal de Santa Catarina

Prof. Dr. George Marconi Lima (Examinador PPGM)
Doutor em Ciências da Computação, University of York, Inglaterra
Professor Universidade Federal da Bahia

Salvador, 06 de novembro de 2009

AGRADECIMENTOS

Agradeço aos meus familiares, principalmente aos meus pais, por todo apoio que foi dado durante minha jornada acadêmica até a realização deste trabalho. Tenho certeza que ainda existe um longo caminho pela frente e tal apoio nunca se esgotará. Agradeço também à Talita Leão por toda a compreensão e companheirismo, aos meus amigos e colegas de trabalho, aos professores, funcionários e alunos do Programa de Pós-Graduação em Mecatrônica, que contribuiram, mesmo que de forma indireta, para que este trabalho pudesse ser construído com sucesso.

Agradeço ao Grupo de Algoritmos e Computação Distribuída (Gaudi) pelos encontros e reuniões que foram realizados e por disponibilizar a infra-estrutura para o ambiente de simulações, através dos computadores existentes no *cluster* do grupo. Meus agradecimentos à FAPESB por me fornecer o apoio financeiro concedido durante a realização desse mestrado. À Talmai Oliveira, por me proporcionar os primeiros contatos com o tema deste trabalho e ser um dos responsáveis por despertar o meu interesse na investigação dos problemas envolvidos nesta pesquisa.

Por fim, guardo o agradecimento mais importante à professora Fabíola Greve pela orientação, paciência e incentivo, ingredientes fundamentais para a realização deste trabalho. A motivação passada pela professora Fabíola, tanto na realização desta dissertação como na escrita dos nossos artigos, foi fundamental durante toda a realização desse mestrado.

RESUMO

O consenso é um problema fundamental em sistemas distribuídos que pode ser utilizado como diretiva básica para o desenvolvimento de aplicações distribuídas tolerantes a faltas. Informalmente, o consenso tem o objetivo de fazer com que todos os processos corretos do sistema decidam por um valor único proposto pelos mesmos. Redes móveis *ad hoc* (ou MANET) são redes dinâmicas constituídas por um conjunto de processos ou nós, onde o canal de comunicação existente entre eles é tipicamente sem fio. Exemplos destas redes são as redes de sensores e sistemas cooperativos compostos de robôs móveis.

O FT-CUP é uma solução para o consenso tolerante a faltas em redes dinâmicas. Nestas redes, o conhecimento prévio dos participantes é uma hipótese muito forte a ser considerada. Por isso, ao contrário do consenso clássico, o FT-CUP não tem esse conhecimento como requisito. De fato, existem poucos trabalhos de consenso que sejam adequados ao contexto de redes dinâmicas. Além disso, poucos deles apresentam uma análise do desempenho dos protocolos propostos. Este trabalho tem como principal objetivo complementar os resultados teóricos alcançados até então para a resolução do FT-CUP e analisar os aspectos práticos da sua realização em redes MANET. Para isso, são propostas implementações para os algoritmos necessários ao FT-CUP e realizadas simulações em cenários realistas. A partir dos resultados obtidos nos experimentos de simulações, chegou-se a um conjunto de parâmetros para os quais é possível a convergência do FT-CUP, definindo-se então as características de uma rede onde o consenso pode ser resolvido de maneira aproximada. Determinou-se também o comportamento do protocolo a partir de diferentes variações de cenários e parâmetros usados nas execuções das simulações.

Palavras-chave: MANET (redes móveis auto-organizáveis), Tolerância a Faltas, Consenso, Simulação

ABSTRACT

Consensus is a fundamental primitive in order to develop distributed fault tolerant applications. Informally, consensus makes all correct processes in the system to decide for a single proposed value. Mobile ad hoc networks (or MANET) are dynamic networks composed by a set of processes or nodes, typically communicating via wireless channels. Some examples are sensor networks and cooperative systems composed of mobile robots.

FT-CUP is a fault tolerant consensus for a dynamic network. In these networks, prior participant knowledge is a strong hypothesis to be considered. Therefore, unlike the classical consensus, FT-CUP does not take into account this requirement. In fact, in a context of dynamic networks, few consensus protocols have been proposed and most of them do not carry an evaluation performance of their protocols. This work complements the theoretical results achieved until now for solving FT-CUP and analyzes the practical aspects of implementing it in a MANET environment. Thus, implementations for the required algorithms are proposed and simulations experiments are conducted in realistic scenarios.

Keywords: MANET (mobile ad hoc networks), Fault Tolerance, Consensus, Simulation

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1

INTRODUÇÃO

A constante evolução tecnológica vem modificando a forma padrão de comunicação entre dispositivos existentes no cotidiano. A praticidade e mobilidade em sistemas distribuídos se tornou fundamental, o que criou a necessidade de evitar o uso de cabos nestes dispositivos. Tarefas que anteriormente eram realizadas através de conexões por cabos, são realizadas através de comunicação sem fio. Aplicações em sistemas distribuídos com requisitos de tolerância a faltas necessitam de um conjunto de protocolos fundamentais para o funcionamento correto. Estes protocolos já foram largamente estudados e definidos para o modelo clássico de sistemas distribuídos. Sendo assim, os protocolos precisam ser revisitados e adaptados ao novo ambiente de computação móvel e sem fio.

Protocolos tolerantes a faltas em sistemas distribuídos podem ser aplicados em diversas áreas, como em sistemas mecatrônicos ou, de forma mais específica, em sistemas compostos por robôs móveis. Sistemas mecatrônicos são compostos pela união de três áreas tecnológicas: mecânica, eletrônica e computação [Ise96]. A computação é a área responsável por prover os algoritmos de controle e processamento das informações envolvidas no sistema. O uso de comunicação sem fio em um sistema formado por um grupo de robôs móveis requer a utilização de mecanismos para garantir que haja cooperação entre os robôs existentes no grupo [GM01]. Este grupo pode, por exemplo, mapear uma determinada área onde ocorreu um acidente em busca de sobreviventes. Para realizar seus objetivos com precisão, é necessário a existência de uma computação confiável que proporcione, entre outras coisas, a coordenação da ação dos mesmos. Os robôs devem se comunicar para concordar em valores obtidos na execução das tarefas, eleger um líder ou delegar determinadas funções para entidades específicas. Para isso, devem existir protocolos tolerantes a faltas, visto que estas são iminentes em ambientes não controlados e em entidades com recursos limitados. Redes de sensores são compostas por nós autônomos

que possuem um meio de comunicação sem fio (*wireless*), formando uma rede entre os sensores. De uma forma geral, são utilizadas para monitorar ambientes de difícil acesso ao ser humano. Estas redes também precisam de protocolos confiáveis para realizar seus objetivos.

Redes de robôs e redes de sensores são consideradas redes móveis auto-organizáveis (*ad-hoc*), ou simplesmente MANET (*mobile ad-hoc network*). Estas são constituídas por nós móveis cuja comunicação dá-se através de canais de comunicação sem fio (*wireless*). Para que um nó possa se comunicar diretamente com outro nó da rede, basta que esse esteja localizado no raio de transmissão do dispositivo de comunicação deste outro nó. Estas redes, ao contrário das redes do modelo clássico de sistemas distribuídos, não possuem infra-estrutura definida, justamente pelas propriedades de mobilidade que possuem. Estas características peculiares das redes MANET dificultam a resolução de problemas fundamentais em sistemas distribuídos.

O consenso é um problema fundamental para realização de computação confiável. Diversos problemas são redutíveis ou equivalentes ao consenso [CT96]. Sendo assim, uma solução para consenso pode ser usada como *middleware* de base para solucionar estes problemas (como difusão atômica (*atomic broadcast*), gestão da filiação ao grupo (*group membership*), eleição, etc.). Informalmente, o consenso tem o objetivo de fazer com que todos os processos corretos do sistema decidam por um valor único entre os valores propostos pelos mesmos. Foi provado que o consenso não pode ser resolvido em ambiente assíncrono na presença de falhas [FLP85]. Sendo assim, abstrações foram propostas para encapsular requisitos temporais e tornar possível a resolução do protocolo. Uma destas abstrações é o detector de falhas, um conjunto de oráculos distribuídos que fornece dicas aos processos sobre quais deles estão falhos [CT96]. Provou-se que a classe de detectores $\diamondsuit\mathcal{S}$ reune as condições de sincronia necessárias e suficientes para a resolução do consenso em redes clássicas, onde o conjunto de participantes é conhecido.

Diversas soluções para o problema do consenso [CT96, BO83, MR01, GR04] utilizam o serviço de difusão confiável para garantir a execução correta do protocolo. Informal-

mente, um protocolo de difusão (*broadcast*) tem como objetivo transmitir uma mensagem para todos os nós da rede. Garantir as propriedades da difusão confiável no ambiente de MANET não é uma tarefa trivial. Por isso, como estudo complementar ao consenso, estuda-se o problema da difusão no contexto das redes MANET. Sendo assim, são descritas algumas das principais soluções para a difusão no ambiente MANET e apresenta-se um trabalho de análise experimental realizado através de simulações, com o objetivo de avaliar o desempenho destes protocolos [dOCG07]. Os resultados tornaram possível concluir que a redundância é fundamental para garantir a confiabilidade do protocolo, ou seja, para garantir uma maior confiabilidade do protocolo, a eficiência deste deve ser sacrificada para garantir a corretude da aplicação. Portanto, protocolos triviais e altamente redundantes, como o *simple flooding* [WC02], podem ser mais adequados em ambientes onde a confiabilidade é fundamental para aplicações em sistemas distribuídos.

Nota-se que, em uma rede clássica, o conhecimento prévio dos participantes é uma hipótese considerada para a resolução do problema do consenso e diversos outros protocolos, como a difusão confiável. Neste contexto, esta hipótese é considerada realista, visto que a estrutura estática da rede permite facilmente este conhecimento prévio. Em uma rede MANET, a mobilidade, a falta de estrutura da rede e as entradas e saídas deliberadas, fazem com que a dinamicidade seja alta, tornando o conhecimento inicial sobre os participantes da rede uma hipótese muito forte e não realista. Sendo assim, considera-se neste trabalho que, no início da execução em uma MANET, um processo não conhece o conjunto de processos que compõem a rede, assim como desconhece a quantidade de processos existentes. Em soluções para o consenso clássico [CT96, MR99], estas informações são essenciais, logo tais soluções não são adequadas ao modelo de redes MANET.

O CUP (*consensus with unknown participants*), definido por [CSS04], é uma variação ao problema do consenso, levando-se em consideração o desconhecimento da rede. Portanto, o CUP é uma proposta para o consenso adequada ao modelo de MANET. A solução para esse problema contempla a definição de uma abstração, os *detectores de participação*. Estes são considerados oráculos distribuídos, associados a cada processo, que retornam um conhecimento parcial dos participantes existentes na rede. Diversas classes para estes

detectores foram propostas. Cada classe estabelece um grafo de conectividade entre os participantes do sistema, definido a partir da relação de conhecimento estabelecida pelo detector. A classe minimal que possibilita resolver o CUP num contexto sem falhas de processos é o OSR (*One Sink Reducibility*) [CSS04].

Posteriormente, a proposta do CUP foi estendida para o FT-CUP (*fault tolerant CUP*), que é o CUP tolerante a faltas [CSS05]. Na solução apresentada são identificados os requisitos de sincronia necessários para a resolução do FT-CUP, considerando-se os requisitos mínimos de conectividade relativos ao detector de participação (os mesmos encontrados em [CSS04], ou seja o OSR). Tal requisito minimal de sincronia resume-se ao detector de falhas perfeito (classe $\mathcal{P}$), uma classe de detector que só pode ser implementada em ambiente síncrono [LFA04].

Greve e Tixeuil apresentam uma proposta alternativa para solucionar o FT-CUP [GT07]. Nesta, os autores consideram os requisitos minimais de sincronia, já identificados para a resolução do consenso no modelo clássico, e buscam os requisitos mínimos para a conectividade do grafo de conhecimento que possibilitam resolver o problema. Como demonstrado em [CT96], o requisito mínimo de sincronia resume-se ao detector de falhas da classe $\diamondsuit\mathcal{S}$. Esses detectores possuem propriedades não-confiáveis e podem ser implementados em sistemas dinâmicos, como MANET, redes de sensores e P2P (*peer-to-peer*). Por isso, esta solução para o FT-CUP é passível de implementação em um modelo assíncrono, estendido com requisitos temporais mais fracos. Além disso, resolve também a versão uniforme para o consenso. Nesta versão, todos os processos que terminam, decidem pelo mesmo valor, sejam estes corretos ou não.

Este trabalho destina-se a avaliar, em termos práticos, o desempenho da solução de Greve e Tixeuil [GT07] para resolver o FT-CUP em redes móveis *ad-hoc*. Para o estudo de protocolos em redes MANET, a análise experimental através de simulações é uma alternativa bastante utilizada, visto que não é fácil obter um ambiente real com as características desejáveis para os testes. Esse estudo experimental tem importantância fundamental como ponto de partida para a avaliação de protocolos, de forma a determi-

nar o seu comportamento e apontar as direções para a otimização e implementação em ambientes reais. Até onde foi pesquisado, não existem trabalhos que avaliam o consenso em redes dinâmicas e que consideram a hipótese do desconhecimento dos participantes da rede. Sendo assim, os algoritmos do FT-CUP foram implementados e simulados em um ambiente previamente selecionado. Com os resultados obtidos, foi possível identificar o comportamento do FT-CUP em redes MANET modeladas através de parâmetros realistas, o que proporcionou verificar a viabilidade de resolução do FT-CUP com tais parâmetros. Um aspecto crítico da abordagem seguida por todos os algoritmos para resolução do CUP e FT-CUP é o fato de que, se os detectores de participação não satisfazem as propriedades identificadas, a corretude do consenso não pode ser garantida. Logo, uma questão fundamental, em qualquer uma das soluções, é a implementação dos detectores de participação. Ocorre que não existem trabalhos que propõem implementações que garantam as propriedades para os detectores de participação. Como o intuito é avaliar as possibilidades de convergência do consenso, mesmo diante de uma rede MANET com uma topologia arbitrária de conhecimento, propõe-se a utilização de implementações simples de detectores de participação, que originam grafos de conhecimento arbitrários e que, não necessariamente, satisfazem as propriedades estabelecidas para a resolução do consenso.

A partir dos resultados obtidos, foi possível identificar um conjunto de valores que tornam possível a convergência do consenso. Desta forma, conseguiu-se identificar as características práticas da rede, onde o consenso pode ser executado corretamente, mesmo considerando-se grafos arbitrários de conhecimento. Foi possível identificar ainda as características do algoritmo que impossibilitam a convergência do consenso em casos específicos.

1.1 ESTRUTURA DA DISSERTAÇÃO

O Capítulo 2 descreve as características fundamentais das redes MANET, ressaltando os seus canais de comunicação e o modelo de simulações para protocolos utilizado em

abiente de MANET. O Capítulo 3 introduz o problema de difusão de mensagens em redes MANET, com a descrição de alguns dos seus principais protocolos e finaliza apresentando uma comparação entre eles, realizada através da ferramenta de simulação de protocolos Network Simulator (NS-2) [ns209]. O Capítulo 4 apresenta a definição clássica do problema do consenso em tolerância a faltas. Mostra ainda os seus resultados de impossibilidade no modelo clássico e as alternativas para sua resolução. Posteriormente, o problema do consenso é abordado no contexto de redes dinâmicas e as soluções apresentadas consideram as características destas redes. O Capítulo 5 complementa o Capítulo 4 com a apresentação do consenso com participantes desconhecidos, um tipo de consenso típico para o ambiente de redes dinâmicas. São descritas as formas de resolução e os algoritmos encontrados para a sua implementação. Apresenta-se ainda neste capítulo o modelo de implementação do FT-CUP proposto neste trabalho. O Capítulo 6 apresenta os resultados preliminares obtidos com as simulações do FT-CUP no ambiente de MANET, utilizando a implementação do FT-CUP proposta no Capítulo 5 e com a proposta de um modelo de simulações adequado para o ambiente utilizado. O Capítulo 7 apresenta os resultados definitivos obtidos com a avaliação de desempenho do FT-CUP em um modelo baseado no modelo proposto no capítulo 6, com a inclusão de melhorias tornando-o mais realista e adequado ao ambiente de MANET. O Capítulo 8 finaliza este trabalho com a apresentação das considerações finais.

1.2 PUBLICAÇÕES

Através das pesquisas realizadas na produção deste trabalho e na colaboração com trabalhos relacionados a este, os seguintes artigos foram desenvolvidos e publicados:

i) COSTA, Victor Franco ; GREVE, F. G. P. ; TIXEUIL, Sébastien. Consenso FT-CUP em Redes MANETs: Uma Abordagem Prática. Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2009, Recife, Pernambuco. Porto Alegre - RS : SBC - Sociedade Brasileira de Computação.

ii) COSTA, Victor Franco ; GREVE, F. G. P. . Implementing Fault-Tolerant Consensus Over Unknown Networks. In: 7th International Information and Telecommunication Technologies Symposium (I2TS), 2008, Foz do Iguaçu, Paraná.

iii) COSTA, Victor Franco ; GREVE, F. G. P. . Aspectos Práticos da Realização do Consenso FT-CUP em Redes Móveis Ad-Hoc. In: VIII Workshop de Teste e Tolerância a Falhas (WTF), 2007, Belém, Pará. Workshop de Testes e Tolerância a Falhas, em conjunto com Simpósio Brasileiro de Redes de Computadores. Porto Alegre - RS : SBC - Sociedade Brasileira de Computação, 2007. v. 1. p. 1-14.

iv) OLIVEIRA, Talmai Brandão de ; COSTA, V. F. ; GREVE, F. G. P. . On the Behavior of Broadcasting Protocols for MANETs Under Omission Faults Scenarios. Third Latin-American Symposium on Dependable Computing (LADC 2007), p. 142-159, 26-28 September, Morelia, Mexico. Also in Lecture Notes in Computer Science, v. 4746, LNCS, Springer-Verlag, 2007.

CAPÍTULO 2

REDES MÓVEIS AD-HOC

Este capítulo introduz o conceito de redes móveis *ad hoc* (MANET). Será definido o modelo dessas redes, com destaque para as suas peculiaridades em relação ao modelo clássico de redes de computadores. Realiza-se ainda uma análise da técnica de simulações para avaliação de protocolos, com ênfase nas ferramentas disponíveis e nas boas práticas de simulações, a fim de proporcionar confiabilidade aos resultados obtidos. Por fim, serão abordadas algumas aplicações reais existentes para redes MANET, onde algoritmos distribuídos tolerantes a faltas podem ser aplicados.

2.1 CARACTERIZAÇÃO

Redes móveis auto-organizáveis *ad hoc*, ou simplesmente MANET, são redes constituídas por um conjunto de processos ou nós, representados por Π, dispostos em um cenário, onde o canal de comunicação existente entre eles é tipicamente sem fio (*wireless*). Com isso, um nó pode se comunicar com outro se este nó se localiza dentro do seu raio de transmissão, o que caracteriza a relação de vizinhança existente entre eles. Por exemplo, através da Figura 2.1(a), pode-se dizer que o nó a tem como conjunto de vizinhos imediatos (um nível) os nós $\{b, c\}$ e como vizinhos de dois níveis (*2-hop* o conjunto $\{d, e\}$.

De acordo com esta regra, a união da informação de conectividade faz com que uma MANET possa ser representada por um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, sendo $\mathcal{V}$ o conjunto vértices ou nós da rede e $\mathcal{E}$ o conjunto de arestas entre os nós, que indica a conectividade entre eles. Uma aresta (x, y) conecta diretamente dois nós x, y neste grafo e representa a relação de vizinhança existente entre x e y. Por exemplo, na Figura 2.1(a), a aresta que conecta os nós a e b indica que b está no alcance de transmissão de a e estes processos podem se

comunicar.

Por definição, um grafo é caracterizado como conexo se existe pelo menos um caminho entre quaisquer nós existentes, implicando na prática em uma rede sem particionamento. Um grafo completamente conexo é aquele que possui uma aresta entre quaisquer pares de vértices, o que possibilita a comunicação direta entre quaisquer pares de nós na rede. É importante ressaltar que, geralmente, o grafo formado pelos nós de uma MANET não é completamente conexo, embora a existência do grafo conexo é essencial para a execução correta de determinados protocolos. A Figura 2.1(a) representa um exemplo de uma rede MANET representada por um grafo, sendo que este é caracterizado como conexo mas não completamente conexo. Porém, uma das causas das impossibilidades de soluções para protocolos nesse ambiente e das execuções incorretas dos protocolos existentes é o particionamento da rede.

Contudo, uma característica das redes sem fio é proporcionar frequentemente conexões assimétricas [KM07, KNG+04]. Por exemplo, na Figura 2.1(b), o nó e está no alcance de transmissão de b mas b não está no alcance de e. Neste caso, a conexão é unidirecional, ou seja, b pode enviar mensagens para e mas a operação inversa não é válida. Portanto, um modelo mais adequado para redes MANET considera o grafo direcionado $\mathcal{G}_{di} = (\mathcal{V}, \mathcal{E})$, como ilustrado na Figura 2.1(b). Neste grafo, as arestas do conjunto $\mathcal{E}$ são direcionadas.

Uma outra característica das MANET refere-se à mobilidade dos nós. Um nó pode mover-se livremente pelo cenário, o que causa alterações nas suas relações de vizinhança ou até mesmo entradas e saídas de nós na rede. Consequentemente, a mobilidade resulta em alterações na topologia da rede, um comportamento exemplificado através Figura 2.1(c). Nesta, o nó c move-se para fora do alcance de transmissão de a e b e a aresta existente entre c e os nós a e b é eliminada.

Diferente do modelo de redes dinâmicas, no modelo clássico de sistemas distribuídos o conjunto de processos Π e a sua cardinalidade é conhecido por cada processo. Além disso, geralmente o grafo de conectividade é completamente conexo, ou seja, quaisquer processos de Π podem se comunicar diretamente. Pelas características apresentadas por

uma MANET, considera-se que o conhecimento prévio dos participantes da rede é uma hipótese muito forte. Sendo assim, um modelo mais realista considera que, inicialmente, cada nó desconhece o conjunto Π e até mesmo a quantidade de nós existente $n = |\Pi|$ [CSS04, CSS05, GT07].

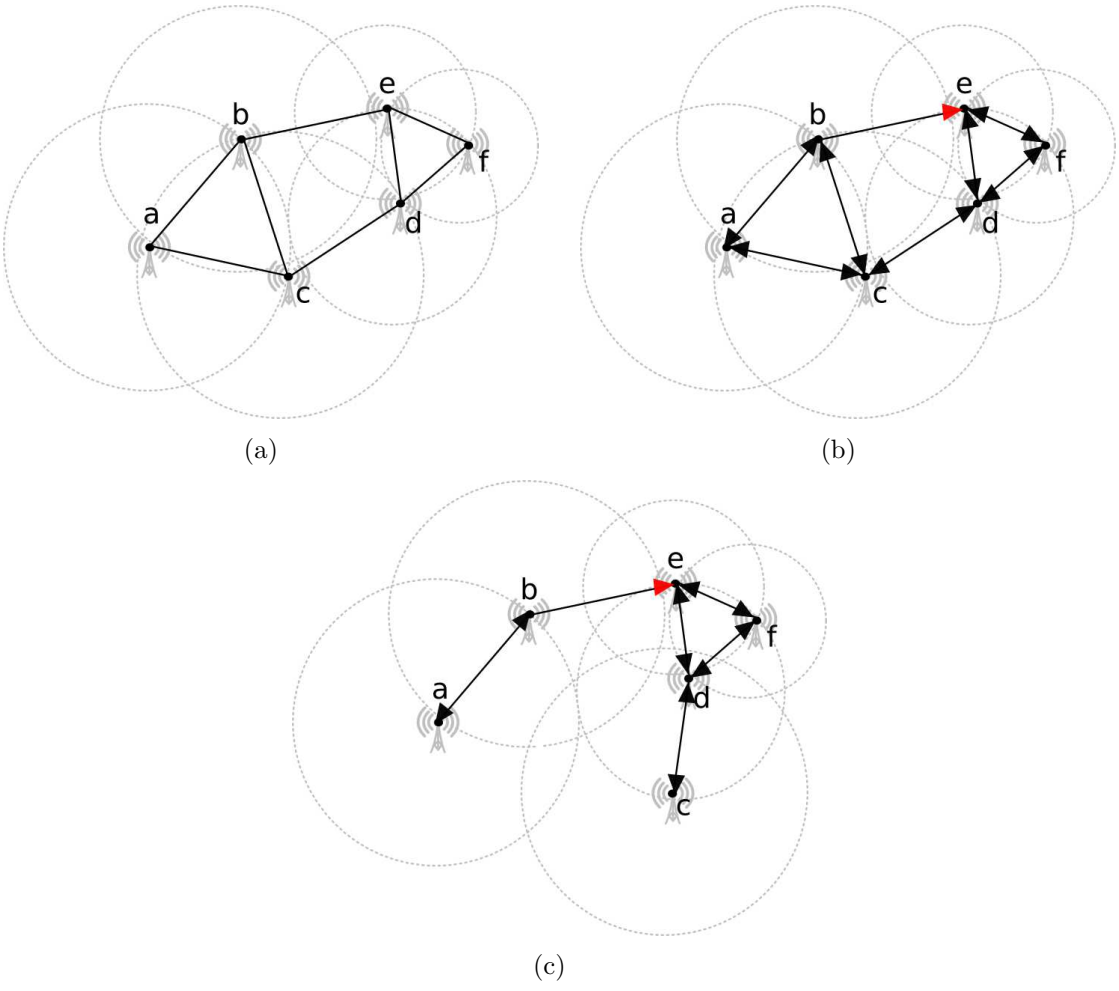


Figura 2.1. Exemplo de MANET. (a) Configuração inicial de uma rede MANET e (c) alteração da configuração após mobilidade

Uma das características desejáveis para os dispositivos que integram redes MANET é a autonomia. Geralmente, os nós que compõem a rede possuem recursos limitados de energia, memória e capacidade de processamento. Além disso, o dispositivo de comunicação possui alcance limitado e o seu uso deve ser otimizado para minimizar o gasto energético do dispositivo. Estas limitações interferem em toda a pilha de protocolos de comunicação e, consequentemente, nos protocolos de alto nível que fornecem os requisitos

de tolerância a faltas para as aplicações. Esses protocolos devem buscar a eficiência, do ponto de vista da otimização do uso dos recursos disponíveis [CCL03, FJL00, RR02].

2.2 CANAIS DE COMUNICAÇÃO

A comunicação em MANET é realizada através de canais sem fio. O meio de comunicação sem fio está sujeito a perdas de mensagens com mais frequência, em relação a comunicação convencional através de cabos. Estas perdas são causadas por, dentre outras coisas, interferências, obstáculos ou ruídos no sinal do canal de comunicação. A interferência inclusive pode ser causada até pelo alto tráfego de mensagens na rede, com os sinais dos nós vizinhos se sobrepondo e prejudicando as transmissões [TNCS02]. Tais perdas são consideradas complicadores para o desenvolvimento de protocolos baseados em troca de mensagens. Pode-se enumerar as características principais dos canais de comunicação em MANET da seguinte forma [BKP03]:

i) Baixa taxa de transmissão;

ii) Alta incidência de erros na troca de mensagens;

iii) Desconexões frequentes.

O padrão de comunicação mais utilizado nas redes sem fio é definido por IEEE 802.11 [CWKS97]. Neste padrão, o acesso ao meio de comunicação é realizado através do método *Carrier Sense Multiple Access with Colision Avoidance* (CSMA/CA). De acordo com este método, antes de iniciar o envio de uma mensagem o nó escuta o canal para identificar se existe uma outra mensagem sendo enviada. Caso não exista transmissão, o nó pode enviar a sua mensagem. Caso contrário, o nó aguarda até que o canal esteja livre e espera ainda por mais um tempo aleatório para poder enviar a sua mensagem. Tal tempo aleatório é útil para diminuir as chances de envios simultâneos.

2.3 TOLERÂNCIA A FALTAS

Assim como no modelo clássico de sistemas distribuídos, a confiança no funcionamento é fundamental para serviços distribuídos em MANET. Uma forma de buscar tal requisito é através da implementação de técnicas de tolerância a faltas. Porém, estas não são técnicas triviais e podem elevar bastante a complexidade dos algoritmos, principalmente no modelo das redes MANET, onde existem novos desafios que aumentam a complexidade.

Um aspecto fundamental para implementar tolerância a faltas é o estabelecimento de um modelo adequado. Este modelo deve refletir o ambiente da aplicação real em que as soluções propostas serão utilizadas. Definições importantes relacionadas ao modelo do sistema abrangem aspectos temporais e de falhas.

2.3.1 Modelo de Falhas

Em MANET, o problema das falhas é bastante iminente já que, como dito anteriormente, a comunicação sem fio é naturalmente sujeita a falhas. A limitação de recursos é uma outra característica que pode levar a falhas. Sendo assim, considera-se no modelo que tanto os processos como os canais de comunicação podem falhar.

