

Anderson Clayton B. Rodrigues

*Estratégias de localização de nós em
aplicações VoIP-P2P sobre redes adhoc*

Manaus – AM

Setembro / 2006

Copyright 2006 Anderson Clayton B. Rodrigues.

Anderson Clayton B. Rodrigues

*Estratégias de localização de nós em
aplicações VoIP-P2P sobre redes adhoc*

Dissertação apresentada ao Programa de
Pós-Graduação em Informática da Universi-
dade Federal do Amazonas para obtenção do
título de Mestre em Informática

Orientador:

Prof. Dr.-Ing. Edjair Mota

MESTRADO EM INFORMÁTICA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS EXATAS
UNIVERSIDADE FEDERAL DO AMAZONAS

Manaus – AM

Setembro / 2006

*Dedico esta dissertação a meus pais e ao meu avô Manuel Barreto,
cujo exemplo de honestidade e trabalho
tem sido um norteador para a minha vida.
Para minha esposa Ketlen, que tem
me dado apoio nos momentos mais difíceis
e mostrado a simplicidade de ter esperança e principalmente ao meu filho Kevin por ser
a razão pela qual este trabalho foi finalizado.*

Agradecimentos

Dedico meus sinceros agradecimentos para:

- o professor doutor Edjair Mota, pela orientação e incentivo;
- a equipe do Laboratório de Voz Sobre IP da UFAM, em especial aos colegas Arlen Nascimento e Alexandre Passito, pela ajuda em diversos momentos;
- a minha esposa Ketlen, pela paciência durante a realização deste trabalho;
- ao meu filho Kevin, por manter acesa minha curiosidade;
- todos os colegas do Mestrado em Informática da UFAM.

*“A atividade da engenharia, enquanto permanecer atividade,
pode levar a criatividade do homem a seu grau máximo;
mas, assim que o construtor pára de construir e se entrincheira
nas coisas que fez, as energias criativas se congelam,
e o palácio se transforma em tumba.”*

Marshall Berman

Resumo

A medida que a mobilidade se torna uma exigência cada vez maior dos usuários de redes de computadores, as aplicações precisam adaptarem-se ao novo ambiente descentralizado, que por vezes, é totalmente imprevisível e mutável. As redes *ad hoc* estão muito difundidas em cenários totalmente descentralizados e desprovidos de infra-estrutura fixa e por isso diversos estudos estão sendo realizados visando a adaptação de aplicações originalmente construídas sobre a luz da arquitetura cliente/servidor. As aplicações de voz sobre IP (VoIP) também precisam se adaptar a nova realidade, e para isso foram adotadas diversas estratégias de descentralização, sendo uma delas a utilização de algoritmos de busca P2P para prover a independência de pontos centralizadores. Este trabalho realiza um estudo do comportamento desse tipo de algoritmo sobre redes *ad hoc* visando identificar os fatores que podem influenciar direta ou indiretamente a performance de uma aplicação VoIP que os utilize como método de localização de seus nós.

Abstract

More and more the demanding for mobility has been wanted by the users who want to have everything no matter where they are. The applications have been modified in order to attend this needs demanded by the new environment without any predictable infrastructure or topology. The adhoc networks are very popular when concerning about non-infrastructures environment and many studies have been started looking for solutions and strategies to understand and adapt the applications originally created to work over client/server architecture. Voice over IP applications also have to be adapted to work properly over this environment and this study try to identify the factors which can impact on application performance. It is made looking at the look up algorithms used by them to find other nodes around the net.

Sumário

Lista de Figuras

Lista de Tabelas

1	Introdução	p. 13
2	Fundamentação Teórica	p. 16
2.1	Telefonia IP (VoIP)	p. 16
2.1.1	Componentes de uma chamada VoIP	p. 17
2.1.2	Etapas de uma chamada VoIP	p. 18
2.1.2.1	Localização	p. 18
2.1.2.2	Configuração	p. 19
2.1.2.3	Transmissão da voz	p. 19
2.1.3	Session Initiation Protocol (SIP)	p. 19
2.1.3.1	Elementos de uma rede SIP	p. 19
2.1.3.2	Funcionamento do SIP	p. 20
2.2	Uma Estratégia de Descentralização	p. 24
2.2.1	Ciclo de vida de um Nó	p. 25
2.2.2	Classificação das Redes P2P	p. 26
2.2.3	Métodos de busca	p. 26
2.2.3.1	Busca Centralizada	p. 27
2.2.3.2	Busca por Inundação (<i>Flooding</i>)	p. 27
2.2.3.3	Busca por Tabela Hash Distribuída (DHT)	p. 30

2.2.4	Exemplo de implementação descentralizada	p. 30
2.3	Aspectos necessários à mobilidade	p. 32
2.3.1	Redes <i>ad hoc</i> Móveis (<i>MANETs</i>)	p. 32
2.3.2	Avaliação de performance em redes <i>ad hoc</i>	p. 34
3	Problema de Interesse	p. 36
3.1	Trabalhos Relacionados	p. 36
3.2	Considerações sobre uma chamada VoIP	p. 37
3.3	Definição do foco	p. 38
3.4	Benefícios esperados	p. 38
4	Abordagem do problema	p. 40
4.1	Escolhendo uma estratégia P2P-VoIP	p. 40
4.2	Construção do Ambiente	p. 41
4.3	Direcionamento dos estudos	p. 43
4.3.1	Hipótese 1	p. 43
4.3.2	Hipótese 2	p. 44
5	Experimentos Realizados	p. 47
5.1	Metodologia	p. 47
5.2	Preparação do Ambiente	p. 48
5.3	Cenário dos experimentos dentro do NS2	p. 51
5.4	Execução dos experimentos	p. 53
5.5	Considerações sobre os resultados	p. 53
6	Considerações Finais	p. 59
6.1	Contribuição	p. 59
6.2	Considerações sobre a estratégia	p. 59

6.3	Trabalhos futuros	p. 60
	Referências	p. 62

Lista de Figuras

1	Rede telefônica com comutação de circuitos	p. 16
2	Aplicações VoIP	p. 18
3	Dois clientes trocando mensagens através de Servidores SIP	p. 20
4	Cliente realizando registro na rede SIP	p. 21
5	Chamada através de um servidor de redirecionamento	p. 22
6	Finalização da chamada	p. 22
7	Redes P2P e Cliente/Servidor	p. 24
8	Estados de um nó	p. 25
9	Arquitetura do Napster	p. 27
10	Método de Busca GnuTella	p. 28
11	Arquitetura de ultra-nó do GnuTella v0.6	p. 29
12	Busca utilizando DHT	p. 30
13	Rede adhoc	p. 33
14	Alcance de uma rede adhoc	p. 33
15	Baixa densidade de nós	p. 43
16	Alta densidade de nós	p. 44
17	Estruturas hierárquicas da rede P2P	p. 45
18	Estruturas hierárquicas da rede P2P após a movimentação de um nó	p. 45
19	Esquema de utilização UML e NS2	p. 50
20	Ilustração de uma rede adhoc e os terminais UML	p. 52
21	Latência de Sessão para 50 nós	p. 54
22	Latência para 25 nós	p. 55

23	Latência para 10 nós	p. 56
24	Vazão para 10 e 50 nós	p. 57
25	Taxa de Perda para 2 a 50 nós	p. 57

Lista de Tabelas

- 1 Lista de parâmetros utilizados na construção do cenário de experimentos p. 51

1 *Introdução*

A demanda mundial por informação e conectividade tornou a Internet, com seus mais de 153,3 milhões de usuários(1, 2), um campo fértil para o desenvolvimento de novas tecnologias de comunicação. Como não poderia deixar de ser, a telefonia também foi impulsionada para utilizar a infra-estrutura fornecida pela Internet para romper as amarras que a seguravam no passado. A telefonia IP foi desenvolvida como uma alternativa à convencional, permitindo a utilização da Internet para a realização de chamadas telefônicas a um custo menor. Algumas empresas atestam que reduziram em mais de 80% seus gastos com telefonia utilizando algum tipo de serviço em telefonia IP(3). Empresas como Skype (4), Google (5) e a brasileira Universo Online (UOL) (6) já fornecem a capacidade de realizar chamadas telefônicas para praticamente todo o mundo a partir de computadores domésticos cobrando taxas, que variam para cada serviço utilizado, mas que ficam abaixo das cobradas pelas operadoras de telefonia convencional principalmente em se tratando de chamadas internacionais.

