UNIVERSIDADE FEDERAL DE PERNAMBUCO

GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO
CENTRO DE INFORMÁTICA

2010.2

ANÁLISE COMPARATIVA DE MECANISMOS DE ENDEREÇAMENTO PARA MANETS

TRABALHO DE GRADUAÇÃO

Aluno	Fernando Rodrigues de Souza Neto	{frsn@cin.ufpe.br}
Orientador	Dr. Djamel Fawzi Hadj Sadok	{jamel@cin.ufpe.br}
Co-orientador	Dr. Eduardo Souto	{esouto@ufam.edu.br}

14 de dezembro de 2010

Resumo

Desde o final do século XX, devido a grandes avanços tecnológicos, a maneira que as pessoas se comunicam mudou drasticamente. Atualmente, dispositivos móveis – como celulares, *pagers* – estão se tornando mais populares a cada ano. Esses aparelhos têm a capacidade de criar redes móveis – também conhecidas como *Mobile Adhoc Networks* (MANET) [9] – sem a existência de nenhuma infraestrutura prévia. Essa funcionalidade permite que as pessoas se comuniquem quase que instantaneamente.

No entanto, apesar de toda esta simplicidade, alguns problemas têm que ser resolvidos antes que a conexão seja alcançada. Dentre eles, está a configuração de endereços de rede. Para isso, diversos protocolos foram propostos para solucionar esta desafiadora tarefa de alocação de endereço em MANETs, entre eles estão *Prime DHCP*, *Dynamic Node Configuration Protocol* (DNCP) e *Prophet Address Allocation Protocol*. Estas soluções distribuem identificadores únicos para cada nó antes que qualquer forma de comunicação possa existir.

Antes de desenvolver qualquer aplicação, um desenvolvedor de redes deve escolher uma das opções disponíveis para desempenhar o processo de alocação necessário. A despeito de suas diferenças, cada protocolo tem suas vantagens e desvantagens, que podem fazê-lo mais adequado às diferentes aplicações. Este trabalho propõe uma análise, através de métricas coletadas de simulações, de alguns dos principais esquemas de alocação de endereços existentes. O objetivo é avaliar a eficiência desses protocolos em diferentes cenários de rede.

Palavras chave: MANET, ad hoc, redes, análise, simulação, dispositivos móveis.

Abstract

Since the late 1900's, as a result of the major technological advances the way people communicate have dramatically changed. Nowadays, mobile devices – e.g. cell phones, pagers – are becoming more popular every year. Those gadgets have the ability to create mobile networks – also known as Mobile Ad Hoc Networks (MANET) [9] – without the existence of any previous infrastructure. This functionality allows people to communicate almost immediately.

However, notwithstanding all that simplicity some issues have to be solved before connection is achieved. Among them is the network address configuration. To do so, several protocols have been proposed to solve this challenging task of address allocation in MANETs, such as Prime DHCP, Dynamic Node Configuration Protocol (DNCP) and Prophet Address Allocation Protocol. These solutions distribute unique identifiers for each node before any form of communication can exist.

Before developing any application, a network designer must choose one of the available options to perform the needed allocation process. Despite their differences, every protocol has its advantages and disadvantages, which can make it more suitable to diverse applications. This works proposes an analysis through metrics collected from simulations of some of the main existing approaches of address allocation. The goal is to evaluate the efficiency of these protocols in different network scenarios.

Keywords: MANET, ad hoc, networks, simulation, analysis, mobile devices.

Agradecimentos

Quando se chega ao final de uma etapa como esta, é salutar refletir e reconhecer os que ajudaram a transformar a longa caminhada em algo menos trabalhoso. Desta forma, agradeço primeiramente a meus pais, Ana e Fernando e a Bel, que me apoiaram e entenderam os diversos percalços que a graduação possa gerar.

Gostaria também de agradecer àqueles professores que contribuíram para minha formação e, em especial, agradeço aos que considero um exemplo de docência, Marcília Campos e Carlos Barros.

Também não poderia deixar de esquecer a pequena, mas forte, turma que por incontáveis vezes se reuniu na Batcaverna para uma revisão de última hora. A todos os engenheiros da Computação de 2006.1 e os que posteriormente se juntaram a nós.

Ao professor Djamel e à professora Judith Kelner pela oportunidade e apoio durante esse tempo no GPRT. Ao professor Eduardo Souto pela imensa ajuda durante este último ano e em especial neste trabalho. Também aos amigos do GPRT – Arara, Léo, Feitosa, Digão, Josias, Janga, Duda, Sidd, Cidão, Pigmeu, Petrônio, Guenzo – que não deixaram o caminho se complicar demasiadamente e pelas idas a Bigo.

Agradecimentos também aos amigos Renan, Perazzo, Gringo, Hugo, João Paulo, Breno, Fanf, Thiago e Marcelinho por todas as conversas e pelos momentos destes últimos anos.

A todos que ajudaram, obrigado!

Fernando Rodrigues

Índice

Lista de figuras

Lista de tabelas

1 Introdução

Comunicações sem fio existem a mais de um século, desde quando o cientista inglês transmitiu sinais de código Morse sobre uma distância de meio milha em 1894 [1]. Por quase quarenta anos [21], na maioria das aplicações – como rádio e televisão –, o processo de comunicação ocorria apenas uma única direção, ou seja, não havia intercâmbio de informações (interatividade) entre o emissor e receptor, uma ponta sempre envia e a outra sempre recebe. Desde o final da década de 1920 e início da década de 1930 – com o desenvolvimento e popularização de aplicações de comunicação sem fio de duas vias –, as comunicações sem fio têm se desenvolvido e estão presentes direta ou indiretamente na vida da população mundial desde o controle remoto de um portão eletrônico até o acesso a Internet via um celular.

Com tais avanços e com a popularização de dispositivos móveis como *notebooks*, celulares, *tablets* a forma de comunicação mudou drasticamente nos últimos anos. Esses dispositivos, cada vez mais imersos no nosso dia-a-dia, proveem capacidades de formar redes utilizando as mais diferentes tecnologias de comunicação sem fio como Bluetooth [2], Wi-Fi [3], Zigbee [4]. A formação dessas redes sem fio não requer a presença de uma pré-existente infraestrutura para serem formadas e são dinâmicas por natureza, sendo comumente chamadas por MANETs, do inglês *Mobile Adhoc Networks*.

Dentro de uma MANET, um dispositivo, também chamado de nó, pode muitas vezes mover-se livremente e randomicamente formando topologias temporárias que depende do reencaminhamento de pacotes para prover comunicação *unicast multi hop*.

Entretanto, antes que qualquer comunicação seja estabelecida é necessário que os nós sejam identificados na rede por um endereço único. Devido às características imprevisíveis e frágeis das MANETs, soluções tradicionais de alocação de endereços como o protocolo DHCP [5] – que depende da existência de um nó coordenador fixo – ou a pré-configuração dos endereços – que não atende aos requisitos de dinamicidade da rede - não podem ser utilizadas.

Para que uma rede móvel esteja consistente na maior parte de sua existência, ela espera que seu esquema de endereçamento possua certas funcionalidades como distribuição de endereços únicos, detecção e correção de partição e união de redes, e recuperação de endereços. Tais funcionalidades auxiliam na manutenção da rede, recuperando endereços sempre que possível e evitando que o espaço de endereçamento seja rapidamente consumido.

Com o objetivo de resolver tais desafios, diversos protocolos foram propostos como *Prime DHCP* [6], *Prophet Address Allocation* [7] e *Dynamic Node Configuration Protocol* [8], entre outros. Diante da variedade de propostas, um desenvolvedor pode se deparar com a situação de ter que escolher um protocolo para utilizar em sua rede. No entanto, tal escolha atualmente é feita baseada

apenas em descrições e características isoladas de cada protocolo devido à ausência de um panorama geral das opções.

1.1 Objetivos

O objetivo geral deste trabalho é produzir uma análise em vários aspectos de alguns dos principais protocolos de alocação de endereços para MANETs. Para tal, cada proposta será implementada e avaliada em diferentes cenários de rede.

Para este trabalho, cada implementação deverá oferecer as funções de distribuição de endereços, recuperação de endereços perdidos e o gerenciamento de eventuais partições e junções, ou *merges*, de topologias na rede.

Por agora, devido a grandes semelhanças (como a escolha de utilizar uma função de alocação distribuída) – o que potencialmente pode dificultar a escolha de um – apenas os protocolos *Prime*, *Prophet* e DNCP serão avaliados. Este trabalho também se propõe a continuar o projeto já iniciado na tese de mestrado do Centro de Informática da Universidade Federal de Pernambuco de Rafael Aschoff [8], que especifica e propõe o protocolo DNCP. Cada avaliação será feita de acordo com a fórmula de análise de desempenho presente em [10].

1.2 Sumário dos capítulos

Este trabalho foi organizado da seguinte forma. A Seção 2, apresenta detalhes sobre alocação de endereços e os principais tipos de abordagens existentes. Na Seção 3, as soluções escolhidas para serem analisadas serão explicadas em detalhes.

A Seção 4 mostra as questões de implementação que foram cruciais no desenvolvimento do trabalho e descreve as métricas de avaliação escolhidas. Na Seção 5, os cenários de simulação e os resultados obtidos serão explicados minuciosamente.

Por fim, a Seção 6 aponta as conclusões obtidas e a direção futura que o trabalho deve tomar.

2 Referencial teórico

Este capítulo apresenta o embasamento teórico deste trabalho, reunindo informações sobre alocação de endereços e descrevendo a importância de esquemas de endereçamento eficientes em redes móveis. Será apresentada uma classificação, os conceitos e protocolos propostos para tentar resolver o problema de endereçamento em MANETs.

2.1 Gerenciamento de endereços

Redes móveis ad hoc têm uma característica intrínseca que é a comunicação direta entre os nós, sem a necessidade de uma unidade central que coordene a rede e exerça o papel de reencaminhar e rotear os pacotes da rede [11]. Diferentemente das redes fixas, nas MANETs todos os nós que formam a rede devem dividir essas tarefas, visto que não possuem uma infraestrutura organizadora para que a conectividade e a comunicação entre todos os nós existam.

Um dos papéis mais importantes nas redes MANETs é o roteamento. Para que uma informação possa ir de um nó fonte para o nó destino que não estão em dentro do alcance de comunicação, os pacotes precisam ser transmitidos de nó para nó até chegar ao seu destino final. Por exemplo, a Figura 2-1 exibe um cenário em que o dispositivo A não tem comunicação direta com o dispositivo D. Assim, para que a comunicação entre eles possa ser estabelecida, dependendo das escolhas feitas pelo protocolo de roteamento utilizado, os nós B ou C devem ser utilizados como roteadores.

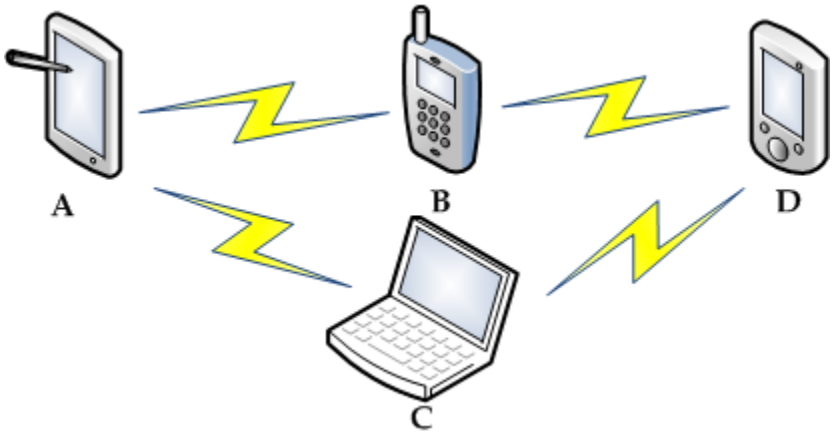

Figura 2-1 Exemplo de uma MANET

No entanto, para que protocolos de roteamento possam achar o melhor caminho entre os dispositivos, cada nó tem que possuir um único identificador dentro da rede [12]. Em diversas abordagens, a identificação escolhida é o

endereço IP, que deve então ser distribuído univocamente para cada nó que deseje ingressar à rede.

Devido às características das MANETs, – onde os dispositivos se auto-organizam criando topologias temporárias e arbitrárias – a distribuição de endereços também tem que seguir as mesmas regras, devendo ser feita de forma automática.

Em MANETs, abordagens que utilizam uma entidade central que tenha controle total do endereçamento também não se mostram efetivas [13]. Este fato ocorre devido às características de mobilidade da MANET, ou seja, com grande chance, um nó que deseje entrar na rede não conseguirá encontrar a unidade central em seu raio de comunicação. Desta forma, forçosamente ele deverá utilizar mensagens multihop desde o início de sua vida na rede, gerando desnecessário *overhead* que poderia ser evitado caso a alocação fosse distribuída e um de seus vizinhos de um salto pudesse alocar endereços.

A Figura 2-2 ilustra um típico processo de obtenção de endereços em uma MANET. Em (a), o nó E que ainda não tem um endereço não pode estabelecer uma conexão direta com os nós já configurados A, B, C e D. Já em (b), o esquema de endereçamento da rede deve garantir a alocação de um endereço único para E, o qual pode ter sido oferecido por qualquer dos outros nós já configurados. Somente após a obtenção do novo endereço, o nó E consegue estabelecer a comunicação com os demais nós da rede.

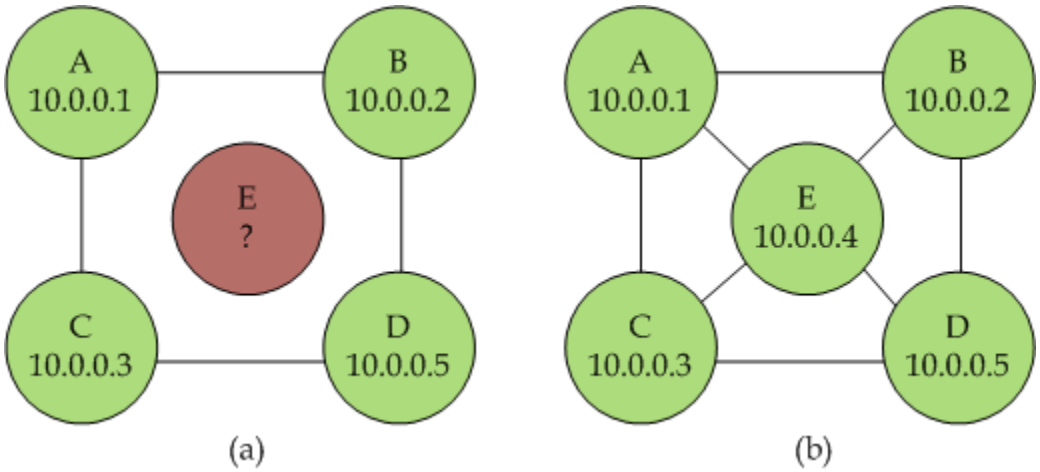

Figura 2-2 (a) Nó E ainda sem identificação entra na rede. (b) Após configuração, o nó E estabelece conexão com os outros nós.

Além de ter que dinamicamente atrelar identificadores aos dispositivos, a rede ainda precisa ter a capacidade de exercer a importante tarefa de manter a unicidade dos endereços a fim de manter a integridade do espaço de endereçamento. Para tal, os nós constituintes da rede devem ter a capacidade de:

(i) Detectar endereços repetidos, que pode ocorrer quando duas redes distintas se juntam ou quando erroneamente, um mesmo endereço é

oferecido para dois nós distintos. O primeiro cenário também é denominado de *merge* de redes.

(ii) Recuperar endereços, que pode ocorrer quando um endereço alocado já não está mais em uso. Por exemplo, devido à partição de uma rede ou no caso de algum nó apresentar funcionamento incorreto.