Nas falhas de processos, existem três modelos que se destacam dentro do contexto de tolerância a faltas: falhas por parada (*crash*), falhas por omissão e as falhas bizantinas [VR00, ALRL04]. O modelo de falhas por parada é largamente utilizado na definição do modelo para a resolução de problemas em tolerância a faltas. Nele, o processo simplesmente para de funcionar, de forma a deixar de participar permanentemente do sistema envolvido. O modelo de falha por omissão sugere que o nó falha por parada ou continua funcionando normalmente mas deixa de receber ou enviar determinadas mensagens. Um exemplo de utilidade desse modelo é simular a instabilidade nas conexões característica de aplicações reais no ambiente de MANET. Nas falhas bizantinas, os processos faltosos podem executar ações arbitrárias sobre o sistema [LSP82]. Enquanto os modelos de falhas por parada e por omissão podem ser caracterizados como benignos, as falhas bizantinas

são consideradas malignas, pois os processos podem agir de forma a prejudicar deliberadamente o comportamento do sistema. O modelo de falhas bizantinas abrange as falhas por parada e por omissão.

Os canais de comunicação podem ser caracterizados como confiáveis ou não-confiáveis. Os canais confiáveis não perdem, duplicam ou modificam as mensagens. Nos canais não-confiáveis, não se pode garantir tais propriedades pois existe a incidência de falhas. Em MANET, um modelo de falhas mais adequado considera canais não-confiáveis. Porém, na existência de canal não-confiável, é possível implementar facilmente canais confiáveis. Basta utilizar um protocolo que forneça garantias em relação a entrega das mensagens [VR00].

2.3.2 Modelo de Sincronia

Um modelo de sincronia, também chamado de temporal, estabelece propriedades de sincronia relacionadas aos processos e canais de comunicação utilizados. Ele pode ser caracterizado como síncrono, assíncrono ou parcialmente síncrono [DLS88]. O modelo síncrono possui limites superiores válidos e conhecidos para tempo de processamento Φ e de transmissão de mensagens Δ entre processos da rede. No modelo assíncrono, os limites Φ e Δ não existem.

O modelo parcialmente síncrono é um modelo intermediário que possui algumas especificações. Dwork *et al.* [DLS88] propõe duas variações. Em uma delas, considera-se que se conhecem os limites Φ e Δ mas eles só serão válidos após um tempo de estabilização T. Uma outra proposta indica que os limites de tempo Φ e Δ não são conhecidos mas existem. Pode-se considerar ainda uma combinação desses dois modelos anteriores e criar um modelo mais fraco de sincronia parcial, onde os limites Φ e Δ não são conhecidos mas existem e serão válidos apenas após um tempo de estabilização T [CT96].

Toda solução de um problema para o modelo assíncrono funciona também no modelo síncrono. Porém, alguns problemas no modelo assíncrono não possuem solução.

Uma forma de encapsular características temporais para resolver problemas no modelo assíncrono é usar abstrações, como a detecção de falhas ou eleição de líder [CT96, MR01]. Tais abstrações serão detalhadas no Capítulo 4.

2.4 SIMULAÇÕES DE PROTOCOLOS EM MANET

Um modo importante para analisar o comportamento de protocolos em sistemas distribuídos é o uso de simulações. Para o estudo de protocolos em redes MANET, esta é uma alternativa bastante utilizada, visto que não é fácil obter um ambiente real com as características desejáveis para os testes. É importante destacar que a técnica de simulação não deve substituir totalmente os experimentos em ambientes reais [KM07, HBG06]. Algumas considerações geralmente realizadas nos modelos de simulações não correspondem as características reais, o que pode levar a resultados inconsistentes. Uma forma de amenizar estes problemas é investigar os modelos e características obtidas em experimentos reais e utilizar nos modelos das simulações [KM07, KNG+04]. A técnica de simulação é muito importante e deve ser utilizada como ponto de partida para as avaliações dos protocolos, para caracterizar o seu funcionamento e apontar as direções para a otimização e implementação em ambientes reais.

Existem diversos simuladores para protocolos de comunicação voltados as redes MANET disponíveis. Uma ferramenta de simulação bastante utilizada é o Network Simulator (NS-2) [ns209]. O NS-2 é direcionado para a área da pesquisa acadêmica, mais especificamente aos temas que envolvem redes de computadores. É muito usado na comunidade científica por ser uma ferramenta robusta, de código aberto e que prevê modificações ou inserções de código, permitindo assim a implementação de diversos protocolos em variados ambientes. Possui um ótimo suporte à simulação de protocolos no ambiente de MANET. Além disto, como muitos trabalhos foram desenvolvidos com implementação de protocolos no NS-2, pode-se utilizar esses protocolos como base para o desenvolvimento de outros protocolos e facilita a realização de comparações entre protocolos de uma mesma família. O NS-2 foi desenvolvido para a plataforma Linux e é um simulador orientado a

eventos discretos, onde a implementação dos protocolos é realizada na linguagem C++. Para a descrição das simulações e configuração dos parâmetros, utilizam-se scripts TCL. A inserção de novos protocolos nesta ferramenta não é algo trivial. Devido à arquitetura do simulador não ser modular, é relativamente complexo estender as funcionalidades existentes no NS-2 e adicionar novos protocolos. Inserir um protocolo no NS-2 requer a modificação de diversos pontos do seu código. Um outro ponto negativo é que, comparado a outros simuladores, essa ferramenta obtém um fraco desempenho em relação ao uso dos recursos computacionais, o que prejudica a escalabilidade dos testes, ou seja, o custo computacional aumenta muito para grandes quantidades de nós na rede [HBG06].

A ferramenta OMNeT++ [Var01, omn09, VH08] é um ambiente de simulações modular orientado a eventos discretos. O OMNeT++ é de uso livre para fins acadêmicos e possui código disponibilizado publicamente. A sua arquitetura é aberta, modular e baseada em componentes e por isso considerado facilmente extensível. O uso de herança, recurso inerente a programação orientada a objetos, facilita a criação de novos protocolos. O OMNeT++ foi desenvolvido para simulação de redes clássicas, baseadas na comunicação com fio. Porém, existe a extensão *Mobility Framework* (MF) [MF09, DSH+03] que possibilita a simulação de redes sem fio e com mobilidade, assim como no modelo das MANETs. Esta extensão fornece um conjunto de protocolos pertencentes ao padrão IEEE 802.11 que permitem o uso da camada física de forma similar as aplicações reais existentes no ambiente de redes sem fio. Além disso, diversos padrões de mobilidade são oferecidos para uso nos testes. Porém, o suporte a redes MANET através do MF não é completo como o existente no NS-2. Nele existem apenas os protocolos básicos para a comunicação IEEE 802.11. Já no NS-2, estão disponíveis diversos protocolos de difusão e roteamento específicos para MANET [HBG06]. Entretanto, o OMNeT++ tem melhor desempenho nas execuções das simulações e fornece um conjunto de ferramentas gráficas que auxiliam o desenvolvedor na criação, validação e análise dos resultados dos protocolos [VH08, XSH08]. Além disso, testes realizados com um mesmo protocolo no OMNeT++ e NS-2 obtiveram resultados similares [XSH08], mostrando que o OMNeT++ é um ambiente tão confiável quanto o NS-2.

Neste trabalho, optou-se por usar o OMNeT++ combinado com a sua extensão *Mobility Framework* para a simulação de cenários de redes MANET. Na Seção 5.3 foi proposta uma implementação para o protocolo de consenso *Fault Tolerant Consensus with Unkwnown Network* (FT-CUP) [GT07], sendo que essa proposta foi aplicada ao OMNeT++ e simulada em diversos cenários, com os resultados apresentados no Capítulo 7.

2.4.1 Padrão de Mobilidade

O padrão de mobilidade define as propriedades e características do movimento realizado pelos nós no cenário do ambiente simulado. Existem diversos padrões de mobilidade utilizados em simulações de redes MANET, que visam uma aproximação das características de determinados ambientes do mundo real. Em alguns padrões, o movimento de cada nó é determinado de forma independente dos demais. Em outros padrões, o movimento é realizado em grupo, ou seja, o movimento de um nó depende dos outros nós. Para a realização de simulações realistas com redes móveis, é necessário basear os testes em padrões de mobilidade adequados, uma vez que a escolha do padrão de mobilidade para as simulações pode ser determinante no comportamento de um determinado protocolo [CBD02].

Um padrão bastante utilizado em simulações de redes MANET é o *Random Waypoint* [CBD02]. Neste padrão os nós movem-se livremente pelo cenário, de acordo com a velocidade previamente definida. Existe também um tempo de pausa definido para os nós. Um nó aguarda por este tempo de pausa e depois se movimenta para a nova posição definida aleatoriamente. Este padrão tem características de mudar bruscamente a direção do movimento de um nó.

Um outro padrão é o *Gauss-Markov* [CBD02]. Neste, são atribuídos inicialmente para cada nó uma direção e velocidade. De tempos em tempos, estes valores são atualizados segundo uma equação baseada em variáveis aleatórias de uma distribuição Gaussiana, que é regulada por um parâmetro de entrada α. Neste padrão, as mudanças de direções são mais suaves, assim como a variação da velocidade.

O *Reference Point Group Mobility* (RPGM) é um padrão que considera movimentos em grupo [CBD02]. Nesse padrão, existe um movimento aleatório dos nós em torno do centro do seu grupo e o movimento aleatório de todo o grupo. Esse é um padrão útil para modelar aplicações onde a formação de grupos de nós na rede se torna evidente. Neste caso, avaliar o comportamento de um protocolo quando existem grupos bem definidos é mais importante do que utilizar um movimento totalmente aleatório que pode ou não ocasionar na formação de grupos.

2.4.2 Modelo de Simulações

Além do padrão de mobilidade escolhido para as simulações, existem outras questões relevantes que devem ser definidas para o modelo de simulações. Criar um modelo de simulação para analisar o comportamento de um protocolo envolve definir:

i) Meio de comunicação;

ii) Cenários;

iii) Parâmetros específicos dos protocolos;

iv) Métricas para análise dos resultados;

v) Método de análise dos resultados.

O funcionamento adequado destes itens é fundamental para a credibilidade dos resultados dos testes. Boas práticas nas definições dos mesmos devem ser consideradas [HBG06, KCC05]. Na definição do meio de comunicação, é importante obter um modelo de propagação do sinal de transmissão coerente com o ambiente real da aplicação testada. As camadas físicas, como a definida no padrão IEEE 802.11, devem ser fieis as implementações reais. Além disso, considerar que os canais de comunicação proporcionam sempre conexões bidirecionais não é uma hipótese realista [KNG+04]. Isso ocorre devido à existência de obstáculos, ruído nos canais de comunicação que atrapalham a propagação do sinal e diferentes alcances de transmissão dos dispositivos.

Os cenários escolhidos devem possuir parâmetros coerentes. Os parâmetros básicos que definem um cenário são: quantidade de nós (n), área do cenário ($w \times h$, onde w é a largura e h a altura), e o raio de alcance de transmissão de cada nó (r). Para garantir a coerência na definição dos parâmetros, aspectos como a densidade de nós na rede e quantidade média de vizinhos por nó devem ser considerados. Define-se a densidade de nós (ND) na rede pela relação entre quantidade de nós e área. Define-se a quantidade média de vizinhos (NC) através da densidade e a área de alcance de transmissão. Tais relações são formalizadas pela seguinte equação:

$$\text{(densidade dos nós) } ND = \frac{n}{w \times h} \qquad \text{(média de vizinhos) } NC = (\pi \times r^2) \times ND \quad (2.1)$$

A definição de cenários aborda ainda os parâmetros específicos do protocolo que se deseja simular. De acordo com o protocolo, tais parâmetros podem depender dos parâmetros básicos definidos anteriormente.

As métricas usadas nas simulações definem as características que se deseja avaliar em um determinado protocolo. Uma definição criteriosa é fundamental para garantir que os objetivos das simulações sejam alcançados. Não existe um conjunto de métricas padrão utilizadas na avaliação de protocolos em MANET. Entretanto, diversos trabalhos de simulações, inclusive no modelo clássico de sistemas distribuídos, usam métricas similares. Um exemplo é a latência [dOCG07, Vol05], métrica que indica a eficiência do protocolo através do tempo gasto na sua execução. Em contrapartida, protocolos específicos necessitam de métricas adequadas ao seu funcionamento, com o objetivo de mostrar que suas definições são válidas em um determinado modelo.

Uma vez que as simulações foram executadas, é necessário aplicar um método estatístico para a análise dos dados coletados através das métricas. Para possibilitar isso, cada conjunto de testes deve ser executado repetidas vezes. Sendo assim, os resultados podem ser apresentados como uma média dessas repetições. Porém, o uso apenas da média não é adequado, é preciso representar também o grau de confiança nessa estimativa. Por isso, deve-se utilizar intervalos para representar o grau de confiança que se pode

atribuir para uma determinada estimativa [KCC05, San99, Urb03]. Quanto menor esse intervalo, mais precisa é a estimativa. Para o cálculo do intervalo, deve-se definir o coeficiente de confiança, geralmente definido por uma porcentagem. Supondo um conjunto de simulações definida pela execução de um protocolo em um determinado cenário, sendo calculada uma estimativa de média de uma métrica específica. Aplicar um intervalo de confiança com coeficiente 95% indica, estatísticamente, que, em 95% das execuções desse conjunto a estimativa estará contida no intervalo de confiança.

Uma aproximação do intervalo de confiança com coeficiente de 95% pode ser calculado da seguinte forma [Urb03]:

$$\bar{x} \pm \left(1.96 \times \frac{s}{\sqrt{n}}\right) \tag{2.2}$$

Sendo s o desvio padrão de uma média $\bar{x}$ e n a quantidade de repetições usadas no cálculo da média.

2.5 APLICAÇÕES

Um exemplo de aplicação de redes MANET são as redes de sensores sem fio [KM07]. Estas redes são úteis em situações bastante específicas, geralmente utilizadas para monitorar ambientes de difícil acesso ao ser humano. Os nós destas geralmente não se movem espontaneamente. Mesmo assim, questões de mobilidade devem ser consideradas pois fenômenos externos podem causar o movimento dos nós da rede, uma vez que o ambiente em que estes se encontram é imprevisível [ASSC05]. As falhas também são frequentes nestas redes, devido a escassez de recursos, principalmente de energia, além do próprio ambiente que a rede se encontra poder levar a falha do nó.

O modelo de MANET pode ser utilizado também em aplicações específicas da robótica, como sistemas cooperativos formados por robôs móveis. Muitas vezes, é necessária a existência de um algoritmo para a resolução de problemas de coordenação, como por exemplo a alocação eficiente de tarefas entre os robôs que formam a rede [GM01]. Nesse caso, um protocolo de consenso adequado ao modelo da aplicação pode ser utilizado como

diretiva básica de implementação do algoritmo de coordenação.

Um outro exemplo de aplicação para MANET são as redes *mesh* [KM07]. Redes *mesh* são compostas por nós munidos de dispositivos de comunicação sem fio, que formam um grafo de conectividade, semelhante aos exibidos na figura 2.1. Nestas redes, um dos nós pode fornecer algum tipo de serviço para os demais, que passam a funcionar como clientes deste. Estas redes podem ser utilizadas para compartilhar o acesso a *Internet* disponível em um nó para os demais nós da rede *mesh*. Um exemplo desse uso pode ser verificado no projeto *One Laptop per Child* (OLPC) [olp09].

Um experimento denominado *Mobile Ad-hoc Network Interoperability And Cooperation* (MANIAC) *Challenge* [SHT+08] foi realizado com o objetivo de evidenciar as características reais de uma rede MANET. A rede do experimento foi formada por um grupo de pessoas munidas de 16 *laptops*. Através disto, foi possível observar a existência frequente de modificações na topologia, causada por entradas e saídas de nós na rede e por mobilidade. Apesar disso, observou-se que a rede manteve altos níveis de conectividade. Observou-se também que as rotas calculadas pelo protocolo de roteamento utilizado foram frequentemente assimétricas. O experimento obteve um resultado ruim de desempenho, apenas um quarto das mensagens foram entregues corretamente aos seus destinos.

2.6 CONCLUSÃO

Este capítulo apresentou uma definição do modelo de redes móveis ad-hoc (MANET). Foram listadas características específicas do ambiente e que diferem do modelo clássico de sistemas distribuídos. Os canais de comunicação encontrados nesse modelo foram definidos e caracterizados, assim como uma breve definição do padrão IEEE 802.11. A questão da tolerância a faltas em MANET foi abordada, com ênfase na necessidade de adequação dos requisitos dos algoritmos ao modelo real de redes MANET. Foram mostrados modelos de falhas de processos e canais, assim como modelos temporais utilizados.

Uma seção de simulações enfatizou o uso dessa técnica para avaliação de protocolos em redes MANET. Duas ferramentas foram apresentadas, assim como uma pequena comparação entre elas. Foram definidos alguns padrões de mobilidade tipicamente utilizados em MANET, considerando-se também outras questões referentes ao modelo de simulações, fundamentais para garantir a credibilidade na realização dos experimentos.

CAPÍTULO 3

DIFUSÃO (BROADCAST) EM MANET E AVALIAÇÃO DE DESEMPENHO

Este capítulo tem como objetivo definir o problema da difusão (*broadcast*) e abordar algumas das suas soluções específicas para o ambiente de redes móveis *ad hoc* (MANET), com ênfase nos seus princípios de funcionamento para resolver o problema. Garantir as propriedades da difusão confiável não é uma tarefa fácil no ambiente de MANET. Um dos requisitos desses protocolos nesse ambiente é o desconhecimento da rede, os nós não conhecem inicialmente os participantes da rede e nem mesmo a quantidade de nós existentes. Por isso, os protocolos apresentados nesse capítulo não garantem as propriedades fundamentais definidas para o problema da difusão confiável e implementam uma versão do problema com objetivo de melhorar a eficiência ou garantir a maior confiabilidade possível. Tais protocolos podem ser caracterizados como de melhor esforço, justamente por possuir estratégias que buscam resolver o problema da melhor forma, embora não ofereçam garantias teóricas da sua resolução. Para finalizar o capítulo, apresenta-se uma análise de desempenho desses protocolos, realizada através de simulações executadas no ambiente de MANET. O objetivo é identificar o impacto das falhas nesses protocolos e definir os seus níveis de confiabilidade e eficiência.

3.1 DEFINIÇÃO

Um protocolo de difusão (*broadcast*) tem como objetivo transmitir uma mensagem para todos os nós da rede, ou seja, se um processo p emite uma mensagem de difusão m, cada processo no conjunto Π, formado por todos os processos da rede, deve receber a mensagem m. O desafio do protocolo de difusão no contexto das redes MANET é solucionar o problema em uma rede assíncrona, com alta dinamicidade e incidência de

falhas, onde o conhecimento prévio dos participantes não é considerado e a conectividade é parcial, conforme modelo descrito na Seção 2.1.

O problema da difusão é fundamental para a tolerância a faltas em sistemas distribuídos. O *broadcast* pode ser utilizado como base para o desenvolvimento de outros protocolos nas MANET. Por exemplo, protocolos de roteamento podem usar o *broadcast* para descobrir uma rota entre um nó emissor e o receptor de uma mensagem [TNCS02] ou pode ser utilizado para solucionar o problema do consenso [Vol05].

Como numa MANET a comunicação entre os nós é realizada por meio de canais de comunicação sem fio, a possibilidade de perdas de mensagens é maior do que em redes convencionais, já que os canais de comunicação utilizados são facilmente suscetíveis a interferências. A interferência na comunicação causada pela "inundação" de mensagens na rede é chamada de *Broadcast Storm Problem* [TNCS02]. Para garantir a eficiência e evitar um alto grau de interferência, alguns protocolos são projetados para reduzir o número de mensagens trafegadas na rede. Porém, esta redução diminui a redundância de mensagens e entra em conflito com as características de confiabilidade do protocolo.

3.2 PROTOCOLOS DE BROADCAST PARA MANET

De forma genérica, os algoritmos de difusão para MANET são classificados em probabilísticos ou determinísticos. Alguns protocolos tem como objetivo principal garantir a eficiência de tal forma que a quantidade de mensagens trafegadas na rede seja a menor possível. Outros protocolos são focados na confiança da entrega das mensagens, mesmo que isso implique em alto tráfego de mensagens. Devido as características de comunicação em MANET, para uma mensagem ser enviada para todos os nós da rede é preciso que um grupo de nós atue como retransmissores (*gateways*) dessa a mensagem. Reduzir a quantidade de mensagens trafegadas implica em reduzir a quantidade de retransmissores. Cada protocolo tem premissas básicas para decidir quais nós devem atuar como *gateway*. Porém, reduzir a quantidade de *gateways* compromete a confiabilidade do protocolo, já que a mobilidade e principalmente as falhas podem causar violações nas propriedades da

difusão.

3.2.1 Simple Flooding

O *Simple Flooding* (inundação simples) é um protocolo de difusão bastante trivial. O seu funcionamento implica em fazer com que todo nó repasse cada nova mensagem que for recebida [WC02, CT96], ou seja, todos os nós da rede atuam como retransmissores. Para isso, é preciso que exista um identificador único da mensagem que possibilite o armazenamento para futura verificação de duplicidade no recebimento.

A quantidade de retransmissões no *flooding* é da ordem de $n - 1$, o que determina uma alta incidência de mensagens desnecessárias trafegando na rede, um agravante para problemas de congestionamento, conflito e interferência [TNCS02]. Em contrapartida, a capacidade de tolerância a faltas é alta, devido a redundância inerente ao protocolo.

3.2.2 Minimum Connected Dominating Set (MCDS)

O *Minimum Connected Dominating Set* (MCDS), ou conjunto dominante minimal conectado, é um problema clássico em teoria dos grafos. O seu problema consiste em encontrar um conjunto mínimo de nós na rede que conecta todos os demais nós, onde tal conjunto é denominado de MCDS [WL99]. A partir disto, propôs-se o uso deste conceito nos protocolos de difusão do ambiente de redes MANET. Sendo assim, para implementar a difusão o MCDS deve ser determinado e apenas os seus nós devem atuar como retransmissores. Por definição, em condições ideais, as retransmissões abrangem todos os nós da rede [WL99].

Um protocolo desse tipo seria o mais eficiente possível, com a emissão do mínimo de mensagens para garantir a difusão, com a quantidade de mensagens igual a quantidade de nós pertencentes ao MCDS. Porém, encontrar o conjunto MCDS é um problema de classe NP completo, ou seja, não existem algoritmos que resolvam o problema com um

tempo de processamento determinístico [GK96, WL99], mesmo numa rede estática.

Mesmo na existência de um algoritmo que calcule o MCDS em um modelo coerente com os requisitos de MANET, existem pontos importantes a se considerar. Devido a existência de mobilidade dos nós, o MCDS encontrado pelo algoritmo é válido apenas por um tempo específico. Além disso, uma falha existente em um membro do conjunto pode prejudicar a execução correta do *broadcast*.

3.2.3 Protocolos Probabilísticos

Protocolos desse tipo apresentam uma abordagem probabilística para determinar se um nó deve retransmitir a mensagem. Para isso, associa-se a cada nó um valor fixo P que indica a probabilidade do nó retransmitir a mensagem aos vizinhos [TNCS02, WC02]. Sendo assim, a cada mensagem recebida, o nó executa uma função que indica se o nó deve ou não retransmitir essa mensagem, baseado no valor de P. Considerando o caso específico de $P = 1$, o protocolo funcionaria exatamente como o *Flooding*.

O protocolo probabilístico proposto por Zhang e Agrawal utilizam a probabilidade P associada à regras para atualização dinâmica do seu valor [ZA05]. Sendo assim, o valor de P varia de forma proporcional à densidade da rede. Em uma rede mais densa, o valor de P tende a diminuir e P tende a aumentar em redes menos densas. Além disso, a variação de P ocorre em locais da rede com densidades diferentes. Isso faz com que os nós de uma área possuam valores de P menores do que os nós de outras áreas da rede. Como consequência disso, o protocolo de adapta a situações de densidade variável em uma mesma rede, o que evita transmissões desnecessárias em determinados pontos da rede e aumenta as retransmissões em outros pontos fundamentais. O mecanismo que faz P variar é o recebimento de mensagens duplicadas. Se o nó recebe uma mensagem duplicada, o valor de P tende a diminuir. Caso o nó não receba mensagens duplicadas por um determinado tempo, P tende a aumentar. Sendo assim, em locais mais densos, os nós vão receber mais mensagens duplicadas e a probabilidade P vai diminuindo e tende à um limite inferior. Se um nó se move para uma área menos densa, P vai aumentando

e tende a um limite superior. Portanto, a probabilidade P se adapta de acordo com a densidade da vizinhança em que o nó se encontra. A escolha dos valores de tempo, valor inicial para P e limites de P são importantes para o desempenho do protocolo [ZA05].

Uma característica desse protocolo é a inexistência de mensagens auxiliares para determinar a variação na probabilidade. As informações indiretas de densidade que o protocolo utiliza são obtidas através das próprias mensagens de difusão enviadas.

3.2.4 Protocolo de Wu e Li

O protocolo de Wu e Li [WL99] utiliza algumas regras para determinar um conjunto de nós que se aproximam do MCDS (Seção 3.2.2). A informação local do conhecimento dos vizinhos é utilizada para determinar se um nó deve retransmitir uma mensagem para garantir as propriedades da difusão. O princípio é simples: se um nó p possui quaisquer dois nós q e r no seu conjunto de vizinhos $N(p)$ que não estejam diretamente conectados entre si ($r \notin N(q)$), este nó deve ser marcado como retransmissor, para garantir que ambos os vizinhos q e r recebam a mensagem. Foi provado que o conjunto formado pelos nós marcados como retransmissores forma um CDS, ou seja, um conjunto que conecta todos os demais nós na rede mas que não é o conjunto mínimo possível. Sendo assim, em [WL99] são propostas ainda duas regras para otimizar o protocolo. Estas regras reduzem ainda mais o número de nós retransmissores na rede:

i) Se o conjunto de vizinhos de um nó p está contido no conjunto de vizinhos $N(q)$ de q e se p e q estão marcados como retransmissores, desmarcar o nó com a menor prioridade (geralmente atribuída pelo identificador do nó). A justificativa é que se um nó já envia a mensagem para um conjunto específico de nós, outro nó não precisa reenviar a mensagem para os mesmos nós.

ii) Se o conjunto dos vizinhos $N(p)$ de um nó p está contido na união do conjunto de dois vizinhos q e r de p ($N(p) \subset N(q) \cup N(r)$), desmarcar p caso tenha a menor prioridade.

Em [WL99] existe a prova de que mesmo com tais regras o conjunto de nós marcados como retransmissores continua sendo um CDS. A informação sobre os conjuntos dos vizinhos de cada nó pode ser implementada usando pacotes auxiliares do tipo HELLO. Este pacote deve ser enviado periodicamente e conter o identificador do nó emissor e o conjunto de vizinhos deste mesmo nó. Assim, cada nó que recebe um pacote $HELLO$ tem a informação de que o nó emissor é seu vizinho e obtém ainda os vizinhos deste nó emissor.

Posteriormente, foram propostas variações ao algoritmo original de Wu e Li com o objetivo de melhorar a eficiência do protocolo. Porém, foram realizadas simulações com estes protocolos que mostraram que o protocolo original possui uma maior confiabilidade [DW04, DW03].

3.2.5 Scalable Broadcast Algorithm (SBA)

O *Scalable Broadcast Algorithm* (SBA) [PL00], ou algoritmo de difusão escalável, é uma alternativa de protocolo que utiliza algumas premissas para determinar um conjunto de nós aproximado do MCDS. O SBA baseia-se na informação dos vizinhos de cada nó, obtida pelo envio periódico de pacotes $HELLO$ para a realização desse cálculo. Além disso, esse cálculo é realizado por cada vizinho que recebe a mensagem, de forma a distribuir a responsabilidade da decisão sobre a retransmissão entre os nós da rede.

Existem duas verificações realizadas por cada nó para decidir sobre a retransmissão da mensagem. Na primeira, no momento do recebimento, o nó verifica se o seu conjunto de vizinhos é um subconjunto do conjunto de vizinhos do nó emissor. Caso positivo, o nó descarta a mensagem. Caso contrário, o nó adiciona os vizinhos do nó emissor ao conjunto de nós que receberam a mensagem e aguarda por um tempo definido. Se novas mensagens chegarem e o conjunto de vizinhos do nó for um subconjunto dos nós que já receberam a mensagem, a mensagem é descartada. Ao término deste tempo, a mensagem é enfim retransmitida.

O problema deste algoritmo é que ele está fortemente vinculado à informação local dos vizinhos. Como o ambiente da rede tem requisito de mobilidade, o conjunto de vizinhos pode ser modificado e se tornar inconsistente ao longo do processo de *broadcast*. Como consequência, um nó pode decidir por não retransmitir uma mensagem baseada na informação inconsistente de vizinhança, o que é prejudicial ao protocolo e causa o não recebimento da mensagem por alguns nós. Considerando falhas dos nós, este problema é ainda mais acentuado.