A despeito de suas aparentes vantagens, a telefonia IP ainda enfrenta alguns obstáculos inerentes à Internet como por exemplo a identificação e endereçamento de nós conectados a rede. Uma vez que nenhuma conexão física é estabelecida entre um nó e outro quando uma chamada é realizada, os nós envolvidos precisam conhecer suas localizações dentro da rede por meio de seus endereços IP. No entanto, não seria prático e nem confortável para os usuários conhecerem números no formato *nnn.nnn.nnn.nnn* toda vez que desejarem "telefonar" para alguém que pode ter seu endereço mudado constantemente nos casos em que os obtenha dinamicamente através de servidores DHCP ou que estejam antecidos por NATs. A solução para contornar este problema e tornar a telefonia IP o mais próxima possível da convencional no que concerne a identificação de um terminal foi a criação de um identificador intermediário que pode até mesmo ser um número de telefone convencional, ou celular ou mesmo um endereço de correio eletrônico (4, 7). Este identificador deve ser vinculado ao endereço IP do nó onde o usuário está conectado, sendo referenciado toda vez que se deseje realizar uma chamada para o mesmo tornando o processo

mais confortável. Apesar de ser um avanço, esta solução ainda traz alguns inconvenientes como o fato do usuário ter ele próprio que atualizar o endereço IP de seus contatos que eventualmente mude. A partir desse ponto a utilização de servidores capazes de traduzir um identificador VoIP para o seu respectivo endereço IP tornou-se uma prática comum levando a maioria dos protocolos a adota-la facilitando ainda mais para os usuários localizarem seus contatos que precisam apenas estarem cadastrados em um servidor para o qual tenham acesso.

A utilização de servidores permitiu o desenvolvimento de outros serviços como segurança, tarifação, correio de voz, detecção de presença, teleconferências, chamada em espera, entre outros. Uma chamada pode ser totalmente encaminhada através do servidor, ou este pode atuar somente como meio de localizar o destino, que após localizado, estabelece uma conexão direta com a origem da chamada. A estratégia de utilização de servidores é diretamente dependente do protocolo de telefonia IP utilizado que pode definir diferentes responsabilidades para um servidor ou distribuí-las em diversos servidores. Segundo nossas pesquisas dois protocolos destacam-se atualmente, o H.323(8) e o SIP(9). Enquanto o primeiro utiliza um servidor chamado *Gatekeeper* especificamente para o controle do acesso a rede e da sinalização entre os pontos, no segundo (9) esta tarefa é realizada pelo servidor *Proxy*.

Em VoIP existem dois momentos cruciais para a realização de uma chamada, a localização do destino e a manutenção desta. Os servidores desempenham um papel importante na primeira permitindo que um cliente obtenha rapidamente uma resposta a respeito do sucesso ou não da sua chamada, caso contrário, este poderia ficar esperando indefinidamente até que percebesse que não obteria resposta. O tempo que um cliente espera até que obtenha o primeiro sinal da sua chamada chama-se **latência de sessão**, e segundo estudos realizados pelo Tesouro Nacional (10) e outro fornecido por (11), o tempo de tolerância médio aceito pelo usuário na abertura de uma aplicação é de 3 à 5 segundos, o que pode também ser aplicado ao tipo de aplicação tratado neste trabalho. Na telefonia convencional este tempo não chega a 3 segundos, daí a expectativa criada pelo usuário sobre a telefonia IP.

Apesar da grande popularidade da arquitetura baseada em servidores, algumas empresas (4, 5) incorporaram a tecnologia P2P às suas aplicações tentando sanar inconvenientes como a dependência excessiva dos clientes em relação aos seus servidores, facilmente observada tanto no protocolo H.323 quanto no SIP. O fato é que caso um servidor venha a falhar todos os clientes que dele dependam ficarão impossibilitados de encontrar uns aos

outros. Superar estas limitações não é simples, e até o momento, segundo nossas investigações, não existe um protocolo padrão, e as soluções apresentadas ainda trazem algum nível de centralização e poucos estudos(12–14) estão sendo desenvolvidos para tornar as aplicações de telefonia IP totalmente descentralizadas.

O objetivo deste trabalho é avaliar as estratégias de busca de nós utilizadas em redes P2P e seus impactos na performance de aplicações VoIP sobre redes *ad hoc*, onde existe mobilidade dos nós. Este trabalho está organizado da seguinte maneira.

No Capítulo 2, uma revisão bibliográfica é feita a respeito das tecnologias envolvidas no estudo incluindo VoIP, P2P e *ad hoc*. Nos Capítulos 3, 4, 5 o problema é delimitado com mais detalhes apresentando trabalhos já realizados sobre o tema, sendo seguido pela metodologia adotada e ambiente onde os experimentos foram realizados. No capítulo 6 são encontrados os resultados obtidos e nossas considerações sobre os mesmos e por fim no capítulo 7 as conclusões são apresentadas juntamente com algumas sugestões para trabalhos futuros.

2 *Fundamentação Teórica*

2.1 Telefonia IP (VoIP)

O conceito aplicado na telefonia convencional não mudou muito desde sua criação e ainda em nossos dias a comutação de circuitos é a base para o seu funcionamento. A Rede de Telefonia Pública Comutada (**RTPC**) é uma rede ponto-a-ponto para transmissão de voz em tempo real que possibilita a transmissão de voz ponto a ponto. Nesse tipo de tecnologia quando alguém realiza uma chamada telefônica um circuito é fechado a partir da origem ao destino da chamada e permanece dedicado até que ela seja terminada, o que encarece o seu custo para o usuário final que paga pela utilização exclusiva dos recursos. Apesar do seu alto custo, a qualidade de voz obtida em redes comutadas é de boa qualidade o que justifica o seu uso durante tantos anos. A figura 1 ilustra as conexões necessárias em uma rede de comutação de circuitos onde cria-se uma **conexão física** entre um ponto e outro.