A Figura 2-3 ilustra três situações críticas quando se refere ao processo de manutenção dos endereços da rede. No primeiro cenário, o nó E, indicando seu mau funcionamento com a cor amarela na ilustração, deixa a rede e sua saída pode ser causada pelos mais diversos fatores, como por exemplo, falta de bateria ou defeito no próprio dispositivo. Caso a saída seja abrupta, muito provavelmente o nó não terá a oportunidade de indicar a sua saída e, consequentemente, o seu endereço ficará inutilizado. Devido à escassez de endereços, um procedimento para a recuperação desse endereço recém-perdido será necessário. Segundo, os nós A e C, que antes pertenciam à mesma rede dos nós B e D, temporariamente deixam a rede, retornando em seguida. Isto pode causar uma inconsistência caso seus endereços sejam recuperados muito cedo. Por fim, os nós F e G que antes compunham a rede 2, efetuam o *merge* com os nós da rede 1. Nesta operação, é necessário verificar a possível existência de endereços iguais nas redes que se juntaram para então poder corrigir as ocasionais repetências.

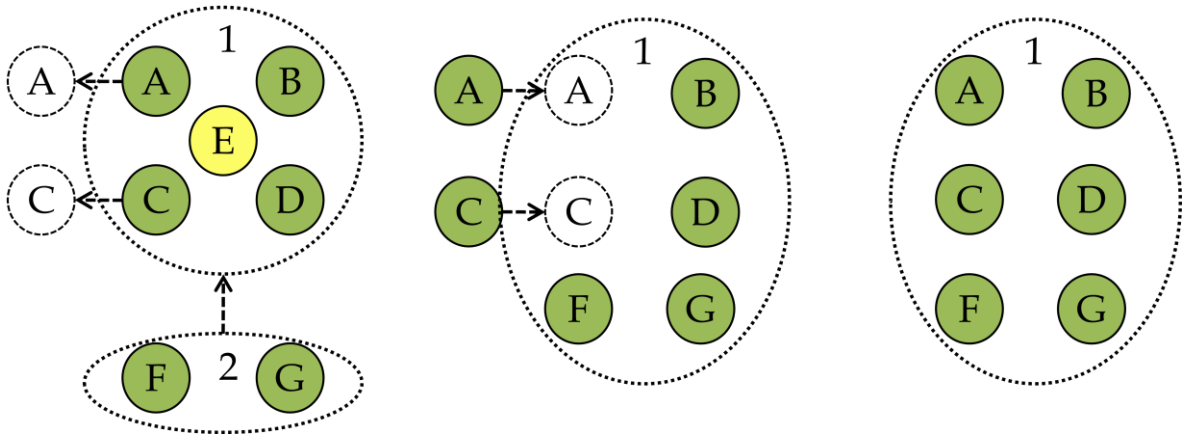

Figura 2-3 Nó E (em amarelo) morrendo, partição e *merge*.

2.2 Classificações e principais abordagens

Como previamente mostrado, a tarefa de alocação correta de endereços para dispositivos constituintes de uma rede móvel ad hoc é uma tarefa crucial para o seu funcionamento correto. Como tal, essa área presenciou muitos estudos que contribuíram para produzir diversas soluções. No entanto, apesar de suas diferenças, as abordagens podem ser divididas em três grupos: s*tateless, stateful* e híbridas [8].

2.2.1 Abordagens *stateless*

Este tipo de solução é caracterizado por não manter informações sobre a topologia da rede ou da situação do espaço de endereçamento. Neste cenário, os nós podem se autoconfigurar apenas com informações locais e também oferecer endereços a nós ingressantes na rede sem garantia de endereço único. Portanto, pela ausência de informações sobre os endereços livres e os já utilizados, é necessária a utilização periódica de algum mecanismo para detectar a repetição de endereços. Tal mecanismo é comumente denominado de DAD, do inglês *Duplicate Address Detection* [23].

Uma simples maneira de obter um endereço de rede é permitir que o nó se autoconfigure através da geração de um identificador randomicamente [14]. Este artifício é o princípio da solução conhecida como *Strong Duplicate Address Detection* (SDAD) [15] que sugere que o nó obtenha dois endereços randomicamente gerados para a sua configuração. O primeiro endereço, a ser utilizado como temporário durante a configuração, será obtido dos primeiros 2047 endereços da rede 169.254.0.0/16. O segundo endereço (que será definitivo caso a configuração seja bem sucedida) será escolhido randomicamente dos 63487 endereços restantes da mesma rede anterior. Ao escolher os dois endereços, o nó envia mensagens *broadcast* na rede com o almejado endereço, na intenção de assegurar que nenhum outro dispositivo já tenha escolhido o mesmo endereço. Em caso de resposta positiva, o processo é reiniciado, senão, o nó é configurado.

Diferente do protocolo SDAD – que procura por endereços duplicados apenas na fase de configuração –, a abordagem descrita em [16] conhecida como *Weak Duplicate Address Detection* (WDAD) estende a busca durante toda a vida da rede, utilizando-se das mensagens trocadas pelos protocolos de roteamento. Para detectar as duplicações, cada nó tem outro identificador único que é enviado em conjunto com o endereço IP, assim ao receber uma mensagem destinada a um IP conhecido, mas com identificador diferente do esperado, um nó detecta e avisa a rede sobre a colisão.

2.2.2 Abordagens *stateful*

Diferentemente da abordagem anterior, uma solução *stateful* conserva informações sobre a situação atual do espaço de endereçamento. Desta forma, os nós componentes da rede podem assegurar que os endereços oferecidos aos requisitantes são novos ou reciclados – através de algum mecanismo de recuperação de endereços. Esta segurança permite que abordagens *stateful* abram mão de procedimentos de detecção de endereços duplicados em uma mesma rede.

As soluções *stateful* ainda podem ser subdivididas em quatro grupos dependendo da forma que escolhem para organizar e manter as informações sobre os endereços [24]. O primeiro subgrupo corresponde às abordagens onde uma unidade central é utilizada para controlar todas as informações em uma única

tabela. Este tipo de abordagem não se encaixa nos requisitos de uma rede MANET porque além do acréscimo de sobrecarga na rede na troca de informações entre a unidade central e os nós comuns, a rede fica susceptível a falha no nó central.

O segundo subgrupo é formado pelas soluções que distribuem a tabela entre todos os nós. Assim, ao manter a tabela em todos os nós, adapta-se a primeira solução para alguns requisitos de redes MANET, no entanto ainda mais sobrecarga é adicionada para poder manter todas as tabelas atualizadas.

Em tese, as melhores alternativas para redes sem fio móveis são tabelas disjuntas e função de alocação distribuída, pois conseguem garantir a unicidade integridade dos endereços distribuídos, evitando o acréscimo de sobrecarga desnecessário para a manutenção do status do espaço de endereçamento. A abordagem que utiliza tabelas disjuntas divide a tabela de todo o espaço de endereçamento em conjuntos disjuntos e as distribui entre os nós. Já as soluções quem utilizam funções de alocação distribuída emprega o uso de uma função que garante a geração de endereços diferentes dependendo dos parâmetros utilizados pelos nós. Desta forma, ambas as abordagens evitam a sobrecarga desnecessária, mantendo as informações congruentes e atualizadas nos nós da rede.

Uma abordagem *stateful* centralizada pode ser encontrada em [17], conhecida como *Agent-Based Addressing* e utiliza um nó coordenador como único responsável para distribuir os endereços aos demais. Periodicamente mensagens de verificação são enviadas pelo coordenador para checar se endereços já distribuídos não estão mais sendo utilizados. Os nós ingressantes devem esperar por uma mensagem de verificação para então estabelecer uma conexão com o nó coordenador com o intuito de requisitar um endereço. Cada coordenador mantém um identificador da rede, ou NID, que será utilizado para detectar *merges*. Ao detectar um *merge*, o coordenador da rede com menor número de nós abrirá mão do seu endereço e da coordenação de sua rede, avisando a todos seus nós para requisitar novamente endereço ao novo coordenador. Caso os nós não recebam as mensagens periódicas de verificação, dinamicamente é assumido que o coordenador não está mais presente na rede e um novo nó se auto-intitula como novo coordenador.

Um exemplo de uma solução adequada para MANETs é conhecida como *Prime DHCP* [6] que utiliza a abordagem de função de alocação distribuída. A função se baseia no fato que cada número inteiro positivo pode ser construído unicamente através da multiplicação de números primos [20]. Assim, cada nó gera endereços distintos e elimina a necessidade mecanismos *DAD*. O primeiro nó, raiz da rede, se configura com identificador 1 e gera identificadores utilizando seu endereço e a função para os nós requisitantes. Mensagens de reciclagem são enviadas pela raiz para verificar se os endereços alocados ainda estão sendo utilizados. Como na abordagem anterior, caso um nó não receba as mensagens de reciclagem durante um determinado tempo, ele se torna a raiz de uma nova rede. Outra utilidade das mensagens de reciclagem é para detectar *merges*, caso uma raiz receba respostas às mensagens de reciclagem de nós diferentes utilizando o mesmo

endereço, ele escolhe um dos conflitantes para reiniciar o processo de requisição de endereço.

Uma recente abordagem, conhecida como *Dynamic Node Configuration Protocol* (DNCP) [8], também utiliza uma função de alocação distribuída. O DNCP se baseia no sistema de numeração binário para poder gerar diferentes conjuntos de endereços a partir de cada nó. Novamente, endereços repetidos não são gerados em uma mesma rede e o nó criador da rede se configura com o identificador 1. Neste protocolo, além de requisições locais destinadas aos nós no alcance da comunicação, mensagens de descoberta remotas podem ser utilizadas na ausência de uma oferta de endereço após tentativas de descobertas locais. Mensagens periódicas de *Hello* são utilizadas para que cada nó possa saber seus vizinhos e tenha uma visão ampla da topologia da rede. Ao contrário do *Prime DHCP*, o processo de recuperação de endereços do DNCP só é iniciado quando um nó ingressante não consegue obter ofertas de endereços nem local nem remotamente. Assim, ao perceber a presença de uma rede – através do recebimento de mensagens de *Hello* – o nó que não recebeu ofertas de endereços após diversas tentativas, inicia o processo de recuperação de endereços na tentativa de encontrar um endereço que foi alocado, mas não está mais sendo utilizado. As mensagens de *Hello* também são utilizadas para detectar *merges*, que são solucionados da seguinte forma: o nó que recebe um *Hello* com identificador de rede diferente do seu, verifica se a outra rede tem mais nós a sua rede (através das estimativas contidas na mensagem de *Hello)*. E em caso positivo ele desiste de seu endereço, avisa aos seus vizinhos e reinicia o processo de requisição de endereços.

Outro método de alocação de endereços que utiliza funções de alocação distribuída é conhecido como *Prophet Address Allocation [7]*, que recebe o nome porque o nó criador da rede, conhecido como profeta, tem conhecimento de quais serão os possíveis endereços a serem oferecidos no início da rede. A peça chave da eficiência desse protocolo é baseada na periodicidade da função escolhida para distribuir os endereços. O algoritmo garante que a probabilidade de dois nós diferentes oferecerem o mesmo endereço é desprezível se a função for escolhida corretamente. Utilizando-se da construção de números inteiros através de número primos e da aritmética modular, o protocolo sugere a utilização de uma função que obtém eficiência muito alta. Bem como o DNCP, o protocolo *Prophet* utiliza-se de identificadores de rede e das mensagens enviadas por protocolos de roteamento para detectar e corrigir eventuais *merges*.

2.2.3 Abordagens híbridas

Este último tipo de solução tenta, como o nome sugere, utilizar as melhores sugestões das abordagens *stateless* e *stateful* para poder obter o melhor desempenho possível da utilização do espaço de endereçamento. A maioria das abordagens híbridas utiliza uma ou mais tabelas distribuídas pela rede em conjunto com

esquemas de detecção de endereços duplicados, sendo assim, algoritmos mais complexos.

Uma abordagem híbrida que mescla mecanismos empregados na abordagem *stateless* SDAD e na *stateful Agent-Based Addressing* se chama *Hybrid Centralized Query-based Autoconfiguration* (HCQA) [18]. No HCQA existe um nó controlador que tem o poder de alocar os endereços que os nós requisitem, chamado de agente. O primeiro nó ao criar a rede se intitula o agente e periodicamente envia um identificador da rede em *broadcast*. Um nó ingressante que queira se configurar, envia uma requisição ao nó agente que fará a verificação de endereços duplicados, utilizando assim um mecanismo DAD. Quando um merge é detectado, os nós agentes se comunicam para identificar os possíveis conflitos e decidir qual dos dois deverá reiniciar o processo de requisição de endereço.

Já a abordagem apresentada em [19], conhecida como *Passive autoconfiguration for mobile ad hoc networks* (PACMAN), é caracterizada pela tentativa de reduzir ao máximo o *overhead* a ser acrescentado na rede pelo protocolo de endereçamento. Um nó que utilize o protocolo PACMAN se autoconfigura com um endereço IP e passivamente aproveita as mensagens provenientes dos protocolos de roteamento para poder identificar os problemas na rede como endereços duplicados. *Merge* e partições são lidados utilizando-se do mecanismo de detecção de endereços duplicados passivo (em inglês *Passive Duplicate Address Detection*, PDAD). Apesar da grande redução do *overhead*, esta abordagem é complexa e depende altamente em mudanças nas mensagens de protocolos de roteamento, podendo levar a inconsistências na rede por grandes períodos de tempo.

2.3 Comentários

Este Capítulo demonstrou as características intrínsecas à MANETs que impõem requisitos mínimos aos protocolos de endereçamento. Além de destacar que é necessário um protocolo descentralizado, que garanta a unicidade de endereços e que lide com os problemas de partição e *merge* de redes. Além disso, um esquema de endereçamento para MANETs deve fornecer um procedimento eficiente de manutenção de endereços com o objetivo de evitar o desperdício de recursos (como largura de banda, energia gasta nas mensagens e de disponibilidade de endereços) dos nós componentes da rede.

Os três protocolos (*Prime*, *Prophet* e DNCP) descritos neste capítulo foram selecionados por atenderem aos requisitos exigidos em cenários de MANETs. Por serem soluções recentes, estre trabalho propõe uma análise comparativa através de simulações.

3 Soluções de endereçamento

Este capítulo detalha o funcionamento das três soluções de endereçamento escolhidas para análise neste trabalho, *Prophet Allocation*, *Prime* DHCP e DNCP.

3.1 *Prophet Allocation*

A alocação *Prophet* é assim conhecida porque o nó criador da rede, conhecido como profeta, pode previamente calcular como será a distribuição de endereço de cada futuro integrante da rede. Denominando o espaço de endereçamento de R, a abordagem sugere a utilização de uma função, $f(n)$, que gera diferenças sequências de inteiros, dividindo do domínio de R.

O estado inicial de $f(n)$ é chamado de semente e define toda a alocação futura. Atualizando corretamente o estado de $f(n)$, diferentes sementes gerarão sequências distintas. Caso R seja suficientemente grande, as sequências geradas a partir de $f(n)$ devem obedecer às seguintes regras para um correto e satisfatório funcionamento:

(i) Em uma determinada sequência, o intervalo de repetição de um número é extremamente longo.

(ii) A probabilidade de que um mesmo número possa ser gerado em sequências diferentes iniciadas por sementes diferentes durante um determinado intervalo é desprezível.

Assim um protocolo de endereçamento pode ser descrito utilizando-se das propriedades das sequências de $f(n)$. O fluxo de configuração e oferta de endereços é mostrado na Figura 3-1.