3.2.6 Dominant Pruning (DP)

O *Dominant Pruning* (DP) [LK01] é um protocolo de difusão determinístico que utiliza a informação de vizinhança com o objetivo de reduzir a quantidade de retransmissões redundantes de uma determinada mensagem. Neste algoritmo, cada nó p que recebe a mensagem escolhe o conjunto mínimo de retransmissores entre os seus vizinhos imediatos $N(p)$, ou seja, os vizinhos de um nível. Esse conjunto deve garantir que a mensagem seja recebida por todos os seus vizinhos de dois níveis. O pacote que contém a mensagem de *broadcast* agrega também a lista dos nós que devem atuar como retransmissores. Sendo assim, no recebimento da mensagem, um nó atua como retransmissor caso esteja contido nessa lista e executa novamente o algoritmo do DP para determinar os seus novos retransmissores.

O problema do protocolo *Dominant Pruning* é considerar que sempre que p envia a mensagem, todos os seus vizinhos em $N(p)$ recebem e garantem que os vizinhos de dois níveis de p também recebam a mensagem. Na presença de falhas, um dos vizinhos de p marcado como retransmissor pode falhar e comprometer toda a execução do protocolo.

3.2.7 Double-Covered Broadcast (DCB)

O *Double-Covered Broadcast* (DCB) [LW07] é um protocolo de difusão determinístico que tem como objetivo principal reduzir a quantidade de retransmissões sem prejudicar a

redundância, fundamental para a capacidade de tolerar falhas dos protocolos de difusão. O DCB baseia-se na informação de vizinhança para determinar o conjunto de nós retransmissores para cada nó que recebe a mensagem. Esse conjunto é determinado através dos vizinhos de um nível do nó de forma que todos os vizinhos de dois níveis possam receber mensagem. Além disso, os vizinhos imediatos devem estar conectados por pelo menos dois transmissores, o próprio nó emissor e um outro nó retransmissor selecionado segundo as regras do algoritmo. As mensagens retransmitidas funcionam como confirmação do recebimento (ACK). Cada nó emissor espera por um tempo pelo envio da mensagem pelos *gateway* e, caso detecte que a mensagem não foi retransmitida corretamente, o nó emissor envia novamente a mensagem até receber a retransmissão de todos os *gateways* selecionados. Esse mecanismo utilizado no DCB garante o aumento na redundância e uma maior garantia na entrega das mensagens.

3.3 AVALIAÇÃO DE PROTOCOLOS DE BROADCAST PARA MANET

Existem diversos trabalhos que realizam comparações de desempenho entre protocolos de *broadcast* para redes MANET. Dentre estes trabalhos, pode-se destacar o desenvolvido por Oliveira *et al.* [dOCG07], onde são realizadas simulações em modelos realistas com o objetivo de avaliar a eficiência e, sobretudo, a capacidade dos protocolos em tolerar falhas. O modelo de simulações utilizado e os resultados obtidos por Oliveira *et al.* [dOCG07] serão apresentados a seguir.

3.3.1 Modelo de Simulações

Os protocolos selecionados para simulações foram: *Simple Flooding*, probabilístico dinâmico, protocolo de Wu e Li, SBA, DP e DCB. A Tabela 3.1 representa os parâmetros utilizados nas simulações destes protocolos no estudo de Oliveira *et al.* [dOCG07], sendo que tais parâmetros foram baseados em estudos anteriores. Dentre os parâmetros descritos, pode-se destacar o padrão de mobilidade *Gauss-Markov*, descrito na Seção 2.4, e

Tabela 3.1. Parâmetros de simulação para os protocolos de difusão

Parâmetros de simulação	
Simulador	NS-2 (2.30)
Quantidade de nós	10 a 100
Área	1300 x 1300 m^2
Alcance *wireless*	250 m
Tempo de simulação	500 s
Repetições	20
Padrão de Mobilidade	Gauss-Markov
Velocidade dos nós	1 m/s
Taxa de difusão	10 msg/s

a taxa de difusão de $10\,msg/s$, que indica que cada nó emite na rede dez mensagens de difusão por segundo.

O modelo de falhas utilizado foi o de falha por omissão, onde nós falhos param de enviar e receber determinadas mensagens, modelo melhor definido na Seção 2.3.1.

Para a análise dos resultados, utilizaram-se as métricas:

i) *Delivery Ratio*. Representa a taxa de cobertura alcançada pelo protocolo, ou seja, a quantidade de nós que recebem a mensagem de difusão, o que indica a confiabilidade dos protocolos;

ii) *Forwarding Ratio*. Contabiliza a quantidade de nós que atuam como retransmissores, que representa a eficiência dos protocolos;

iii) *End-to-End Delay*. Mede o tempo gasto na execução dos algoritmos de difusão selecionados.

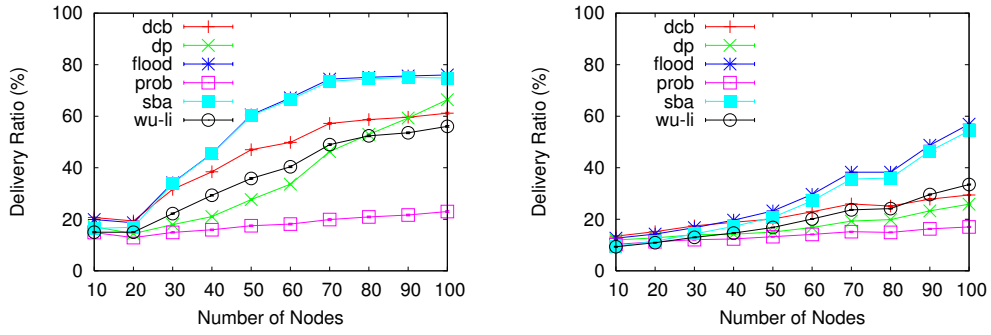
3.3.2 Análise dos Resultados

Conforme informado na Tabela 3.1, as simulações são executadas por 20 repetições em cada cenário testado. Desta forma, para garantir a credibilidade na apresentação dos resultados, os gráficos mostram a média para cada métrica associada ao intervalo de

confiança de 95%, como sugerido na Seção 2.4.

Os gráficos apresentados nas Figuras 3.1, 3.2 e 3.3 foram selecionados dentre os apresentados por Oliveira *et al.* [dOCG07]. Tais gráficos representam os resultados das métricas para cenários livre de falhas (0%) e com 50% dos nós falhos (por omissão), em função da quantidade de nós existentes na rede. De acordo com o modelo de falhas proposto, para o caso de 50% de falhas, a cada 10 segundos uma seleção aleatória é realizada e metade dos nós da rede falham por omissão durante este tempo, ou seja, tais nós não enviam ou recebem mensagens durante 10 segundos, até que uma nova seleção seja realizada.

Delivery Ratio. Através da análise dos gráficos da Figura 3.1, foi possível identificar que o *Simple Flooding* obteve o melhor desempenho, ou seja, foi o protocolo que possuiu a maior taxa de entrega de mensagens, caracterizando-o como o de maior confiabilidade dentre os selecionados. O SBA obteve desempenho similar ao *Flooding*, sendo levemente inferior em alguns casos. O algoritmo probabilístico dinâmico obteve o pior desempenho e os demais protocolos tiveram resultados intermediários. Além disso, comparando os gráficos 3.1(a) e 3.1(b), mostrou-se que a quantidade de falhas injetadas afeta consideravelmente o desempenho dos protocolos. A taxa de entrega dos protocolos é maior para uma quantidade de falhas menor, sendo que exemplica-se este comportamento através do cenário livre de falhas (Figura 3.1(a)).

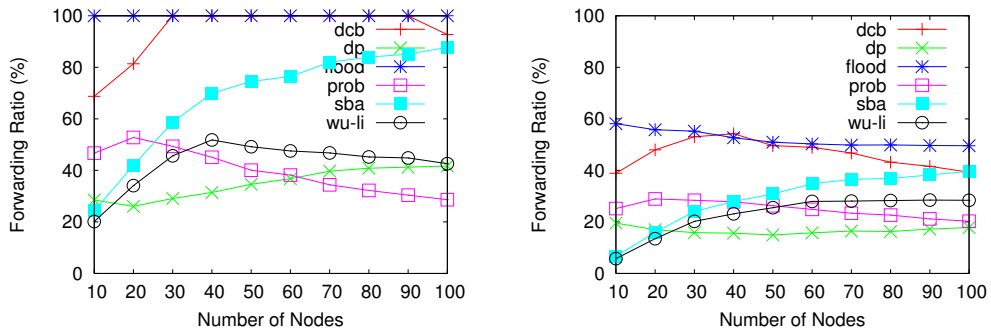


(a) 0% de falhas (b) 50% de falhas

Figura 3.1. Taxa de entrega dos protocolos para 0% (a) e 50% (b) de falhas

Forwarding Ratio. Através da descrição presente na Seção 3.2.1, sabe-se que no *Flo-*

oding todos os nós atuam como retransmissores. Sendo assim, conforme o esperado e confirmado pela Figura 3.2, o *Flooding* apresenta a maior quantidade de retransmissores dentre os protocolos selecionados. O DCB apresenta resultados similares ao *Flooding*, uma consequência da suas características de redundância. O probabilístico dinâmico apresentou uma baixa quantidade de retransmissores, o que justificativa o resultado ruim na taxa de entrega de mensagens. O SBA apresenta em geral resultado um pouco melhor que o *Flooding* e o DCB. Além disso, comparando os gráficos 3.2(a) e 3.2(b), mostrou-se que a quantidade de falhas injetadas afeta na quantidade de retransmissores existentes, de forma similar ao desempenho da métrica anterior.

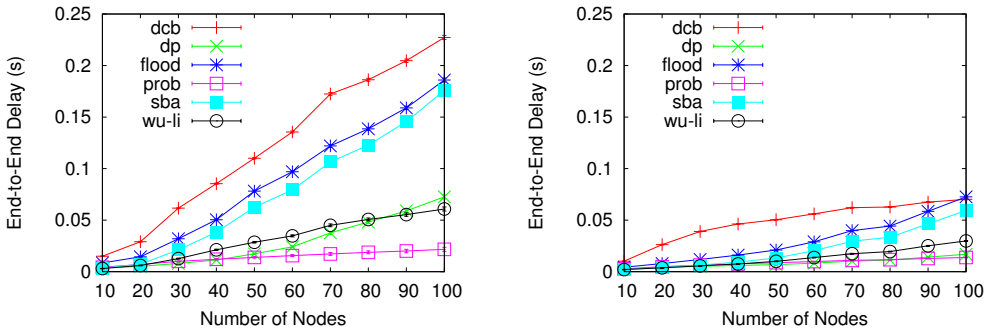


(a) 0% de falhas (b) 50% de falhas

Figura 3.2. Quantidade de retransmissores para 0% (a) e 50% (b) de falhas

End-to-End Delay. Os resultados apresentados na Figura 3.3 evidenciaram que o protocolo com maior tempo de execução é o DCB, seguido pelo *Flooding* e o SBA. O probabilístico obteve o menor tempo, porém, esse resultado não pode ser considerado relevante, uma vez que o seu desempenho na métrica *Delivery Ratio* foi muito inferior em relação aos demais protocolos. Através da comparação dos gráficos 3.3(a) e 3.3(b), mostrou-se que também nesta métrica a quantidade de falhas afeta o tempo de execução do protocolo, causando um tempo inferior, justificado pelos resultados das métricas anteriores.

Portanto, através da análise dos resultados, pode-se concluir que a redundância é fundamental para garantir a confiabilidade do protocolo. Em cenários com existência de falhas, uma maior quantidade de retransmissores é essencial para garantir uma maior



(a) 0% de falhas (b) 50% de falhas

Figura 3.3. Tempo de execução dos protocolos para 0% (a) e 50% (b) de falhas

taxa de entrega de mensagens de difusão entre os nós da rede. Sendo assim, a eficiência pode ser sacrificada em detrimento da confiabilidade, em casos onde a execução correta do protocolo de difusão é fundamental para a aplicação.

CAPÍTULO 4

O PROBLEMA DO CONSENSO

Este capítulo apresenta o consenso, um problema fundamental para tolerância a faltas em sistemas distribuídos. O capítulo é iniciado com a definição do problema do consenso e os resultados formais sobre as suas possibilidades de resolução no modelo clássico de sistemas distribuídos. Posteriormente, descreve-se as principais estratégias de resolução do consenso no ambiente de redes estáticas. Após apresentar o consenso clássico, considera-se o mesmo problema do consenso estendido com requisitos próprios do modelo de redes dinâmicas, como o ambiente de redes MANET definido no Capítulo 2. Por fim, serão apresentados trabalhos que realizam a análise de desempenho de soluções para o consenso na prática, demonstrando os resultados obtidos através de simulações.

4.1 CONSENSO CLÁSSICO

O consenso é um problema fundamental para a tolerância a faltas, uma vez que diversos problemas desta área são redutíveis a ele, como por exemplo a difusão atômica [CT96]. Informalmente, o consenso tem o objetivo de fazer com que todos os processos corretos do sistema decidam por um valor único entre os valores propostos pelos mesmos. Sendo assim, a definição formal do consenso clássico consiste nas seguintes propriedades [CT96]:

- *Terminação*. Todo processo correto decide por um valor;
- *Validade*. Se um processo decide por um valor, ele foi proposto por algum processo;
- *Acordo*. Todos os processos corretos decidem pelo mesmo valor.

Uma variação do consenso clássico é o *consenso uniforme*, no qual a propriedade de acordo é válida tanto para os processos corretos como para processos falhos:

- *Acordo Uniforme*. Todos os processos (corretos ou não) decidem pelo mesmo valor.

4.1.1 Modelo Clássico (FLP)

Considera-se como modelo um sistema distribuído formado por um conjunto finito Π de $n > 1$ processos ou nós, que se comunicam através de canais confiáveis (sem perdas de mensagens) e de forma assíncrona (sem que sejam feitas hipóteses temporais sobre as ações dos processos e canais). Considera-se que $f < n$ processos podem falhar por parada (*crash*), através de um colapso brusco ou saída deliberada (*switched off*). Considera-se ainda a existência canais de comunicação confiáveis entre quaisquer par (p, q) de processos no sistema, de forma que qualquer processo p pode enviar mensagens para um processo q de forma confiável [FLP85].

4.1.2 Resolução

Sabe-se que, para garantir a execução correta de algoritmos distribuídos, as propriedades de progressão (*liveness*) e segurança (*safety*) devem ser satisfeitas [Lam77]. Sendo assim, pode-se relacionar a propriedade de terminação do consenso com a progressão e as propriedades de validade e acordo com a segurança do sistema.

Um resultado teórico importante para o consenso é a impossibilidade FLP, uma prova formal de que o consenso não pode ser resolvido em ambientes assíncronos na presença de falhas de processos, mesmo que os canais sejam confiáveis [FLP85]. Na existência dessa impossibilidade, diversos autores propuseram alternativas para resolver o consenso. Deste modo, soluções probabilísticas e determinísticas foram apresentadas, utilizando como modelo ambientes estendidos com requisitos temporais ou impondo restrições nas propriedades definidas para o consenso. Soluções probabilísticas resolvem o consenso com uma associação de uma probabilidade para a sua convergência, ou seja, existe a possibilidade da ocorrência de execuções em que o consenso não é resolvido [BO83]. As soluções determinísticas usam ambientes estendidos com requisitos temporais, com a

utilização de um modelo de sincronia mais restrito do que o modelo assíncrono. Muitas vezes, propostas nesta linha utilizam abstrações como os detectores de falhas e eleição de líder [CT96]. Todas essas soluções propõem algoritmos que solucionam o consenso através da execução em rodadas, sendo que diversas rodadas do protocolo podem ser executadas até que se obtenha o acordo sobre uma determinada estimativa. Dentre estas soluções, pode-se observar que o serviço de difusão confiável é essencial para garantir a execução correta e a validade das propriedades do consenso.

4.1.3 Paxos

Paxos [Lam98, Lam01] é um protocolo clássico proposto por Lamport que resolve o consenso na presença de falhas, com o suporte de um protocolo para eleição de líder. Mostrou-se posteriormente que o detector de líder utilizado no Paxos corresponde ao Ω [BDFG03].

Para explicar o Paxos, define-se um conjunto de três classes de agentes: *proposals*, *acceptors* e *learners*. Cada processo pode assumir o papel de um ou mais tipos de agentes durante a execução do protocolo. Define-se a seguir a execução do protocolo Paxos de forma simplificada. Inicialmente, cada agente *proposal* envia a sua estimativa sobre o valor para os agentes *acceptors*. Os *acceptors* escolhem o maior valor de estimativa e a enviam por mensagens de resposta para os agentes *proposal*. Caso um agente receba uma resposta da maioria dos *acceptors* contendo a sua própria estimativa, ele envia para os *acceptors* uma confirmação da aceitação da sua estimativa pela maioria. Os *acceptors* finalizam aceitando essa estimativa como resposta dos agentes *proposal*. Para terminar a execução do algoritmo, os agentes do tipo *learner* enviam mensagens de requisição sobre o valor decidido e os *acceptors* respondem com o envio da estimativa concretizada na etapa anterior.

4.1.4 Oráculo Aleatório

A resolução do consenso através de um oráculo aleatório foi proposta inicialmente por Ben-Or [BO83]. No seu algoritmo, os processos iniciam a rodada compartilhando as suas estimativas e, em cada rodada que não se chegue a um acordo sobre algum valor estimado, cada processo chama o seu oráculo e gera uma nova estimativa, de forma aleatória e dentro de um conjunto de possíveis valores. Sendo assim, associa-se ao protocolo uma probabilidade de terminação, o que caracteriza este consenso como probabilístico [Asp03]. O protocolo proposto é capaz de tolerar $f < n/2$ falhas e, no pior caso, o seu tempo de convergência é exponencial (em relação a quantidade de processos n).

Uma evolução para a resolução do consenso probabilístico foi proposta por Ezhilchelvan *et al.* [EMR01]. No seu protocolo, ao contrário do protocolo de Ben-Or [BO83], os processos podem propor valores arbitrários, sem necessidade de conhecimento inicial do conjunto de valores possíveis. A principal diferença em relação ao algoritmo de Ben-Or é a existência da difusão inicial das estimativas de cada nó. Sendo assim, cada nó forma um conjunto das estimativas possíveis de todos os nós da rede e a escolha aleatória de novos valores para as estimativas é realizada sobre esse conjunto.

4.1.5 Detecção de Falhas

O detector de falhas (FD) é uma abstração de sincronia proposto por Chandra e Toueg [CT96] que funciona como um oráculo associado aos processos distribuídos, informando-os sobre as suspeitas de falhas ocorridas nos componentes dos sistemas. De certa forma, os detectores de falhas foram criados para abstrair os requisitos temporais necessários para solucionar problemas como o consenso e a difusão atômica. Ele permite com que algoritmos assíncronos sejam propostos para resolver problemas fundamentais de tolerância a faltas de forma independente das características de sincronia do ambiente. O detector de falhas é definido pelas propriedades de completude e exatidão [CT96].

A completude garante que todos processos que falham serão suspeitos.

Tabela 4.1. Classes de Detectores de Falhas

Completude	Exatidão			
	Forte	Fraca	Forte Após um Tempo	Fraca Após um Tempo
Forte	$\mathcal{P}$	$\mathcal{S}$	$\Diamond\mathcal{P}$	$\Diamond\mathcal{S}$
Fraca	$\mathcal{Q}$	$\mathcal{W}$	$\Diamond\mathcal{Q}$	$\Diamond\mathcal{W}$

i) *Completude forte.* Todo processo falho será suspeitado por todos os processos corretos;

ii) *Completude fraca.* Um processo correto suspeitará de todos os processos falhos.

A exatidão faz restrições em relação aos erros do detector.

i) *Exatidão forte.* Nenhum processo é suspeito antes que falhe;

ii) *Exatidão fraca.* Existe pelo menos um processo correto que nunca será suspeitado;

iii) *Exatidão forte após um tempo.* Em algum tempo futuro, todo processo correto não será suspeito por nenhum processo correto;

iv) *Exatidão fraca após um tempo.* Em algum tempo futuro, algum processo correto não será suspeito por nenhum processo correto;

A combinação destas propriedades define as classes dos detectores $\mathcal{P}$ (perfeito), $\mathcal{S}$ (forte), $\Diamond\mathcal{P}$ (perfeito após um tempo), $\Diamond\mathcal{S}$ (forte após um tempo), $\mathcal{Q}$ (quase perfeito), $\mathcal{W}$ (fraco), $\Diamond\mathcal{Q}$ (quase perfeito após um tempo) e $\Diamond\mathcal{W}$ (fraco após um tempo) [CT96]. Tais classes e suas propriedades encontram-se representadas na Tabela 4.1.

O detector minimal para resolver o consenso num sistema clássico em ambientes sujeitos à falhas e com uma maioria de processos corretos ($f < n/2$) é o $\Diamond\mathcal{S}$ [CHT96]. Sendo assim, mesmo um detector não-confiável como o $\Diamond\mathcal{S}$ é suficiente para resolver o consenso na presença de falhas. Mostrou-se ainda que o consenso uniforme também pode ser solucionado com o uso do detector $\Diamond\mathcal{S}$. Portanto, o $\Diamond\mathcal{S}$ é o requisito mínimo de

sincronia necessário para resolver o consenso no modelo clássico de sistemas distribuídos na presença de falhas.

Como os detectores de falhas abstraem requisitos de sincronia do sistema, nenhuma destas classes de detector pode ser implementada em ambiente assíncrono. Porém, Larrea *et al.* [LFA04] mostrou que os detectores mais fracos $\diamondsuit\mathcal{P}$, $\diamondsuit\mathcal{Q}$, $\diamondsuit\mathcal{S}$ e $\diamondsuit\mathcal{W}$ podem ser implementados em um modelo temporal parcialmente síncrono com uma maioria de processos corretos. Entretanto, os detectores perpetuais $\mathcal{P}$, $\mathcal{Q}$, $\mathcal{S}$ e $\mathcal{W}$ não podem ser implementados nem mesmo em ambientes parcialmente síncronos.

4.1.6 Detecção de Líder

Assim como o detector de falhas, o detector de líder é uma abstração de sincronia que funciona como um oráculo associado a cada processo da rede. A definição do detector de líder informa que, após um tempo, um único processo correto em Π é designado como líder [MR01].

Formalmente, um detector de líder Ω [CT96] possui uma função LEADER que satisfaz a seguinte propriedade:

- *Eventual Leadership*. Após um tempo, todos processos que executam a função LEADER obtêm o mesmo processo p como retorno.

O detector de líder que satisfaz essa propriedade é equivalente ao $\diamondsuit\mathcal{S}$ e pode ser utilizado para resolver o consenso em ambiente assíncrono com $f < n/2$ [CHT96]. Além disso, mostrou-se que o protocolo de detecção de líder utilizado no Paxos [Lam98] para resolver o consenso é equivalente ao detector do tipo Ω [BDFG03].

4.1.7 Consenso Indulgente Genérico

Guerraoui e Raynal [GR04] propuseram um algoritmo genérico para resolução do consenso em que três tipos de abstrações podem ser utilizadas de maneira modular,

ou seja, generaliza os protocolos que são associados a abstrações. Além disso, essas abstrações não violam as propriedades de segurança do consenso. Sendo assim, pode-se dizer que o consenso proposto é indulgente em relação as abstrações utilizadas.

As abstrações que podem ser utilizadas no protocolo são: o detector de falhas $\diamondsuit S$, o detector de líder Ω e um oráculo aleatório. O detector de líder e o detector de falhas são oráculos equivalentes que possibilitam soluções determinísticas para o consenso, já o oráculo aleatório caracteriza um consenso probabilístico. Tais abstrações são necessárias apenas para garantir a propriedade *liveness* (terminação) do consenso e, por isso, não afetam *safety* (validade e acordo). Para qualquer que seja a abstração utilizada, o consenso indulgente genérico possui o limite para as falhas tolerável de $f < n/2$.

Para possibilitar a utilização dos três oráculos sem haver modificação na estrutura básica do algoritmo, uma outra abstração LAMBDA foi definida. São propostas três implementações para LAMBDA, cada uma deve ser utilizada especificamente com um dos oráculos. É importante ressaltar que estas implementações de LAMBDA não substituem os oráculos, são usadas em conjunto com um destes. Descreve-se a seguir o algoritmo utilizado no consenso genérico, que sintetiza também a execução do consenso para os três oráculos: aleatório, detecção de falhas e detecção de líder, descritos nas Seções 4.1.4, 4.1.5 e 4.1.6, respectivamente.

Conforme observado no Algoritmo 1, o protocolo genérico de Guerraoui e Raynal [GR04] é executado em rodadas que são subdivididas em duas fases. Na primeira fase (linha 5), executa-se a função LAMBDA na forma $est2_i \leftarrow lambda(r_i, est1_i)$, sendo r_i a rodada atual do processo i, $est1_i$ a estimativa inicial e o valor retornado pela função é a nova estimativa $est2_i$. Assim como LAMBDA é utilizada no algoritmo principal desta solução para o consenso, os oráculos são utilizados na implementação de cada uma das três variações de LAMBDA. A execução da abstração LAMBDA associada ao seu oráculo deve garantir que, em uma rodada r, todos os processos que invocaram *lambda* obtém o mesmo valor para $est2$. Na segunda fase (a partir da linha 7), todos os processos enviam mensagens de difusão, que contém as suas estimativas $est2_i$ obtidas na fase anterior, e aguardam por $n-f$

Algoritmo 1 Consenso Indulgente Genérico [GR04]

```
1: est1_i ← v_i; r_i ← 0
2: loop
3:    r_i ← r_i + 1
4:    ========= Fase 1 da rodada r_i ==========
5:    est2_i ← lambda(r_i, est1_i)
6:    ========= Fase 2 da rodada r_i ==========
7:    broadcast phase2(r_1, est2_i)
8:    wait until (phase2(r_1, est2) mensagens recebidas de n − f processos)
9:    let rec_i = {est2 | phase2(r_1, est2)}
10:   if rec_i = {v} then
11:      R_broadcast decide(v)
12:      return v
13:   else if rec_i = {v, ⊥} then
14:      est1_i ← v
15:   else if rec_i = {⊥} then
16:      est1_i ← ⊥
17:   end if
18: end loop
19:
20: upon receive decide(v): return v
```

destas mensagens. Caso as mensagens recebidas contenham um único valor v, este valor é utilizado como resultado do consenso e uma nova difusão é realizada para que a decisão seja realizada por todos os processos. Caso não ocorra as condições necessárias para a decisão, uma nova rodada se inicia com a execução das fases descritas anteriormente.

4.1.8 O Modelo Heard-Of (HO)

Um modelo denominado *Heard-Of* (HO) [CBS07] foi proposto com o intuito de unificar as noções de sincronia e falhas que os algoritmos para o consenso se baseiam, além de estender o modelo de falhas para englobar tanto falhas de processos como falhas dos canais de comunicação. Para isso, define-se uma notação $HO(p, r)$ que indica o conjunto de processos escutados por um processo p em uma rodada r, ou seja, os processos que enviaram mensagem para p naquela rodada. A existência de uma falha de comunicação no envio de uma mensagem de q para p em uma rodada r, ou até mesmo a falha por

parada no processo q, faz com que q não esteja presente no conjunto representado por $HO(p, r)$. Sendo assim, tanto características de sincronia como as falhas, de processos ou de canais, encontram-se indistintamente caracterizadas nesse modelo.

No modelo *Heard-Of*, um predicado estabelecido sobre o conjunto $HO(p, r)$ torna possível a resolução do consenso, sendo que o modelo HO pode ser aplicado em soluções já existentes para o consenso. A aplicação do modelo ao consenso probabilístico [BO83] é representada pelo algoritmo *UniformVoting*. O algoritmo *LastVoting* representa o Paxos [Lam98] aplicado ao modelo HO. Cada um desses algoritmos define o seu predicado que deve ser satisfeito para sua execução correta.

4.2 CONSENSO DINÂMICO

Esta seção destina-se a abordagem do consenso em redes dinâmicas. A definição do problema do consenso em redes dinâmicas abrange as mesmas propriedades descritas para o consenso no modelo clássico (Seção 4.1). Porém, novas características introduzidas pelo modelo de redes dinâmicas implicam em requisitos adicionais que devem ser considerados para a resolução do problema. Algumas destas características são: comunicação sem fio, o grafo de conectividade formado pelos nós da rede não é completamente conexo, mobilidade dos nós, entradas e saídas de nós. No Capítulo 2 pode ser encontrada uma definição mais detalhada das características encontradas em redes dinâmicas, mais especificamente em redes MANET. Os protocolos descritos a seguir buscam resolver o consenso considerando algumas destas características na definição do seu modelo de sistema. Porém, todas as soluções apresentadas nesta seção consideram o conhecimento inicial do conjunto de participantes Π. Como visto na Seção 2.1, essa hipótese é muito forte para o ambiente de MANET.

4.2.1 Paxos para MANET

Como descrito na Seção 4.1.8, o modelo *Heard-Of* (HO) pode ser combinado com soluções já existentes para o consenso para resolver o consenso na presença de falhas de canais ou de processos. O algoritmo *LastVoting* proposto em [CBS07] representa a adaptação do Paxos ao modelo HO e define um predicado que deve ser satisfeito para garantir a execução correta do algoritmo.