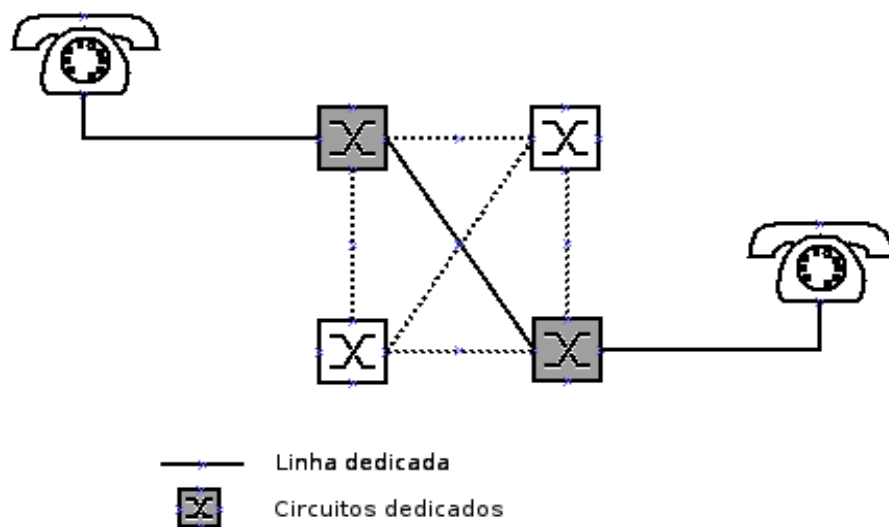


Figura 1: Rede telefônica com comutação de circuitos

A Voz Sobre IP (**VoIP**) é uma alternativa à telefonia convencional que faz uso de redes comutadas por pacote, como a Internet, para a transmissão de voz de um ponto

a outro. Nesta tecnologia a voz é codificada e particionada em pequenos pacotes que são enviados através da rede, sem que para isso, os recursos sejam dedicados a uma determinada chamada maximizando a utilização do meio de transmissão que pode servir a diversas chamadas simultaneamente. O principal atrativo para a utilização de Voz Sobre IP é a diminuição significativa dos custos das chamadas. Em vez de pagar por duas contas (a de conexão à Internet e a telefônica) ou de se pagar mais caro por chamadas interurbanas, o usuário necessita arcar apenas com os custos da conexão a Internet, o que ele normalmente já faz (15). A realização de uma chamada utilizando Voz Sobre IP obedece a uma sequência de etapas e faz uso de protocolos que regem o funcionamento e a participação de componentes que desempenham papéis específicos na transmissão da voz de um ponto a outro da rede. Neste capítulo faremos uma revisão bibliográfica sobre os fatores relevantes na realização de uma chamada VoIP.

2.1.1 Componentes de uma chamada VoIP

Podemos classificar os elementos envolvidos em uma chamada VoIP em três grupos:

- **Clientes** - realizam e recebem as chamadas dentro da rede e também são responsáveis pela codificação/decodificação da voz através de **Codecs**, que são dispositivos ou softwares capazes de codificar a voz em sinais digitais ou analógicos. Um cliente pode ser um telefone convencional, um PC ou terminais telefônicos específicos para telefonia IP.
- **Servidores** - são responsáveis por diversas atividades como a localização e autenticação de clientes, prestação de serviços especializados como secretária eletrônica e armazenamento de mensagens de voz.
- **Tradutores** - são responsáveis pela comunicação com a rede pública de telefonia **RPTC**. Essas entidades são capazes de entender os protocolos utilizados tanto na rede IP quanto na convencional.

Existem várias combinações possíveis para esses elementos na realização de uma chamada. A Figura 2 ilustra três das mais utilizadas. Na primeira, uma chamada é feita entre dois telefones localizados na rede pública que fazem uso de tradutores (**gateways**) para acessarem a rede IP (**Internet**). O próximo cenário mostra uma chamada entre um telefone localizado na rede pública e um computador conectado a rede IP e finalmente no último cenário é mostrada uma chamada entre dois computadores conectados a rede IP.

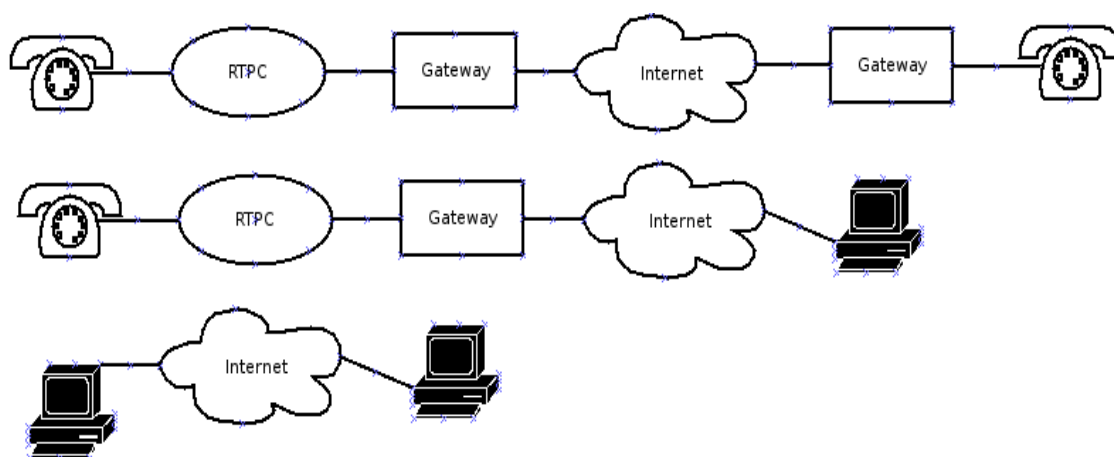


Figura 2: Aplicações VoIP

A arquitetura de uma rede de telefonia IP é diretamente dependente do protocolo utilizado, dos quais destacam-se atualmente, o SIP (9) da IETF (Internet Engineering Task Force) e o H.323 da ITU-T (International Telecommunication Union - Telecommunication) que é um protocolo onde toda a inteligência é alocada nos servidores centrais, ao contrário do SIP que delega algumas, mas não todas, as responsabilidades aos seus clientes(16), tornando-o mais próximo de um modelo descentralizado.

2.1.2 Etapas de uma chamada VoIP

O estabelecimento de uma chamada utilizando telefonia por IP segue três etapas primárias que devem ser executadas em uma sequência específica. Cada uma delas tem um papel crítico, e a não execução de uma delas pode levar ao cancelamento da chamada. Essas etapas são:

2.1.2.1 Localização

Quando queremos realizar uma chamada telefônica convencional informamos o número desejado e as conexões físicas são estabelecidas até o seu destino. Na telefonia por IP este princípio não se aplica já que nenhuma conexão física é estabelecida e o destino tem de ser encontrado utilizando-se outras formas. Existem duas maneiras de se localizar um nó na telefonia IP: fornecer o IP do destino da chamada, ou utilizar servidores para o armazenamento do endereço IP dos participantes da rede VoIP e configurando os demais elementos da rede para os consultarem toda vez que desejarem realizar uma chamada.

2.1.2.2 Configuração

Mesmo após o destino já ter sido localizado, as partes envolvidas na chamada sabem pouco uma da outra e necessitam trocar informações para garantir que tenham condições de estabelecer a transmissão de voz. Dentre os parâmetros que devem ser estabelecidos está o tipo de **codec** utilizado em ambos os clientes. Um Codec pode ser um *software* ou *hardware* especializado em codificar/decodificar a voz para/de bits de forma que possa ser transmitida através da rede de dados.

2.1.2.3 Transmissão da voz

Antes de ser efetivamente transmitida, a voz é codificada em bits, que são organizados em pacotes, que podem ser transmitidos diretamente entre um cliente e outro, ou passarem por um servidor que atua, entre outros papéis, como multiplexador em uma conferência por voz.

2.1.3 Session Initiation Protocol (SIP)

O SIP(9) foi desenvolvido visando o estabelecimento, controle e finalização de sessões multimedia através da Internet. Entre outras características vantajosas, é baseado em mensagens de texto, característica que incorporou de um protocolo largamente utilizado na Internet, o *Hyper Text Transport Protocol* (HTTP)(17).

2.1.3.1 Elementos de uma rede SIP

Os elementos de uma rede SIP podem ser classificados em dois grupos: clientes e servidores SIP.