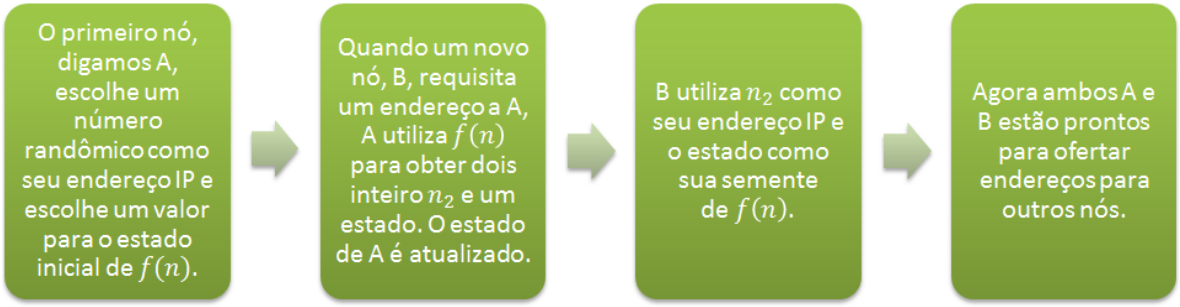

Figura 3-1 Fluxo de um algoritmo de autoconfiguração

Uma maneira de garantir a unicidade dos endereços mesmo com probabilidade desprezível de repetição é utilizar as informações do nó profeta para calcular quando aconteceriam os conflitos e informar aos nós de antemão, antes que os conflitos ocorram. Caso muitos conflitos ocorram, o nó profeta pode escolher outra semente e reiniciar o processo. Além disso, o protocolo *Prophet*

afirma que reclamação de endereços é desnecessária, pois as sequências são cíclicas e os endereços serão eventualmente – depois de um largo período de tempo – oferecidos novamente.

Partições de rede não produzem nenhum problema, pois – quando alguns nós saem da rede, os demais nós continuarão gerando endereços diferentes devido às características previamente descritas das sequências de $f(n)$. Os *merges* de redes distintas são solucionados utilizando uma propriedade de cada rede. Cada nó profeta gera no início da rede um identificador denominado *Network Identificator* ou NID que é repassado a cada nó ainda no processo de alocação. Quando um nó detecta a presença de uma rede com NID distinto da sua, verifica se o valor de seu NID é menor que o da outra rede, em caso positivo, inicia o processo de requisição na segunda rede e informa aos seus vizinhos para que façam o mesmo, caso contrário esta mensagem será ignorada. A máquina de estados do algoritmo de solução de junção de redes pode ser verificada na Figura 3-2.

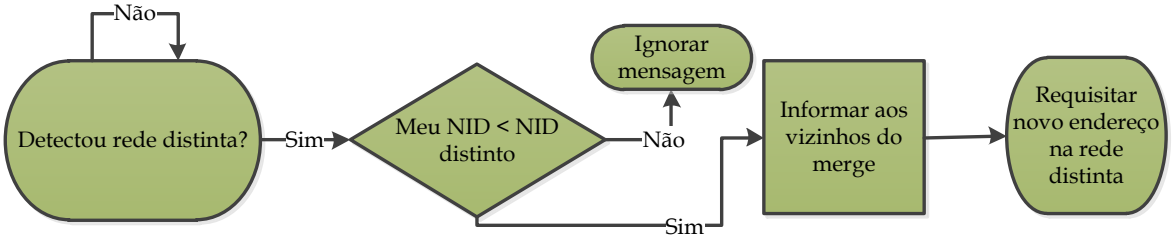

Figura 3-2 Algoritmo de solução de *merge*

No entanto, o ponto chave do algoritmo é a eficiência da função $f(n)$, afinal são suas características que garantirão a integridade dos endereços e a periodicidade das sequências. Como modelar uma função que satisfaça à risca os dois requisitos descritos acima é extremamente custoso e complexo, a sugestão é utilizar uma função que as satisfaça aproximadamente. Assim, a sugestão é utilizar a seguinte propriedade dos números inteiros positivos:

$$n = \prod_{i=1}^{k} p_i{}^{e_i}, \text{ onde } p_i \text{ são primos e obedecem } p_1 < p_2 < \cdots < p_k.$$

Os expoentes são números naturais. Assim, qualquer número n pode ser descrito através de uma k-tupla $(e_1, e_2, \cdots, e_k)$ e o intuito da alocação de endereços passa a ser gerar diferentes k-tuplas para cara nó.

Por exemplo, caso $k = 4$, o primeiro nó escolhe um número randômico a como seu endereço e $(0,0,0,0)$ como semente. Desta forma, pode-se descrever $f(n)$ de tal forma que cada nó seja identificado por $\big(endereço, (e_1, e_2, e_3, e_4)\big)$. Onde $endereço = \big((a + 2^{e_1}3^{e_2}5^{e_3}7^{e_4}) \bmod r\big) + 1$ e r equivale ao tamanho do espaço de endereçamento. As regras para atualização dos estados são:

(i) O elemento sublinhado da tupla é acrescentado de uma unidade.

(ii) O estado inicial do novo nó é copiado do nó ofertante e o elemento sublinhado é deslocado para a direita.

A Figura 3-3 mostra a geração de identificadores únicos e atualização dos estados para uma rede com quatro nós.

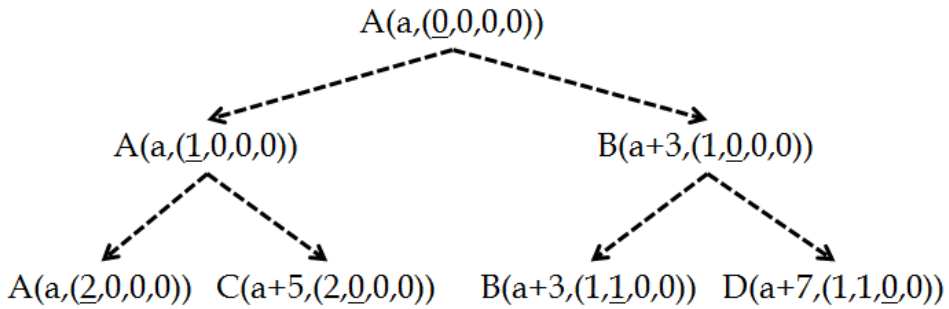

Figura 3-3 Geração de endereços e atualização dos estados

Em cenários reais, o r deve ser bem maior e consequentemente também k, produzindo assim tuplas maiores. Para acelerar o algoritmo, *arrays* de primos podem ser pré-calculados para a utilização na alocação.

O protocolo *Prophet* é descrito em seis passos bem definidos, podendo ser verificados como uma máquina de estados na Figura 3-4. Ao iniciar, o nó muda seu estado de **Não inicializado** para **Em espera**. Neste processo, o nó inicia o envio de requisições de endereços em *broadcasts* de apenas um salto, enviando também na mensagem seu endereço MAC para que a resposta possa ser efetuada em *unicast*. Caso não receba nenhuma resposta, o nó permanece **Em espera** e repete a requisição por até k vezes. Se uma oferta for recebida, o nó configura seu endereço, seu estado inicial e o NID, todos contidos na resposta. Se nenhuma resposta for recebida, o nó cria uma nova rede e se torna profeta. Em ambos os cenários, o estado é mudado para **Configurado**. Durante este estado, o nó enviará periodicamente mensagens de *Hello* que podem ser encapsuladas nas mensagens de roteamento, responde a requisições de endereços e atualiza seu estado de acordo com o processo descrito anteriormente. Caso uma mensagem de *Hello* seja recebida e contenha uma NID distinta, o nó passa para o estado de **Solução de merge** onde executa os passos já descritos, ao final, volta ao estado **Configurado**. Caso deseje sair da rede, o nó retorna ao estado **Não inicializado**.

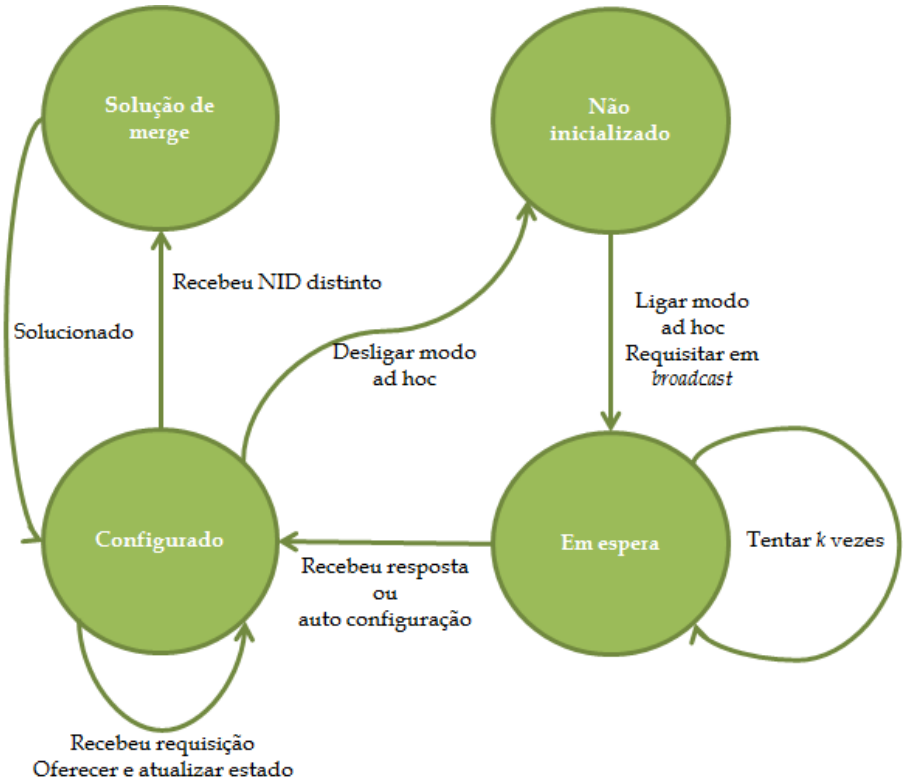

Figura 3-4 Máquina de estados do protocolo *Prophet*

3.2 *Prime* DHCP

Bem como a abordagem anterior, o protocolo de endereçamento *Prime* DHCP se utiliza da propriedade de geração única de números inteiros positivos através da multiplicação de números primos. Assim uma função de alocação distribuída, denominada na proposta de *Prime Number Address Allocation* (PNAA), é sugerida para distribuir, sem conflitos, os endereços aos nós que desejem participar da rede. A função de distribuição é descrita da seguinte forma:

(i) O primeiro nó da rede, chamado de raiz, se autoconfigurará com o identificador igual a 1. Então, poderá distribuir identificadores com números primos em ordem crescente.
(ii) Para um nó comum, ou seja, não raiz, ele poderá oferecer endereços correspondentes à multiplicação de seu endereço com números primos, começando pelo maior primo componente da fatoração de seu identificador.

Desta forma, dois nós não podem gerar endereços iguais, eliminando assim a necessidade de qualquer mecanismo de duplicação local. A oferta de endereços pode ser feita com pouca complexidade, utilizando apenas um vetor de números primos e que cada nó tenha registrado o seu estado, ou seja, o último número

primo utilizado. Um exemplo da árvore de alocação de endereços utilizando PNAA pode ser vista na Figura 3-5.

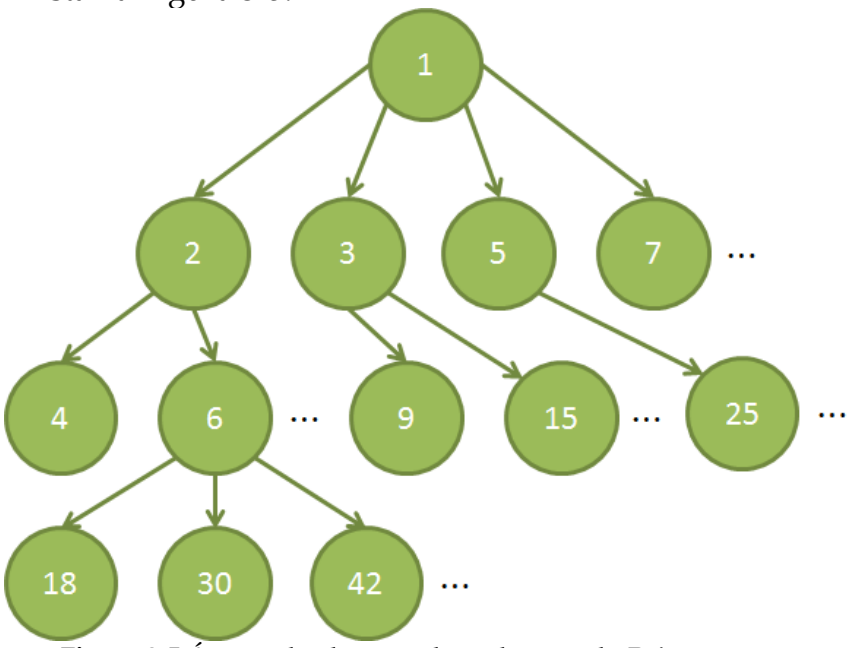

Figura 3-5 Árvore de alocação de endereços do *Prime*

O protocolo de endereçamento segue regras bem definidas para poder garantir que os endereços sejam bem distribuídos. Quando um nó deseja ingressar em uma MANET, ele enviará via *broadcast* uma mensagem de descoberta de servidores denominada *DHCP_Discover*. Cada nó já previamente configurado que receber as mensagens de descoberta passará a funcionar como um DHCP *Proxy* para o nó requisitante. No entanto, ao invés de reencaminhar as mensagens diretamente, cada nó *Proxy* calcula o seu próximo endereço a ser oferecido e o envia em uma mensagem *DHCP_Offer* ao nó requisitante. Se o nó não puder oferecer o endereço, ele reencaminhará a mensagem de descoberta ao seu pai – o nó que lhe forneceu endereço. Quando o nó requisitante receber as repostas com ofertas, escolherá o menor endereço ofertado – para prevenir que a árvore cresça rapidamente – e enviará uma mensagem de *DHCP_Request* em *broadcast*. Essa mensagem só será propagada na rede caso haja necessidade, ou seja, caso a oferta tenha sido feita através do ancestral de algum nó no raio de comunicação.

Ao receber um *DHCP_Request*¸ o nó que ofertou o respectivo endereço atualiza seu estado atual e envia por fim um *DHCP_Ack* em *unicast* para o requisitante, finalizando o processo de alocação de endereço. É previsto na especificação do protocolo uma mensagem para caso o nó deseje sair "educadamente" da rede, sendo chamada de *DHCP_Release*. Essa mensagem deve ser enviada em *unicast* para o seu pai, para acelerar a recuperação de endereços e evitar que endereços sejam perdidos. Desta forma, na próxima alocação o pai do nó que deixa a rede pode oferecer o endereço recentemente liberado. Caso o nó a sair seja a raiz da rede, ele avisa ao seu último filho alocado para que ele se torne a

nova raiz. No entanto, apesar de toda esta descrição, não é obrigatório e muito menos esperado que os nós saiam da rede "educadamente".

Exceções dos tipos: saída sem aviso, perda de pacotes, *merge* de redes e partições podem ocorrer. O protocolo, com o intuito de solucionar perda de pacotes, sugere que as mensagens sejam reenviadas após um tempo caso nenhuma resposta seja recebida. Partições não necessitam que nenhum mecanismo se dedique a resolvê-las, pois mesmo separados, os nós não alocam endereços iguais. Para evitar a perda de endereços devido a saídas sem aviso, o nó raiz da rede envia periodicamente mensagens de *DHCP_Recycle* que devem ser propagadas na rede e respondidas por cada nó da rede com seu status de alocação atual. Assim, a raiz pode reconstruir a árvore de alocação e identificar eventuais saídas não reportadas. Caso ocorram, a raiz avisará sobre a saída aos pais de cada endereço recuperado no processo.

Merges são identificados utilizando-se também um identificador de rede, gerado pela raiz no momento de criação da rede. Assim ao receber *DHCP_Recycle* de nós de outras redes, a raiz poderá identificar possíveis conflitos com sua rede. Além de escolher qual dos nós conflitantes deverá reiniciar a requisição de endereço. Como dito antes, partições não geram problemas de conflito, no entanto a rede particionada precisará ainda de nova raiz, assim, cada nó que não receber diversas mensagens de *DHCP_Recycle*, deverá auto intitular-se como nova raiz após um tempo inversamente proporcional ao seu endereço, o que visa transformar os nós com maior endereço (e possivelmente nenhum filho) na nova raiz.

3.3 *Dynamic Node Configuration Protocol* (DNCP)

O protocolo DNCP se propõe a focar em dois importantes pontos da tarefa de prover e garantir a conectividade em uma rede móvel sem fio: alocação e manutenção de endereços. Portanto, o protocolo se propõe tanto a distribuir identificadores únicos para cada nó componente da rede quanto a preservar a consistência do espaço de endereçamento durante toda a vida da rede.