O modelo HO torna-se útil na definição de soluções para o consenso em redes MANET, onde considerar falhas de canais é importante, uma vez que as características dessas redes indicam que tais falhas são frequentes (Seção 2.2). Sendo assim, Borran *et al.* [BPS08] propôs uma solução para o consenso em redes MANET que utiliza o algoritmo *LastVoting* combinado com uma camada de comunicação adequada ao ambiente dessas redes. Essa camada de comunicação deve garantir a validade do predicado definido para o *LastVoting*, de forma a proporcionar a eleição de um processo líder e a difusão de mensagens.

Portanto, estender o Paxos para o modelo HO e utilizar uma camada de comunicação adequada faz com que um protocolo de consenso do modelo clássico possa ser adaptado a algumas características de redes dinâmicas, como a mobilidade e falhas frequentes nos canais de comunicação.

4.2.2 Consenso Probabilístico para MANET

Vollset [Vol05] propõe uma solução para o consenso em MANET que é uma adaptação do consenso probabilístico de Ezhilchelvan *et al.* [EMR01] utilizada em conjunto com um protocolo de difusão confiável adequado ao modelo de MANET. No modelo utilizado para solucionar o consenso e a difusão confiável, a mobilidade dos nós é considerada e o conhecimento inicial do conjunto Π é um requisito fundamental para a execução dos algoritmos. São propostas ainda otimizações para o consenso utilizado, com o objetivo de adaptá-lo as características das redes MANET.

Vollset [Vol05] concluiu que a solução probabilística do consenso é adequada ao ambiente de redes MANET, uma vez que soluções que utilizam oráculos como os detectores de falhas introduzem novas mensagens na rede, aumentando a latência de execução do consenso. Além disso, tais oráculos geralmente utilizam temporizadores na sua implementação, o que é difícil de definir em um ambiente dinâmico. Outra vantagem é que a solução probabilística é executada de forma completamente descentralizada, sem a necessidade de um coordenador, que em MANET pode estar inacessível devido as características da rede, como a possibilidade de particionamento entre os nós.

Bonnet *et al.* [BEV06] propõe resolver o consenso para MANET com a subdivisão do problema em duas etapas. Na primeira etapa executa-se uma versão mais fraca do consenso, onde existe apenas a garantia de que ao menos um processo correto decide, ou seja, a propriedade de terminação do consenso (definida na Seção 4.1) é modificada para:

- *Terminação Fraca*. Pelo menos um processo correto decide por um valor.

Na segunda etapa, executa-se um protocolo de difusão confiável para disseminar a decisão obtida na primeira etapa, obtendo o consenso com as suas propriedades fundamentais (descritas na Seção 4.1).

Nesta solução, existem dois tipos de propriedades definidas para os protocolos de difusão e consenso utilizados:

i) $\diamondsuit\mathcal{Q}$. Em algum tempo futuro, a transmissão de mensagens referente à uma execução do protocolo (consenso ou difusão) termina;

ii) $\diamondsuit\mathcal{R}$. Em algum tempo futuro, os nós descartam a mensagem de difusão.

Estas propriedades garantem uma melhor eficiência no uso dos canais de comunicação e da memória dos processos, características desejáveis para os protocolos desenvolvidos para redes MANET. Associando-se as propriedades aos protocolos utilizados na proposta de Bonnet *et al.* [BEV06], pode se dizer que a primeira etapa do consenso usa um protocolo de difusão do tipo $\diamondsuit\mathcal{R}$-*broadcast* e a segunda utiliza o $\diamondsuit\mathcal{Q}$-*broadcast*. Para finalizar,

mostrou-se que, utilizando um protocolo de difusão da classe $\diamondsuit\mathcal{R}$-broadcast, a solução encontrada em [EMR01] resolve a versão mais fraca do consenso e com a propriedade $\diamondsuit\mathcal{Q}$, com a quantidade de falhas $n > 4f$ e que pode ainda ser otimizada para tolerar a quantidade de falhas $3f < n \leq 4f$.

4.2.3 Consenso Hierárquico

Com o crescimento da quantidade de processos existentes na rede, uma preocupação evidente é tornar o consenso escalável. Para isso, deve-se focar em reduzir a quantidade de mensagens trocadas entre os nós, ou seja, melhorar a eficiência do protocolo. Uma das alternativas para melhorar a eficiência dos protocolos é o estabelecimento de uma hierarquia entre os nós. Wu *et al.* [WCYR07] propõe um consenso hierárquico para redes MANET, onde um conjunto de nós H de hierarquia superior é selecionado e os demais nós $(\Pi - H)$ estão associados a um nó do conjunto H.

Nesta proposta, utiliza-se o detector de falhas eventualmente perfeito $\diamondsuit\mathcal{P}$ e a quantidade máxima de falhas tolerável é o menor valor entre a metade da quantidade de nós da rede $(n/2)$ e a cardinalidade $|H|$, ou seja, $f < min(|H|, n/2)$. Portanto, para tolerar $f < n/2$ falhas, o conjunto H deve possuir cardinalidade mínima de $f + 1$ ($|H| \geq f + 1$). Para iniciar o conjunto H, pode-se utilizar uma função aleatória ou basear-se em alguma métrica que indique um conjunto de nós privilegiados. Considera-se no modelo do sistema que cada par de nós (p, q) está conectado por um canal confiável.

Em linhas gerais, a execução do protocolo é baseada na realização do acordo entre os nós do conjunto H. Para cada rodada, um coordenador é selecionado neste conjunto, que envia sua estimativa para os nós em H. A partir disso, os nós que não pertencem a H enviam suas estimativas para o seu nó associado e o acordo é realizado entre os nós do conjunto H. Com um valor de decisão obtido, uma mensagem de difusão é enviada para que todos os nós da rede possam decidir por este mesmo valor. Se em algum momento o coordenador ou algum nó em H for suspeitado, executa-se um procedimento de recuperação para a substituição deste nó e uma nova rodada é iniciada.

4.2.4 Detectores de Falhas para Redes Dinâmicas

Devido as características peculiares das redes MANET, os protocolos propostos para detecção de falhas e eleição de líder encontradas para o modelo clássico não são adequadas para o modelo de redes dinâmicas. Entretanto, existem propostas para estes oráculos que consideram algumas das características encontradas nos ambientes das redes dinâmicas. Nesta linha, pode-se destacar no contexto desse trabalho os detectores que visam solucionar o problema sem considerar o conhecimento inicial dos participantes da rede.

4.2.4.1 Detecção de Líder. Jiménez *et al.* [JAF06] mostraram que o detector de líder Ω pode ser implementado mesmo sem considerar o conhecimento inicial dos participantes. Além disso, eles propuseram também uma solução para a implementação deste detector. Nesse algoritmo, os nó enviam a cada período de tempo η uma mensagem de difusão contendo um conjunto *punish*, composto do par (v, q), sendo v a quantidade de vezes que o processo q foi suspeitado. Desse modo, ao receber uma destas mensagens de um processo q, p atualiza o conjunto $punish_p$ com as informações recebidas de q. Além disso, registra o tempo t_q em que a mensagem foi recebida. Se após o tempo η a partir de t_q nenhuma nova mensagem de q for recebida, p remove q do conjunto $punish_p$. Em cada alteração realizada sobre o conjunto *punish*, determina-se um novo líder através da relação $leader_p \leftarrow min\{punish_p\}$.

4.2.4.2 Detecção de Falhas. Para os detectores de falhas, Sens *et al.* [SABG07] mostraram que o detector $\diamondsuit\mathcal{S}$ pode ser implementado sem a necessidade do conhecimento inicial dos participantes, basta que algumas propriedades de comportamento dos processos sejam satisfeitas. Sens *et al.* [SABG07] propuseram ainda uma implementação para o $\diamondsuit\mathcal{S}$ nesse contexto. Nesta proposta, foi utilizado um mecanismo baseado em perguntas e respostas (*Query-Response*) entre os nós.

Para garantir a terminação do protocolo, define-se uma variável d de forma que, se

$|range_p|$ representa o tamanho do conjunto dos vizinhos de um processo p, o valor de d é igual ao menor valor de $|range_p|$ entre os processos de Π, relação representada pela Equação 4.1.

$$d = min(|range_p|), \forall p \in \Pi \tag{4.1}$$

Descreve-se a seguir o algoritmo proposto por Sens *et al.* [SABG07] para implementação do detector. Inicialmente, os nós enviam periodicamente mensagens do tipo QUERY, sendo que estas mensagens são utilizadas para obter o conjunto *known*, um subconjunto de Π. Cada nó aguarda por $d - f$ mensagens do tipo RESPONSE para decidir quais nós estão no conjunto dos suspeitos, formando o conjunto de mensagens de resposta rec_from. Após esta espera, o conjunto de nós suspeitos é incrementado com os nós que pertencem a *known* e não enviaram mensagens RESPONSE, ou seja, não pertencem a rec_from. Além disso, para garantir as propriedades do detector $\diamondsuit\mathcal{S}$, um conjunto *mistake* composto de nós suspeitados erradamente é incluído na mensagem QUERY.

4.3 AVALIAÇÃO DE PROTOCOLOS DE CONSENSO NA PRÁTICA

Existem diversos trabalhos que fazem a avaliação do protocolo de consenso na prática, utilizando ferramentas de simulações definidas sobre modelos realistas. Em um destes trabalhos, Urban *et al.* [UHSK04] realizam uma comparação entre o Paxos e o consenso de Chandra-Toueg [CT96]. A principal métrica utilizada para análise dos resultados foi a latência de execução. A quantidade de processos utilizada foi de $n = 3$ e $n = 7$, com a quantidade de falhas máxima de 3, respeitando a restrição $f = n/2$. Na análise dos resultados para o cenário de execução na ausência de falhas e de falsas suspeitas, o desempenho dos protocolos foi equivalente. A maior diferença entre os protocolos ocorreu nos cenários com maior frequência de falsas suspeitas, onde o desempenho do Paxos foi melhor. A justificativa para esse resultado é que no Paxos existe uma menor quantidade de rodadas em execução concorrente, o que ocasiona em menor contenção (bloqueio) no protocolo. Portanto, Urban *et al.* [UHSK04] concluiu que o Paxos é mais adequado para

ambientes onde falsas suspeitas ocorrem com maior frequência.

Wu *et al.* [WCYR07] apresentam uma avaliação do consenso hierárquico para MANET, descrito na Seção 4.2.3. Neste trabalho, as métricas utilizadas indicam o desempenho em relação ao tempo de execução (latência) e do custo das mensagens. Para os parâmetros específicos da rede MANET, utilizou-se a quantidade de nós n variando de 10 a 100, com o propósito de testar a escalabilidade do protocolo, sendo que tais nós se movem segundo o padrão de mobilidade *Random Waypoint* (descrito na Seção 2.4). A quantidade de falhas injetadas variou de 10% a 50% sobre n. Através da análise dos resultados para o cenário de $f = 10\%$, mostrou-se que, com o uso de um conjunto H maior, aumenta-se a quantidade de rodadas na execução do consenso e diminui-se a latência para a decisão. A quantidade total de mensagens trocadas entre os nós durante as simulações neste cenário não apresentou variações significativas com o aumento de $|H|$. Neste caso, as variações mais significativas ocorreram no aumento da quantidade de nós n. Nas simulações com $f = 50\%$ a quantidade de rodadas e a latência apresentaram variações bruscas, alcançando valores significativamente mais elevados do que para menores valores de f. Portanto, concluiu-se com esta avaliação do consenso hierárquico em [WCYR07] que a quantidade de nós utilizada nos cenários e a quantidade de falhas f afetam significativamente o desempenho do protocolo. Além disso, mostrou-se que, comparado com outros protocolos, o protocolo proposto pode ser mais eficiente do ponto de vista da latência e da complexidade de mensagens.

O protocolo Paxos adaptado ao ambiente de MANET, apresentado na Seção 4.2.1, é avaliado por Borran *et al.* [BPS08] em cenários sem falhas de processos ou de canais de comunicação. As falhas que acontecem nos cenários são ocorrências naturais da interferência e colisões de pacotes nas redes MANET. Cenários interessante testados neste trabalho relacionam-se com a localização do coordenador do protocolo na área onde os nós estão dispostos nas simulações. Foram comparados os resultados entre as simulações com o coordenador em um dos cantos do cenário e com o coordenador no centro do cenário. Para o meio de comunicação sem fio, um nó no canto do cenário produz menos colisões de pacotes mas necessita de mais saltos para a execução do roteamento das mensagens. Para

um nó no centro do cenário, a quantidade de colisões é maior mas o roteamento é executado com mais facilidade. Os resultados das simulações realizadas evidenciaram que a localização do coordenador no cenário não afeta o desempenho do protocolo. Isto se deve as características de colisões e quantidade de saltos necessários no roteamento, que são compensadas tanto para o coordenador no canto como no centro do cenário. Mostrou-se também que, a partir de um determinado nível de perda de mensagens, o desempenho do protocolo decresce significativamente. Quanto à mobilidade, em redes densas, esta é irrelevante para o desempenho do protocolo. Entretanto, em redes mais esparsas, a mobilidade interfere negativamente no desempenho. Borran *et al.* [BPS08] concluiram com esta avaliação que a abordagem do Paxos para MANET pode ser aplicada em redes reais, combinadas com parâmetros similares aos avaliados nas simulações.

4.4 CONCLUSÃO

Este capítulo apresentou o consenso clássico e algumas alternativas para resolver este problema. Foram descritas soluções probabilísticas e determinísticas, sendo que estas últimas incluem o uso de detectores de falhas e algoritmos de eleição de líder.

Posteriormente, o problema do consenso foi estendido para atender aos requisitos existentes no ambiente de redes dinâmicas, como as redes MANET. Foram apresentadas soluções para o consenso nesse modelo que consideram algumas características das redes MANET como a mobilidade, a conectividade parcial e falhas frequentes nos canais de comunicação. Assim como as soluções para o consenso clássico, as soluções para o consenso em redes dinâmicas apresentadas nesse capítulo consideram inicialmente o conhecimento do conjunto de participantes Π. Porém, mostrou-se que existem propostas para detectores de falhas e algoritmos de eleição de líder que não consideram o conhecimento de Π como hipótese. No capítulo seguinte complementam-se os estudos sobre os protocolos de consenso para o ambiente de redes dinâmicas, onde será destacado o problema do consenso com participantes desconhecidos.

CAPÍTULO 5

CONSENSO EM REDES DESCONHECIDAS

Este capítulo detalha o problema do consenso com participantes desconhecidos, conhecido como CUP (*Consensus with Unknown Participants*) [CSS04] e a sua versão tolerante a faltas FT-CUP (*Fault Tolerant Consensus with Unknown Participants*) [CSS05, GT07]. Serão apresentadas duas soluções para o FT-CUP com diferentes requisitos. Uma solução considera os requisitos mínimos de conectividade necessários para a resolução do CUP e encontra quais as condições mínimas de sincronia necessárias [CSS05]. Outra solução incorpora os requisitos mínimos de sincronia do consenso clássico e encontra quais os requisitos minimais de conectividade [GT07].

5.1 CONSENSO COM PARTICIPANTES DESCONHECIDOS E DETECTORES DE PARTICIPAÇÃO

As redes MANET possuem características que são complicadores para a resolução dos problemas fundamentais em sistemas distribuídos, como por exemplo o consenso. A mobilidade, a falta de estrutura da rede e as entradas e saídas deliberadas, faz com que a dinamicidade seja alta, dificultando o conhecimento inicial sobre os participantes da rede. Isto faz com que seja necessário novos conceitos e abstrações que permitam a resolução do consenso em redes dinâmicas. Sendo assim, para todas as soluções apresentadas nesse capítulo, utiliza-se o modelo descrito a seguir.

5.1.1 Modelo para Sistemas com Participantes Desconhecidos

Considera-se um sistema distribuído formado por um conjunto finito Π de $n > 1$ processos, que se comunicam através de canais confiáveis (sem perdas de mensagens) e de

forma assíncrona (sem que sejam feitas hipóteses temporais sobre as ações dos processos e canais). Quando o problema define a possibilidade da existência de falhas (FT-CUP), considera-se que $f < n$ processos podem falhar por parada (*crash*), através de um colapso brusco ou saída deliberada (*switched off*). Considera-se ainda a existência de um protocolo de roteamento confiável tal que, se p_i conhece p_j, p_i é capaz de enviar mensagens para p_j de forma confiável.

5.1.2 Consenso com Participantes Desconhecidos (CUP)

Cavin *et al.* [CSS04] introduzem o problema do consenso em redes MANET, onde os participantes da rede não são conhecidos inicialmente, problema denominado por CUP (*Consensus with Unknown Participants*). As propriedades para tal problema são as mesmas do consenso clássico, descritas na Seção 4.1. Porém, é importante ressaltar que no CUP os processos não conhecem o conjunto de participantes da rede Π, diferentemente do consenso clássico. Além disso, não são consideradas falhas dos processos nessa definição do problema.

Para resolver o CUP, foi necessário definir um novo tipo de oráculo, com objetivo de prover informações parciais sobre os participantes do sistema. Por isso, definiu-se a abstração detectores de participação, detalhada em seguida.

5.1.3 Detectores de Participação

Os detectores de participação (PD) são oráculos distribuídos que fornecem informações sobre os participantes do sistema. Tais detectores surgem para abstrair requisitos de conectividade no consenso em redes desconhecidas e tornar possível a resolução do consenso em um modelo que não considera o conhecimento prévio do conjunto de participantes Π, nem mesmo a sua cardinalidade.

Logo, seja PD_p o detector de participação associado ao processo p, quando consultado

por p, PD_p retorna um subconjunto de processos de Π com os quais ele pode colaborar. Além disso, estes detectores devem satisfazer as seguintes propriedades:

i) *Inclusão da informação.* A informação retornada por PD_p não decresce ao longo do tempo;

ii) *Exatidão da informação.* PD_p não comete erros.

A informação retornada por todos os PD associados aos processos enriquece o sistema com um grafo de conectividade por conhecimento. Este grafo é orientado, pois a relação de conhecimento não é necessariamente bidirecional. Assim, se $q \in PD_p$, então não necessariamente $p \in PD_q$. A partir das características do grafo de conhecimento, algumas classes de detectores de participação foram propostas por [CSS04] para resolução do CUP.

Considera-se o grafo não direcionado $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, onde $\mathcal{V} = \Pi$ e a aresta $(p, q) \in \mathcal{E}$, se $q \in PD_p$ ou $p \in PD_q$. Para o grafo direcionado $\mathcal{G}_{di} = (\mathcal{V}, \mathcal{E})$, onde $\mathcal{V} = \Pi$ e a aresta $(p, q) \in \mathcal{E}$, se $q \in PD_p$. [CSS04] definem quatro classes de detectores a partir desses grafos:

- *Conexo (CO).* O grafo $\mathcal{G}$ é conexo;
- *Fortemente conexo (SCO).* O grafo $\mathcal{G}_{di}$ é fortemente conexo;
- *Completo (FCO).* O grafo $\mathcal{G}_{di}$ é completo, ou seja, para todo $p, q \in \Pi$ existe a aresta $(p, q) \in \mathcal{E}$;
- *Redutível a Único Poço (OSR).* O grafo $\mathcal{G}$ é conexo e a redução de $\mathcal{G}_{di}$ para suas componentes fortemente conexas possui apenas um poço [1] (*sink*).

De acordo com o grau de conectividade do grafo estabelecido pelo detector, o consenso pode ou não ser resolvido. Sendo assim, para resolver o problema do CUP, Cavin *et al.* [CSS04] provaram que:

[1] Um poço (ou sorvedouro) é um vértice (ou nó) com grau de saída 0. Deste modo, uma componente poço não possui arestas com destino em nós de outras componentes.

i) O detector $PD \in CO$ é necessário, mas não é suficiente para resolver o consenso;

ii) $PD \in SCO$ e $PD \in FCO$ são equivalentes e suficientes, mas não necessários;

iii) $PD \in OSR$ é suficiente e necessário, sendo portanto o detector minimal para resolver o CUP.

5.1.4 Fault Tolerant CUP (FT-CUP)

Cavin *et al.* [CSS05] apresentam uma evolução do seu trabalho CUP [CSS04], considerando a ocorrência de falhas nos processos. Esse problema recebeu o nome de FT-CUP (*Fault Tolerant Consensus with Unknown Participants*). Para resolver esse novo problema, foram consideradas as mesmas condições de conectividade minimais identificadas anteriormente para o CUP, representadas pelo detector $PD \in OSR$. Sendo assim, foi provado que o detector de falhas minimal para resolver o FT-CUP nestas condições de conectividade é o detector perfeito ($FD \in$ classe $\mathcal{P}$), definido na Seção 4.1.5.

Para que o protocolo execute corretamente, a propriedade $PD - FD \in OSR$ deve ser satisfeita. Isto significa que, mesmo com falhas em alguns processos, o grafo de conhecimento deve continuar satisfazendo os requisitos do detector OSR. No caso do FT-CUP, as propriedades do detector de falhas são aplicadas sobre o conjunto de processos encontrado pelo detector de participação. Como o detector perfeito não pode ser implementado em um ambiente parcialmente síncrono, a implementação da solução proposta em [CSS05] é difícil de ser garantida no modelo adotado para redes MANET, onde condições de sincronia mais fortes não são adequadas. Além disso, outra desvantagem desta abordagem é que, mesmo considerando detectores perfeitos, o *consenso uniforme* não pode ser resolvido quando $PD \in OSR$. Deste modo, uma nova abordagem para solucionar o FT-CUP foi proposta por Greve e Tixeuil [GT07]. Esta solução será detalhada a seguir.

5.2 FT-CUP COM REQUISITOS MINIMAIS DE SINCRONIA

Greve e Tixeuil [GT07] realizaram uma análise entre os requisitos de conectividade por conhecimento e sincronia necessários para se resolver o FT-CUP. O interesse desta análise é identificar uma solução para o FT-CUP a partir das condições de sincronia minimais para resolver o consenso numa rede clássica, que são representadas pelo detector de falhas $\diamondsuit\mathcal{S}$ (Seção 4.1.5). Portanto, considerando-se esse detector, buscou-se quais os requisitos mínimos relacionados à conectividade do grafo, definido pelo detector de participação, que possibilitariam resolver o consenso. Sendo assim, novas classes de detectores de participação foram propostas:

- *k-Conexo (k-CO).* O grafo $\mathcal{G}$ não direcionado é k-conexo;
- *k-Fortemente Conexo (k-SCO).* O grafo $\mathcal{G}_{di}$ é k-fortemente conexo;
- *k-Redutível a Único Poço (k-OSR).* O grafo $\mathcal{G}$ é conexo. Reduzindo o grafo $\mathcal{G}_{di}$ às componentes k-fortemente conexas, existe apenas uma componente poço. Entre quaisquer pares de componentes existem pelo menos k caminhos disjuntos.

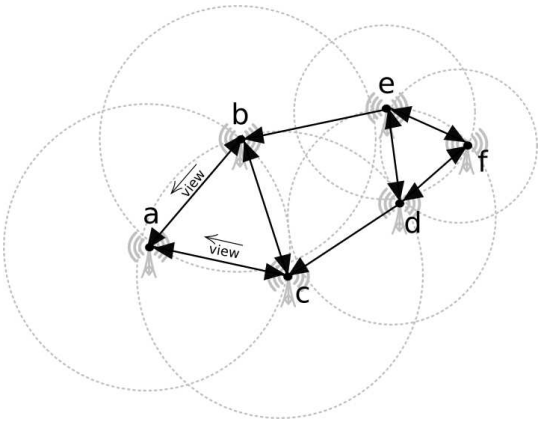
Uma componente G_c de um grafo G_{di} é k-fortemente conexa se, para qualquer par (p, q) de nós em G_c, existem k caminhos disjuntos entre p e q. Uma componente G_s do grafo G_{di} é considerada componente poço quando não existem caminhos partindo de nós em G_s para qualquer outro nó de G_{di}. Sendo assim, um grafo k-OSR formado por apenas uma componente corresponde a um grafo do tipo k-SCO.

Greve e Tixeuil [GT07] demonstram que, nas condições de sincronia estabelecidas através do detector de falhas $\diamondsuit\mathcal{S}$, o detector k-OSR é suficiente e necessário para resolver o consenso, na presença de $f < k < n$ falhas, e apresentam algoritmos que resolvem, não somente o consenso FT-CUP, mas também o FT-CUP uniforme com este detector. Para tanto, propõem-se três algoritmos, COLLECT, SINK e CONSENSUS. Cada um deles é executado em sequência, individualmente por cada processo do sistema.

5.2.1 Algoritmo Collect – Expansão do Conhecimento da Rede

Este algoritmo é usado pelo processo p para expandir o seu conhecimento sobre os participantes da computação, a partir de uma busca no grafo. No início do algoritmo, o processo p consulta o seu detector de participação para obter PD_p. Em seguida, num procedimento de descoberta do grafo, p irá requisitar novas informações de conhecimento (visões) a novos processos identificados, até que nenhuma nova informação possa ser obtida. O algoritmo executa em rodadas. Em cada rodada $r > 0$, p contata todos os nós que ele não conhece e que acabou de tomar conhecimento na rodada $r - 1$. Na rodada 0, p conhece apenas ele mesmo; na rodada 1, ele conhece todos os nós que são retornados pelo seu PD_p; na rodada 2, ele conhece quem os processos em PD_p conhecem e assim sucessivamente, até que não possa mais incrementar o seu conhecimento. Este conhecimento acumulado dos participantes é armazenado em Π_p e para evitar bloqueio, em cada rodada espera-se por $|\Pi_p| - f$ respostas dos nós. Ao término do algoritmo, ele terá armazenado em Π_p o conjunto maximal de participantes que ele pode alcançar no grafo de conhecimento.

A Figura 5.1 representa um exemplo da execução do algoritmo COLLECT, onde o nó a recebe as mensagens que contém as visões dos nós b e c.

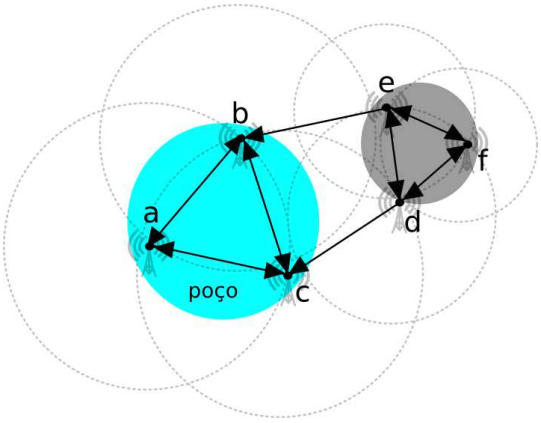


(a)

Figura 5.1. Exemplo de execução do algoritmo COLLECT.

5.2.2 Algoritmo Sink – Determinação da Componente Poço

Através deste algoritmo, cada nó pode saber se pertence a componente poço (definida pelo grafo k-OSR). Nesse algoritmo, cada nó p envia mensagens para todos os nós em Π_p (encontrados pelo COLLECT), visando comparar o seu conjunto com o de cada processo em Π_p. Seja $q \in \Pi_p$ um desses processos. Se q possui o conjunto $\Pi_q = \Pi_p$, então q responde ACK à solicitação de p, senão ele responde NACK. Se $|\Pi_p| - f$ nós respondem com ACKs, o nó p deduz que ele pertencente à componente poço, já que apenas na componente poço os nós possuem a mesma visão do conjunto de processos no sistema. Se o nó p recebe algum NACK, ele deduz que não pertence à componente poço. Note que se o grafo for k-SCO, todos os nós devem ser considerados pertencentes à componente poço, pois esta componente coincide com o próprio grafo do sistema. A Figura 5.2 representa um exemplo da execução do algoritmo SINK, com a formação de duas componentes, sendo uma delas a componente poço.



(a)

Figura 5.2. Exemplo de execução do algoritmo SINK.

5.2.3 Algoritmo Consensus – Execução do Consenso Clássico

Este último algoritmo a ser executado é responsável pela realização do acordo. Inicialmente, cada nó executa o SINK para obter uma visão parcial do sistema e decidir se pertence à componente poço. Ele admite dois comportamentos para os nós. Apenas

os nós identificados como participantes da componente poço executam a fase agreement para obter o consenso. Sendo assim, qualquer consenso subjacente, baseado em algum oráculo (*detector de falhas, eleição de líder* ou *aleatório*), é suficiente para resolver o consenso, desde que exista uma maioria de processos corretos na componente poço [CHT96]. Precisamente, a componente poço deve possuir pelo menos $2f + 1$ nós. Os outros nós (pertencentes as componentes k-fortemente conectas) não participam do consenso. Eles apenas enviam uma mensagem para os processos conhecidos requisitando o valor do consenso e aguardam até que a componente poço obtenha o acordo e e seus nós enviem mensagens de resposta sobre o valor decidido no consenso. A Figura 5.3 ilustra um exemplo da execução do algoritmo CONSENSUS. Encontram-se representados o início da execução (a), com as execuções da fase agreement pela componente poço e da fase request pela outra componente, e o término da execução (b), com os nós da componente poço enviando as mensagens de resposta.