Um cliente é um elemento localizado nas bordas de uma rede SIP e é responsável por originar requisições, além de enviar e receber tráfego multimedia. Ele pode ser um telefone SIP, um software sendo executado em um computador ou mesmo um *gateway* que fornece acesso a rede de telefonia convencional RPTC(18). Todo cliente é identificado por uma URI (Uniform Resource Identifier) na forma `sip:user@domain` e deve registrar seu IP em um Servidor de Registro (*Registrar*) conhecido na rede. A Figura 3 mostra um cenário mais realista onde dois clientes comunicam-se através de servidores SIP, que são elementos localizados no centro da rede SIP e auxiliam em tarefas como localização, autenticação e bilhetagem de clientes. A RFC 3261(9) define 3 tipos de Servidores SIP:

Proxy, Registrar e Redirect.

- **Servidores *Proxy*** - sua principal função é receber mensagens e as encaminhar para o outro destino. No entanto, também pode ser utilizado para realizar diversas tarefas administrativas como controle de acesso, filtragem de chamadas, bilhetagem, validação de *host*, etc.
- **Servidores de Registro (*Registrar*)** - recebe requisições de registro dos clientes e atualiza suas informações em um banco de dados.
- **Servidores de Redirecionamento (*Redirect*)** - recebe requisições de um cliente ou *proxy* e responde com uma mensagem de redirecionamento, que carrega a localização para onde a requisição deve ser encaminhada.

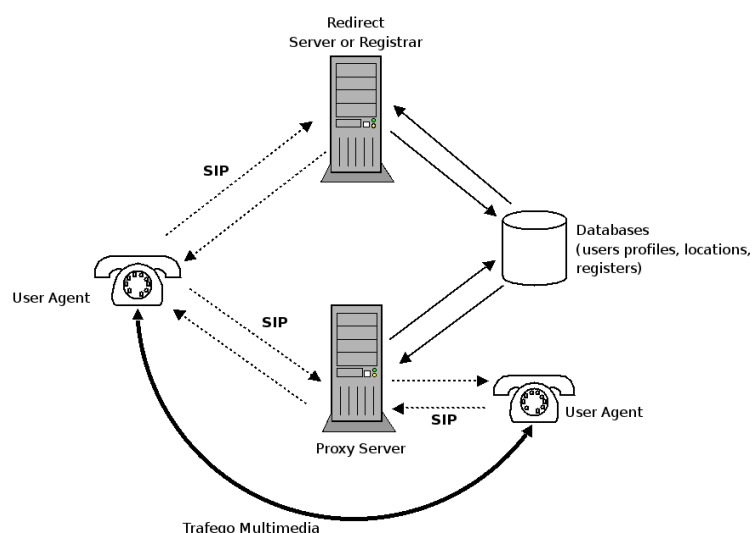


Figura 3: Dois clientes trocando mensagens através de Servidores SIP

2.1.3.2 Funcionamento do SIP

Freqüentemente a melhor forma de aprender sobre o funcionamento de um protocolo é olhar para um exemplo de sua utilização(19). Existem dois grupos de mensagens dentro do SIP: requisições e mensagens informativas. A diferença básica entre elas é o fato de que mensagens informativas, como denunciado pela sua denominação, tem o propósito apenas de informar sobre alguma determinada situação ou status, enquanto requisições são utilizadas para iniciar, encerrar ou controlar uma sessão. As mensagens informativas tem sua primeira linha iniciada por um número, na forma [1-6]XX que identifica seu tipo específico, já as requisições iniciam com o nome do método a ser executado.

Quando um cliente SIP deseja entrar na rede, ele envia uma requisição **REGISTER** para um servidor de registro válido, nela está o endereço IP no qual ele pode ser encontrado por outros membros da rede. O servidor de registro pode ser acessado tanto diretamente, como também através de um servidor *proxy* que conhece sua localização. Na Figura 4 um cliente realiza seu registro na rede através de um servidor *proxy* que conhece a localização de um servidor de registro. Para isso, o cliente inicia enviando uma requisição do tipo **REGISTER**, que é respondida pelo *proxy* com uma mensagem 100 Trying indicando que está tentando localizar um servidor de registro para encaminhar a requisição, este por sua vez, ao receber a requisição, informa que o registro foi realizado com sucesso enviando uma mensagem 200 OK ao *proxy*, que a encaminha ao cliente.

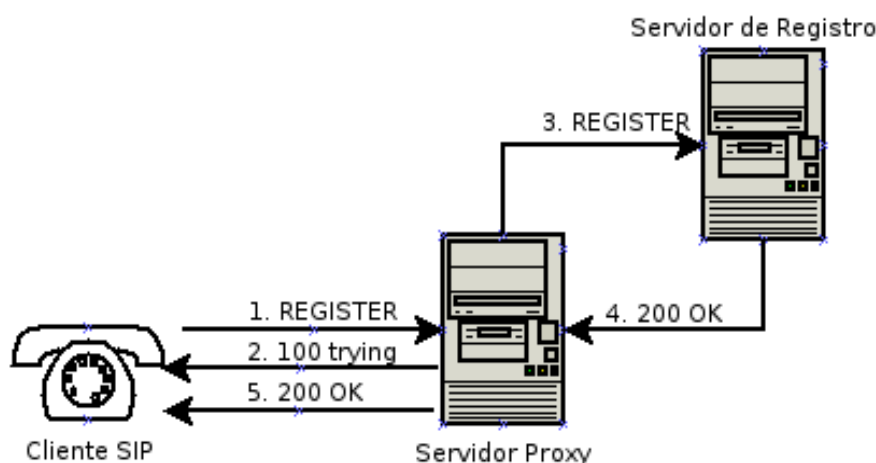


Figura 4: Cliente realizando registro na rede SIP

Os servidores *proxy* são freqüentemente utilizados como autenticadores e podem negar ou conceder a entrada de um determinado cliente impedindo que o registro seja efetuado e deixando-o no anonimato perante os demais, que não poderão encontrá-lo. O registro precisa ser renovado de tempos em tempos caso contrário o servidor assume que o cliente não está mais conectado a rede e informa a todos que o consultarem que este não pode mais ser alcançado.

A realização de uma chamada SIP é sempre iniciada com o envio de uma requisição **INVITE** para a URI do destino, caso conheça o endereço IP do mesmo, ou para um servidor *proxy* que o conheça. Caso o servidor não conheça a URI solicitada, encaminha a requisição para um servidor de redirecionamento esperando que o mesmo encontre o destino da chamada. Caso servidor de redirecionamento reconheça a URI ele retorna seu endereço IP ao *proxy* para que a chamada seja encaminhada.