Semelhante aos outros protocolos previamente descritos, DNCP também sugere a utilização de uma função de alocação distribuída. Desta vez, em vez da propriedade de formação única de números inteiros positivos através da multiplicação de números primos, a função se baseia na notação posicional dos números e também na aritmética binária.

A função definida pelo protocolo, chamada de *Binary Number Generator* (BNG), é adaptável para qualquer notação posicional. No entanto para facilitar e acelerar o cálculo em computadores foi escolhido o sistema binário. Cada nó pertencente a uma rede que utilize o protocolo DNCP será marcado por dois números identificadores, chamados de *seed* e *status*. Assim, quando um nó já configurado oferecer um endereço, ele fornecerá tanto o *seed* quanto *status* ao novo nó. A distribuição de endereços funciona da seguinte maneira:

(i) O nó criador da rede, também conhecido como nó raiz, se autoconfigurará com uma *seed* igual a 1 e um *status* igual à zero. Este nó também atribuirá um número randômico como o NID.

(ii) A oferta de endereços é feita baseada na *seed* – valor constante – e na situação atual do *status*. A função é descrita a seguir, onde n é a *seed* e s é o status: $f(n,s) = (n + 2^s, s + 1)$. Assim, a tupla resultante da função é ofertada ao nó requisitante e o *status* do nó ofertante é incrementado por um.

Na Figura 3-6 podemos ver a árvore de alocação de endereços utilizando a função BNG.

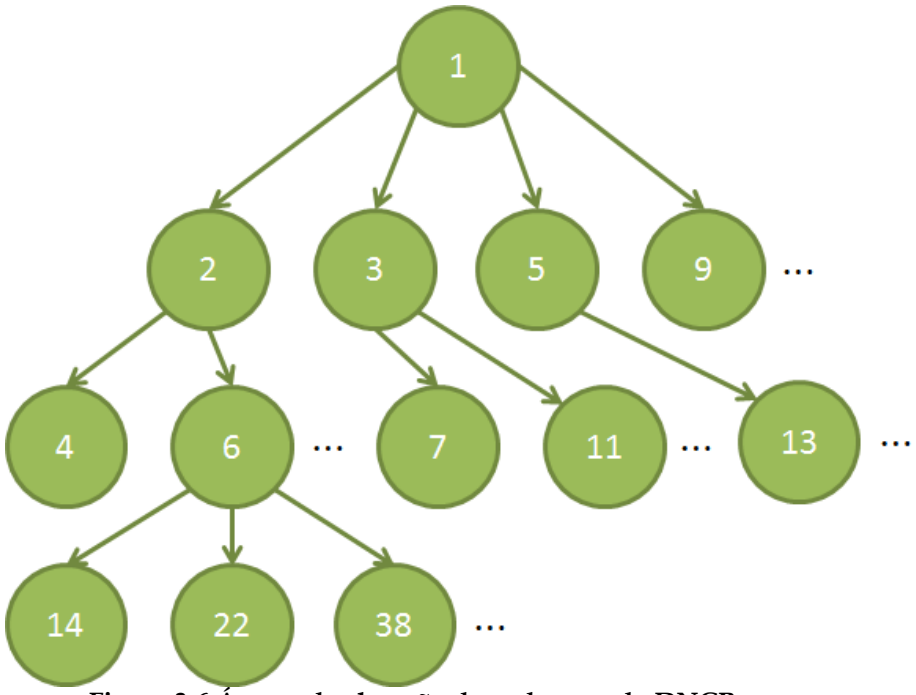

Figura 3-6 Árvore de alocação de endereços do DNCP

O processo de requisição descrito pelo protocolo DNCP, com o intuito de sempre encontrar um endereço disponível caso exista, prevê já para as primeiras fases requisições de endereço locais, remotas e inclusive recuperação de endereços inutilizados. Um nó que deseje conectar-se à MANET protocolada pelo DNCP primeiramente envia uma mensagem em *broadcast* chamada *Server Discover* na tentativa de encontrar nós já configurados em seu alcance de comunicação. Os nós que receberem uma mensagem *Server Discover* e puderem oferecer um endereço, calculam o valor a ser oferecido utilizando a previamente descrita função BNG e respondem diretamente ao nó requisitante através de uma mensagem chamada *Address Offer*. Possivelmente, um nó requisitante recebe mais de uma oferta de endereço. Quando isso ocorre, o nó requisitante deve escolher o menor *seed* para evitar que árvore de alocação cresça rapidamente. Após a escolha da melhor *seed*,

uma mensagem em *unicast* de *Address Request* ao nó ofertante, que ao recebê-la checará se o endereço requisitado é o que foi ofertado, em caso positivo respondendo com uma mensagem de *Address Reply* e atualizando seu *status.* Por fim, o nó requisitante configura-se com o endereço ofertado mediante o recebimento do *Address Reply*.

Como as mensagens de *Server Discover* podem ser perdidas, elas devem ser por k vezes. Caso o nó requisitante encontre um servidor, mas não receba o *Address Reply*, o valor de k deve ser reiniciado. Ainda assim, é possível que um nó não consiga encontrar servidores com endereços ainda não alocados no seu alcance de comunicação. Neste caso, o nó deve iniciar o processo de busca por servidores remotos, enviando mensagens denominadas *Remote Server Discover,* que serão reencaminhadas pelos nós que não tiverem mais endereços para ofertar. O resto da comunicação é semelhante à descoberta local, no entanto, é feita remotamente, utilizando nós intermediários como *proxys*.

No caso de uma rede com um número de nós se aproximando ao tamanho do espaço de endereçamento, é possível que um nó não receba ofertas nem locais nem remotas, impedindo assim a sua configuração. Neste caso, os nós que detectarem a presença de uma rede – utilizando-se das mensagens de *Hello,* que são enviadas pelos nós configurados, periodicamente– poderá iniciar o mecanismo de recuperação de endereços. Ao retardar este processo até o momento de necessidade, o protocolo evita adicionar *overhead* desnecessário previamente na rede. A mensagem denominada *Reclamation Request* é inicialmente enviada em *broadcast* pelo nó não configurado e então propagada por todos os outros nós da rede. Os nós configurados então devem produzir uma mensagem chamada *Reclamation Reply* com informações da sua *seed* e do seu *status* e enviá-la diretamente ao nó que iniciou o processo de reclamação de endereços. Ao receber as respostas, o nó deverá calcular os valores de *status*, s_0, de cada nó, o que pode ser feito com a Equação 3-1.

$$s_o = min\{j | 2^j \geq n\}, j \in \aleph$$

Equação 3-1 - Função para calcular *status* inicial

Depois de todos os cálculos, a árvore de alocação deve ser reconstruída para poder identificar eventuais faltas, ou nós que deveriam estar presentes na rede e não responderam, identificando assim a possibilidade de um endereço a ser recuperado. Caso exista algum endereço disponível, o nó escolhe o menor deles para ser o seu e encaminha aos demais nós da rede qual é a situação atual da rede, informando possíveis outros endereços que não estão mais sendo utilizados. No entanto, se não houver nenhum endereço a ser recuperado, o nó parte para a criação de uma nova rede, utilizando o processo de autoconfiguração. Isto também ocorreria caso nenhuma mensagem de *Hello* fosse detectada após a busca não sucedida por servidores locais e remotos. A Figura 3-7 mostra a máquina de estado de todo o processo de busca por endereços de um nó ingressante na rede.

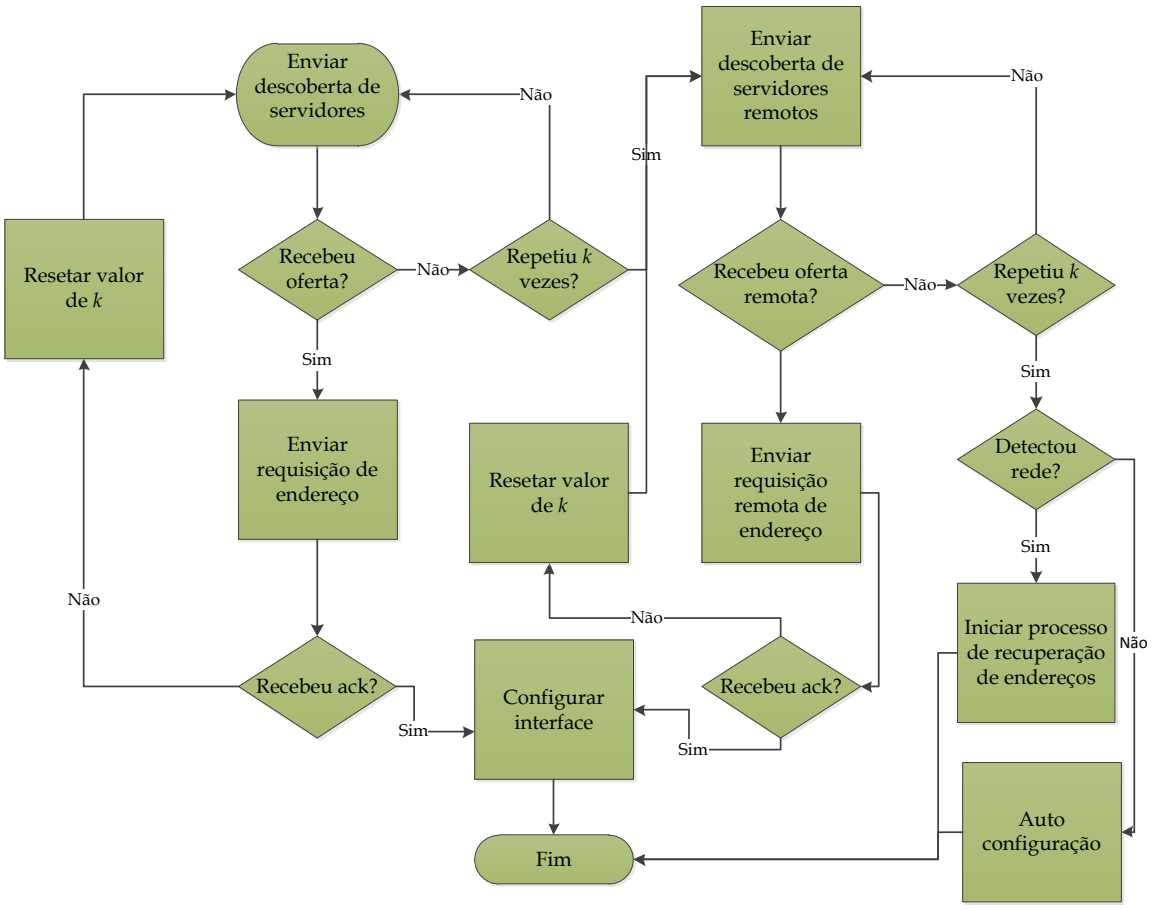

Figura 3-7 – Máquina de estados da busca de servidores

Os *merges* são tratados a partir da análise de mensagens *Hello* e de estimativas do tamanho da rede. Ao receber uma mensagem de *Hello*, o nó irá checar se o NID da mensagem é igual ao seu. Em caso positivo, atualizará a tabela de nós vizinhos e a estimativa do tamanho da rede. Caso não seja, o *merge* é detectado e o nó faz a seguinte checagem: se o tamanho da outra rede for maior que o da sua rede, ele liberará seu endereço e iniciará a requisição de um novo endereço na rede maior. Caso seja menor, o nó não fará nada. Por fim caso as duas redes tenham o mesmo tamanho, as seguintes restrições têm que ser checadas para decidir qual dos nós liberará seu endereço:

(i) O *status* do nó que enviou o *Hello* é maior que o meu *status*.
(ii) *Status* iguais, mas o *seed* do nó que enviou o *Hello* é maior.
(iii) *Status* e *seeds* iguais, mas o NID do outro nó é maior.

O processo de solução de *merge* é tratado em cada nó individualmente, evitando que diversos nós comecem a requisitar endereços novos na rede detectada e gerem um congestionamento temporário na rede, o que ocasionalmente aumentaria a probabilidade de conflitos e perdas de pacotes.

3.4 Comentários

Neste capítulo as descrições de três protocolos de endereçamento *stateful* que oferecem todas as funcionalidades necessárias para um bom desempenho em redes móveis sem fio foi apresentada. Todos os protocolos utilizam funções de alocação distribuída, o que elimina a necessidade de mecanismos DAD.

Todos os protocolos defendem as suas escolhas, justificando ora para simplificar ou acelerar o processo de alocação ora para evitar o acréscimo de sobrecarga na rede. Nos próximos capítulos, essas escolhas serão postas a prova através de suas implementações em um simulador de rede. As avaliações serão realizadas em ambientes e condições iguais para cada protocolo, sendo assim então possível uma comparação justa dessas abordagens.

4 Metodologia de avaliação

Este capítulo descreve a metodologia usada para a avaliação de desempenho dos algoritmos descritos no Capítulo 3. Serão apresentadas o conjunto de métricas escolhidas para qualificá-los, detalhes de implementação e os cenários de avaliação.

4.1 Questões de implementação

Atualmente é possível encontrar diferentes tipos de redes de dispositivos computacionais, desde as redes cabeadas até as inúmeras redes sem fio, como as redes de celular, mesh, WiMax e de sensores.

Montar uma infraestrutura experimental dessas redes que possa ser utilizada para testá-las é muito custoso tanto em recursos quanto em tempo. Assim, os projetistas de arquiteturas, protocolos e serviços em redes têm cada vez mais usados simuladores no desenvolvimento dos seus trabalhos com o objetivo de economizar tempo e serem mais eficazes nas suas soluções. Dentre os simuladores de domínio público e efetivamente utilizados na simulação de redes e protocolos, destaca-se o *Network Simulator 3*, conhecido como NS-3 [25].

O NS-3 é um simulador de rede de eventos discretos desenvolvido para testes de sistemas de Internet e com foco primário em ajudar pesquisas e ensino. Totalmente desenvolvido em C++ e utilizando-se de técnicas de engenharia de software, este simulador está sobre a licença GNU de software livre, possibilitando assim uma grande contribuição de seus usuários no desenvolvimento e solução de problemas.

Neste trabalho, as simulações foram feitas usando um notebook HP modelo dv7-1020us [26], com sua configuração padrão e utilizando o sistema operacional Ubuntu, versão 10.04 [27] com versão de *kernel* 2.6.32.15. Os protocolos de alocação de endereços foram implementados em sua totalidade na linguagem C++.

A Tabela 4-1 mostra o valor dos parâmetros compartilhados pelos três protocolos.

Tabela 4-1 - Valor dos parâmetros comum a todos os protocolos

Parâmetro	Valor
Retry number	3s
Wait time	0.5s
Hello interval	5s
Espaço de endereçamento	10.0.0.0/24

O parâmetro *Retry number* diz respeito a quantas vezes o nó tentará encontrar um servidor. O *Wait time* é um valor parâmetro de quanto tempo se espera por uma resposta após o envio de uma mensagem, podendo o intervalo específico de espera variar em múltiplos de *Wait time*. Já o *Hello interval* se refere ao intervalo entre mensagens de *Hello*, denominadas de *Recycle* no protocolo *Prime* DHCP. O último parâmetro comum a todos os protocolos é o espaço de endereçamento, que define que os nós poderão oferecer endereços de 10.0.0.1 a 10.0.0.255.

Além desses parâmetros descritos, cada protocolo também possui parâmetros específicos que são descritos abaixo:

(i) *Prophet Allocation*: O valor inicial da *seed* foi escolhido como 1.
(ii) *Prime* DHCP: Valor da constante de espera para tornar-se a raiz caso várias mensagens de *Recycle* não sejam recebidas foi escolhido como 1000.
(iii) DNCP: O número de tentativas de busca de servidores remotos é igual a 3. O intervalo de espera por respostas a uma mensagem de *Reclamation* foi definido como 4s.