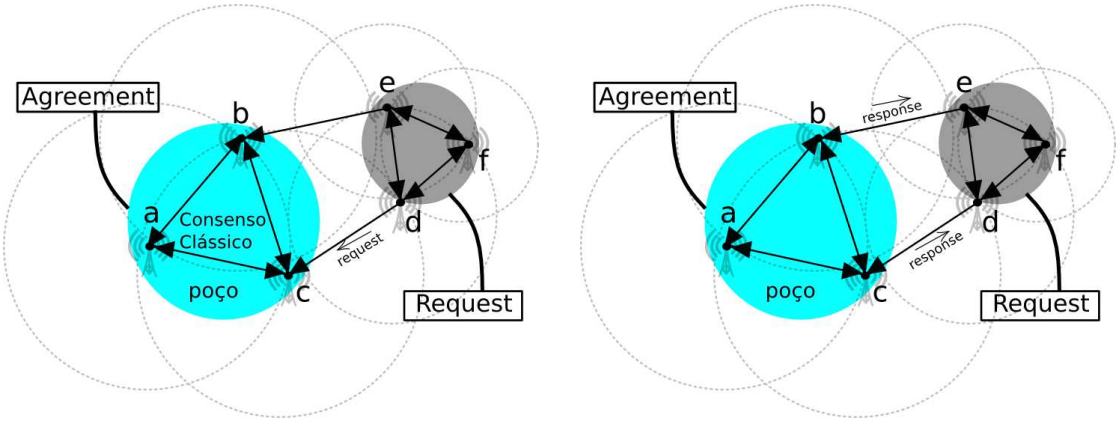


Figura 5.3. Exemplo de execução do algoritmo CONSENSUS, representando a execução do consenso clássico no poço (a) e o envio da resposta para as demais componentes (b).

5.2.4 Restrições e Custo Computacional

As restrições para a convergência da solução de Greve e Tixeuil [GT07] são as seguintes. Devem existir ao menos $2k + 1 < n$ processos na componente poço, e f deve satisfazer $f < k/2$. Tais restrições decorrem das condições para solucionar o consenso

clássico com detectores não-confiáveis, que exigem uma maioria correta de processos no sistema [CHT96].

O custo computacional dos algoritmos (quantidade de mensagens e latência para decidir) depende do grafo de conhecimento $\mathcal{G}_{di}$ formado pelos processos e este depende da topologia da rede. Embora uma análise teórica, no pior caso, possa ser efetuada, ela só terá significado se acompanhada de uma análise real, baseada numa topologia específica de MANET, que este trabalho está se propondo a fazer. Assim, COLLECT tem latência máxima equivalente ao diâmetro do grafo; a quantidade de mensagens será proporcional ao número de nós alcançáveis em $\mathcal{G}_{di}$. Em SINK, esses valores também dependem do número de nós alcançáveis. Logo a complexidade de mensagens fica em $O(n^2)$ para esses algoritmos, sendo n a quantidade de nós na rede. O CONSENSUS tem a sua complexidade dependente do consenso clássico subjacente.

5.3 IMPLEMENTAÇÃO DO CONSENSO FT-CUP SOBRE MANET

Apresenta-se neste capítulo uma proposta de implementação da solução para FT-CUP definida por Greve e Tixeuil [GT07]. Foram definidos módulos representando os algoritmos do FT-CUP e os seus protocolos auxiliares necessários para a implementação em ambiente de MANET. Tal proposta foi realizada com o objetivo de proporcionar sua utilização em um ambiente de simulações, de forma a possibilitar a avaliação do comportamento do protocolo no contexto das redes MANET. Sendo assim, define-se nessa seção as escolhas realizadas para os protocolos utilizados juntamente com o FT-CUP e alguns dos seus parâmetros fundamentais, além do modelo de falhas utilizado para a execução do protocolo.

5.3.1 Módulos do FT-CUP

O modelo para a implementação do FT-CUP descrito a seguir representa o algoritmo proposto por Greve e Tixeuil [GT07], descrito na Seção 5.2. Nesta proposta de

implementação, o FT-CUP foi representado através de uma abordagem modular, onde a comunicação entre os módulos é realizada através de troca de mensagens por interfaces pré-definidas. Os módulos do FT-CUP e suas relações encontram-se representados na Figura 5.4(a). Os seguintes módulos são definidos: PD, COLLECT, SINK e CONSENSUS. Cada um dos módulos representa, respectivamente, os algoritmos *participant detector*, *collect*, *sink* e *underlying classical consensus*. Em uma camada inferior ao FT-CUP na pilha de redes, são encontrados os módulos auxiliares ROUTER e MAC. Estes representam, respectivamente, o algoritmo de roteamento e a camada MAC do protocolo de comunicação [TG01].

Os protocolos selecionados para roteamento e difusão de mensagens são baseados no *Simple Flooding*, descrito na Seção 3.2.1. Em um estudo recente apresentado na Seção 3.3, protocolos de difusão para redes MANET foram analisados e o *Simple Flooding* obteve os melhores resultados para a taxa de entrega das mensagens de difusão [dOCG07]. Neste protocolo, a confiabilidade e a tolerância a faltas são altas devido a redundância intrínseca do *Flooding*. Entretanto, este protocolo pode causar o problema do *broadcast storm problem* [TNCS02], o que pode ocasionar um aumento da latência na difusão. Como o objetivo desse trabalho, através de uma análise experimental, é avaliar mais diretamente o grau de convergência do consenso, preferiu-se adotar o *simple flooding* por propósitos de simplificação e altos níveis de confiabilidade. Caso o objetivo seja o de avaliar mais diretamente a latência média do consenso, pode-se optar pela adoção de outro protocolo de *broadcast* confiável mais eficiente [dOCG07].

A Figura 5.4(b) representa o detalhamento do módulo CONSENSUS, encontrado na Figura 5.4(a). O módulo CONSENSUS_MEDIATOR é responsável por determinar quando um nó é um participante ativo ou passivo do protocolo de consenso clássico. De acordo com o algoritmo do FT-CUP, se o nó é ativo, este pertence à componente poço e faz parte da execução do consenso clássico, de outra forma é considerado passivo, e apenas espera por uma decisão. Para o algoritmo do consenso clássico, foi utilizado o protocolo GENERIC_CONSENSUS proposto por Raynal e Guerraoui [GR04], definido na Seção 4.1.7. Cada um dos oráculos – *leader, random* e *FD (failure detector)* – foi implementado por

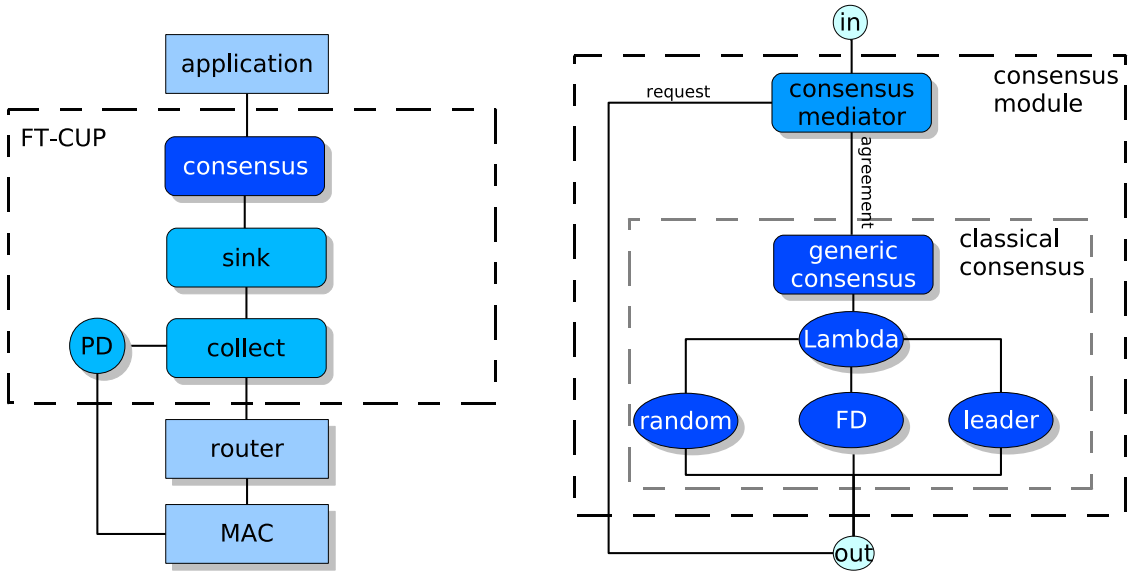


Figura 5.4. Módulos do FT-CUP (a) e específicos do módulo CONSENSUS (b).

algoritmos adequados ao modelo de redes MANET. A exceção é o *random* que consiste apenas em uma função que retorna aleatoriamente um valor 0 ou 1. O módulo RANDOM implementa o oráculo aleatório de Ben Or's de forma trivial [BO83].

O módulo FD implementa um detector de falhas assíncrono para redes móveis e desconhecidas, proposto recentemente por Sens *et al.* [SABG07], descrito na Seção 4.2.4. Este protocolo implementa um detector da classe $\diamondsuit\mathcal{S}$ e não utiliza quaisquer alternativas baseadas em tempo para detectar as falhas. Além disso, o algoritmo não requer conhecimento sobre a composição do sistema e nem mesmo sobre a quantidade de nós existentes. Seu princípio básico é a difusão de informações de suspeitas de falhas sobre a rede, baseados em trocas periódicas de pares de mensagens *query-response* entre processos vizinhos.

O módulo LEADER implementa um algoritmo de eleição de líder proposto por Jimenez *et al.* [JAF06], descrito na Seção 4.2.4. Este algoritmo implementa o detector de líder da classe Ω sem a necessidade do conhecimento prévio sobre os participantes, sendo portanto adequado ao modelo de MANET.

5.3.2 Detectores de Participação

O detector de participação é uma abstração fundamental para a execução correta do FT-CUP. Porém, não existem algoritmos propostos para tais detectores que satisfaçam as propriedades da classe k-OSR. Além disso, não existem garantias e provas teóricas sobre a possibilidade de implementação deste detector no ambiente proposto para o FT-CUP.

Sendo assim, um dos objetivos deste trabalho é avaliar a possibilidade de convergência do consenso mesmo utilizando detectores de participação que não satisfazem necessariamente as propriedades do detector da classe k-OSR. Por isso, propõe-se a utilização de detectores triviais, com a finalidade de testar o FT-CUP em situações adversas. Os detectores propostos a seguir usam o recurso da difusão local inerente aos dispositivos de comunicação sem fio (recurso oferecido pela camada MAC de comunicação) para obter uma visão local constituída pelos vizinhos imediatos ou adicionando ainda os vizinhos de dois ou mais níveis de comunicação. Estas implementações simples foram propostas com o intuito de analisar o FT-CUP em uma rede com um conhecimento restrito sobre os nós.

5.3.2.1 PD-1hop O detector proposto, denominado PD-1hop, retorna para cada processo um conjunto de vizinhos que estão inseridos no seu alcance de transmissão. Como descrito no Algoritmo 2, ele inicia enviando mensagens de broadcast local do tipo *hello*. Os vizinhos recebem estas mensagens *hello* e adicionam o nó emissor em PD_p, o seu conjunto Π estimado. O algoritmo termina quando um tempo pré-definido expira e $|PD_p| \geq k$. Durante este tempo, os nós continuam enviando e recebendo mensagens *hello*. Isso garante que o conjunto PD_p retornado pelo detector é composto de pelo menos k vizinhos.

Uma desvantagem no uso deste detector é que o algoritmo depende do parâmetro de conectividade k, embora isso não garanta a formação de grafos k-SCO ou k-OSR. Sendo assim, em condições de rede onde não se pode satisfazer a condição de k vizinhos para todos os nós, o algoritmo do FT-CUP é bloqueado. Uma outra proposta para este

detector é realizar uma pequena variação no algoritmo, que o torna totalmente temporal. Tal variação consiste em eliminar na linha 11 do algoritmo a dependência do parâmetro k, de forma que a condição seja apenas $|PD_p| \geq timeout$.

Algoritmo 2 PD-1hop

```
 1: PD_p ← ∅
 2:
 3: init:
 4: loop
 5:    send hello(p)
 6:    wait for δ
 7: end loop
 8:
 9: receive a message from node q:
10: PD_p ← PD_p ∪ {q}
11: if |PD_p| ≥ k and timeout then
12:    return PD_p
13: end if
```

5.3.2.2 PD-2hop Uma outra proposta de detector é denominada de PD-2hop. Como evidenciado no Algoritmo 3, o seu funcionamento é similar ao PD-1hop, a diferença básica é que os nós enviam nas mensagens *hello* o seu conjunto Π estimado. Dessa forma, este detector obtém o conhecimento de pelo menos dois níveis (2 *hops*) em relação ao grafo de comunicação. De forma similar ao PD-1hop, alterações na condição da linha 12 do algoritmo podem ser realizadas para torná-lo dependente apenas do tempo δ.

Este detector foi usado nos estudos preliminares e foi possível observar que, para valores de k menores, o desempenho de PD-1hop e PD-2hop é semelhante. Além disso, como o algoritmo COLLECT do FT-CUP tem igualmente o papel de obter o conhecimento mais abrangente sobre a rede, optou-se por não adotar o PD-2hop nos experimentos finais.

Algoritmo 3 PD-2hop

```
1: PD_p ← ∅
2:
3: init:
4: loop
5:    send hello(p, PD_p)
6:    wait for δ
7: end loop
8:
9: receive a message from node q:
10: PD_p ← PD_p ∪ {q}
11: PD_p ← PD_p ∪ PD_q
12: if |PD_p| ≥ k and timeout then
13:    return PD_p
14: end if
```

5.3.3 Modelo de Falhas

Processos falham por parada (*crash*). Porém, no modelo de simulações proposto neste projeto, as falhas ocorrem apenas após os processos executarem o detector de participação. De acordo com o FT-CUP, se um processo falhar antes ou durante a construção do grafo de conhecimento, ele simplesmente não participa do grafo e portanto não é considerado parte do sistema. Por isso, falhas ocorrem apenas após a execução do detector de participação. Esse modelo é utilizado para o consenso com o oráculo RANDOM e o LEADER. Para o consenso com o oráculo FD, define-se uma nova característica sobre o modelo de falhas do sistema. Depois de determinada a componente poço no FT-CUP e durante a execução do consenso clássico, as falhas ocorrem apenas após os processos interagirem pelo menos uma vez com outros processos vizinhos através de mensagens do tipo QUERY. Este é um conceito primário para a corretude do detector de falhas proposto por Sens *et al.* [SABG07], ou seja, para que o processo seja suspeito, é necessário interagir com outros processos para serem reconhecidos por estes. Caso nenhuma mensagem do tipo QUERY seja recebida, nenhum processo jamais será conhecido pelos outros e, por isso, nenhum processo será suspeito.

CAPÍTULO 6

TESTES PRELIMINARES DO FT-CUP

Este capítulo apresenta os resultados dos testes preliminares encontrados no estudo do protocolo FT-CUP através de simulações. Para isso, o FT-CUP foi implementado no ambiente de simulações OMNeT++ [Var01], de acordo com o modelo proposto na Seção 5.3. Serão apresentados em seguida os cenários utilizados nas simulações, descrevendo tanto os parâmetros específicos de redes MANET como os parâmetros necessários ao FT-CUP. As simulações executadas nestes cenários produzem os resultados das métricas selecionadas para análise, com os seus resultados apresentados em forma de gráficos. A análise dos resultados encontra-se subdividida entre métricas relacionadas à execução do FT-CUP até o módulo SINK e as métricas adequadas à reflexão sobre o módulo CONSENSUS. A união destes resultados descreve o desempenho geral do FT-CUP, sendo que a análise dos resultados dos testes preliminares é finalizada com a apresentação das conclusões obtidas.

6.1 CENÁRIOS DE TESTES

Definem-se a seguir os parâmetros dos cenários necessários para definição dos conjuntos de simulações que serão executadas. Estes parâmetros são de fundamental importância pois interferem diretamente no desempenho do FT-CUP. Por isso, a definição dos cenários foi baseada em trabalhos relacionados [Urb03, dOCG07] e em reflexões e críticas realizadas sobre trabalhos de simulações para protocolos no ambiente de MANET [HBG06, KCC05]. Tais observações sobre simulações em MANET encontram-se melhor explicadas na Seção 2.4.

Tabela 6.1. Parâmetros de simulação para os testes preliminares

Parâmetros de simulação	
Simulador	OMNeT++ (3.4)
Quantidade de nós	10 à 50
Área	300 x 300 m^2
Alcance *wireless*	100 m
Tempo de simulação	20 s
Repetições	10
Padrão de Mobilidade	Random Waypoint
Velocidade dos nós	de 20 à 50 m/s
Tempo de pausa	até $2s$

6.1.1 Ambiente MANET

A Tabela 6.1 descreve os parâmetros gerais utilizados nos cenários das simulações. Para a realização das simulações, os nós foram dispostos em um cenário com *área* de 300x300m^2, sendo a *quantidade de nós* entre 10 e 50. Cada nó possui um dispositivo de comunicação com um *alcance de transmissão* constante e com um raio de 100m. Estes valores foram selecionados com o objetivo de obter uma densidade razoável para a execução do protocolo. Sendo assim, calcula-se a quantidade média de vizinhos de cada nó pertencente a rede segundo a Equação 2.1, que depende da quantidade de nós, da área e do alcance de transmissão. Por exemplo, para uma rede com 10 nós, a quantidade média esperada de nós vizinhos é de 3.49. Para 50 nós, a média é de 17.45 vizinhos. Foram realizados testes com uma rede mais esparsa, em cenários com áreas maiores. Entretanto, nestes cenários a taxa de convergência do consenso foi muito baixa, consequência da baixa conectividade e dos frequentes particionamentos ocorridos nesta rede mais esparsa. Os valores escolhidos para as simulações apresentadas nesta seção possibilitaram uma relação de conectividade entre os nós suficiente para evitar o particionamento da rede.

Cada simulação foi executada por um *tempo de simulação* de 20 segundos, o que correspondeu a maior latência do FT-CUP encontrada nas simulações preliminares, utilizadas para definição adequada do modelo. O *modelo de mobilidade* utilizado foi o *Random Waypoint* com a *velocidade* variando no intervalo de 20 a 50 m/s e *tempo de pausa* até

$2s$. Esta velocidade foi selecionada para que, devido à densidade, a interferência da mobilidade na convergência do FT-CUP não seja relevante, de forma a reduzir a possibilidade de particionamentos. Isso possibilita o protocolo de roteamento e de difusão a executar corretamente e entregar de forma confiável as mensagens. Entretanto, esta velocidade interfere na latência do protocolo, pois um pode mover-se para fora do alcance de transmissão de outro, aumentando o tempo de entrega de mensagens pela camada de roteamento.

O ambiente de simulações utilizado para implementação e testes do protocolo FT-CUP foi o OMNeT++ (versão 3.4) [Var01], com a utilização do *Mobilty Framework* [DSH+03] como simulador do ambiente de redes MANET. Maiores detalhes sobre este ambiente de simulações utilizado estão descritos na Seção 2.4. O *Mobilty Framework* possui algumas alternativas para simular as camadas físicas e de enlace para os nós. Nos testes preliminares, foi utilizado para estas camadas um módulo denominado *CoreTestNic*. Tal módulo não implementa camadas tão realistas quanto as definidas no modelo 802.11 [CWKS97], sendo que questões como ruído e perda de mensagens não são consideradas. Por isso, considera-se que as simulações apresentadas nesta seção são executadas em condições perfeitas em relação aos canais de comunicação. Nos testes finais, apresentados na seção seguinte, optou-se por utilizar a implementação das camadas do padrão 802.11 para simular ambientes de aplicações reais de redes MANET.

6.1.2 Ambiente FT-CUP

A solução proposta por Greve e Tixeuil [GT07] para o FT-CUP requer a definição prévia de alguns parâmetros. Os principais parâmetros são: k (*conectividade do grafo*), f (*quantidade máxima de falhas*) e n (*quantidade total de nós na rede*). Seguindo a restrição $f < k < n$, estes parâmetros foram combinados com o objetivo de analizar o seu impacto na execução do FT-CUP e determinar os melhores valores para a convergência do consenso. Para possibilitar a injeção de falhas na rede, define-se ainda o parâmetro Pf, uma porcentagem sobre as falhas f e que determina quantos nós efetivamente falham

por parada durante a execução.

Os valores selecionados para k foram: $k = n/5$ e $k = n/10$, correspondendo a 20% e 10% sobre n, respectivamente. Considera-se que estes valores caracterizam uma rede com baixa e média conectividade, respectivamente. Note que o valor máximo de k é $(k = n - 1)$. Porém, este valor não é utilizado por requisitar a existência de um grafo fortemente conexo, correspondendo a uma rede com conectividade total, como definido para o modelo clássico na Seção 4.1.1. Portanto, neste cenário o problema do consenso possui características similares ao consenso no modelo clássico, o que não corresponde aos objetivos deste trabalho. O parâmetro f foi fixado em $(f = k - 1)$. Isto corresponde ao máximo de falhas toleradas (de acordo com a relação $f < k < n$). A porcentagem de falhas Pf foi definida como: $Pf = 0$ (0% de falhas), $Pf = (k - 1)/2$ (50% de falhas) e $Pf = k - 1$ (100% de falhas).

Os detectores usados nas simulações foram o PD-1hop e PD-2hop, definidos na Seção 5.3.1, com a utilização de *timeout* de $100ms$. A maioria dos resultados apresentados nesta seção foram obtidos com a utilização do PD-1hop. O PD-2hop foi usado apenas para efeito de comparação do resultado entre os detectores. Por isso, apenas nestes gráficos de comparação encontra-se a indicação de qual detector foi utilizado. Nos demais gráficos usou-se o detector PD-1hop, embora esta informação esteja omitida.

A combinação dos parâmetros k e Pf define conjuntos de simulações. Cada simulação foi repetida por 20 vezes, usando um intervalo de confiança de 95% para a apresentação dos resultados, calculados de acordo com a Equação 2.2. São apresentados nos gráficos a seguir os valores médios calculados nestas repetições.

6.2 ANÁLISE DA CONVERGÊNCIA DO FT-CUP

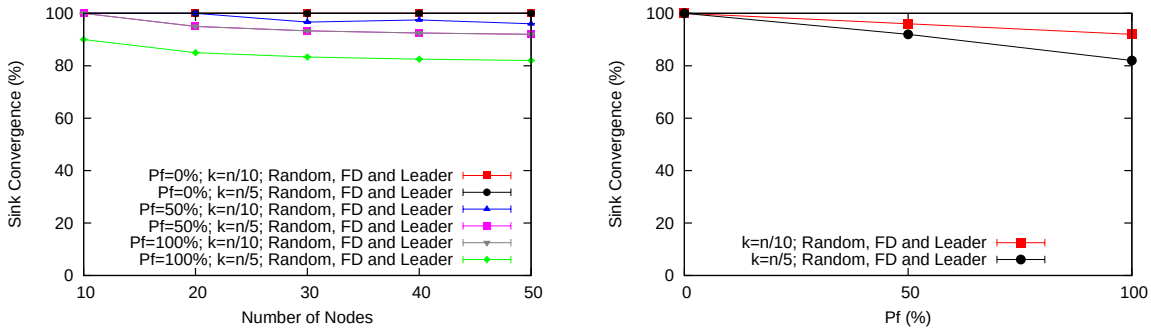
Nesta seção são apresentados os resultados para as métricas relacionadas ao desempenho do FT-CUP, com o foco em mostrar o desempenho dos seus algoritmos específicos, a saber PD, COLLECT e SINK. Portanto, para a análise dos resultados, foram definidas

as métricas:

- *Convergência para poço.* Representa a porcentagem dos nós que fazem parte do poço e que conseguem determinar que estão na componente poço;
- *Latência do* SINK. Indica a média dos tempos para execução do algoritmo SINK.

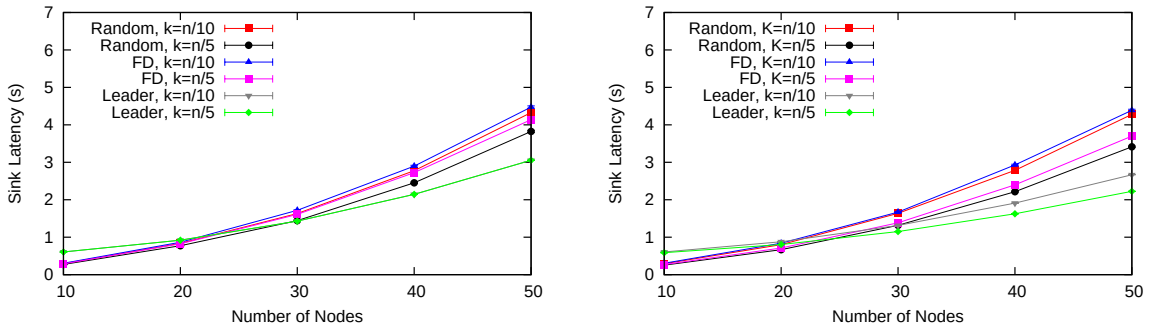
6.2.1 Convergência para Poço

A Figura 6.1 representa os resultados da convergência para poço, ou seja, a porcentagem de nós identificados como pertencentes a componente poço. O desempenho de cada oráculo Random, FD and Leader não interfere nos resultados desta métrica e são apresentados indistintamente nos gráficos da figura. Para cada oráculo utilizado, foram executados dois conjuntos de simulações com os parâmetros $k = n/10$ e $k = n/5$. Os gráficos na Figura 6.1 representam os resultados desta métrica para cenários sem falhas $Pf = 0\%$ e cenários com a presença de falhas dos nós, nos valores de $Pf = 50\%$ e $Pf = 100\%$. Para cenários sem falhas $Pf = 0\%$, a convergência foi total (100%) em todos os casos simulados, ou seja, todos os nós foram caracterizados como pertencentes a componente poço, o que define uma formação do grafo de conhecimento composto de apenas uma componente. As execuções com $Pf = 50\%$ e $Pf = 100\%$ representam os cenários com falhas por parada dos nós. Os resultados evidenciam uma forte dependência com o valor de k utilizado. A convergência para $k = n/10$ é maior do que com $k = n/5$. Isso ocorre porque a capacidade de tolerar falhas é inferior para um menor valor de k, uma vez que a quantidade de falhas tolerável é definida por $f = k - 1$. Além disso, como esperado, pode-se observar que a convergência para poço decresce com o aumento da porcentagem de falhas, relação melhor observada através do gráfico 6.1(b).



(a) $Pf = 0\%$, 50% e 100%; $k = n/10$ e $k = n/5$ (b) 50 nós

Figura 6.1. Convergência para poço com $Pf = 0\%$ (0 falhas), $Pf = 50\%$ ($(k-1)/2$ falhas) e $Pf = 100\%$ ($k-1$ falhas) com $k = n/10$ e $k = n/5$ em função da quantidade de nós (a) e para 50 nós em função de Pf (b).



(a) $Pf = 0\%$ (0 falhas) (b) $Pf = 100\%$ ($k-1$ falhas)

Figura 6.2. Latência do SINK para $Pf = 0\%$ (a) e $Pf = 100\%$ (b).

6.2.2 Latência do sink

A Figura 6.2 representa os resultados obtidos nas simulações para a latência do algoritmo SINK. Os gráficos (a) e (b) representam os cenários livre de falhas $Pf = 0\%$ e com falhas $Pf = 100\%$, respectivamente. O cenário intermediário com a quantidade de falhas de $Pf = 50\%$ foi omitido devido a similaridade com os outros resultados ($Pf = 0\%$ e $Pf = 100\%$). Como esperado, a latência aumenta com o aumento da quantidade de nós na rede. Isto ocorre devido à alta complexidade de mensagens associada com os algoritmos envolvidos na resolução do FT-CUP (COLLECT, SINK e CONSENSUS). Tal complexidade é, para o pior caso, de $O(n^2)$. Porém, a latência diminui com o aumento da porcentagem de falhas Pf, justamente devido a existência de menos nós envolvidos

na execução dos algoritmos do FT-CUP. Pode-se observar ainda que a latência é menor para um valor de k maior ($k = n/5$). A variação na latência do SINK existente na comparação entre Random, FD e Leader deve-se as mensagens acrescentadas pelos oráculos (FD e Leader) e pelas características da execução do consenso clássico associado a eles. Um consenso clássico terminado em menos tempo impede que eventuais nós que ainda não chegaram na execução do módulo CONSENSUS terminem a execução do SINK, uma vez que a execução da simulação é concluída antes disso ocorrer, o que leva a estas variações vistas na latência.