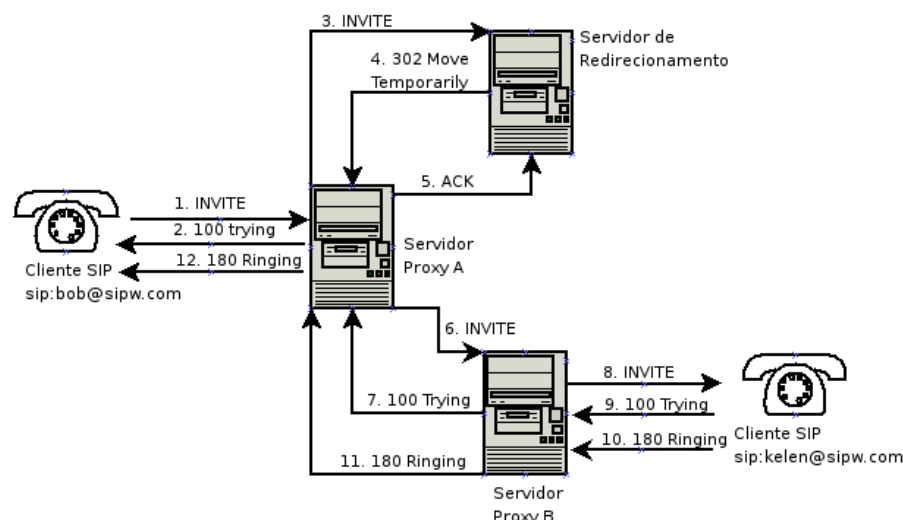


Figura 5: Chamada através de um servidor de redirecionamento

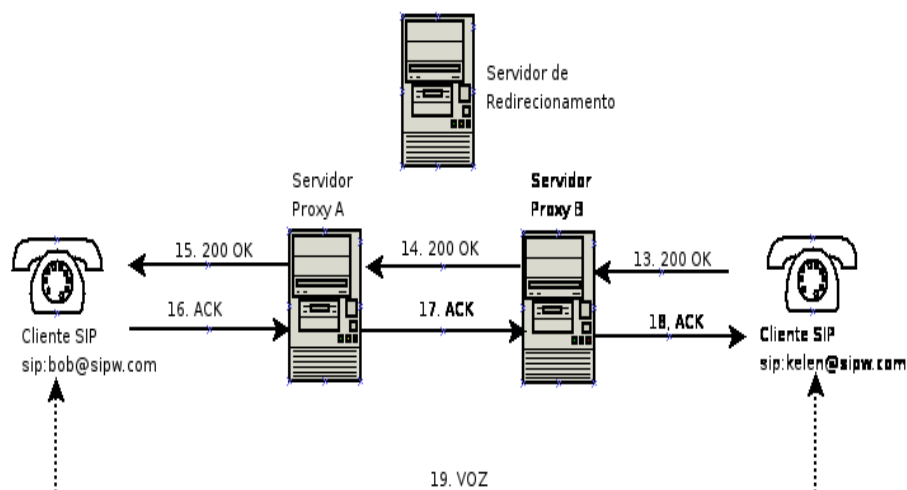


Figura 6: Finalização da chamada

A Figura 5 ilustra a sequência de mensagens trocadas para que uma chamada seja completada com o auxílio de um servidor *proxy* e um servidor de redirecionamento. O cliente SIP `sip:bob@sipw.com` deseja realizar uma chamada para `sip:kelen@sipw.com`, para isso envia uma requisição **INVITE** para o servidor *Proxy A* onde está cadastrado, e recebe uma mensagem **100 Trying** indicando que sua chamada está sendo processada.

O *Proxy A*, por sua vez, não reconhece a URI contida e a encaminha para um servidor de redirecionamento que conhece quem pode fornecer a localização exata do destino da chamada; para isso responde com uma mensagem **302 Moved Temporarily** que contém esta informação sendo confirmada seu recebimento através de um **ACK**. De posse desta informação o *Proxy A* encaminha o **INVITE** para a localização indicada pelo servidor

de redirecionamento, que é outro servidor *proxy*, o qual conhece a URI na requisição repassando-a ao cliente SIP final.

Após receber a requisição, `sip:kelen@sipw.com` responde com `180 Ringing` sinalizando que está esperando o aceite da chamada. Quando isto acontece, é enviada uma mensagem `200 OK`. Que é respondida com um `ACK` confirmando que ambos estão prontos para iniciar a transmissão de voz que é feita diretamente entre os clientes sem a necessidade dos servidores como ilustrado na Figura 6.

Existem diversas configurações possíveis para que uma chamada seja encaminhada até o seu destino dependendo dos serviços que serão oferecidos na rede, no entanto, a exposição dos mesmos está além do escopo deste trabalho. Mais detalhes sobre a sinalização e os serviços do protocolo SIP podem ser encontrados na RFC 3261 (9).

2.2 Uma Estratégia de Descentralização

A seção anterior tornou clara a dependência intrínseca do protocolo SIP em relação a arquitetura cliente/servidor e a sua fragilidade em caso de falha em um dos servidores. A busca por uma alternativa a essa limitação passa pela utilizações de arquiteturas descentralizadas.

As redes Peer-to-Peer (P2P) tornaram-se uma referência tratando-se da descentralização de aplicações. O termo *Peer* é a palavra em latim para **igual**. E desde modo, caracteriza indivíduos "considerados como iguais em relações as suas funções, situações ou características físicas" (20). Portanto, o termo **peer-to-peer** denomina a interação entre elementos semelhantes e com os mesmos direitos e responsabilidades. Este raciocínio lógico leva a formulação de um conceito bastante plausível sobre as Redes P2P que são "redes com dois ou mais nós, os quais possuem as mesmas responsabilidades, direitos e papéis no universo compartilhado onde se encontram". Sua principal característica é o descarte do uso de um nó central, fazendo com que os recursos tornem-se descentralizados e que a rede tenha uma maior tolerância a falhas. As redes P2P tornaram-se populares através de programas que possibilitaram o compartilhamento de arquivos (principalmente tipo .mp3) através da Internet como o Napster (21) e GnuTella (22). No entanto, este não é o seu único propósito. Esse tipo de rede pode ser caracterizada por permitir o compartilhamento de **recursos** independente de suas naturezas podendo ser arquivos, espaço em disco ou mesmo capacidade de processamento.

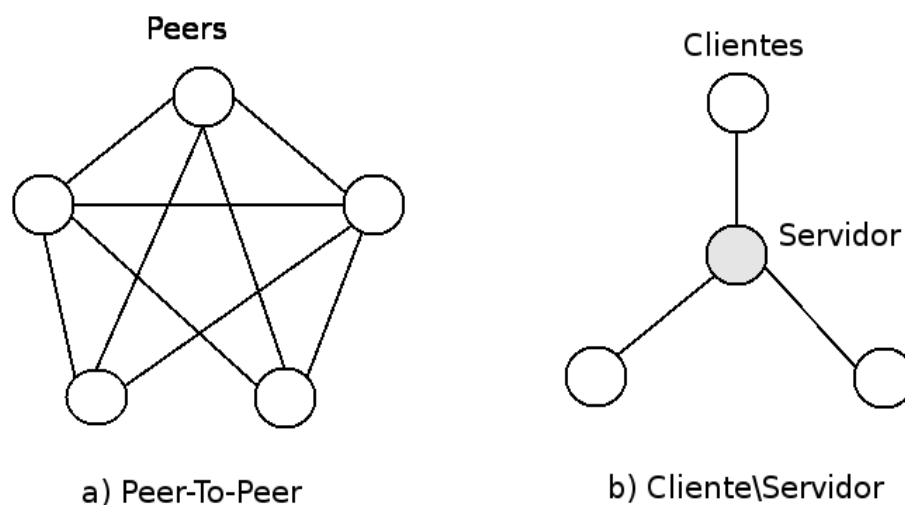


Figura 7: Redes P2P e Cliente/Servidor

A Figura 7 (a) mostra uma típica rede P2P onde todos os nós são equivalentes, podendo tanto atuar como clientes como servidores. Esse tipo de nó também é chamado

de *Servent* por Rudiger em (20). Na Figura 7 (b) já não se percebe a igualdade entre os nós, visto que existe a separação nítida e formal entre os clientes e o servidor.