4.2 Métricas utilizadas

Com o intuito de comparar as três abordagens escolhidas, algumas métricas comuns na área de redes sem fio foram escolhidas. Para uma que uma análise estatística possa ser feita com o devido rigor matemático necessário, é necessária a determinação de quantas coletas para cada métrica são necessárias para obter o resultado dentro de um intervalo de confiança determinado. O professor indiano de Ciências da Computação e Engenharia Raj Jain propõe a Equação 4-1 em [10] a para determinar o número de simulações baseado na média e no desvio padrão amostral.

$$n = \left(\frac{100.z.s}{r.\bar{x}}\right)^2$$

Equação 4-1 - Função para determinar número de simulações

Assim, a partir de um pequeno número de amostras, é possível determinar o número necessário de simulações utilizando a função acima, onde $\bar{x}$ é a média amostral, s o desvio padrão amostral, r é o intervalo de confiança desejado e z é o valor do ponto do percentil de uma distribuição normal para o intervalo de confiança desejado.

Em todos os cenários que serão descritos mais adiante foram rodadas vinte simulações preliminares para obter as médias e desvios padrões necessários para os cálculos acima descritos.

4.2.1 Latência

Para estar apto a comunicar-se com os demais componentes da rede, um nó precisa obter um endereço o mais rápido possível. O tempo gasto entre o surgimento na rede e a configuração completa, será denominado como a latência do nó. O objetivo é estabelecer comparações substanciais para cenários críticos com necessidades similares a sistemas de tempo-real.

4.2.2 Trocas de endereços

Dependendo do protocolo, um nó pode ser forçado a trocar seu endereço devido a *merges*¸ a não responder uma mensagem de *DHCP_Recycle* ou até por duplicações nas seqüências numéricas do protocolo *Prophet*. Portanto, ao perder seu endereço atual, o nó também perde sua conexão direta com os nós da rede, o que pode ocasionar perdas de pacotes. Além disso, a troca de endereços também irá requer um acréscimo de *overhead* causado pelo processo de requisição de um novo endereço.

Assim, redes com menor número de trocas de endereços conseguem, em geral, manter a conectividade de seus nós por mais nós, além de evitar congestionar muito a rede com mensagens de alocação.

4.2.3 Sobrecarga de mensagens

Como dito anteriormente, redes móveis ad hoc, em sua maioria, têm recursos de bateria, de largura de banda e de alcance de sua antena bastante restritos. Desta forma, energia deve ser economizada a todo tempo para evitar que o nó morra previamente.

Seus recursos devem ser majoritariamente utilizados para troca de mensagens não relativas ao funcionamento do protocolo de endereçamento. Assim, quanto mais sobrecarga introduzida pela quantidade de informação trocada proveniente das mensagens de controle do protocolo de endereçamento, pior será o protocolo. Além de ocupar o meio de comunicação por mais tempo, estará gastando recursos preciosos do nó.

4.2.4 Criação de redes

Na execução dos protocolos, foi dada a todos os dispositivos a capacidade de criar uma rede nova. No entanto, esta situação (no caso ótimo) só deveria acontecer caso não exista nenhum outro nó configurado no raio de comunicação do nó ou se a rede não possuir nenhum endereço para oferecer.

Essa métrica compara o número de redes criadas com o número de redes remanescentes ao final do tempo de execução. A meta é determinar se a rede tem a capacidade de lidar com *merges* eficientemente.

4.2.5 Escalabilidade

A energia consumida por mensagens *broadcast* é diretamente proporcional ao número de nós da rede. Como a latência está relacionada diretamente proporcional ao diâmetro da rede, que por sua vez é proporcional ao número de nós, quanto mais *multihop broadcasts* sejam necessários na configuração de um nó, pior será sua escalabilidade [7].

4.3 Cenários de avaliação

Para poder testar as capacidades de cada protocolo em situações diferentes, dois cenários diferentes foram propostos. O primeiro cenário é denominado de estático e tem a intenção de verificar as funcionalidades básicas de distribuição de endereços. Nesse cenário, os nós são distribuídos em uma matriz e não possuem a capacidade de se mover. No entanto, ainda é necessário testar as funções de solução de junção e partição de redes e recuperação de endereços. Para isso, um cenário móvel foi elaborado e implementado. O objetivo é simular um ambiente onde as condições extremas de mobilidade ocorrem, aumentando a probabilidade de tais eventos ocorrerem.

Em ambos cenários, o tempo de simulação foi fixado em 400 segundos e os nós foram configurados com interfaces Wi-Fi com alcance de transmissão igual a 50 metros e vazão de 1 Mb/s. O atraso e perda na propagação do sinal são modelados de acordo com dois modelos do NS-3, chamados *Constant Speed Propagation* e *Friis Propagation Loss* respectivamente [29]. Para efetuar o roteamento das informações na rede, o protocolo OLSR [28] foi escolhido com seus parâmetros padrões.

4.3.1 Cenário estático

O primeiro cenário de certo modo facilita alguns fatores para os mecanismos em estudo. O cenário é denominado de estático, pois os nós têm posições fixas durante a sua vida na rede.

Nesse cenário, os nós são organizados em uma matriz, sendo alocados obedecendo a uma função espiral, como pode ser observado na Figura 4-1, onde o número de cada nó representa a ordem de nascimento na rede. Este esquema foi desenvolvido com o intuito de proporcionar a um nó ingressante na rede, o maior

número de vizinhos possível, evitando assim a necessidade de requisições remotas desde cedo na rede.

Como dito antes, o alcance de comunicação das interfaces sem fio dos nós é de 50 metros, justamente a distância entre nós diagonalmente separados, como ilustrado na Figura 4-1 entre os nós 7 e 21. Desta maneira, cada dispositivo na rede pode ter três, cinco ou oito vizinhos no alcance de sua antena, sendo estes o caso do nó 21 – que tem como vizinhos 7, 20 e 22 -, do nó 20 – tendo como vizinhos 6, 7, 19, 21 e 22 – e do nó 7 – conseguindo comunicar-se em um salto com 1, 6, 8, 19, 20, 21, 22 e 23 – respectivamente.

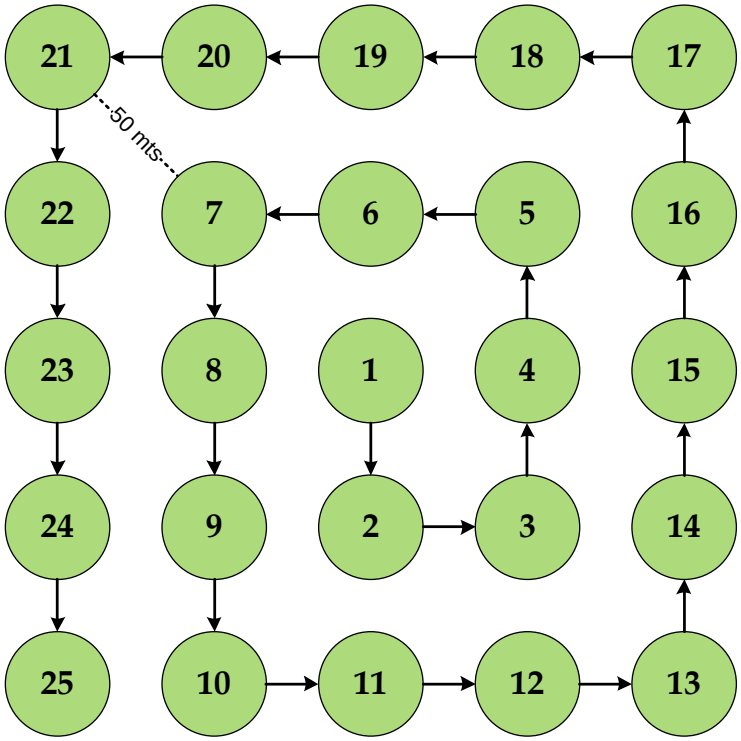

Figura 4-1 - Nós organizados em espiral

4.3.2 Cenário móvel

Diferente do cenário anterior, neste cenário móvel os nós podem mover-se livremente dentro de uma região retangular. Com esta capacidade, os nós forçam situações de junções e partições de redes, testando assim a eficiência, estabilidade e robustez dos protocolos sob tais condições comuns a MANETs.

A movimentação dos nós é feita de acordo com o um modelo de mobilidade presente no simulador NS-3, denominado *Random Walk 2D Mobility Model* [29]. Este modelo determina funciona da seguinte maneira:

(i) No início o nó escolhe randomicamente uma direção e uma velocidade randômica e anda nesta direção até os limites da região retangular.
(ii) Ao atingir o limite da região, o nó pára por um determinado tempo e então reinicia o processo.

Nos testes, os parâmetros escolhidos foram de uma região retangular equivalente a um quadrado de lado igual a 300 metros, o tempo de espera foi de 8 segundos e que a velocidade de movimentação possa variar de acordo com uma distribuição uniforme de 0 a 5 metros por segundo.

Como neste cenário os nós podem mover-se, perde-se o sentido da utilização de um esquema de posicionamento em uma matriz obedecendo a uma espiral, como proposto no cenário estático. Desta forma, foi elaborado um novo mecanismo de posicionamento dos novos nós na rede, descrito abaixo:

(i) O primeiro nó nascerá no centro do retângulo e escolherá uma velocidade e direção de acordo com o modelo descrito anteriormente.
(ii) Os nós seguintes irão nascer após um intervalo de tempo que varia de acordo com uma distribuição uniforme de 5 a 10 segundos após o nascimento do nó anterior. Quanto à posição, o nó será posicionado randomicamente no alcance de comunicação do nó anterior, respeitando os limites do retângulo.

Este modelo foi pensado e desenvolvido com o intuito de assegurar a presença de ao menos um servidor no alcance de comunicação de um nó ingressante na rede.

A Figura 4-2 ilustra o nascimento de um nó na rede seguindo o esquema de posicionamento descrito anteriormente. Na ilustração, o nó A, em verde, já está configurado e move-se na direção da seta que dele sai. O alcance de comunicação do nó A está demonstrado pela circunferência pontilhada ao seu redor, e é dentro desta área que o nó ainda não configurado B, em vermelho, será posicionado e da mesma forma escolherá uma velocidade e direção como indicado pela seta.

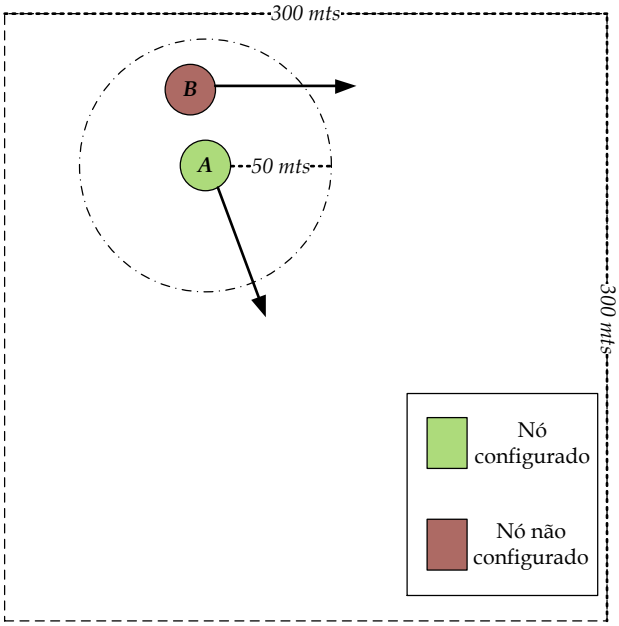

Figura 4-2 - Nascimento de um nó no cenário dinâmico

Desta forma, sempre que um novo nó nascer, ele terá um servidor disponível no seu alcance de comunicação por determinado período de tempo dependente dos posicionamentos dos nós, das velocidades e das direções de seus movimentos.

4.4 Comentários

Este capítulo descreveu os métodos utilizados nas fases de implementação, simulação e análise dos protocolos *Prophet*, *Prime* e DNCP. A importância de se escolher um simulador de rede se deve à facilidade que ele permite ao desenvolvedor de mudar rapidamente os parâmetros bem como ajustar os cenários de teste.

Cada métrica utilizada dos protocolos tem o seu propósito e importância na análise e comparação da capacidade e eficiência das soluções quando testadas em cenários diferentes.

5 Resultados obtidos

Este capítulo descreve os resultados obtidos para cada métrica usando os cenários descritos no capítulo anterior. Também são feitas análises e comparações sobre o desempenho de cada protocolo analisado.

5.1 Cenário estático

Esta subseção apresenta os resultados das diferentes métricas quando medidas no cenário estático descrito em 4.3.1.

5.1.1 Latência

A Figura 5-1 apresenta o tempo médio para que um nó possa obter um endereço usando os diferentes protocolos.

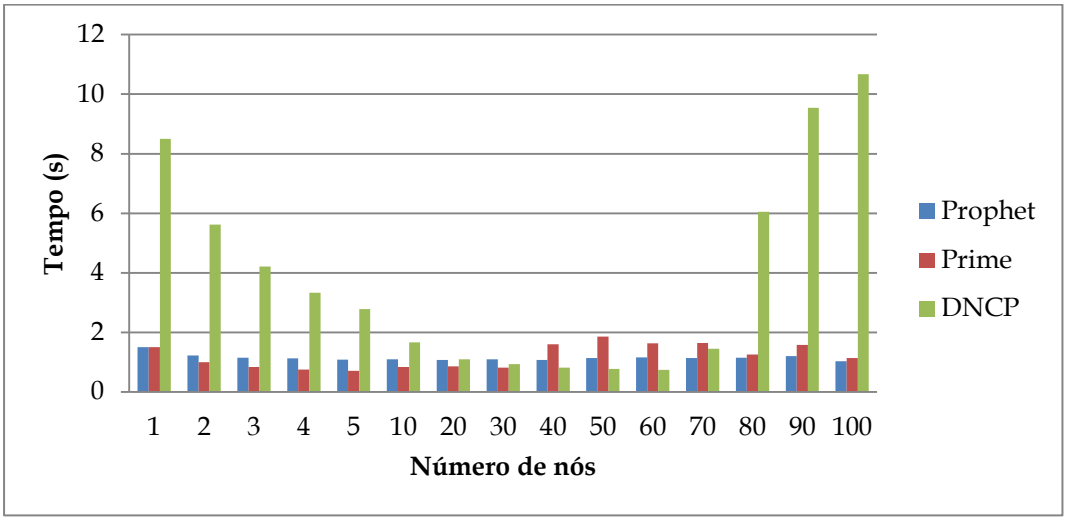

Figura 5-1 - Latência no cenário estático

A Figura 5-1 mostra que os protocolos *Prophet* e *Prime* obtiveram um bom resultado geral, ficando abaixo da marca de dois segundos em todas as populações. O protocolo DNCP apresenta uma grande latência para pequenas quantidades de nós devido ao mecanismo de busca diferenciada por servidores locais e remotos. No DNCP, o primeiro nó na rede levará 8,5s para poder se autoconfigurar, o que tem grande influência na média geral quando o número de nós é reduzido. À medida que o número de nós aumenta a latência do protocolo DNCP diminui até atingir 0.7s com 60 nós. Em seguida, o tempo de espera por um endereço voltar a aumentar devido à topologia da rede. Pois assim, os nós vizinhos

aos nós ingressantes passam a não poder oferecer endereços, forçando a requisição de endereços remotamente ou até, em casos extremos, criação de outras redes (seguida de *merge)*.

Esse atraso excessivo é evitado pelo protocolo *Prime* que não diferencia requisições locais e remotas, desta forma caso um nó receba uma requisição de endereço e não puder respondê-la, reencaminha imediatamente a requisição ao seu nó ancestral na árvore de alocação.

5.1.2 Trocas de endereços efetivadas

O número de trocas de endereços efetivadas, ou seja, quando um nó já configurado é forçado a requisitar um endereço novo é apresentado na Figura 5-2. O número de nós inicia em 10, pois em experimentos preliminares observou-se que não ocorreram trocas de endereço em nenhum dos três protocolos para populações menores que este valor.