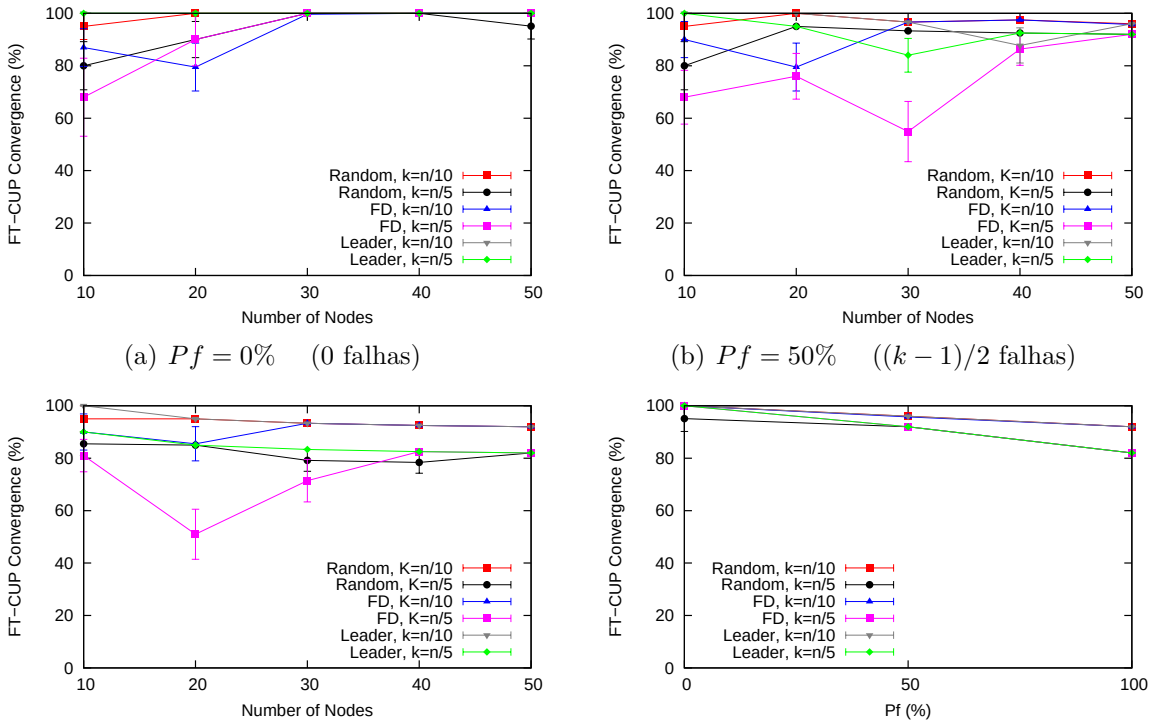
6.3 ANÁLISE DO DESEMPENHO DO FT-CUP COM OS ORÁCULOS

Na seção anterior, foram apresentados os resultados para as simulações do FT-CUP até a execução do seu algoritmo SINK. Nesta seção, serão apresentados os resultados do FT-CUP até o término da sua execução, ou seja, até a execução do algoritmo CONSENSUS, o módulo que utiliza o consenso genérico de Guerraoui e Raynal [GR04] para permitir o uso de três variações para o consenso clássico, conforme descrito no modelo apresentado na Seção 5.3. Sendo assim, para a análise dos resultados, foram selecionadas as métricas:

- *Convergência do FT-CUP.* Refere-se à porcentagem de nós que terminam o consenso com uma decisão;
- *Latência do FT-CUP.* Mostra a média dos tempos gastos para a convergência do FT-CUP.

6.3.1 Convergência do FT-CUP

A Figura 6.3 representa a porcentagem de nós que obtém uma decisão no consenso FT-CUP, ou seja, aqueles nós que terminaram a execução do consenso com a obtenção de um valor para a decisão. Mesmo em cenário sem falhas, representado através do gráfico 6.3(a), é possível observar que nem todos os protocolos alcançam a taxa de 100% de



(a) $Pf = 0\%$ (0 falhas)

(b) $Pf = 50\%$ ($(k-1)/2$ falhas)

(c) $Pf = 100\%$ ($k-1$ falhas)

(d) 50 nós

Figura 6.3. Convergência do FT-CUP com $Pf = 0\%$ (a), $Pf = 50\%$ (b), $Pf = 100\%$ (c) em função da quantidade de nós e para 50 nós (d) em função de Pf.

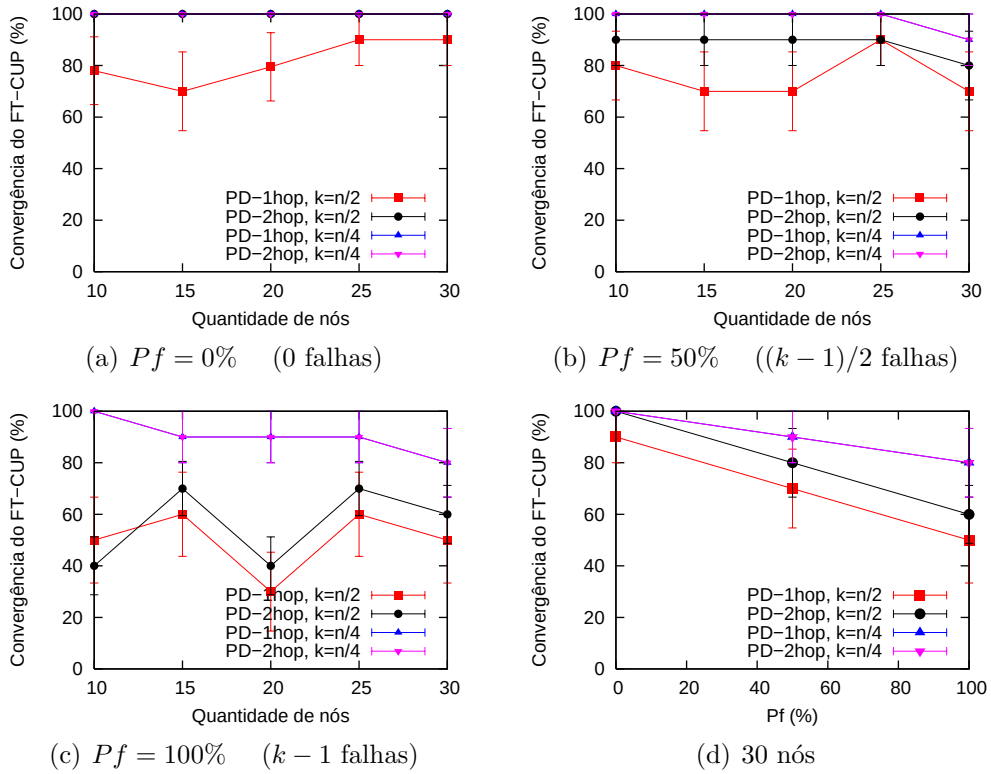
convergência. Neste cenário, o FT-CUP com o consenso clássico utilizando o Leader obteve os melhores resultados. Em quase todas as execuções, ele obteve 100% de convergência, para $k = n/10$ e $k = n/5$. Para os demais protocolos, observa-se que o desempenho é melhor com o aumento da densidade da rede, chegando a obter 100% de convergência entre 30 e 40 nós. Com 50 nós, para o Random com $k = n/5$, a convergência decresce para 95%. Isso provavelmente ocorre devido a uma maior quantidade de nós participando da execução do consenso clássico e as características do consenso aleatório de possuir a probabilidade de terminação associada a quantidade de nós na rede.

As Figuras (b) e (c) representam os resultados de convergência para $Pf = 50\%$ e $Pf = 100\%$, respectivamente. Como esperado, essa convergência diminui com o aumento da porcentagem de falhas Pf. Esta relação pode ser melhor observada no gráfico (d). Para $k = n/10$, os oráculos Leader e Random tiveram desempenho similar e em geral melhor do que o oráculo FD. A variação do FT-CUP com FD e $k = n/5$ apresentou

os piores resultados para essa métrica. A implementação do detector não-confiável FD para MANET [SABG07] (Seção 4.2.4) produz uma alta quantidade de falsas suspeitas, devido a questões de mobilidade dos nós. Além disso, o protocolo é assíncrono e pode ser bloqueado uma vez que a combinação de parâmetros utilizados não correspondem com as características dos cenários usados nas simulações. Por exemplo, existe uma condição de espera por $d - f$ mensagens, onde d representa o grau mínimo de nós na rede, como definido na Equação 4.1. Os parâmetros d e f são constantes definidas como requisitos para a execução do algoritmo, sendo que se seus valores não são suficientes, a execução do protocolo é bloqueada. Sendo assim, o oráculo FD possui características que podem complicar e comprometer a terminação do consenso.

6.3.1.1 Comparação entre PD1-hop e PD-2hop. Em simulações anteriores, o FT-CUP foi testado também com o detector de participação PD-2hop. A Figura 6.4 representa os resultados destas simulações para $k = n/2$ e $k = n/4$ com a quantidade de nós variando de 10 à 30 nós. Nestas simulações, apenas o consenso com oráculo Random foi utilizado. Os demais parâmetros para os cenários usados nestas simulações são os mesmos definidos na Seção 6.1.

Através do gráfico 6.4(a), observa-se que, mesmo na ausência de falhas, para $k = n/2$ e PD-1hop, a convergência oscilou em torno de 80%. Conclui-se que, para ambientes com maiores necessidades de conectividade ($k = n/2$), deve ser usado um detector mais forte, que oferece mais garantias sobre as propriedades da classe a que pertence, de forma a melhorar a convergência do FT-CUP. Para $k = n/4$, o desempenho foi similar para ambos os detectores, ficando exatamente em 100% de convergência na ausência de falhas. A causa disto é o bom desempenho na convergência para poço, com resultados similares aos gráficos da Figura 6.1, reforçando a hipótese de que, para uma conectividade menor, um detector simples possa ser utilizado. No gráfico 6.4(b) são mostrados os resultados para $Pf = 50\%$. Nesse gráfico, a convergência cai para 90% com $k = n/2$ e PD-2hop, mantendo-se em 100% para $k = n/4$ até 25 nós. Para $k = n/2$ e PD-1hop, a convergência oscila em torno de 70%. Apenas no gráfico 6.4(c), onde $Pf = 100\%$, a



(a) $Pf = 0\%$ (0 falhas) (b) $Pf = 50\%$ ($(k-1)/2$ falhas)

(c) $Pf = 100\%$ ($k-1$ falhas) (d) 30 nós

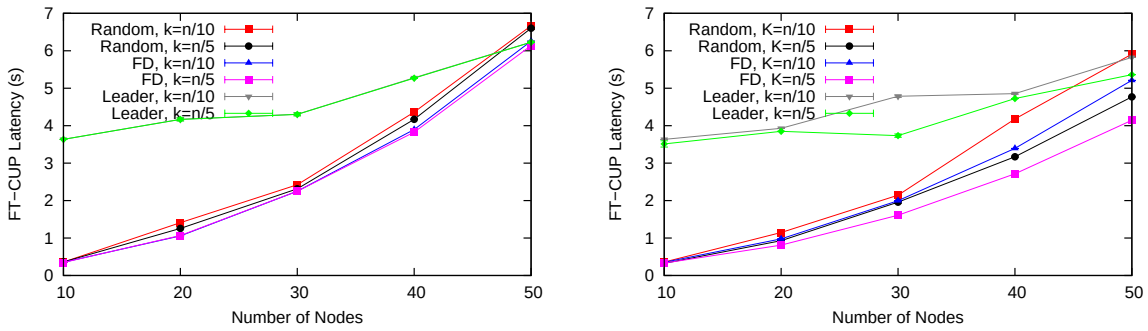
Figura 6.4. Convergência do FT-CUP com $Pf = 0\%$ (a), $Pf = 50\%$ (b), $Pf = 100\%$ (c) em função da quantidade de nós e para 30 nós (d) em função da quantidade de falhas.

taxa de convergência para $k = n/4$ diminui significativamente, assim como para $k = n/2$. O gráfico 6.4(d) representa a taxa de convergência em função da porcentagem de falhas, levando-se em consideração 30 nós da rede. Por esse gráfico pode-se observar melhor a dependência da convergência com o parâmetro k e o detector de participação utilizado.

6.3.2 Latência do FT-CUP

A Figura 6.5 representa os resultados de latência obtidos nas simulações, contabilizando os tempos desde a execução do detector de participação até o término do FT-CUP. Serão apresentados dois cenários, execuções na ausência de falhas ($Pf = 0\%$) e com falhas ($Pf = 100\%$). Os resultados para o cenário intermediário ($Pf = 50\%$) foram omitidos devido as similaridades com os outros ($Pf = 0\%$ e $Pf = 100\%$).

Com a exceção do consenso com Leader, o comportamento da latência do FT-CUP e



(a) $Pf = 0\%$ (0 falhas) (b) $Pf = 100\%$ ($k - 1$ falhas)

Figura 6.5. Latência do FT-CUP $Pf = 0\%$ (a) e $Pf = 100\%$ (b)

seus oráculos foi similar aos resultados apresentados para latência do SINK (Figura 6.5). A latência do FT-CUP com os oráculos FD e Random cresce rapidamente com o aumento da quantidade de nós na rede. Porém, a latência com Leader cresce gradualmente, alcançando resultados similares aos outros oráculos apenas com 50 nós. Os resultados indicam que o consenso com Leader obtém um comportamento melhor para redes mais densas. Para redes mais esparsas, nos resultados obtidos nas execuções com quantidade de nós entre 10 e 40, a latência do Leader é consideravelmente maior do que no consenso com FD e Random. Questões como mobilidade e conectividade parcial da rede podem ocasionar modificações constantes na definição do líder, aumentando o tempo gasto na obtenção do consenso. O consenso FT-CUP com FD obteve os melhores resultados nesta métrica. Na abordagem determinística, quando o oráculo associado é executado de forma correta, a terminação do consenso é garantida com o requisito de apenas a maioria dos processos corretos. Para o oráculo aleatório, o consenso pode não terminar ou até executar muitas rodadas até obter a decisão, uma influência das propriedades probabilísticas do consenso aleatório.

6.4 LIÇÕES APRENDIDAS

Serão listados em seguida resultados obtidos com as simulações preliminares realizadas neste capítulo. Estas observações definem condições e características de redes MANET

onde o consenso FT-CUP pode ser resolvido e o comportamento do protocolo nos cenários estudados, segundo o modelo inicialmente adotado para este trabalho.

i) A escolha do detector de participação e do parâmetro k é essencial para a convergência do FT-CUP;

ii) As simulações evidenciaram que, em MANET, mesmo um detector simples como o PD-1hop pode gerar um grafo de conhecimento suficiente e que se aproxima dos requisitos teóricos para resolver o FT-CUP;

iii) Na existência de requisitos de conectividade por conhecimento maiores (maior valor de k) e, como consequência, para tolerar maiores quantidades de falhas, um detector de participação com premissas mais fortes (como o PD-2hop) deve ser utilizado para a obtenção de melhores resultados;

iv) Em cenários sem falhas, a taxa de convergência foi de 100% e alcançou 80% nos cenários com maior quantidade de falhas;

v) Um menor requisito para a conectividade por conhecimento, como a utilização de $k = n/10$, apresenta resultados melhores em comparação com valores maiores como $k = n/5$;

vi) O consenso com detector de falhas apresentou a melhor latência porém, possui as piores taxas de convergência do protocolo;

vii) Inversamente, o consenso com detector de líder apresentou a pior latência e as melhores taxas de convergência. Isso aconteceu em geral nas redes mais esparsas (< 50 nós). Apesar das simulações estarem limitadas para até 50 nós, para uma rede mais densa (≥ 50), os gráficos indicam uma tendência para que as latências sejam menores do que os valores dos outros oráculos;

viii) O consenso com oráculo aleatório obteve convergência similar ao do detector de líder e latência similar ao consenso com detector de falhas.

As considerações obtidas através das simulações deste capítulo são importantes para o entendimento inicial do protocolo e para a aplicação de novos testes mais direcionados ao FT-CUP, de forma a proporcionar uma maior variação de cenários, inclusive aqueles que simulam condições adversas de conectividade e falhas. Sendo assim, realizou-se uma reflexão sobre o modelo de simulações e as métricas utilizadas neste capítulo que ocasionou em definições e simulações mais realistas, com novos resultados, apresentados no Capítulo 7.

CAPÍTULO 7

AVALIAÇÃO FINAL DO DESEMPENHO DO FT-CUP

Esse capítulo tem como objetivo realizar uma análise de desempenho da solução proposta por Greve e Tixeuil [GT07] para o FT-CUP, através de simulações em cenários realistas definidos por um modelo adequado e com características de redes dinâmicas, como as redes MANET (Seção 2.1). Assim como no capítulo anterior, o FT-CUP foi implementado no ambiente de simulações OMNeT++ [Var01], de acordo com o modelo proposto na Seção 5.3. No Capítulo 6, foram apresentados os resultados obtidos em estudos preliminares, realizados sobre a proposta de um modelo inicial para a execução das simulações. Baseado no entendimento do comportamento do protocolo no ambiente previamente proposto e em críticas sobre o modelo utilizado, realizou-se uma reflexão sobre o modelo de simulações e foram propostas melhorias, com o intuito de se aproximar das características reais das redes MANET e possibilitar uma análise mais profunda dos protocolos utilizados no FT-CUP. Com esse novo modelo, novas simulações foram executadas, o que proporcionou resultados mais completos e realistas sobre o problema e a obtenção de parâmetros e características da resolução do FT-CUP em redes MANET.

7.1 CENÁRIOS DAS SIMULAÇÕES

De forma semelhante à definição dos parâmetros para os cenários dos testes preliminares na Seção 6.1, definem-se a seguir os parâmetros necessários para as execuções das simulações deste trabalho. Tais parâmetros foram revisados e adequados ao modelo realista de redes MANET, com uma maior abrangência de situações e cenários possíveis. Assim como na Seção 6.1, os trabalhos relacionados [Urb03, dOCG07] e críticas sobre trabalhos de simulações para protocolos no ambiente de MANET [HBG06, KCC05] (Seção 2.4 tiveram influência na definição dos cenários.

Tabela 7.1. Parâmetros de simulação

Parâmetros de simulação	
Simulador	OMNeT++ (4.0)
Quantidade de nós	10 à 50
Área	300x300, 400x400 e 500x500 m^2
Alcance *wireless*	$125m$ e aleatório ($25m$, $50m$, $125m$, $250m$)
Tempo de simulação	50 s
Repetições	30
Padrão de Mobilidade	Random Waypoint
Velocidade dos nós	de 0 à 10 m/s
Tempo de pausa	até $2s$

7.1.1 Ambiente da MANET

A Tabela 7.1 descreve os parâmetros gerais utilizados nas simulações. A *quantidade de nós*, os valores para a *área* e *alcance do dispositivo de comunicação wireless* foram selecionados visando definir a densidade da rede. Para a realização das simulações, os nós foram dispostos em um cenário com um determinado valor de *área*, sendo a *quantidade de nós* entre 10 e 50. Para definir a quantidade média de vizinhos de cada nó pertencente a rede, utilizou-se a Equação 2.1, que tem como dependência a quantidade de nós, a área e o alcance de transmissão. Portanto, pode-se dizer que a densidade da rede está fortemente relacionada ao valor médio de vizinhos por nó. Sendo assim, define-se neste capítulo algumas variações para a densidade da rede, com a utilização do valor da área do cenário como o parâmetro que caracteriza uma determinada densidade. Como constatado nos testes preliminares do Capítulo 6, em cenários de rede muito esparsa, o consenso não apresentou boas taxas de convergência, dado que frequentemente ocorria o particionamento da rede. Porém, é possível investigar o desempenho do FT-CUP em redes mais esparsas do que as redes utilizadas nas simulações anteriores, visto que, a partir de determinada densidade, obtém-se boas taxas de convergência.

Foram selecionados três valores de densidade: esparsa (área de 500x500m^2), normal (área de 400x400m^2) e densa (área de 300x300m^2). Inicialmente, definiu-se a utilização do *raio de alcance de transmissão* fixado no valor de $125m$. Porém, como ca-

racterizado na Seção 2.1, as redes MANET apresentam frequentemente conexões assimétricas [KM07, KNG+04]. Sendo assim, para simular cenários onde possam ocorrer conexões unidirecionais no grafo de conhecimento gerado pelo detector de participação, define-se um cenário onde os nós têm alcances de transmissão diferentes, o que naturalmente proporciona o surgimento destas conexões. Neste cenário, denominado como rede híbrida, atribui-se aleatoriamente para cada nó um valor de alcance de transmissão, dentre os valores previamente selecionados: $25m$, $50m$, $125m$ e $250m$. Com isso, o alcance médio de transmissão calculado na rede híbrida é de $112.5m$. Para esta rede, foi utilizado apenas a área de 500x500m^2, a mesma área da rede esparsa. Esses valores selecionados para a área e alcance de transmissão permitem uma conectividade entre os nós suficiente para evitar o particionamento frequente.

Apresenta-se a seguir os resultados esperados para a quantidade média de vizinhos por nó em alguns dos cenários utilizados neste trabalho, calculados através da Equação 2.1. No cenário de rede densa, para 10 nós, espera-se que cada nó tenha em média 5.45 vizinhos. Para 50 nós, neste mesmo cenário, a média é de 27.27 vizinhos. Para a rede normal, com 10 nós, espera-se que cada nó tenha em média 3.07 vizinhos. Com 50 nós, a média é de 15.34 vizinhos. Na rede esparsa, para 10 nós, espera-se que cada nó tenha em média 1.96 vizinhos. Para 50 nós, a média é de 9.82 vizinhos. Para a rede híbrida calculou-se que, com 10 nós, espera-se uma média de 1.59 vizinhos. Para 50 nós, a média é de 7.95 vizinhos.

O *tempo de duração* definido para cada simulação é de 50 segundos, tempo suficiente para que houvesse uma convergência para um valor de decisão. O *padrão de mobilidade* usado é o *Random Waypoint* com a *velocidade* variando de 0 à 10 m/s e *tempo de pausa* de até $2s$. Esta velocidade foi selecionada com o intuito de minimizar a interferência da mobilidade na convergência do consenso.

7.1.2 Ambiente do FT-CUP

Para os parâmetros específicos do FT-CUP, f (*máximo de falhas tolerável*) e Pf (porcentagem real de falhas sobre f), considera-se uma combinação de seus valores, a fim de analisar o impacto no FT-CUP e determinar as melhores configurações possíveis para o consenso. Dado que a verdadeira conectividade da rede, em termos de conhecimento, não pode ser pré-definida, escolheu-se não fixar um valor para k, como foi feito nos testes preliminares apresentados no Capítulo 6. Nas simulações, utiliza-se apenas uma quantidade f de falhas na rede, correspondente a uma porcentagem dos nós que podem falhar. O valor de k é atribuído pelo detector de participação, com a obtenção do seu valor através da cardinalidade do conjunto de participantes ($|PD|$) retornado por cada nó. Portanto, o valor de k é obtido de forma dinâmica e depende dos parâmetros definidos para a rede MANET, como a área, o alcance do dispositivo de comunicação e a quantidade de nós. Espera-se nesse caso, que $f < k$, embora tal relação não possa ser garantida.

O parâmetro f é definido como uma porcentagem sobre n. Este parâmetro é fornecido como entrada para a execução das simulações, visto que os algoritmos do FT-CUP dependem dele para executar corretamente. Os valores considerados foram as porcentagens 0%, 30% e 50% em relação à n. Sendo assim, existem cenários que toleram uma quantidade de falhas que obedece os valores $f = 0.1$, $f = 0.3$ e $f = 0.5$ em relação à n. O Pf foi definido como um percentual em relação à f. Os valores considerados foram de 0%, 50% e 100% em relação à f, fazendo com que existam cenários sem falhas ($Pf = 0$), com metade das falhas possíveis ($Pf = 0.5$) e com o máximo de falhas tolerável ($Pf = 1$). O detector utilizado nas simulações é o PD1-hop, definido na Seção 5.3.1, sendo usado um *timeout* de $2s$. Decidiu-se não utilizar o detector PD2-hop nestas simulações pois tal detector tem características redundantes ao algoritmo COLLECT e o objetivo destas simulações é avaliar o FT-CUP nas condições fracas em relação à conectividade por conhecimento.

A combinação dos parâmetros f, Pf e da densidade da rede definem conjuntos de simulações. Cada conjunto é simulado por 30 vezes, sendo que o que se vê nos gráficos são os valores médios obtidos através dessas repetições e o seu intervalo de confiança asso-

ciado, calculado atráves da Equação 2.2, com a utilização de um coeficiente de confiança de 95%.

7.2 ANÁLISE DA CONVERGÊNCIA DO FT-CUP

Nesta seção são apresentados os resultados para as métricas relacionadas ao desempenho do FT-CUP, com o foco em mostrar o desempenho dos seus algoritmos específicos, a saber PD, COLLECT e SINK. Sendo assim, os resultados apresentados nesta seção são independentes dos oráculos utilizados no consenso clássico do módulo CONSENSUS e os gráficos mostrados representam os resultados para qualquer um dos oráculos utilizados. Para a análise dos resultados, foram definidas as métricas:

- *Participantes Detectados*. Refere-se à porcentagem média dos nós encontrados pelos detectores de participação associados a cada nó;
- *Participantes Coletados*. Representa a quantidade de nós coletados durante a execução do algoritmo COLLECT;
- *Convergência para poço*. Representa a porcentagem dos nós que fazem parte do poço e que conseguem determinar que estão na componente poço;
- *Latência do* SINK. Indica a média dos tempos para execução do algoritmo SINK.

7.2.1 Detector de Participação

A Figura 7.1 representa os resultados de simulação para a métrica *participantes detectados*. Esta traduz efetivamente os valores de média de vizinhos esperados apresentado na Seção 7.1 e calculados segundo as considerações realizadas na Seção 2.4.2. A variação em relação a esses valores é consequência da mobilidade dos nós. Esta mobilidade interfere no conhecimento dos nós, uma vez que novos vizinhos podem se mover para o alcance de transmissão de novos nós. Pode-se observar quatro curvas no gráfico: *densa*

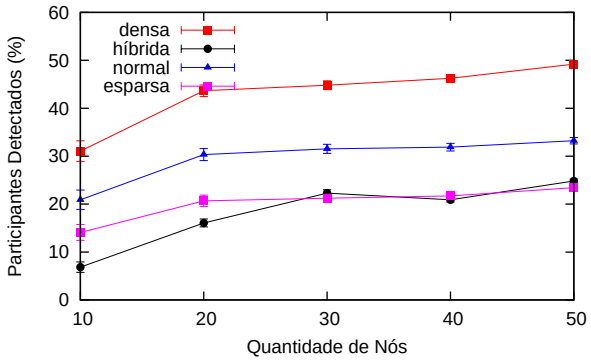


Figura 7.1. Resultado do detector PD-1hop em função da quantidade de nós.

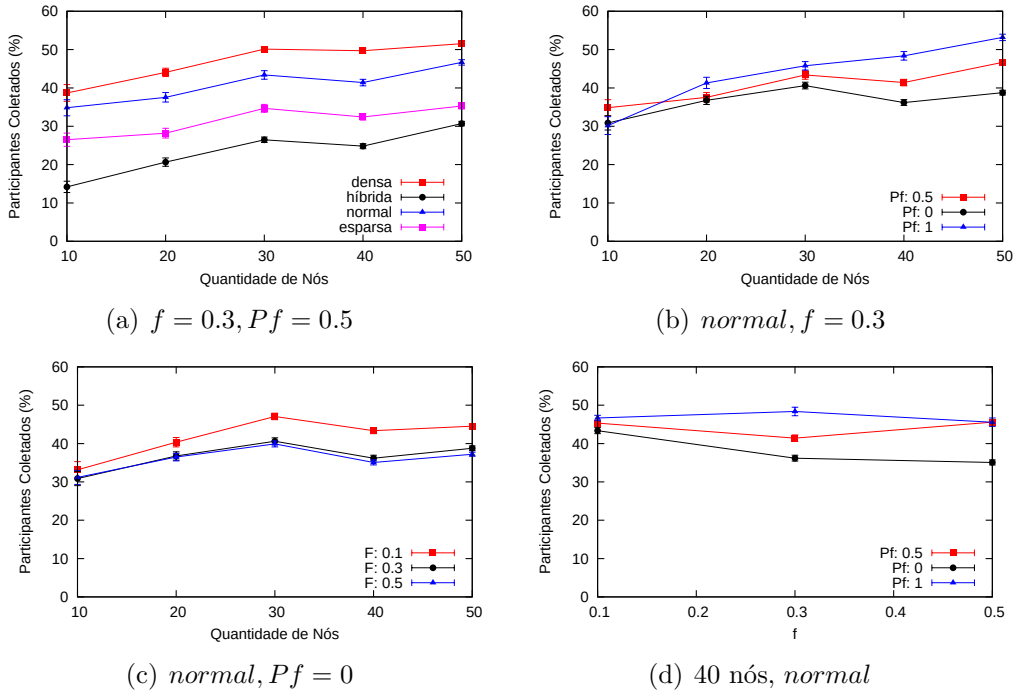
(alta densidade), *normal* (densidade normal), *esparsa* (baixa densidade) e *híbrida* (nós com diferentes alcances de transmissão). Neste gráfico, é visto que o desempenho do detector de participação é rigorosamente dependente da densidade da rede. A medida que se aumenta a densidade, aumenta-se o grau de conhecimento retornado por PD. No cenário híbrido, o resultado é semelhante ao resultado para a rede esparsa.