2.2.1 Ciclo de vida de um Nó

As redes P2P são famosas por facilitar a escalabilidade dos seus nós, isto é, a inclusão ou retirada de nós não prejudica a operacionalização da rede. Isto é resultante de um ciclo de vida bem definido de cada nó, onde existem etapas que garantem a sua participação efetiva na rede. Essas etapas são:

1. **Conexão (*join*)**, nesta fase o nó precisa informar sua presença aos demais participantes da rede e se autenticar caso seja necessário;
2. **Operacionalização**, nesta fase o nó pode realizar buscas e também ser consultado por outros nós interessados em seus recursos.
3. **Desconexão (*disjoin*)**, é muito importante que o nó informe aos demais que está deixando a rede visando que medidas possam ser tomadas para minimizar o impacto da sua saída.

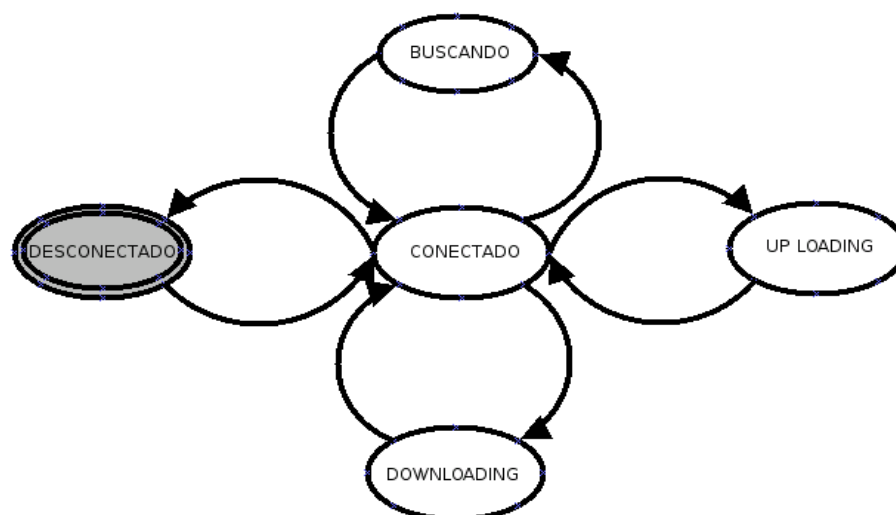


Figura 8: Estados de um nó

A Figura 8 ilustra os diversos estados que um nó em uma rede de compartilhamento de arquivos pode assumir. Ele inicia fazendo sua conexão na rede, entrando no estado de **CONECTADO** que o habilita a realizar buscas, a compartilhar seus arquivos (**UPLOADING**) e a também obter arquivos de outros nós (**DOWNLOADING**). O

estado **DESCONECTADO** é informado devido relevância da operação **DESCONECTAR** que cuida para que a rede tome conhecimento que o nó não está mais disponível.

2.2.2 Classificação das Redes P2P

As redes P2P podem ser classificadas segundo seu nível de descentralização (23) em:

- **Descentralizadas** - também apontada como P2P *pura*, não possui nenhum ponto de centralização e todos os nós são totalmente independentes. Podemos considerar o GnuTella(22) como um exemplo para esse modelo.
- **Semicentralizadas** - também apontada como P2P *híbrida*, pode possuir um ou mais nós centralizadores que geralmente são utilizados para a localização dos recursos e validação dos demais nós. Depois de localizados, a comunicação passa a ser direta entre um nó e outro. Um bom exemplo desse modelo é o Napster (21).

E quanto ao modo como seus nós estão organizados em:

- **Estruturadas** - organizam seus nós segundo algum método não aleatório, de maneira que eles sejam identificados deterministicamente utilizando-se de algum método matemático. Como exemplo temos CAN (24), Tapestry (25), Pastry (26), Chord (27), Kademlia (28) e Viceroy (29).
- **Desestruturadas** - organizam seus nós de maneira aleatória sem que os mesmos tenham qualquer conhecimento prévio sobre topologia da rede. Como exemplo temos Freenet (30), Gnutella (22), FastTrack/KaZaA (31), BitTorrent (32) e Overnet (33).

2.2.3 Métodos de busca

Os métodos de busca utilizados para encontrar os recursos compartilhados na rede também são uma forma de classificação comum e utilizada por Twosley (34) em seu estudo. Nesta sessão serão apresentados 3 métodos de busca, dando ênfase às suas estratégias de interação entre os nós.

2.2.3.1 Busca Centralizada

A busca é focada em um nó central que é responsável por indicar onde o recurso pode ser encontrado e no qual todos os nós que desejarem entrar na rede devem antes registrar seus recursos, depois que o recurso é localizado a comunicação passa a ser direta entre os nós interessados. O Napster inovou, visto que possibilitou a troca de arquivos entre vários usuários de forma direta entre a fonte e o destino. Contudo, os participantes não são capazes de descobrir a localização dos arquivos desejados de forma distribuída já que dependem de um servidor central. Quando um usuário deseja se conectar ao sistema, deve conectar-se ao servidor central e então fornecer a lista dos seus arquivos que estará compartilhando. Outro usuário conectado fará uma busca por arquivos que será respondida pelo servidor central. Eventualmente um arquivo disponibilizado anteriormente poderá ser listado como uma resposta para a busca do segundo usuário que poderá optar por iniciar uma transferência desde. Os arquivos são transferidos utilizando-se o protocolo HTTP diretamente entre a fonte e a origem sem intervenção do servidor. Pode-se notar que não é uma arquitetura puramente P2P, isto porque embora a transferência dos arquivos ocorra de forma distribuída, as buscas são respondidas de uma forma cliente/servidor. A Figura 9 ilustra a relação entre clientes e servidores dentro da rede Napster.

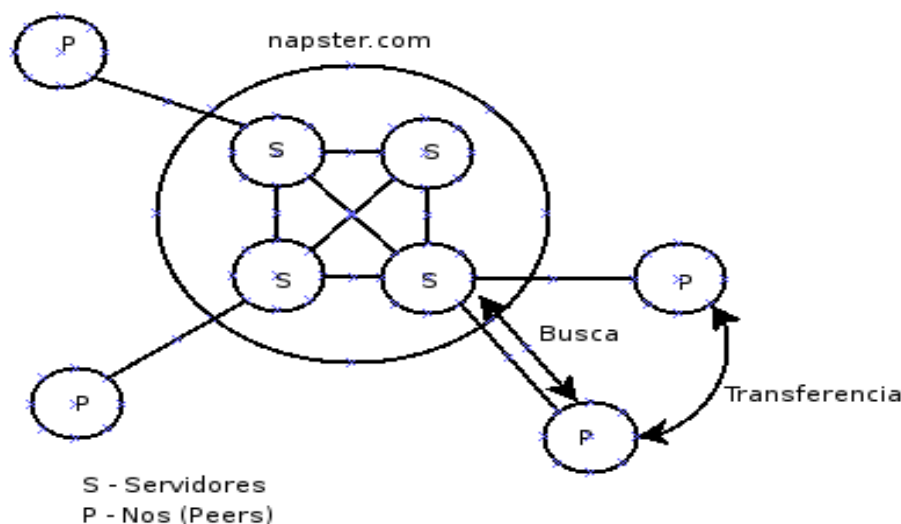


Figura 9: Arquitetura do Napster

2.2.3.2 Busca por Inundação (*Flooding*)

A busca é executada através de envio de consultas aos nós mais próximos ao nó que realiza a busca, e esta é repassada para os nós próximos a estes até que a rede seja completamente percorrida (inundada). Um exemplo de aplicação desse método é

o software de compartilhamento de arquivos chamado GnuTella(22). Na rede GnuTella cada nó é chamado de *servent* pela especificação do protocolo GnuTella e pode atuar tanto como cliente como servidor. Desta forma, eles possuem uma interface servidora para responder buscas dos pares e também possuem uma interface cliente para realizar suas próprias buscas. Isto causa uma alta redundância e o serviço não será interrompido se vários nós estiverem desconectados ou com problemas. Em (22) encontra-se uma descrição completa da versão 0.4 do protocolo GnuTella.