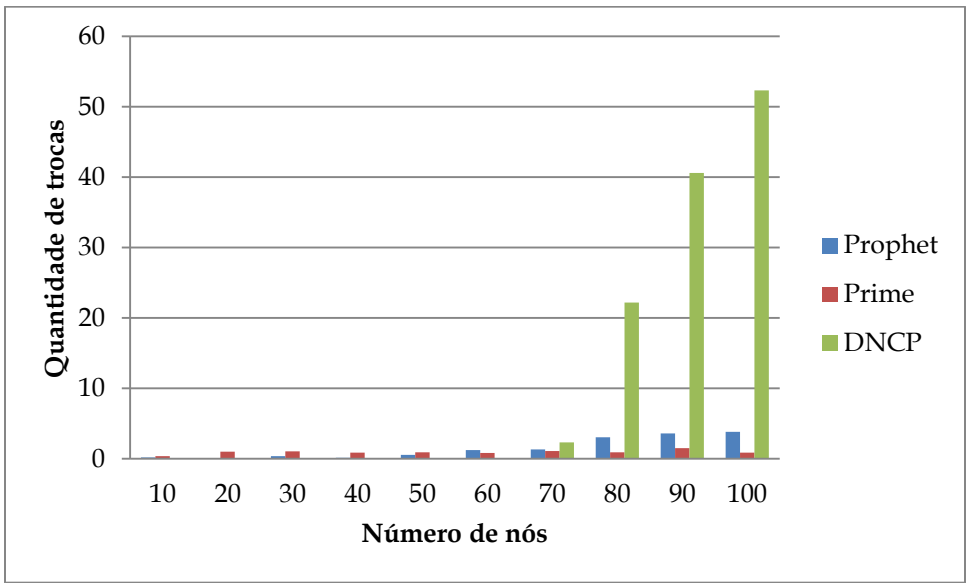

Figura 5-2 - Troca de endereços no cenário estático

Esta métrica corrobora o fato mostrado na latência. O protocolo DNCP apresenta um aumento no tempo de configuração quando o número de nós ultrapassa 70 justamente porque nós ingressantes que não conseguirem obter endereços primeiramente, se autoconfiguram – criando suas redes próprias redes. Desta forma, quando o *merge* for detectado e resolvido, os nós participantes terão que deixar seus endereços e solicitar um novo endereço a rede já pré-existente.

Para valores menores que 70 nós, os protocolos *Prophet* e *Prime* apresentaram resultados piores que o DNCP. Enquanto o DNCP não registrou nenhuma troca de endereço nestas condições, o protocolo *Prime* em média causou que um nó tivesse que trocar seu endereço – provavelmente a uma ausência de

resposta aos *DHCP_Recycle*. Por sua vez o protocolo *Prophet* demonstrou uma troca de endereço crescente proporcionalmente ao número de nós, chegando a uma média de quase quatro endereços com 100 nós, o que indica que apesar da baixa probabilidade de repetições nas ofertas de endereços, as sequências de inteiros da função de distribuição podem apresentar resultado não satisfatório quando o número de nós se aproxima do período repetição.

5.1.3 Sobrecarga de mensagens

O gráfico da Figura 5-3 mostra o número total de bytes das mensagens de controle enviadas em cada protocolo enquanto o gráfico presente na Figura 5-4 mostra o total de bytes enviados por mensagens periódicas como a mensagem *Hello* nos protocolos *Prophet* e DNCP e *DHCP_Recycle* no protocolo *Prime*.

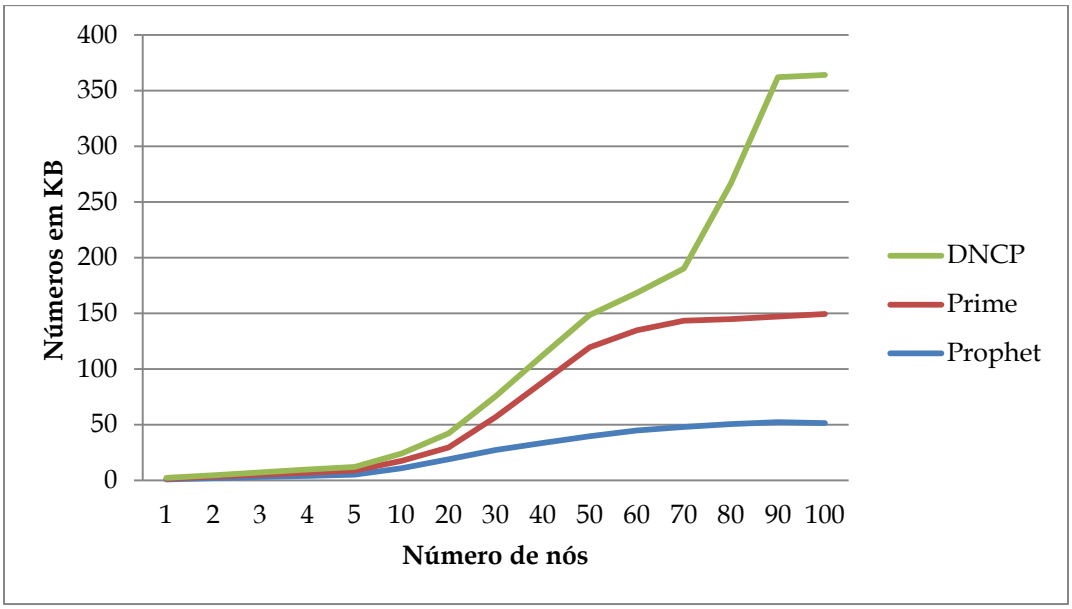

Figura 5-3 - Sobrecarga total em Kbytes no cenário estático

Como é esperada, a sobrecarga das mensagens trocadas nos processos de alocação e manutenção de endereço aumenta paralelamente ao número de nós do cenário.

O protocolo DNCP apresenta novamente uma mudança de comportamento quando o número de nós ultrapassa 70, isso se deve aos processos de *merge* de redes que requer uma considerável troca de mensagens para poder ser resolvido. O protocolo *Prophet* apresenta o menor *overhead*, pois utiliza menos mensagens no processo de distribuição de endereços. Já o protocolo *Prime* acompanha o DNCP até 70 nós, quando se estabiliza em torno de 150 Kbytes trocados.

Quando a análise é feita apenas nas mensagens periódicas, vê-se uma clara vantagem do protocolo *Prime* que apenas envia mensagens deste tipo de sua raiz. Em outras palavras, o número de mensagens periódicas trocadas não depende do

tamanho da rede, se relacionando apenas com o número de redes criadas. Já os protocolos *Prophet* e DNCP apresentam comportamento semelhante. No entanto, o primeiro introduz mais sobrecarga na rede devido ao tamanho de suas mensagens *Hello*, que incluem mais informações e, portanto são maiores quando comparadas com as mensagens periódicas do protocolo DNCP.

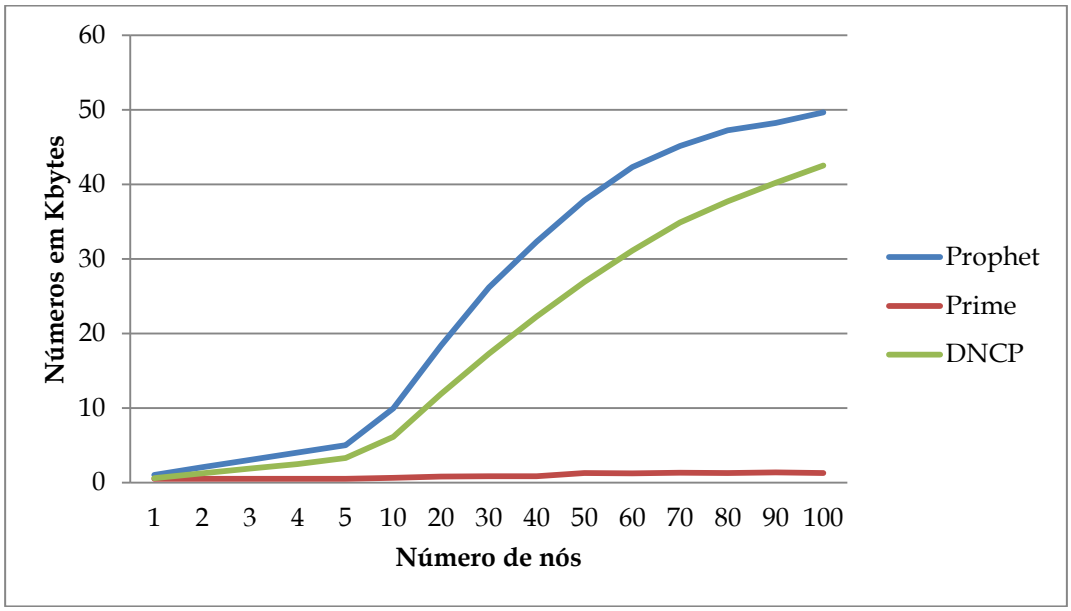

Figura 5-4 - Sobrecarga de mensagens periódicas em Kbytes

5.1.4 Criação de redes

As figuras abaixo mostram o número de redes criadas, Figura 5-5, e quantas redes distintas existiam no final da simulação na Figura 5-6.

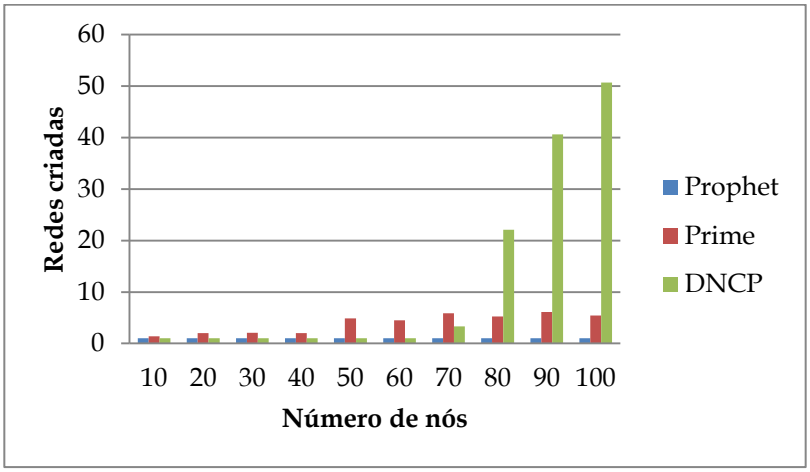

Figura 5-5 - Número de redes criadas no cenário estático

Como indicado pelas demais métricas, o protocolo DNCP apresentou um número de redes criadas grande quando o número de nós ultrapassa a marca de 70. Isso se deve à falha na requisição de endereços, pois seus vizinhos não podiam oferecer endereços sem ultrapassar o espaço de endereçamento. No entanto, apesar da criação de novas redes, o mecanismo de detecção e *merge* de redes distintas se mostrou eficiente, pois ao final das simulações, apenas uma rede existia independente do número de nós.

Já o protocolo *Prime*, mesmo criando poucas redes, não conseguiu efetuar uma junção eficiente das mesmas. Tal fato pode ter sido causado por uma escolha no algoritmo de solução de *merge* que diz que ao detectar um *merge*, todos os nós da rede devem ser avisados ao mesmo tempo. Assim, uma grande quantidade de mensagens de requisição seria trocada em um curto espaço de tempo, aumentando o congestionamento da rede e consequentemente aumentando a probabilidade de perdas de pacotes, o que torna o mecanismo falho.

Como dito em sua descrição, as funções de distribuição de endereços *Prophet* são cíclicas, e, portanto um nó sempre poderá oferecer um endereço, mesmo que este já tenha sido oferecido antes. Desta forma, um nó apenas se autoconfigurará se não houver nenhum vizinho por perto, o que só ocorre no cenário estático com o nó profeta. Assim, como visto nos gráficos, o protocolo *Prophet* cria e mantém apenas uma rede durante todo o tempo de simulação.

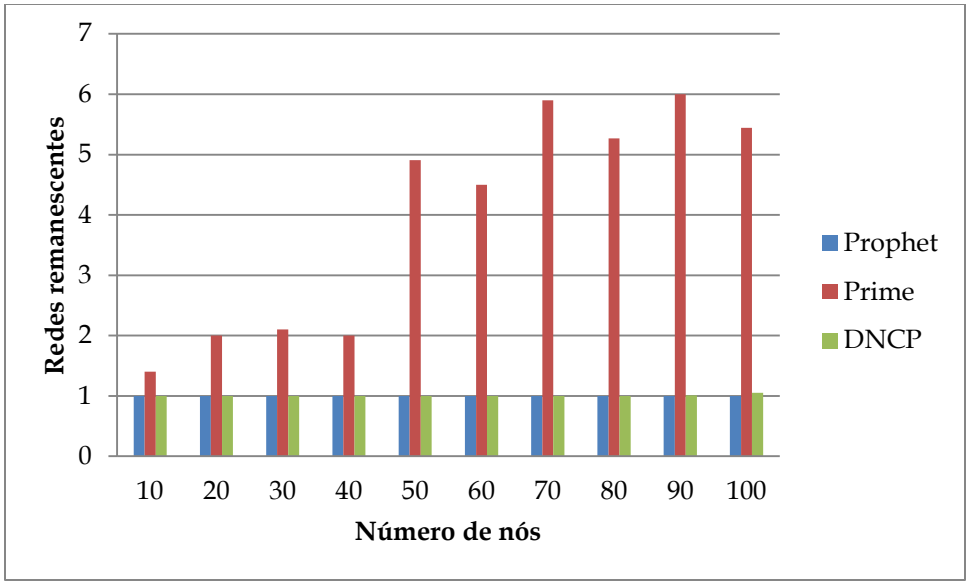

Figura 5-6 - Número de redes remanescentes no final das simulações do cenário estático

5.1.5 Escalabilidade

A Figura 5-7 mostra o número de mensagens em *broadcast* que foram reencaminhas em *broadcast*, ou seja, a quantidade de *multihop broadcast* para uma rede composta por 70 a 100 nós.

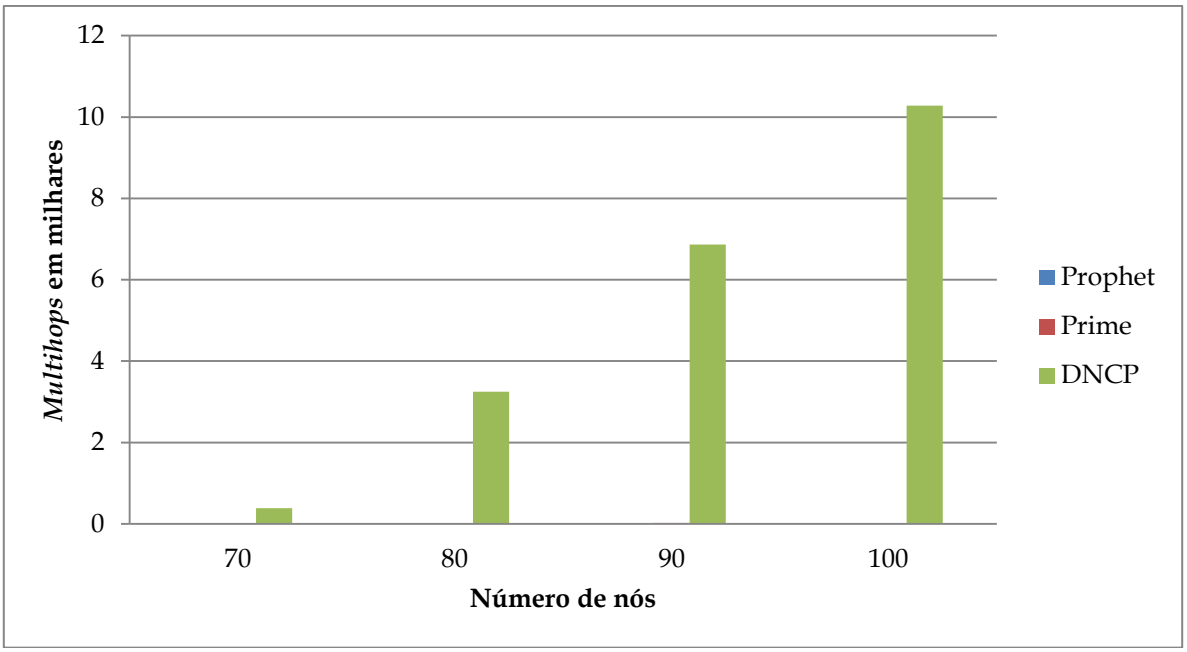

Figura 5-7 - Número de mensagens *multihop broadcast* no cenário estático

O gráfico mostra apenas números a partir de 70 nós, pois antes disso não foram coletados números relevantes (mais que 10 mensagens). Em toda descrição do protocolo *Prophet* não há nenhum tipo de mensagem que deva ser reencaminhado em *broadcast*. Desta forma, o protocolo não registrou nenhuma mensagem deste tipo.