Sabe-se que o detector de participação PD-1hop, descrito através do Algoritmo 2, não apresenta dependência dos parâmetros do FT-CUP f e Pf e que o modelo de falhas descrito na Seção 5.3.3 garante que um nó falha apenas após a execução do detector de participação. Por isso, os valores de f e Pf não interferem no desempenho do detector de participação PD-1hop e os gráficos gerados por todos os valores destes parâmetros são semelhantes ao da Figura 7.1.

7.2.2 Desempenho do Algoritmo collect

A Figura 7.2 representa os resultados de simulação para a métrica *participantes coletados*. Esta métrica indica a porcentagem média dos nós coletados na execução do algoritmo COLLECT, através da visão da rede obtida por cada nó ao término da sua execução.

Através do gráfico 7.2(a), pode-se observar que a densidade da rede tem forte influência sobre o desempenho do COLLECT, ou seja, quanto maior a densidade da rede,



(a) $f = 0.3, Pf = 0.5$ (b) $normal, f = 0.3$

(c) $normal, Pf = 0$ (d) 40 nós, $normal$

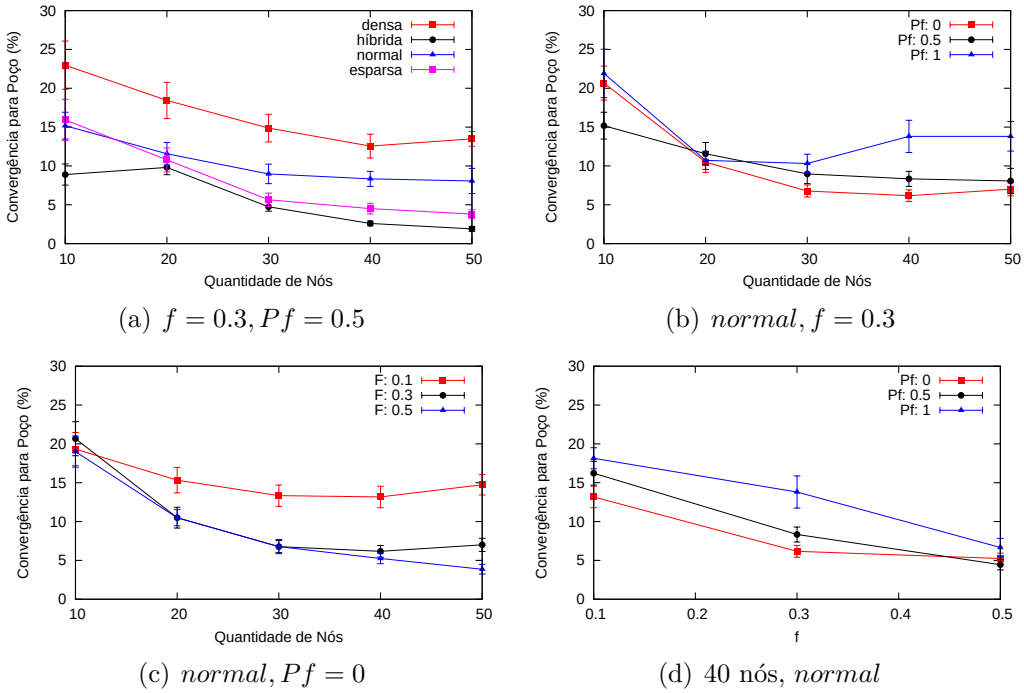
Figura 7.2. Quantidade de nós conhecidos com o COLLECT em função da quantidade de nós (a), (b), (c) e para 40 nós em função de f (d).

maior a porcentagem de nós coletados. Como esperado, em comparação com a métrica anterior *participantes detectados*, pode-se dizer que o COLLECT fornece uma expansão do conhecimento da rede. Entretanto, esta expansão tem mais influência em redes mais esparsas, uma vez que o detector PD-1hop já fornece uma boa visão sobre a rede para redes mais densas.

Diferentemente da métrica anterior *participantes detectados*, os parâmetros do FT-CUP f e Pf tem influência direta nos resultados desta métrica, comportamento explicitado através dos gráficos (b), (c) e (d) da Figura 7.2. A quantidade de participantes coletados é maior para um menor valor de f, sendo que a diferença é relevante apenas para $f = 0.1$. Para $f = 0.3$ e $f = 0.5$, os resultados são similares. Este comportamento decorre da característica do COLLECT, que possui uma espera de $|\Pi_p| - f$ mensagens para prosseguir na sua execução. Com um valor de f menor, cada nó espera por mais mensagens para prosseguir, o que possibilita um maior conhecimento sobre os nós da rede. Inesperadamente, é visto através do gráfico 7.2(b) que a porcentagem de nós coletados é

maior para um maior valor de Pf. Com isso, conclui-se que, quanto mais próximo for a quantidade real de falhas do número de falhas esperadas no protocolo, melhor o desempenho do COLLECT. Portanto, pode-se garantir que é importante adequar o parâmetro f para as aplicações reais de acordo com as características existentes na rede MANET utilizada. O gráfico 7.2(d) representa a comparação das curvas obtidas com a execução do COLLECT em relação as métricas f e Pf de forma simultânea e fortalece as considerações sobre os gráficos anteriores.

7.2.3 Convergência para Poço



(a) $f = 0.3, Pf = 0.5$ (b) $normal, f = 0.3$

(c) $normal, Pf = 0$ (d) 40 nós, $normal$

Figura 7.3. Convergência para poço em função da quantidade de nós (a), (b), (c) e para 40 nós em função de f (d).

A Figura 7.3 representa a métrica *convergência para poço*. Os valores representados nos gráficos são oriundos da execução do algoritmo SINK do FT-CUP.

O gráfico 7.3(a) compara os resultados desta métrica em relação às configurações de densidade utilizadas. Neste gráfico pode-se perceber que quanto maior a densidade da

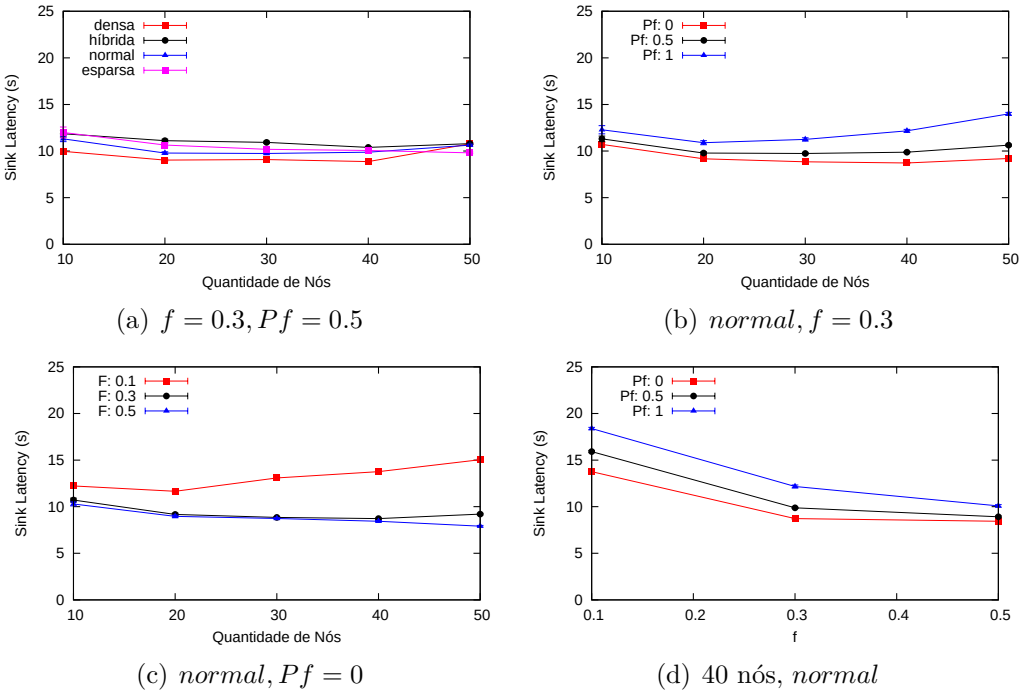
rede, maior a porcentagem de nós caracterizados como participantes da componente poço. Observe também que, em um mesmo valor de densidade, a porcentagem diminui com o aumento dos nós. Através da análise anterior do resultado do detector de participação, é visto que, uma maior densidade tem como consequência o maior grau de conectividade. Sendo assim, é maior a probabilidade da componente poço ter um maior número de participantes.

O gráfico 7.3(b) compara os resultados em relação aos valores de Pf utilizados. Quanto maior o valor de Pf, maior a porcentagem de convergência. Isso ocorre devido à estratégia do algoritmo COLLECT e SINK em que cada nó p espera por $|\Pi_p| - f$ mensagens. Sendo assim, quando efetivamente os f nós falham ($Pf = 1$), ocorre a situação em que cada nó recebe as respostas esperadas dos nós corretos da rede. Quando $Pf = 0$, um nó p sempre ignora mensagens de f nós corretos. Estes f nós poderiam aumentar o conhecimento de p (através do COLLECT) e fazer com que este identifique-se como participante do poço.

O gráfico 7.3(c) representa a métrica em questão comparando os resultados dos valores de f utilizados. Através dele, observa-se que quanto menor o f, maior a convergência, ou seja, quando menos mensagens são ignoradas, a convergência para poço é maior. Isso confirma a justificativa anterior em relação ao gráfico 7.3(b). O gráfico 7.3(d) sintetiza essas observações comparando os valores de f e Pf para o valor fixo de 40 nós.

7.2.4 Latência do sink

A Figura 7.4 representa a métrica *latência do* SINK. Os valores encontrados nos gráficos representam a média do tempo gasto na execução do algoritmo SINK do FT-CUP. Como pode-se observar nos resultados desta métrica para os testes preliminares (Seção 6.2), os resultados de latência do SINK apresentam pequenas variações na comparação entre os oráculo utilizados no módulo CONSENSUS. Porém, nas simulações deste capítulo, estas variações são irrelevantes e em nada acrescentam na interpretação dos resultados obtidos e encontram-se omitidas nos gráficos, onde os resultados apresentados referem-se



(a) $f = 0.3, Pf = 0.5$ (b) $normal, f = 0.3$

(c) $normal, Pf = 0$ (d) 40 nós, $normal$

Figura 7.4. Latência do SINK em função da quantidade de nós (a), (b), (c) e para 40 nós em função de f (d).

ao oráculo Random.

No gráfico 7.4(a), a latência do SINK apresenta valores altos e ligeiramente melhores para redes mais densas. Este comportamento pode ser justificado pelo uso do *Simple Flooding* como protocolo de roteamento, sendo que um protocolo mais eficiente poderia garantir melhores resultados. Através do gráfico 7.4(b), observa-se que a latência é menor para um menor valor de Pf. O gráfico 7.4(c) apresenta as curvas para as simulações com diferentes valores de f, sendo que a latência é maior para o valor de $f = 0.1$ e semelhante para $f = 0.3$ e $f = 0.5$. Assim como no algoritmo COLLECT, o SINK também possui a espera $|\Pi_p| - f$, que causa uma latência maior quando um menor valor de f é utilizado pois implica em uma quantidade maior de mensagens a receber antes que a execução possa prosseguir. No gráfico (d) da Figura 7.4 observa-se melhor as considerações sobre a influência de f e Pf representadas também nos gráficos (b) e (c).

7.3 ANÁLISE DO DESEMPENHO DO FT-CUP COM OS ORÁCULOS

Na seção anterior, foram apresentados os resultados para as simulações do FT-CUP até a execução do seu algoritmo SINK. Nesta seção, serão apresentados os resultados do FT-CUP até o término da sua execução, ou seja, até a execução do algoritmo CONSENSUS, o módulo que utiliza o consenso genérico de Guerraoui e Raynal [GR04] para permitir o uso de três variações para o consenso clássico, conforme descrito no modelo apresentado na Seção 5.3. Sendo assim, para a análise dos resultados, foram selecionadas as métricas:

- *Convergência do FT-CUP.* Refere-se à porcentagem de nós que terminam o consenso com uma decisão;
- *Verificação do Acordo.* Representa a porcentagem das execuções em que a propriedade acordo do consenso clássico (definida na Seção 4.1) não é violada, ou seja, ao término da execução, apenas um valor é decidido por todos os nós;
- *Latência do FT-CUP.* Mostra a média dos tempos gastos para a convergência do FT-CUP.

Para cada métrica, serão mostrados os resultados do FT-CUP com o oráculo aleatório Random comparando os efeitos dos parâmetros f, Pf e densidade da rede e, posteriormente, a comparação dos resultados entre os oráculos Random, FD e Leader.

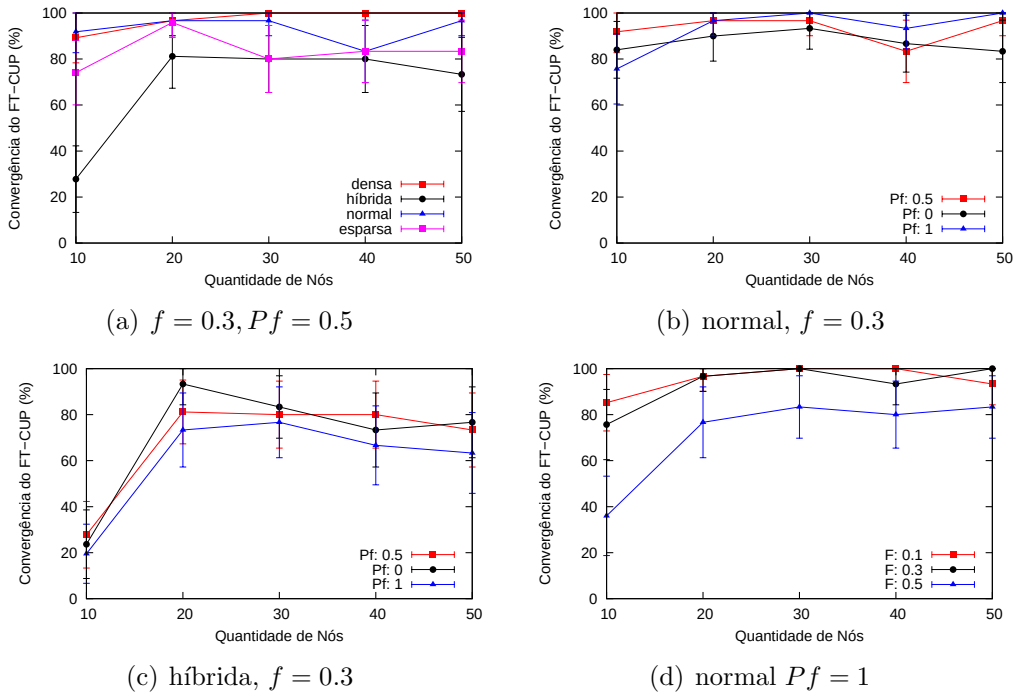
7.3.1 Verificação da Propriedade Terminação

Como descrito na Seção 4.1, a propriedade terminação do problema do consenso é definida da seguinte forma:

- *Terminação.* Todo processo correto decide por um valor.

Portanto, mostram-se nesta seção os resultados da métrica que se relaciona com a terminação do consenso, a *convergência do FT-CUP*, referente a quantidade de processos

que decidem por um valor nas simulações (dentro do tempo máximo da execução de cada simulação).



(a) $f = 0.3, Pf = 0.5$ (b) normal, $f = 0.3$

(c) híbrida, $f = 0.3$ (d) normal $Pf = 1$

Figura 7.5. Convergência do FT-CUP com o Random em função da quantidade de nós.

Oráculo Aleatório (Random). Na Figura 7.5 é mostrada a métrica *convergência do FT-CUP* para o oráculo Random. Observando o gráfico 7.5(a), é visto que, quanto maior a densidade maior a taxa de convergência. Isso ocorre devido ao maior grau de conectividade na rede.

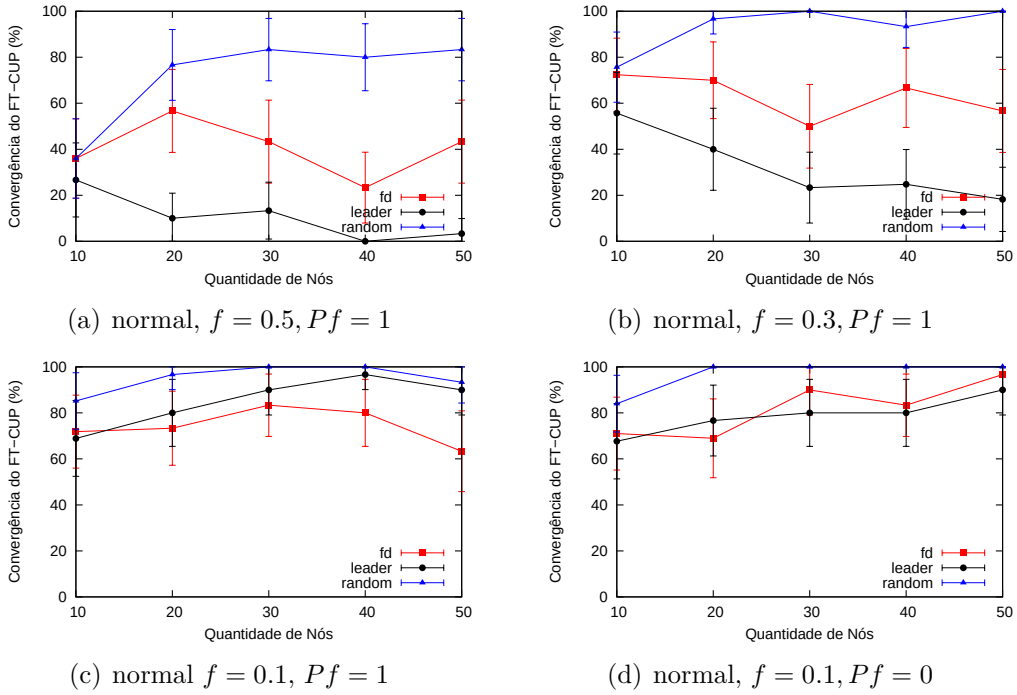
Os gráficos 7.5(b) e 7.5(c) representam as curvas de convergência para os valores de Pf na densidade normal e híbrida, respectivamente. Pode-se observar nestes gráficos que uma quantidade de falhas menor (menor valor de Pf) produz um melhor desempenho, comportamento contrário ao ocorrido na métrica convergência para poço (Seção 7.2.3). Neste caso, pode-se concluir que, como esperado, a ocorrência de falhas interfere na execução do consenso. Isto ocorre porque as falhas podem ocorrer em nós participantes da componente poço e ocasionar em bloqueios que impedem a execução normal do consenso clássico. Observando as curvas dos gráficos 7.5(b) e 7.5(c), conclui-se que, a depender da densidade da rede, o valor de Pf causa um determinado grau de inter-

ferência na convergência. Para redes com maior densidade, o valor de Pf interfere de forma menos significativa na convergência. Sendo assim, pode-se dizer que, em redes densas, é possível executar o FT-CUP mesmo na presença da quantidade máxima de falhas, limitada pelo valor de f. Além disso, em alguns casos a convergência é maior para um maior valor de Pf, assim como acontece na convergência para poço. Conclui-se disso que, não adianta usar o FT-CUP com uma capacidade de tolerar falhas (representada aqui pelo valor de f) maior do que a quantidade de falhas esperada no ambiente. Essa definição inadequada pode ser prejudicial nos casos de execuções livres de falhas ou com falhas significativamente abaixo do esperado. O gráfico 7.5(d) representa as curvas de convergência para os valores de f na densidade normal onde é visto que quanto menor o valor de f, maior a convergência.

Comparação entre os Oráculos. Na Figura 7.6 mostram-se os resultados da métrica *convergência do FT-CUP* para a comparação entre os oráculos Random, FD e Leader. O gráfico 7.6(a) apresenta os resultados para $f = 0.5$ e $Pf = 1$. Pode-se observar que o desempenho do FT-CUP com o Random é muito superior aos demais. O desempenho do oráculo FD foi apenas razoável e o oráculo Leader apresentou resultados ruins. Comparando-se os gráficos (a), (b) e (c) da Figura 7.6, pode-se observar que a convergência é maior para um menor valor de f. O desempenho com os oráculos FD e Random melhora bastante com a diminuição de f. Portanto, pode-se atribuir ao resultado ruim destes oráculos a bloqueios ocorridos no protocolo, devido a dependência do parâmetro f na execução dos seus algoritmos. No gráfico 7.6(d), encontram-se os resultados para $f = 0.1$ na ausência de falhas ($Pf = 0$). É visto que o Random obtém convergência quase total enquanto os oráculos FD e Leader obtém desempenho inferior, em torno de 80%.

7.3.2 Verificação da Propriedade Acordo

A propriedade acordo do problema do consenso, descrita na Seção 4.1, é definida da seguinte forma:



(a) normal, $f = 0.5, Pf = 1$ (b) normal, $f = 0.3, Pf = 1$

(c) normal $f = 0.1$, $Pf = 1$ (d) normal, $f = 0.1, Pf = 0$

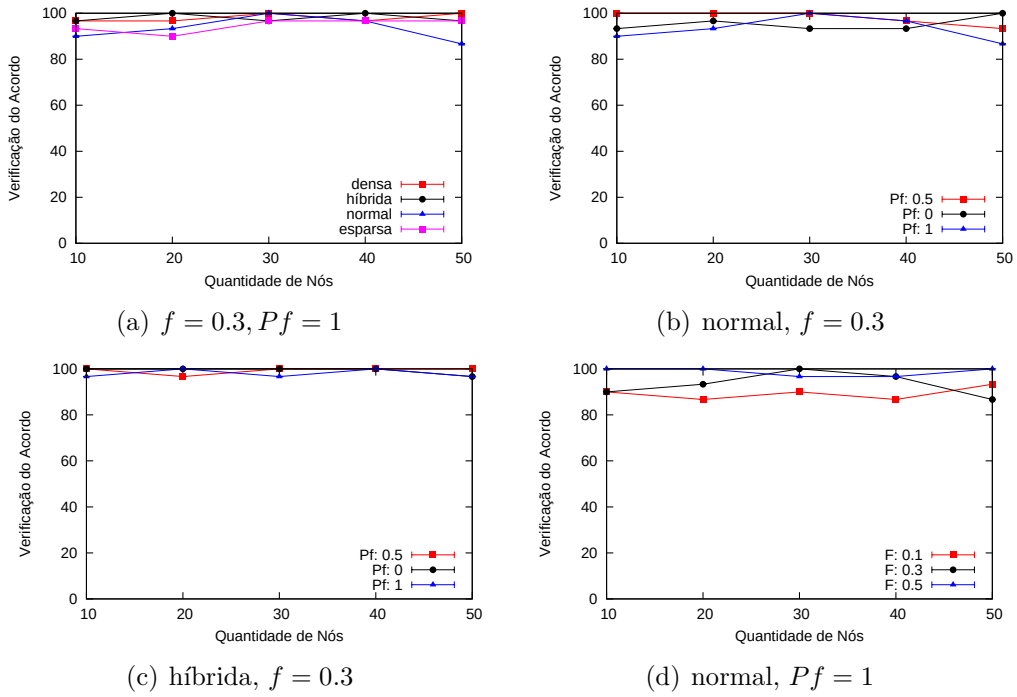
Figura 7.6. Convergência do FT-CUP em função da quantidade de nós.

- *Acordo*. Todos os processos corretos decidem pelo mesmo valor.

Portanto, mostram-se nesta seção os resultados da métrica *verificação do acordo*, que contabiliza as execuções do FT-CUP que ocorrem sem a violação da propriedade acordo, ou seja, onde todos os processos decidem pelo mesmo valor nas simulações executadas.

Oráculo Aleatório (Random). Na Figura 7.7 encontra-se representado os resultados para a métrica *verificação do acordo* para o oráculo Random. Em todas as situações (dos gráficos (a), (b), (c) e (d) e dos gráficos aqui omitidos), a porcentagem média das execuções que não violam a propriedade acordo superou o valor de 85%. A formação de mais de uma componente poço é a principal razão para que o acordo seja violado. Outra possibilidade seria um particionamento da rede, fazendo com que existam dois ou mais grupos de nós completamente isolados. Observe que a ocorrência destes fenômenos é pequena porém não é irrelevante.

Os gráficos (a), (b), (c) e (d) da Figura 7.7 evidenciam que o comportamento desta métrica frente as diversas variações dos parâmetros usados nas simulações é bastante



(a) $f = 0.3, Pf = 1$ (b) normal, $f = 0.3$

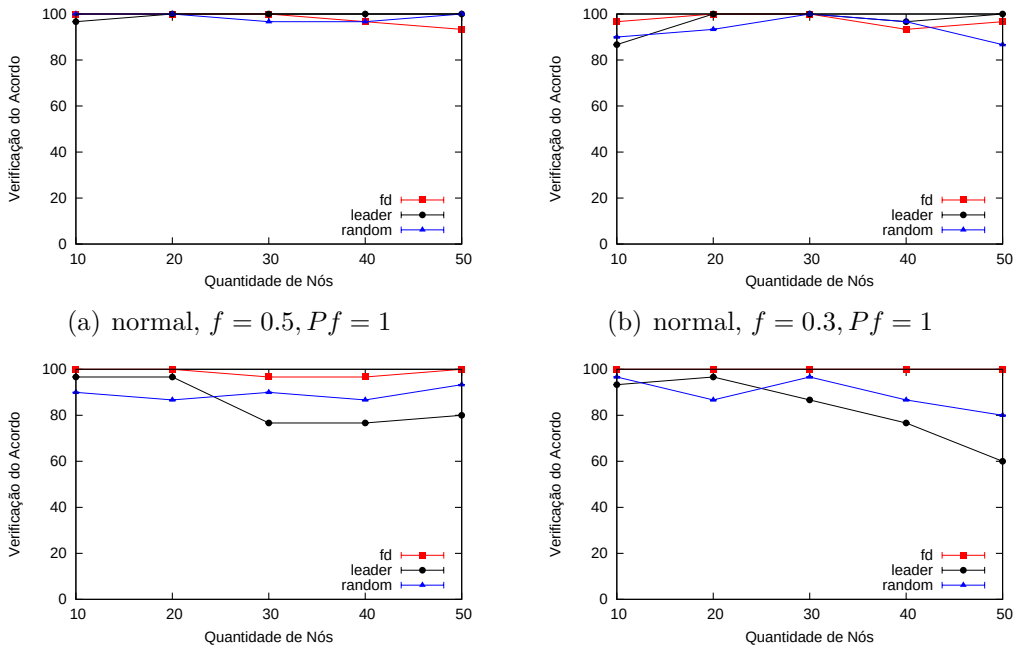
(c) híbrida, $f = 0.3$ (d) normal, $Pf = 1$

Figura 7.7. Verificação do acordo para o oráculo Random, em função da quantidade de nós.

semelhante. A diferença mais relevante encontrada seria para a variação de f, observada no gráfico 7.7(d). Quanto maior o f, menor a quantidade de execuções que violam o acordo.

Comparação entre os Oráculos. Na Figura 7.8 encontram-se representados os resultados para a métrica *verificação do acordo* para a comparação entre os oráculos Random, FD e Leader. O gráfico 7.8(a), que representa os resultados para $f = 0.5$ e $Pf = 1$, pode-se observar que, com todos os oráculos, quase todas as execuções do FT-CUP ocorreram sem a violação da propriedade acordo. Neste cenário, o oráculo Leader obteve o melhor resultado, com a violação do acordo ocorrendo apenas na rede composta por 10 nós. No gráfico 7.8(b), que representa os resultados para $f = 0.3$ e $Pf = 1$, o oráculo Leader apresenta desempenho ligeiramente inferior, assim como os demais oráculos. Para o gráfico 7.8(c), que possui os parâmetros $f = 0.1$ e $Pf = 1$, a diferença de desempenho entre os oráculos torna-se mais evidente. O FD obteve o melhor desempenho neste cenário, aproximando-se de 100% das simulações executadas corretamente em relação ao acordo. Pode-se dizer que o valor de f encontrado neste cenário torna-se adequado às características do pro-

tocolo de detecção de falhas utilizado, o que não ocorre para o detector de líder. Para o oráculo Leader, o cenário com o maior valor de falhas possíveis $f = 0.5$ apresentou os melhores resultados. O oráculo Random apresentou comportamento semelhante, devido às características probabilísticas do consenso realizado com este oráculo. Portanto, Leader e Random possuem desempenho melhor para um maior valor de f enquanto que o FD tem melhor desempenho para um menor valor de f. Através do gráfico 7.8(d), para $f = 0.1$ e na ausência de falhas ($Pf = 0\%$), observa-se que o desempenho dos oráculos foi similar ao obtido no mesmo cenário, mas na presença de falhas ($Pf = 1$). A execção é o resultado do oráculo FD, que obteve desempenho perfeito.