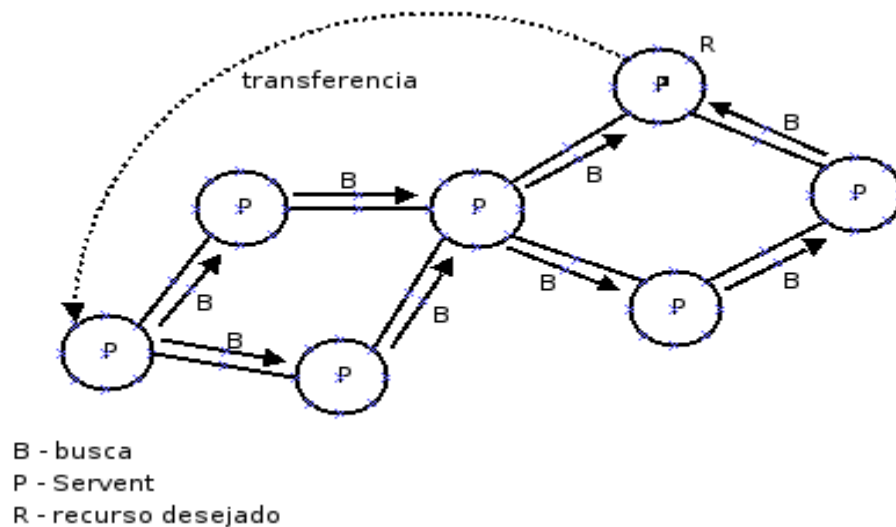


Figura 10: Método de Busca GnuTella

Para se conectar à rede, um *servent* estabelece uma conexão com outro nó que já esteja conectado e realiza buscas através da propagação da mensagem pelo nós. Para conter a inundação é utilizado um campo que controla a quantidade de nós atingidos por uma mensagem e barra a mesma quando o número máximo é atingido. A Figura 10 ilustra o método de busca utilizado em uma rede GnuTella.

Em sua mais recente versão 0.6, o protocolo do GnuTella incorporou um conceito chamado *ultrapeer* ou ultra-nó. Em suas versões anteriores as pesquisas eram direcionadas a todos os nós da rede, o que funcionava para aqueles que tiram banda passante suficiente para suportar a inundação. Os ultra-nós são utilizados de organizar um esquema hierárquico categorizando os nós em **folhas** e **ultra-nós**. Uma folha mantém apenas um pequeno número de conexões abertas, estas são sempre ligando-a a ultra-nós. Um ultra-nó atua como um *proxy* filtrando as pesquisas que chegam das folhas ligadas a ele.

Um ultra-nós só repassa uma pesquisa para uma folha ligada a ele quando considera que ela possa ser atendida, e as folhas não tem acesso as pesquisas repassadas entre os ultra-nós. Quando um ultra-nó conclui que a pesquisa não pode ser atendida pelas suas

folhas, ele repassa essa pesquisa para os demais ultra-nós da rede. Assim, o número de tráfego é diminuído substancialmente já que apenas as mensagens dos ultra-nós inundarão a rede. Visto que a rede Gnutella é descentralizada, a eleição de um ultra-nó não é feita através de um servidor, mas sim pelo próprio nó que segue alguns critérios para tal. Esses critérios estão detalhadamente descritos na RFC do GnuTella 0.6 (35) e serão apenas citados abaixo para um melhor entendimento do conceito. São eles:

- **Não esteja protegido por *firewall***
- **Sistema operacional robusto**, esteja sendo executado em um sistema operacional com suporte a multiportas, multiprocessamento, etc.
- **Largura de Banda suficiente**, pelo menos 15 KB/s.
- **Tempo de amadurecimento (*uptime*)**, um nó não pode se tornar um ultra-nó logo que se torne ativo. Isto deve ocorrer depois de algumas horas.
- **Memória e CPU suficientes**, visto que administrará várias folhas o ultra-nó precisa de capacidade de processamento e memórias suficientes para tal.

Estes critérios definem o mínimo necessário para que um nó seja eleito um ultra-nó, isto não quer dizer que atendendo esses requisitos ele necessariamente é um ultra-nó. A Figura 11, mostra como está organizada a estrutura hierárquica do conceito de ultra-nós.

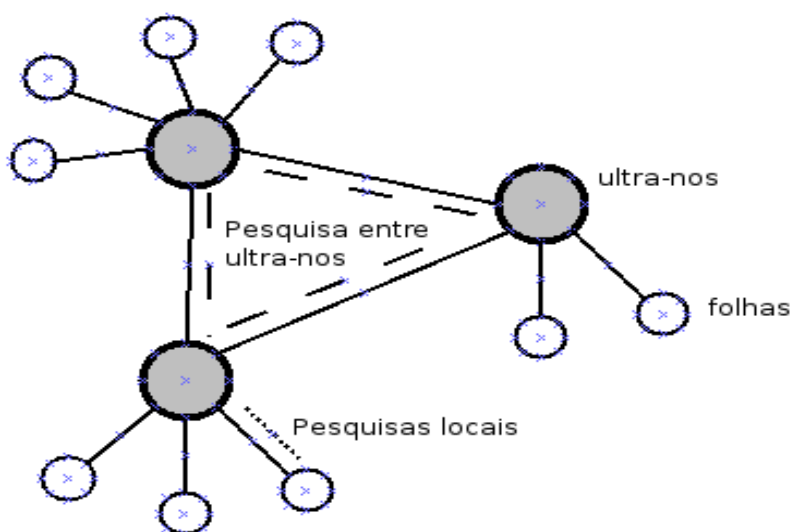


Figura 11: Arquitetura de ultra-nó do GnuTella v0.6

2.2.3.3 Busca por Tabela Hash Distribuída (DHT)

Tabelas *hash* (ou tabelas de espalhamento, ou tabelas de dispersão) são tabelas que utilizam uma função *hash* (ou função de dispersão) para realizar o mapeamento de diversas chaves em seus valores que são guardados como pares em um vetor. O valor da chave é utilizado como entrada da função *hash* que provê um inteiro como resultado. Este inteiro é utilizado como índice no vetor para se encontrar a chave em questão(36). Nessas redes os nós são autônomos e independentes e utilizam uma tabela *hash* para realizar as buscas. Ao entrar na rede cada nó recebe um identificador aleatório único a medida que recursos são publicados (compartilhados) um identificador é associado a cada um baseado em uma função de *hashing* e encaminhado para o nó segundo uma regra específica. Quando um nó realiza uma busca, a mesma função *hash* é aplicada ao nome do recurso desejado para encontrar sua localização na rede.