O protocolo *Prime* apresenta poucas mensagens deste tipo, em torno de 8 mensagens iniciando quando o número de nós passa de 10. Já o protocolo DNCP apresenta o mecanismo de busca remota, que faz que os nós reencaminhem as mensagens de *Server Discover* em *broadcast*. Com a falta de endereços disponíveis nas redondezas, os nós ingressantes na rede têm que iniciar o processo busca remota de endereço, aumentando consideravelmente a quantidade de *multihops broadcast*, chegando a mais de 10 mil mensagens quando o número de nós é igual a 100.

5.2 Cenário móvel

5.2.1 Latência

A Figura 5-8 apresenta o tempo médio para que um nó possa obter um endereço usando os diferentes protocolos.

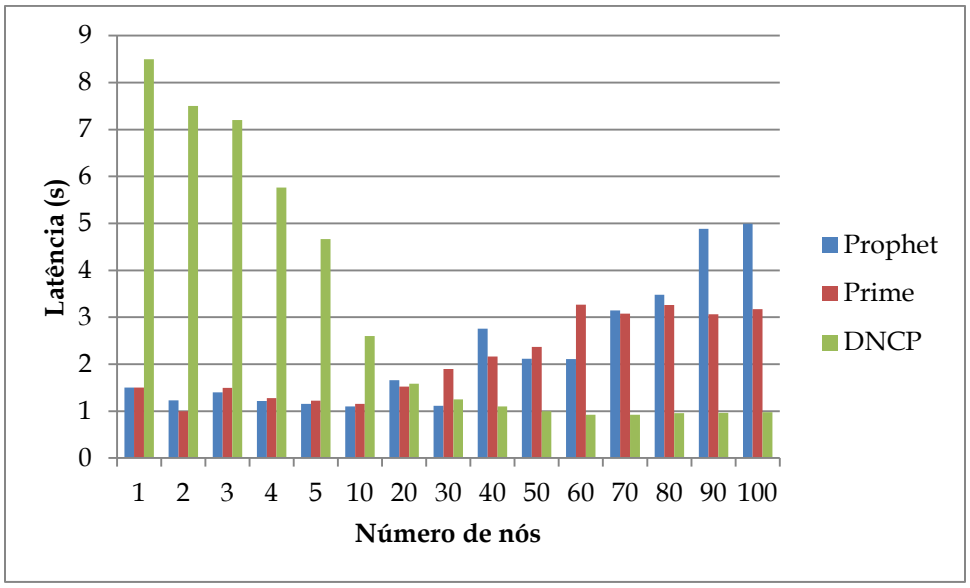

Figura 5-8 - Latência no cenário móvel

Ao contrário dos resultados obtidos no cenário estático, o protocolo DNCP apresenta melhora no tempo médio que cada nó tarda para receber um endereço. Isso se deve ao fato que, diferentemente do primeiro cenário, aqui os nós se movem livremente e, portanto mais servidores podem estar ao alcance de comunicação de um nó ingressante, eliminando assim a necessidade de requisição remota de endereço. Os altos valores quando o número de nós é pequeno ainda se deve ao tempo de 8,5s que o nó criador da rede toma para se autoconfigurar.

Os protocolos *Prophet* e *Prime* apresentam um crescimento na latência devido ao número de trocas de endereços quando o número de nós aumenta como pode ser visto na próxima subseção.

5.2.2 Troca de endereços

A quantidade de endereços trocados pode ser visto em Figura 5-9. Como observado anteriormente, o DNCP não apresentou nenhum valor significativo de trocas de endereços – tendo seu valor médio igual a 0.3 trocas efetivadas para uma rede formada por 100 nós. Isso é consequência do maior número de nós configurados na vizinhança de um nó ingressante.

Este acréscimo de nós no alcance de comunicação é um fator prejudicial ao protocolo *Prophet*. Como o protocolo não utiliza nenhum tipo de mensagem de confirmação do recebimento de endereço, o seguinte fato pode acontecer: um nó ingressante envia mensagem de requisição de endereço para a sua vizinhança que contém 10 nós configurados, então todos os nós configurados ofertam um endereço e assumem que aquele endereço será utilizado, como apenas um é escolhido os demais endereços são perdidos. Assim, rapidamente o espaço de endereçamento será utilizado e as funções atingiram o máximo, reiniciando o ciclo

de ofertas. Este reinício é a causa de alguns conflitos de endereços repetidos em uma mesma rede, o que causará a troca de endereço.

Enquanto o protocolo *Prime* apresenta valores condizentes com os problemas decorrentes da mobilidade dos nós. Respostas aos *Recycle* do nó raiz podem não ser enviadas ou até perdidas por momentâneas desconexões na rede, o que causaria o nó a reiniciar o processo de requisição de endereço.

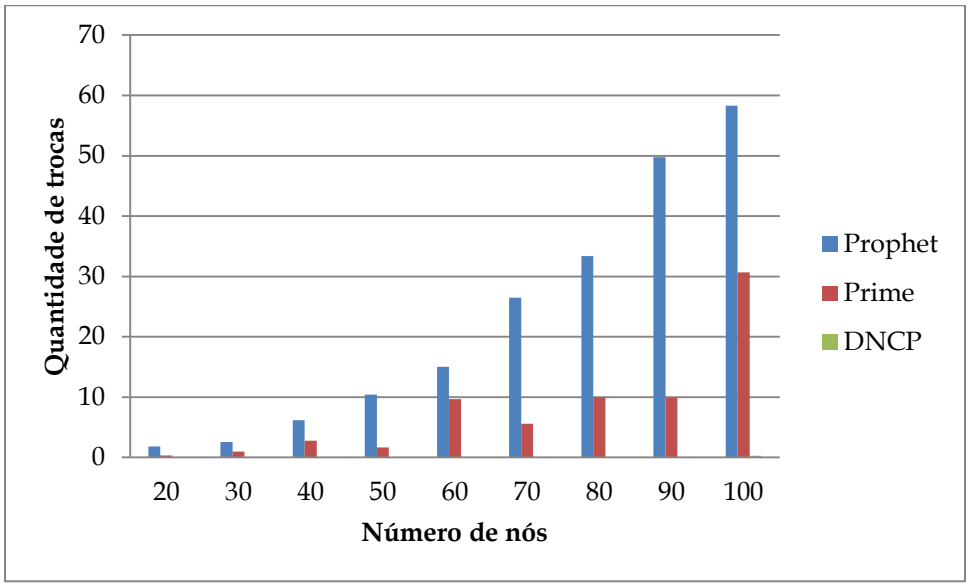

Figura 5-9 - Troca de endereços no cenário móvel

5.2.3 Sobrecarga de mensagens

Os gráficos abaixo, Figura 5-10 e Figura 5-11, mostram a sobrecarga introduzida em Kbytes no total de mensagens e nas mensagens periódicas, respectivamente.

Como esperado, devido ao número de trocas de endereço, o protocolo *Prophet* introduziu uma maior sobrecarga, chegando a mais de 200 Kbytes quando o número de nós é igual a 100.

Enquanto o protocolo *Prime* introduziu *overhead* na rede para sanar os problemas causados pelas perdas de pacote e das consequentes trocas de endereço. No entanto, como o *Prime* apresentou menores trocas de endereço quando comparado com o *Prophet*, a sobrecarga adicionada apresentada também foi inferior, atingindo quase 130 Kbytes nos experimentos com 100 nós. Ainda assim, o número de mensagens periódicas é inferior, pois continua sendo enviado apenas pela raiz da rede.

O protocolo DNCP apresentou o melhor comportamento no que se refere à sobrecarga introduzida. Como não foram necessárias muitas trocas de endereço, o *overhead* não cresceu a grandes taxas, estabilizando-se em apenas 50 Kbytes quando o número de nós é igual a 100.

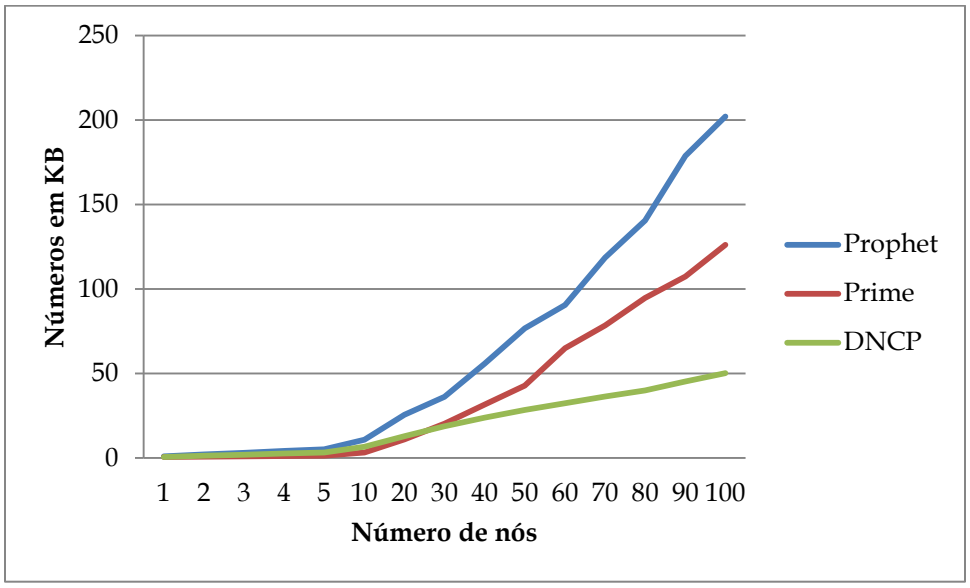

Figura 5-10 - Sobrecarga total em Kbytes no cenário móvel

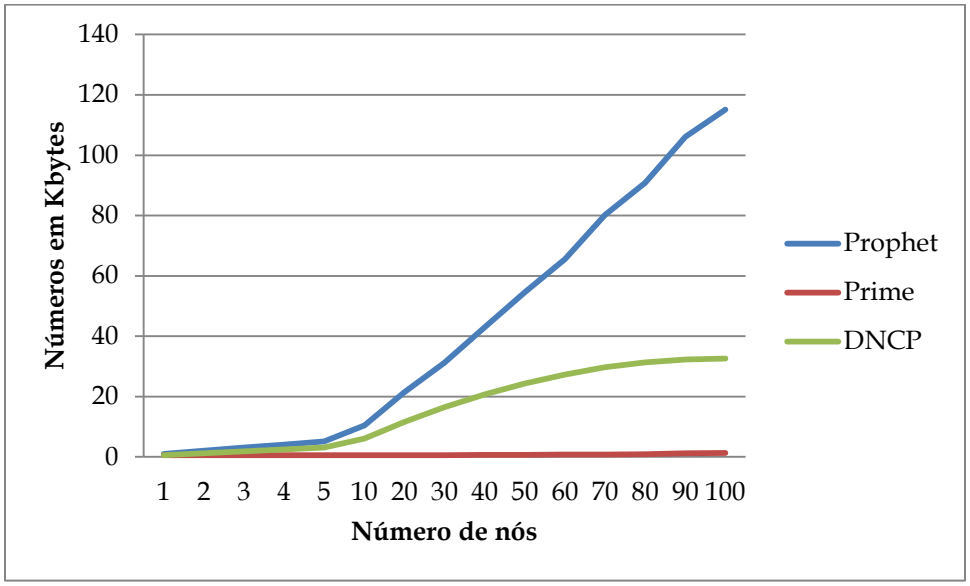

Figura 5-11 - Sobrecarga de mensagens periódicas em Kbytes

5.2.4 Criação de redes

Nas Figura 5-12 e 5-13 são apresentados os números de redes criadas e remanescentes, respectivamente. Os gráficos se iniciam quando o número de nós é igual a 30, pois antes disso apenas uma rede era criada e mantida até o final das simulações.

Os protocolos *Prophet* e DNCP apresentaram desempenho excelente neste quesito. O primeiro atingiu tais números pelo fato já apresentado anteriormente, ou seja, como todos os nós podem oferecer endereços baseados em funções cíclicas,

o espaço de endereçamento é reaproveitado. Assim, apenas o nó que não encontrar nenhum vizinho no seu alcance de comunicação irá se autoconfigurar. Já o DNCP obteve tal desempenho devido ao aumento da disponibilidade de servidores no alcance de comunicação de nós ingressantes, o que acelera o processo de configuração e previne a criação de novas redes.

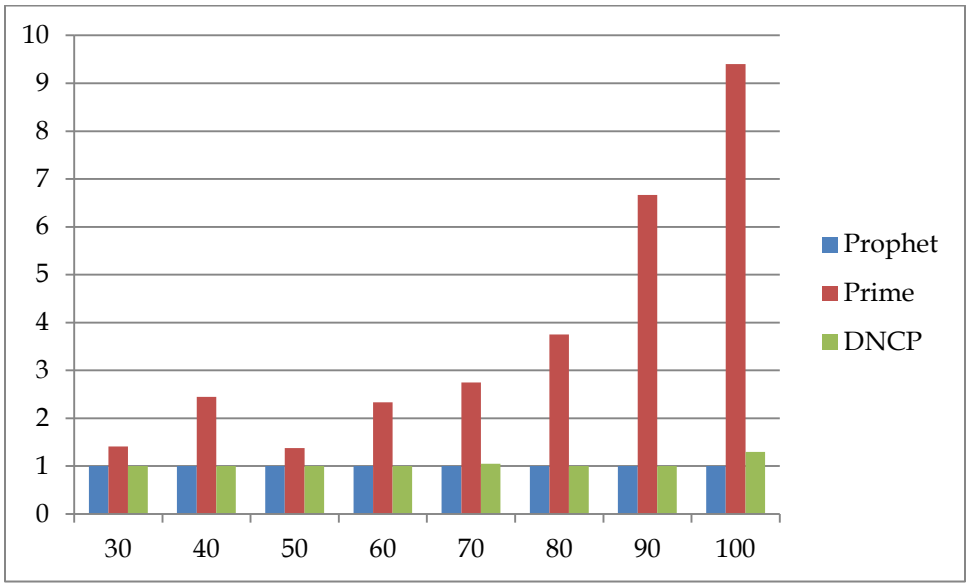

Figura 5-12 - Número de redes criadas no cenário móvel

O protocolo *Prime* apresentou desempenho ruim pelas mesmas razões do cenário estático. À medida que os nós que não receberam ofertas de endereços, se autoconfiguram – criando novas redes –, o mecanismo de correção de junção não se mostrou eficiente, causando uma presença média de mais que 4 redes simultâneas no final das simulações.