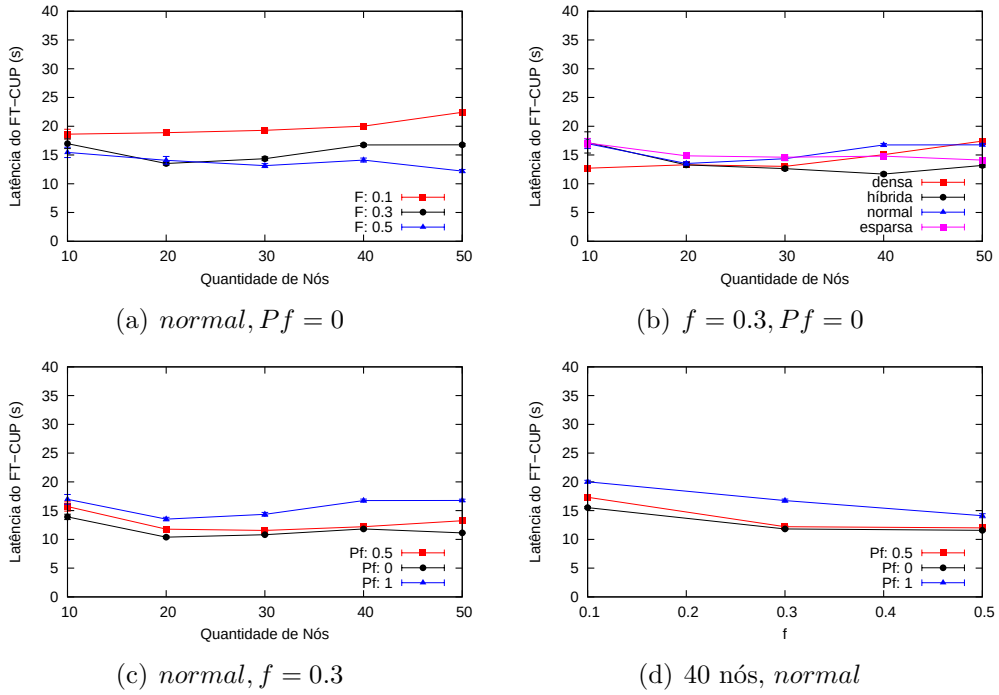
(a) normal, $f = 0.5, Pf = 1$ (b) normal, $f = 0.3, Pf = 1$

(c) normal $f = 0.1$, $Pf = 1$ (d) normal, $f = 0.1, Pf = 0$

Figura 7.8. Verificação do acordo em função da quantidade de nós.

7.3.3 Latência do FT-CUP

Oráculo Aleatório (Random). A Figura 7.9 representa os resultados da métrica *latência do FT-CUP* para o oráculo Random. Essa latência varia em torno de 15 segundos. O protocolo de roteamento utilizado (*flooding*) faz com que exista um alto tráfego de men-



(a) $normal, Pf = 0$ (b) $f = 0.3, Pf = 0$

(c) $normal, f = 0.3$ (d) 40 nós, *normal*

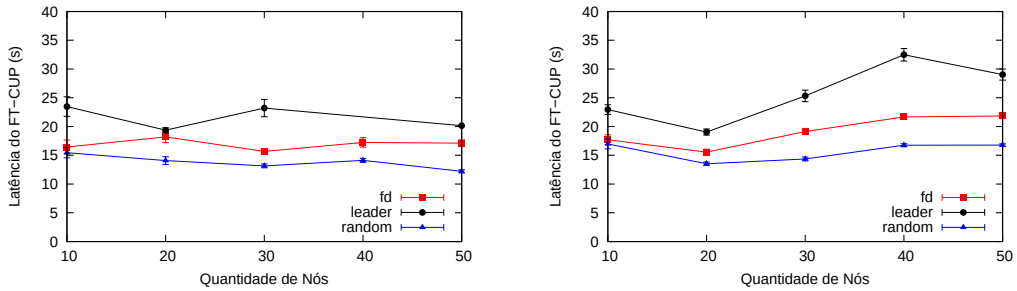
Figura 7.9. Latência do FT-CUP em função da quantidade de nós (a), (b), (c) e para 40 nós em função de f (d).

sagens na rede, ocasionando tempos altos para o FT-CUP. Isso acontece também devido a alta complexidade dos algoritmos do FT-CUP (COLLECT, SINK e CONSENSUS), que é, no pior caso, $O(n^2)$. Portanto, esta métrica deve ser analisada apenas qualitativamente. Acredita-se que com o uso de um protocolo de roteamento mais adequado para MANET este tempo diminua significativamente.

No gráfico 7.9(a) é visto que a latência aumenta com o menor valor de f, uma vez que para menores valores de f temos uma maior componente poço, o que aumenta a complexidade do consenso. Usando o mesmo raciocínio, no gráfico 7.9(b), a latência é maior para uma rede mais densa e no gráfico 7.9(c) a latência é maior para um maior valor de Pf. O gráfico 7.9(d) evidencia a diminuição da latência com o aumento de f.

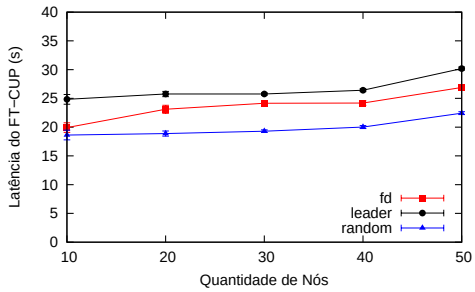
Comparação entre os Oráculos. A Figura 7.10 representa os resultados da métrica *latência do FT-CUP* para a comparação entre os oráculos Random, FD e Leader. Os gráficos (a), (b), (c) e (d) representam os resultados para $f = 0.5$ e $Pf = 1$, $f = 0.3$ e $Pf = 1$, $f = 0.1$ e $Pf = 1$, $f = 0.1$ e $Pf = 0$, respectivamente. Para todos estes

cenários os comportamentos são similares. Nesta figura, observa-se que a latência é, em geral, ligeiramente menor para maiores valores de f. Valores diferentes para f apresentam variações mais relevantes na latência para o FT-CUP com o oráculo Leader. Comparando-se os gráficos (c), cenário com falhas, e (d), ausência de falhas, é visto que em cenários sem falhas a latência do FT-CUP é menor, uma vez que menos rodadas do consenso são executadas e a convergência para a decisão ocorre mais rapidamente.

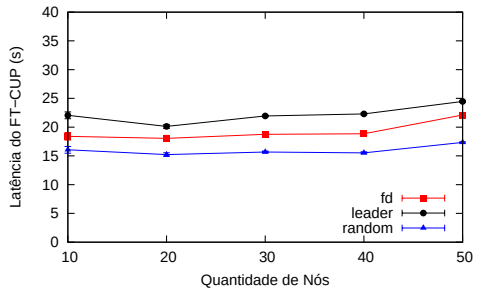


(a) normal, $f = 0.5, Pf = 1$

(b) normal, $f = 0.3, Pf = 1$



(c) normal $f = 0.1$, $Pf = 1$



(d) normal, $f = 0.1, Pf = 0$

Figura 7.10. Latência do FT-CUP em função da quantidade de nós .

7.4 CONCLUSÕES

A análise dos resultados das simulações, apresentada nesse capítulo, proporcionou enumerar algumas conclusões obtidas na execução da solução para o FT-CUP de Greve e Tixeuil [GT07] sobre o ambiente de MANET, entre os cenários definidos também neste capítulo. Os itens a seguir definem as condições práticas onde o FT-CUP pode ser resolvido em uma MANET, além de características importantes observadas no comportamento do FT-CUP.

i) A escolha do detector de participação e do parâmetro f é essencial para a convergência do FT-CUP;

ii) Através das simulações pôde-se observar que mesmo um detector simples constrói um grafo de conhecimento que se aproxima dos requisitos teóricos para resolver o FT-CUP, representados por um grafo k-fortemente conexo com apenas uma componente poço;

iii) Para o oráculo Random do consenso probabilístico, em redes com densidade alta, a convergência do FT-CUP obteve valores próximos a 100%, mesmo na presença de falhas. Conclui-se que, a partir de determinada densidade temos os requisitos necessários para a resolução do FT-CUP, mesmo na presença de uma quantidade significativa de falhas;

iv) Os oráculos FD e Leader do consenso determinístico não tiveram bons resultados para a convergência do FT-CUP em cenários com valores maiores de f. A dependência de f no algoritmo destes oráculos é determinante para a ocorrência de bloqueios na execução do protocolo e prejudicial a converrgência do FT-CUP.

v) É possível resolver o FT-CUP mesmo com densidades menores. Nestes casos, deve-se tolerar menos falhas utilizando para f um valor mais adequado;

vi) Em alguns casos, a convergência do FT-CUP é maior para um maior valor de Pf. Conclui-se disso que o parâmetro f, o indicador da capacidade de tolerar falhas do protocolo, deve ser adequado à quantidade de falhas esperada no ambiente.

vii) Para o oráculo Random, em todas as simulações, mesmo nas redes mais esparsas, a porcentagem das execuções que não violam a propriedade de acordo superou o valor de 85%, sendo que nas redes densas atingiu 100% na maioria das situações. Conclui-se que, de acordo com a densidade da rede, o FT-CUP pode ser resolvido de maneira aproximada sem a violação desta propriedade.

viii) O oráculo FD apresentou o melhor resultado em relação à execução do consenso sem violação do acordo.

CAPÍTULO 8

CONSIDERAÇÕES FINAIS

O FT-CUP é uma variação do problema do consenso para redes dinâmicas, como as redes MANET. O FT-CUP não considera o conhecimento prévio dos participantes como requisito para sua resolução e utiliza uma abstração, o detector de participação, para obter o conhecimento parcial sobre a rede necessário para garantir o acordo. Este trabalho estudou os aspectos práticos da implementação do FT-CUP proposto por Greve e Tixeuil [GT07] no ambiente de MANET. Propõe-se uma abordagem modular para a implementação do FT-CUP, onde um algoritmo de consenso clássico e um protocolo de difusão são utilizados como parte de sua solução, além dos seus algoritmos específicos. Para o consenso clássico, o consenso genérico de Guerraoui e Raynal [GR04] foi utilizado. Esta solução permite o uso das principais abstrações para resolução do consenso em ambientes assíncronos: detector de falhas, eleição de líder e oráculo aleatório. Protocolos de difusão adequados ao ambiente de MANET foram estudados e apresentou-se um trabalho de análise experimental desses protocolos, realizada através de simulações. Através de reflexões sobre os resultados desta análise, optou-se por utilizar como protocolo de difusão no FT-CUP o *Simple Flooding*, justamente por possuir alta redundância e, consequentemente, alta capacidade de tolerar faltas. Implementações simples para o detector de participação (PD1-hop e PD2-hop) foram utilizadas para avaliar o comportamento do FT-CUP em redes com conhecimento bastante limitado sobre os nós participantes.

8.1 SUMÁRIO

O Capítulo 2 descreve as características fundamentais das redes MANET que são utilizadas durante todo este trabalho, incluindo definições sobre os seus canais de comunicação. Descrevem-se as características de tolerância a faltas específicas ao modelo de

MANET, como o modelo de falhas dos processos e canais de comunicação, e o modelo de sincronia, adequados ao ambiente destas redes. Aborda-se também o método de simulações para análise do desempenho de protocolos no ambiente de redes MANET, com a descrição de ferramentas de simulações existentes e boas práticas para garantir resultados confiáveis.

O Capítulo 3 introduz o problema da difusão em redes MANET e descreve os protocolos *Simple Floodind* [WC02, CT96], probabilístico dinâmico [ZA05], protocolo de Wu e Li [WL99], *Scalable Broadcast Algorithm* (SBA) [PL00], *Dominant Pruning* (DP) [LK01] e o *Double-Covered Broadcast* (DCB) [LW07]. Tais protocolos são executados em ambientes de simulação para caracterizar o comportamento frente ao modelo de falhas por omissão. Conclui-se com os resultados que a redundância é fundamental para garantir a confiabilidade do protocolo de difusão e que a eficiência pode ser sacrificada em detrimento da confiabilidade, em casos onde a execução correta do protocolo de difusão é fundamental para a sua aplicação.

O Capítulo 4 apresenta a definição do problema do consenso em tolerância a faltas e mostra os seus resultados de impossibilidade no modelo clássico (FLP) [FLP85]. Apresentam-se algumas alternativas para sua resolução, como o consenso com detector de líder (Paxos) [Lam98], consenso probabilístico [BO83, EMR01], consenso com detecção de falhas [CT96] e um consenso genérico [GR04] que funciona munido de qualquer oráculo aleatório, de eleição de líder ou detecção de falhas. Posteriormente, o problema do consenso é abordado no contexto de redes dinâmicas e as soluções apresentadas consideram as características destas redes. Tais soluções são o Paxos para MANET [BPS08], consenso probabilístico para MANET [Vol05] e consenso hierárquico [WCYR06], apresentando-se ainda implementações para detectores de falhas e líder específicos ao ambiente de MANET [JAF06, SABG07].

O Capítulo 5 complementa o Capítulo 4 com a apresentação do consenso com participantes desconhecidos, um tipo de consenso típico para o ambiente de redes dinâmicas. Inicialmente, descreve-se o CUP (*Consensus with Unknown Participants*) [CSS04], o con-

senso com participantes desconhecidos que não tolera falhas nos processos, e a abstração dos detectores de participação, sendo que o detector mínimo necessário para a resolução do CUP é o OSR. Posteriormente, descreve-se o FT-CUP (*Fault Tolerant Consensus with Unknown Participants* [CSS05], a versão do CUP com a possibilidade de falhas nos processos. A solução inicial [CSS05] apresenta a resolução para os mesmos requisitos mínimos de conectividade do CUP e encontra como requisitos mínimos de sincronia o detector de falhas $\mathcal{P}$. A solução de Greve e Tixeuil [GT07] resolve o FT-CUP considerando o detector de falhas $\diamondsuit\mathcal{S}$ e encontra como requisito mínimo de conectividade o detector de participação k-OSR. Finaliza-se este capítulo com a descrição da proposta deste trabalho para o modelo de implementação do FT-CUP [GT07] no ambiente de MANET. A implementação do protocolo é realizada de forma modular, subdividindo os módulos entre os algoritmos necessários ao FT-CUP e os seus protocolos auxiliares. Propõe-se neste capítulo implementações simples para os detectores de participação, como o PD1-hop e PD-2hop. Define-se ainda o modelo de falhas utilizado nas simulações deste trabalho.

O Capítulo 6 apresenta os testes preliminares das simulações do FT-CUP de Greve e Tixeuil [GT07], implementado seguindo o modelo descrito na Seção 5.3. Apesar destes testes terem sido realizados sobre um modelo inicial de simulações, foram consideradas as boas práticas na definição dos parâmetros de simulações e buscou-se cenários realistas. Através da análise dos resultados, foi possível concluir que a escolha do detector de participação e do parâmetro k é essencial para a convergência do FT-CUP e que é possível resolver o consenso mesmo com um detector de participação simples, como o PD1-hop. Sendo assim, o modelo de simulações proposto neste capítulo inspirou o modelo das simulações finais, onde foram adicionadas melhorias que possibilitaram uma visão mais realista do problema FT-CUP em MANET.

O Capítulo 7 apresenta os resultados finais de avaliação de desempenho do FT-CUP, implementado seguindo o modelo descrito na Seção 5.3. Este capítulo aproveitou-se do trabalho previamente realizado no Capítulo 6, de forma a fortalecer o modelo de simulações e se aproximar mais das características reais de redes MANET. Sendo assim, um novo modelo de simulações foi utilizado, o que definiu novos conjuntos de simulações

a se realizar. Com estas simulações, acrescentam-se aos resultados que, de acordo com a densidade da rede, o FT-CUP pode ser resolvido de maneira aproximada sem a violação desta propriedade e que a escolha do oráculo utilizado em conjunto com o FT-CUP é importante tanto para garantir a terminação quanto a propriedade acordo do consenso.

8.2 CONCLUSÕES

A abordagem prática conduzida no estudo do FT-CUP proporcionou concluir que a escolha do detector de participação e do parâmetro f é essencial para a convergência do FT-CUP. Através das simulações pode-se observar que mesmo um detector simples, como o PD-1hop, é capaz de construir um grafo de conhecimento que se aproxima dos requisitos teóricos para resolver o FT-CUP e que possibilitou a execução correta do consenso no ambiente das redes MANET. Em relação aos cenários de densidade da rede, tem-se que, a partir de determinada densidade, os requisitos necessários para a resolução do FT-CUP são alcançados, mesmo na presença de uma quantidade significativa de falhas. Porém, é possível resolver o FT-CUP mesmo com densidades menores, basta adequar o parâmetro f, de forma a tolerar uma quantidade menor de falhas. Além disso, de acordo com a densidade da rede, o FT-CUP pode ser resolvido de maneira aproximada sem a violação da propriedade acordo.

8.3 TRABALHOS FUTUROS

Para trabalhos futuros, pretende-se simular o consenso com implementações diferentes para o detector de falhas e eleição de líder, adequados ao ambiente de MANET, de forma a comparar com os resultados obtidos neste trabalho. Deve-se considerar ainda outras implementações para o protocolo de roteamento e difusão de mensagens, para verificar se a latência do FT-CUP pode ser reduzida com o uso de protocolos mais eficientes. É importante também verificar a escalabilidade do protocolo e simular o FT-CUP em redes com grandes quantidades de nós.

Pretende-se ainda investigar mecanismos para garantir uma melhor taxa de convergência do FT-CUP e garantir que, mesmo em cenários desfavoráveis, a propriedade de acordo não seja violada.

REFERÊNCIAS

[ALRL04] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, 2004.

[Asp03] James Aspnes. Randomized protocols for asynchronous consensus. *Distrib. Comput.*, 16(2-3):165–175, 2003.

[ASSC05] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. In *SBRC 2005: Brazilian Symposium on Computer Networks*, pages 330–398, mai 2005.

[BDFG03] Romain Boichat, Partha Dutta, Svend Frolund, and Rachid Guerraoui. Deconstructing paxos. *SIGACT News*, 34(1):47–67, 2003.

[BEV06] François Bonnet, Paul Ezhilchelvan, and Einar Vollset. Quiescent consensus in mobile ad-hoc networks using eventually storage-free broadcasts. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 670–674, New York, NY, USA, 2006. ACM Press.

[BKP03] C. Basile, M. Killijian, and D. Powell. A survey of dependability issues in mobile wireless networks. Technical report, LAAS CNRS, Toulouse, France, feb 2003.

[BO83] Michael Ben-Or. Another advantage of free choice (extended abstract): Completely asynchronous agreement protocols. In *PODC '83: Proceedings of the second annual ACM symposium on Principles of distributed computing*, pages 27–30, New York, NY, USA, 1983. ACM.

[BPS08] Fatemeh Borran, Ravi Prakash, and André Schiper. Extending paxos/lastvoting with an adequate communication layer for wireless ad hoc networks. In *SRDS '08: Proceedings of the 2008 Symposium on Reliable Distributed Systems*, pages 227–236, Washington, DC, USA, 2008. IEEE Computer Society.

[CBD02] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2:483–502, 2002.

[CBS07] Bernadette Charron-Bost and André Schiper. The Heard-Of Model: Computing in Distributed Systems with Benign Failures. Technical report, 2007. Replaces TR-2006: The Heard-Of Model: Unifying all Benign Failures.

[CCL03] Imrich Chlamtac, Marco Conti, and Jennifer Liu. Mobile ad hoc networking: Imperatives and challenges. *Ad Hoc Networks*, 1(1), 2003.

[CHT96] Tushar Deepak Chandra, Vassos Hadzilacos, and Sam Toueg. The weakest failure detector for solving consensus. *J. ACM*, 43(4):685–722, 1996.

[CSS04] David Cavin, Yoav Sasson, and André Schiper. Consensus with unknown participants or fundamental self-organization. In *Third Int. Conf. on Ad hoc Net. and Wireless (ADHOC-NOW 2004)*, pages 135–148, Vancouver, Canada, July 2004.

[CSS05] David Cavin, Yoav Sasson, and André Schiper. Reaching agreement with unknown participants in mobile self-organized networks in spite of process crashes. Technical Report 2005026, Ecole Polytechnique Federale de Lausanne, 2005.

[CT96] Tushar Deepak Chandra and Sam Toueg. Unreliable failure detectors for reliable distributed systems. *J. ACM*, 43(2):225–267, 1996.

[CWKS97] B.P. Crow, I. Widjaja, L.G. Kim, and P.T. Sakai. Ieee 802.11 wireless local area networks. *Communications Magazine, IEEE*, 35(9):116–126, Sep 1997.

[DLS88] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. *J. ACM*, 35(2):288–323, 1988.

[dOCG07] Talmai Brandão de Oliveira, Victor Franco Costa, and Fabíola Greve. On the behavior of broadcasting protocols for manets under omission faults scenarios. In *Lecture Notes in Computer Science*, volume 4746, pages 142–159. Springer, September 2007.

[DSH+03] Witold Drytkiewicz, Steffen Sroka, Vlado Handziski, Andreas Köpke, and Holger Karl. A mobility framework for omnet++. In *3rd International OMNeT++ Workshop*, january 2003.

[DW03] Fei Dai and Jie Wu. Distributed dominant pruning in ad hoc networks. In *Proceedings of the IEEE International Conference on Communications (ICC 2003)*, volume 1, pages 353–357, 2003.

[DW04] Fei Dai and Jie Wu. Performance analysis of broadcast protocols in ad hoc networks based on self-pruning. *IEEE Trans. Parallel Distrib. Syst.*, 15(11):1027–1040, 2004.

[EMR01] Paul Ezhilchelvan, Achour Mostefaoui, and Michel Raynal. Randomized multivalued consensus. *IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, 0:0195, 2001.

[FJL00] Magnus Frodigh, Per Johansson, and Peter Larsson. Wireless ad-hoc networking - the art of networking without a network. *Ericsson Review*, (4):248–263, 2000.

[FLP85] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, 1985.

[GK96] Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. In *ESA '96: Proceedings of the Fourth Annual European Symposium on Algorithms*, pages 179–193, London, UK, 1996. Springer-Verlag.

[GM01] Brian P. Gerkey and Maja J. Mataric. Principled communication for dynamic multi-robot task allocation. In *ISER '00: Experimental Robotics VII*, pages 353–362, London, UK, 2001. Springer-Verlag.

[GR04] Rachid Guerraoui and Michel Raynal. The information structure of indulgent consensus. *IEEE Trans. Comput.*, 53(4):453–466, 2004.

[GT07] Fabiola Greve and Sebastien Tixeuil. Knowledge connectivity vs. synchrony requirements for fault-tolerant agreement in unknown networks. In *DSN '07: Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 82–91, Washington, DC, USA, 2007. IEEE Computer Society.

[HBG06] Luc Hogie, Pascal Bouvry, and Frédéric Guinand. An overview of manets simulation. In *Proceedings of the First International Workshop on Methods and Tools for Coordinating Concurrent, Distributed and Mobile Systems (MTCoord 2005)*, volume 150, pages 81–101, Namur, Belgium, mar 2006. Electronic Notes in Theoretical Computer Science.

[Ise96] R. Isermann. Modeling and design methodology for mechatronic systems. *IEEE/ASME Transactions on Mechatronics*, 1(1):16–28, March 1996.

[JAF06] Ernesto Jiménez, Sergio Arévalo, and Antonio Fernández. Implementing unreliable failure detectors with unknown membership. *Inf. Process. Lett.*, 100(2):60–63, 2006.

[KCC05] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. Manet simulation studies: the incredibles. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(4):50–61, 2005.

[KM07] Wolfgang Kiess and Martin Mauve. A survey on real-world implementations of mobile ad-hoc networks. *Ad Hoc Netw.*, 5(3):324–339, 2007.

[KNG+04] David Kotz, Calvin Newport, Robert S. Gray, Jason Liu, Yougu Yuan, and Chip Elliott. Experimental evaluation of wireless simulation assumptions. In *MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 78–82, New York, NY, USA, 2004. ACM.

[Lam77] L. Lamport. Proving the correctness of multiprocess programs. *IEEE Trans. Softw. Eng.*, 3(2):125–143, 1977.

[Lam98] Leslie Lamport. The part-time parliament. *ACM Transactions on Computers Systems*, 16(2):133–169, 1998.

[Lam01] Leslie Lamport. Paxos made simple. *ACM SIGACT News*, 32(4):18–25, December 2001.

[LFA04] Mikel Larrea, Antonio Fernandez, and Sergio Arevalo. On the implementation of unreliable failure detectors in partially synchronous systems. *IEEE Transactions on Computers*, 53(7):815–828, 2004.

[LK01] H. Lim and C. Kim. Flooding in wireless ad hoc networks. *Computer Communications*, 24(3):353–363, February 2001.

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[LW07] Wei Lou and Jie Wu. Toward broadcast reliability in mobile ad hoc networks with double coverage. *IEEE Transactions on Mobile Computing*, 6(2):148–163, 2007.

[MF09] Mobility framework. http://mobility-fw.sf.net, Jan 2009.

[MR99] Achour Mostéfaoui and Michel Raynal. Solving consensus using Chandra-Toueg's unreliable failure detectors: A general quorum-based approach. In *13th Int. Symp. on Distributed Computing (DISC'99)*, volume 1693 of *LNCS*, pages 49–63, sep 1999.

[MR01] A. Mostefaoui and M. Raynal. Leader-based consensus. *Parallel Process Letters*, 11(1):95–107, March 2001.

[ns209] The network simulator. http://www.isi.edu/nsnam/ns, Jan 2009.

[olp09] One laptop per child. http://laptop.org, Jun 2009.

[omn09] Omnet++. http://www.omnetpp.org, Jan 2009.

[PL00] Wei Peng and Xi-Cheng Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing*, pages 129–130, Boston, Massachusetts, 2000.

[RR02] Ram Ramanathan and Jason Redi. A brief overview of ad hoc networks: challenges and directions. *IEEE Communications Magazine*, 40(5):20–22, 2002.

[SABG07] Pierre Sens, Luciana Arantes, Mathieu Bouillaguet, and Fabíola Greve. Asynchronous implementation of failure detectors with partial connectivity and unknown participants. *CoRR*, abs/cs/0701015, 2007.

[San99] Susan M. Sanchez. Abc's of output analysis. In *WSC '99: Proceedings of the 31st conference on Winter simulation*, pages 24–32, New York, NY, USA, 1999. ACM.

[SHT+08] Vivek Srivastava, Amr B. Hilal, Michael S. Thompson, Jawwad N. Chattha, Allen B. MacKenzie, and Luiz A. DaSilva. Characterizing mobile ad hoc networks - the maniac challenge experiment. In *WiNTECH '08: Proceedings of the third ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*, pages 65–72, New York, NY, USA, 2008. ACM.

[TG01] T. Tang and M. Gerla. Mac reliable broadcast in ad hoc networks. In *Proc. of the IEEE Military Communications Conference*, pages 1008–1013, 2001.

[TNCS02] Yu-Chee Tseng, Sze-Yao Ni, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. *Wireless Networks*, 8(2–3):153–167, 2002.

[UHSK04] Peter Urban, Naohiro Hayashibara, Andre Schiper, and Takuya Katayama. Performance comparison of a rotating coordinator and a leader based consensus algorithm. In *SRDS '04: Proceedings of the 23rd IEEE International Symposium on Reliable Distributed Systems*, pages 4–17, Washington, DC, USA, 2004. IEEE Computer Society.

[Urb03] Péter Urbán. *Evaluating the Performance of Distributed Agreement Algorithms: Tools, Methodology and Case Studies*. PhD thesis, École Polytechnique Fédérale de Lausanne, Switzerland, August 2003. Number 2824.

[Var01] András Varga. The OMNeT++ discrete event simulation system. In *European Simulation Multiconference (ESM'2001)*, Prague, Czech Republic, june 2001.

[VH08] András Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and*

systems & workshops, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Vol05] Einar Wiik Vollset. *Design and Evaluation of Crash Tolerant Protocols for Mobile Ad Hoc Networks*. PhD thesis, University of Newcastle Upon Tyne, Newcastle, England, sep 2005.

[VR00] Paulo Veríssimo and Luís Rodrigues. *Distributed Systems for Systems Architects*. Kluwer Academic Publishers, 2000.

[WC02] Brad Williams and Tracy Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 194–205, Lausanne, Switzerland, jun 2002.

[WCYR06] Weigang Wu, Jiannong Cao, Jin Yang, and Michel Raynal. A hierarchical consensus protocol for mobile ad hoc networks. In *PDP '06: Proceedings of the 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP'06)*, pages 64–72, Washington, DC, USA, 2006. IEEE Computer Society.

[WCYR07] Weigang Wu, Jiannong Cao, Jin Yang, and Michel Raynal. Design and performance evaluation of efficient consensus protocols for mobile ad hoc networks. *IEEE Transactions on Computers*, 56(8):1055–1070, 2007.

[WL99] Jie Wu and Hailan Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 7–14, Seattle, Washington, United States, aug 1999.

[XSH08] Xiaodong Xian, Weiren Shi, and He Huang. Comparison of omnet++ and other simulator for wsn simulation. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, pages 1439–1443, June 2008.

[ZA05] Qi Zhang and Dharma P. Agrawal. Dynamic probabilistic broadcasting in manets. *Journal of Parallel and Distributed Computing*, 65(2):220–233, 2005.