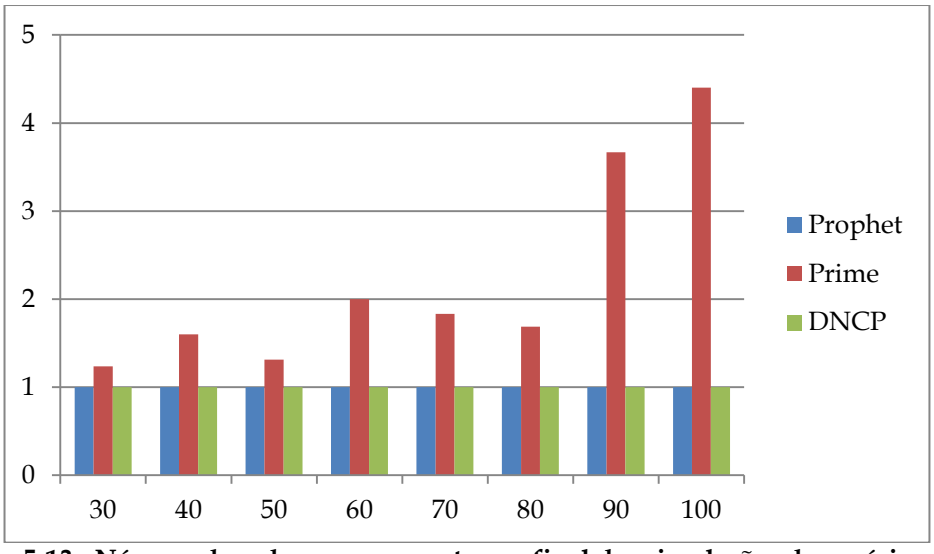

Figura 5-13 - Número de redes remanescentes no final das simulações do cenário móvel

5.2.5 Escalabilidade

A Figura 5-14 apresenta o número de mensagens *multihop broadcast* coletadas nas execuções do cenário móvel com diferentes números de nós. Os números de mensagens deste tipo para o protocolo *Prime* e para o DNCP ficou abaixo de 10 para cenários com menos de 70 nós. O protocolo *Prophet* não utiliza mensagens deste tipo.

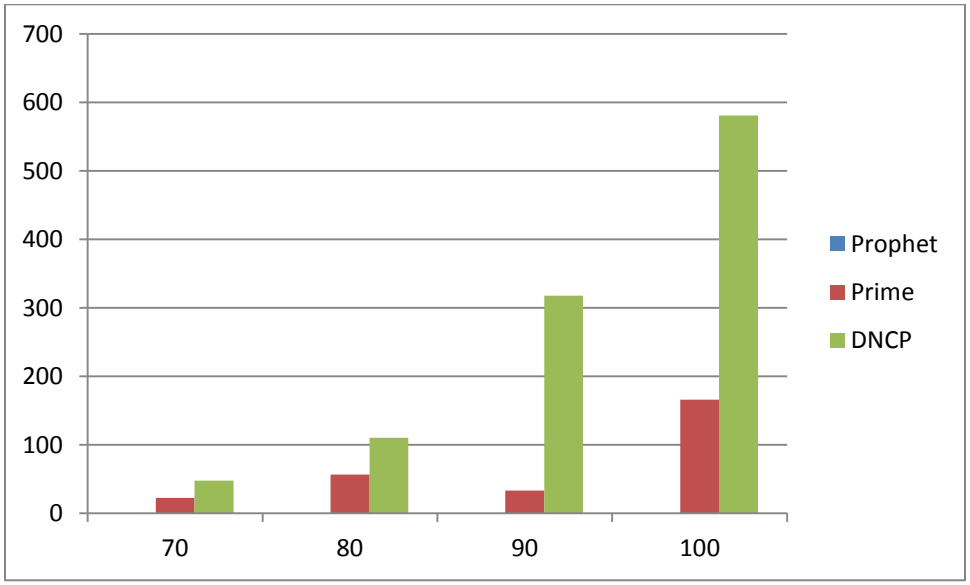

Figura 5-14 - Número de mensagens *multihop broadcast* no cenário móvel

O protocolo DNCP ainda apresenta um número grande de mensagens *broadcast multihop* pelo fato de disponibilizar a capacidade de requisição remota de endereços, o qual passa a ser muito utilizado quando o número de nós se aproxima ao espaço de endereçamento. Enquanto o *Prime* apresenta um número menor de mensagens deste tipo, pois a requisição remota é feita automaticamente em conjunto com a local.

5.3 Comentários

Este capítulo apresentou os resultados obtidos a partir das coletas de informações feitas nas simulações dos protocolos. Em cada cenário, estático e móvel, foram analisadas cinco métricas, sendo elas: latência, troca de endereços efetivadas, sobrecarga de mensagens, criação de redes e escalabilidade.

Utilizando os conceitos de ruim/médio/bom, o desempenho de cada protocolo foi sumarizado para o cenário estático na Tabela 5-1 e para o cenário móvel na Tabela 5-2.

Tabela 5-1 - Desempenho dos protocolos no cenário estático

	Cenário estático		
	Prophet Allocation	***Prime* DHCP**	**DNCP**
Latência	Bom	Bom	Médio
Troca de endereços	Bom	Bom	Médio
Sobrecarga de mensagens	Médio	Médio	Ruim
Criação de redes	Bom	Ruim	Médio
Escalabilidade	Bom	Bom	Ruim

Como esperado e previamente comentado, os protocolos obtiveram diferentes desempenhos. Ao analisar o cenário estático, vê-se que o protocolo DNCP mostrou um desempenho não satisfatório. Quando o número de nós cresce e a quantidade de nós servidores no alcance de comunicação é pequena, a solução DNCP inicia o processo de requisição remota, que introduz uma sobrecarga alta na rede e aumenta a latência média.

No entanto, ainda no cenário estático, pode-se verificar a eficácia do mecanismo de solução de junções do protocolo DNCP, melhor que os descritos no protocolo *Prophet* e *Prime*. Apesar um número grande de redes criadas, devido à autoconfiguração de nós que receberam ofertas de endereços, apenas uma remanescia no final das simulações. Desempenho tal que não foi obtido pelo protocolo *Prime* que mesmo criando menos redes que o DNCP, não se mostrou eficaz na solução da junção das mesmas, permitindo, no pior caso, a existência de até seis redes simultâneas no final das simulações.

Quanto à escalabilidade, corroborou-se a idéia das demais métricas. Ao enviar muitas mensagens *multihop broadcast*, o protocolo DNCP não se mostrou adequado para redes com muitos nós, pois quanto mais nós, maior seria o gasto de energia com mensagens deste tipo. Em dissonância, o protocolo *Prophet* não utiliza em nenhum momento mensagens *multihop broadcast* e a solução *Prime* utilizou-se de poucas mensagens deste tipo no cenário estático.

Em geral, o protocolo *Prophet* se mostrou mais adequado para o cenário estático, em seguida o protocolo *Prime* e então o DNCP. Ainda assim, eram necessários estudos de cenários mais condizentes com a realidade de MANETs, e como pode ser visto na Tabela 5-2, o comportamento dos protocolos se mostrou diferente no cenário móvel.

Tabela 5-2 - Desempenho dos protocolos no cenário móvel

	Cenário móvel		
	Prophet Allocation	***Prime* DHCP**	**DNCP**
Latência	Ruim	Médio	Médio
Troca de endereços	Ruim	Médio	Bom
Sobrecarga de mensagens	Ruim	Ruim	Bom
Criação de redes	Bom	Ruim	Bom
Escalabilidade	Bom	Médio	Ruim

O cenário móvel, mais próximo de situações reais, mostrou diferenças de comportamento dos protocolos quando comparado com o cenário estático. Por exemplo, a latência média dos endereços *Prophet* e *Prime* apresenta um aumento considerável quando o número de nós cresce, no entanto o DNCP obteve resultado diferente, obtendo latências menores que um segundo quando o número de nós é maior que 50.

Esse aumento na latência percebido pelos protocolos *Prophet* e *Prime* se deve a quantidade de endereços que tiveram que ser trocados. No caso do *Prophet* é efeito da periodicidade das funções de alocação de endereço e a ausência de uma mensagem que confirme que realmente será utilizado, como explicado anteriormente.

A sobrecarga de mensagens foi outro fator crítico para o protocolo *Prophet* no cenário móvel. O processo de trocas de endereços gera muitas mensagens de controle do protocolo e assim os protocolos *Prophet* e *Prime* obtiveram o pior resultado, inserindo cada um uma sobrecarga em torno de 200KB e 125KB respectivamente. Já o DNCP, por não ter necessitado efetuar muitas trocas de endereços, introduziu apenas 50 KB de sobrecarga no pior caso.

Diferentemente do cenário estático, o protocolo DNCP não criou muitas redes no cenário móvel devido à maior presença de servidores no alcance de comunicação, diminuindo a necessidade de requisições remotas e autonconfigurações. O protocolo *Prophet* novamente cria apenas uma rede e a solução *Prime* mesmo criando poucas redes, não consegue fazer a junção das mesmas em uma só.

A métrica que ofereceu comportamento mais semelhante nos dois cenários foi a escalabilidade. Os protocolos DNCP e *Prime* enviaram uma quantidade de mensagens *multihop broadcast* proporcional ao número de nós, enquanto *Prophet* seguiu sem enviar mensagens deste tipo.

6 Conclusões e Direções futuras

Este trabalho descreveu e avaliou três protocolos de endereçamento – *Prime Allocation Protocol*, *Prime* DHCP e *Dynamic Node Configuration Protocol* – sob as mesmas condições e em diferentes cenários com o intuito de apontar vantagens e desvantagens de cada um, que o fazem mais adequados a diferentes aplicações e cenários.

6.1 Conclusões

Após a análise e estudo das métricas colhidas nas simulações e apresentadas anteriormente, as seguintes observações podem ser feitas:

(i) *Prophet Allocation*: Foi visto que o protocolo se comporta bem quando não existem muitos vizinhos no alcance direto de comunicação de um nó requisitante. Caso esta situação ocorra, os endereços podem ser perdidos fazendo com que as funções de endereçamento de cada nó cheguem ao final de seu ciclo prematuramente. Assim, cenários com um espaço de endereçamento grande ou com grandes dimensões espaciais (diminuindo a probabilidade de muitos vizinhos diretos) são favoráveis a este protocolo.

(ii) *Prime* DHCP: O *Prime* mostrou-se efetivo na rápida distribuição de endereços, obtendo uma latência pequena para os novos nós da rede. Isto se deve em parte ao processo de requisição remota que é feita automaticamente caso o nó que recebeu uma requisição não possa ofertar um endereço. No entanto, sua solução de *merges* é simplória e ineficiente, pois causa um congestionamento temporário da rede, causando crescimento da latência e perda de pacotes. Assim, cenários onde a rápida obtenção de endereço é essencial e/ou as dimensões espaciais são reduzidas, diminuindo a probabilidade de *merges*, são favoráveis ao *Prime* DHCP.

(iii) DNCP: O *Dynamic Node Configuration Protocol* apresenta um engenhoso algoritmo de manutenção de *merges* e partições que se provou ser o mais eficaz dentre os três pesquisados. Assim, a conectividade entre os nós é obtida mais rapidamente e mantida por mais tempo. No entanto, quando o espaço de endereçamento é pequeno ou o número de nós se aproxima a ele, uma grande quantidade de mensagens *broadcast multihop* é enviada na tentativa de se obter um endereço. O envio de tal tipo de mensagem é prejudicial

à escalabilidade da rede, devido ao grande gasto de energia. Desta forma, projetistas de aplicações em cenários de emergência – onde possivelmente poucos nós existem e a ocorrência de partições e *merges* é mais frequente – devem utilizar esse protocolo buscando a manutenção da conectividade durante toda a vida da rede.

6.2 Direções futuras

Após as análises, é visto que nenhum dos protocolos estudados é ideal para todas as situações e cenários diferentes. Desta forma dois pontos de consideração são levantados.

Primeiro, a análise pode ser estendida a outras abordagens, inclusive de outros tipos, como, por exemplo, a solução *stateless* SDAD. Assim, da mesma forma que foram apontados cenários ideias para cada protocolo aqui estudado, poderia ser feito o mesmo para os novos protocolos estudados. Com novos cenários e aplicações, a ajuda aos desenvolvedores de redes ad hoc sem fio seria melhorada.

Outro ponto considerado é o desenvolvimento de um novo protocolo que mesclaria as abordagens bem sucedidas dos três protocolos aqui apresentados e buscaria obter as vantagens de cada um. Assim, o novo protocolo a ser especificado seria recomendado a diferentes tipos de cenário.

7 Referências

[1] Spartacus Educational. (2010) Wireless Telegraphy. [Online] http://www.spartacus.schoolnet.co.uk/FWWwireless.htm

[2] Bluetooth SIG. Bluetooth Specification Version 2.0+ EDR. (2004)

[3] IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. Technical report, (2007)

[4] IEEE Standard for Information technology—Telecommunications and information exchange between systems— Local and metropolitan area networks—Specific requirements – Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). Technical report, (2009)

[5] Dynamic Host Configuration Protocol. Request for Comments (RFC) 2131. (1997)

[6] Y.-Y. Hsu and C.-C. Tseng. "Prime DHCP: A Prime Numbering Address Allocation Mechanism for MANETs", IEEE Communication Letters, 9(8), (2005)

[7] H. Zhou, L. M. Ni, and M. W. Mutka, "Prophet Address Allocation for Large Scale MANETs", In Proceedings of 22nd Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM'03), vol. 2, pp. 1304-1311, (2003)

[8] R. Aschoff, "DNCP: Dynamic Node Configuration Protocol", Tese de dissertação de mestrado em Ciências da Computação – CIn – UFPE. (2010)

[9] Wikipedia, the free encyclopedia. (2010) Mobile ad hoc network. [Online] http://en.wikipedia.org/wiki/Mobile_ad_hoc_network

[10] R. Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling", New York, NY: Wiley- Interscience, ISBN: 0471503361, (1991).

[11] G. Aggelou, "Mobile ad hoc networks: from wireless LANs to 4G networks", New York, NY: McGraw-Hill Professional Engineering, pp. xiii, (2005).

[12] H. Karl and A. Willig, "Protocols and architectures for wireless sensor networks", Chichester, England: John Wiley & Sons, pp. 181-183, (2005).

[13] C.E. Perkins and T. Jagannadh, "DHCP for mobile networking with TCP/IP", ISCC, pp. 255, IEEE Symposium on Computers and Communications (ISCC'95), (1995).

[14] N. Choi et al, "Random and linear address allocation for mobile ad hoc networks", pp. 2231, IEEE Wireless Communications and Networking Conference (WCNC'05), (2005).

[15] C. Perkins, R. Wakikawa, J. M. E. Belding-Royer and Y. Sun, "IP address autoconfiguration for ad hoc networks", IETF Network working group, (2001).

[16] N. H. Vaidya, "Weak duplicate address detection in mobile ad hoc networks", in procedures of ACM MobiHoc, (2002).

[17] M. Günes and J. Reibel, "An IP address configuration algorithm for Zeroconf mobile multihop ad hoc networks", Proc. Int'l Workshop on Broadband Wireless ad hoc networks and services, (2002).

[18] Y. Sun and E. M. Belding-Royer, "Dynamic address configuration in mobile ad hoc networks", UCSB technology report, 2003-11, (2003).

[19] K. Weniger, "PACMAN: Passive autoconfiguration for mobile ad hoc networks", IEEE Journal on selected areas in communications, vol. 23, issue 3, pp. 507-519, (2005).

[20] J. Misra, "Unique prime factorization theorem", (2006).

[21] Wikipedia – The Online Encyclopedia. (2010). History of two-way radio [Online] http://en.wikipedia.org/wiki/Two-way_radio#History

[22] A. Yousef, A. Diab and A. Mitschele-Thiel, "Performance evaluation of stateful address auto-configuration protocols in Ad Hoc networks"

[23] N. Moore, "Optimistic duplicate address detection (DAD) for IPv6", Request for comments (RFC) 4429. (2006)

[24] F. Haro, "IP address assignment schemes for mobile ad hoc networks", Report for the University of Catalunya (2006).

[25] The ns-3 network simulator – Official website. (2010). [Online] http://www.nsnam.org/

[26] CNET Reviews – dv7-1020us specifications. (2010). [Online] http://reviews.cnet.com/laptops/hp-pavilion-dv7-1020us/4507-3121_7-33182260.html

[27] Ubuntu homepage – Ubuntu 10.04 Lucid Lynx. (2010). [Online] http://www.ubuntu.com/files/LTS_ISV_Desktop_Datasheet_05.pdf

[28] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)", ITEF Network Working group – RFC3623, (2003).

[29] The ns-3 network simulator – Class list. (2010) [Online] http://www.nsnam.org/doxygen-release/annotated.html