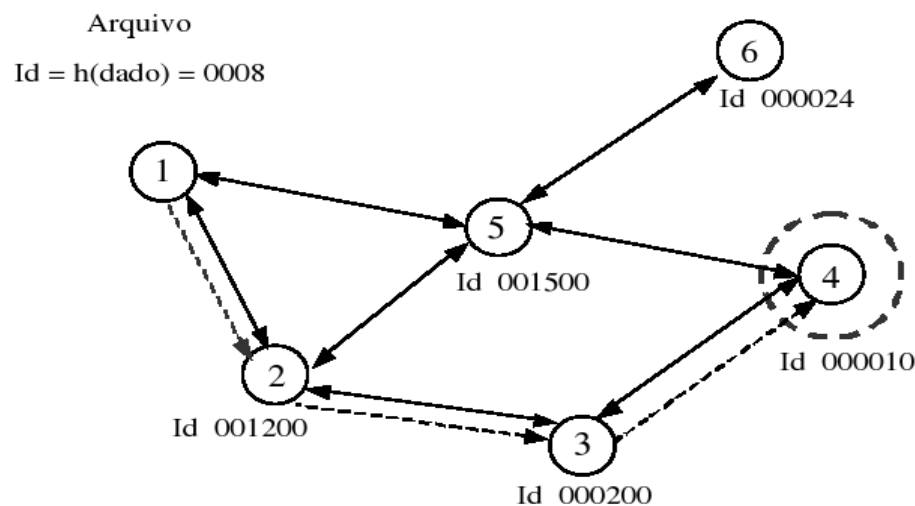


Figura 12: Busca utilizando DHT

Segundo nossas pesquisas um algoritmo chamado Chord(27) tornou-se muito popular pela sua eficiência nas buscas, simplicidade de implementação e facilidade de acoplamento com outras aplicações (37) e será apresentado com maiores detalhes na próxima seção pela sua relevância nos experimentos realizados neste trabalho. Para maiores informações sobre os demais algoritmos aconselha-se a leitura das referências citadas nesta seção.

2.2.4 Exemplo de implementação descentralizada

O Skype é um cliente VoIP-P2P de protocolo proprietário desenvolvido em 2003 pela KaZaa(38) que permite aos seus usuários a realização de chamadas de voz para telefones

fixo ou celulares e envio de mensagens de texto para outros usuários Skype. Foi desenvolvido partindo de uma aplicação P2P de compartilhamento de arquivos muito popular chamada **KaZaa** e incorporou várias de suas características como a estratégia de relacionamento entre os nós. Existem dois tipos de nós na rede Skype, nós ordinários (SC) e super-nós (SN). Um nó ordinário é um cliente capaz de realizar chamadas e manter conversações através de textos, enquanto um super-nó é responsável por um ou mais nós ordinários que devem se registrar em um deles para que possam participar da rede. Qualquer nó que possua um IP público e tenha capacidade de memória e processamento pode se tornar um super-nó. A rede Skype utiliza um servidor de autenticação central onde a senha e os nomes de usuários são armazenados garantindo-se suas unicidades.

Todo cliente Skype precisa armazenar uma lista contendo o endereço IP e o número porta de SN conhecidos chamada de *host cach* de modo a facilitar sua entrada na rede e a busca por outros nós. A busca por um SC necessariamente passa pela consulta a seu SN, que lhe fornece 4 (2^2) endereços de nós para consulta, caso não obtenha sucesso, o SC solicita mais 8 (2^3) endereços e assim continua indefinidamente até que o nó desejado seja encontrado. Segundo um estudo feito por Salman(39), a condição de falha não é clara e uma busca pode gastar bastante tempo até que seja dada como encerrada.

Embora o Skype seja visto como uma aplicação inerentemente P2P, ele ainda apresenta alguns pontos de centralização em seus super-nós e principalmente no seu servidor de autenticação. Isto significa que não poderia ser utilizado em uma rede sem conexão externa, tornando-o inadequado para utilização em redes sem infra-estrutura definida como as *ad hoc*, visto que nem sempre é garantido a conexão com todos nós. Outro ponto negativo a seu respeito é que seu protocolo é proprietário tornando-o incompatível com outras aplicações baseadas em protocolos abertos.

2.3 Aspectos necessários à mobilidade

As soluções apresentadas para proporcionar descentralização às aplicações VoIP baseadas no protocolo SIP citadas na seção anterior tem como base redes infra-estruturadas e cabeadas. Este tipo de rede proporciona um ambiente estável e com recursos computacionais confortáveis de maneira que a performance não seja uma diretiva prioritária quando da implantação dessas aplicações. No entanto, a busca por mobilidade tem levado a um crescimento considerável de dispositivos móveis, que tornaram-se mais acessíveis aos usuários fazendo com que a demanda por serviços móveis acompanhassem essa tendência (40, 41). A mobilidade foi resultado do aparecimento das redes sem fio (*wireless*) que a princípio ainda mantinha algumas dependências das redes fixas, mas posteriormente tornou-se completamente independente na forma das redes *ad hocs* ou MANETs (*Mobile ad hoc Networks*).

2.3.1 Redes *ad hoc* Móveis (*MANETs*)

São redes *Wireless* sem a necessidade de infra-estrutura, onde os nós comunicam-se diretamente entre si. São geralmente utilizadas em ambientes ou situações onde não é possível a instalação de infra-estrutura computacional como em catástrofes, campos de batalha e terrenos geograficamente hostis. A figura 13 ilustra 3 nós (A, B e C) que juntos formam uma *MANET*, cada nó possui uma área de cobertura que indica até onde o seu sinal pode ser detectado. Os nós **A** e **B** estão ambos dentro da área de cobertura do outro, permitindo que se comuniquem diretamente, já o nó **C** está apenas acessível a **B** e caso queira se comunicar com A deve fazê-lo através deste. Como pode ser percebido a responsabilidade de encaminhar as mensagens é dos próprios nós e durante a escrita desse trabalho constatamos a existência de muitos algoritmos de roteamento voltados para esse fim, destacando-se (42–48).

A mobilidade é o maior diferencial nas *MANETs* tornando possível que seus nós se movimentem livremente dentro da área de cobertura da rede, permitindo assim, uma enorme flexibilidade em suas topologias física e lógica. Esse tipo de rede é de fácil instalação visto que não precisa de uma infra-estrutura previamente disponível, e uma vez que não existe uma figura centralizadora elas possuem uma maior tolerância a falhas já que não dependem unicamente de um nó. Apesar das vantagens, existe um preço a ser pago, pois os nós são geralmente equipamentos portáteis, leves e com pouca autonomia para permitir que seus usuários possam se locomover facilmente, o que impõe algumas

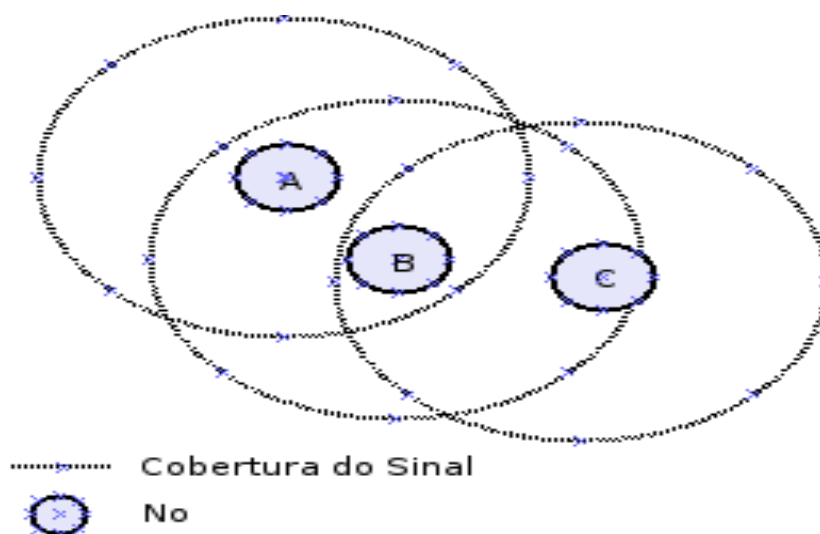


Figura 13: Rede adhoc

limitações como por exemplo: a utilização de uma banda passante reduzida, que chega em ambientes fechados na ordem de Mbps, interferências do ambiente e taxa de erros no enlace maior(49).

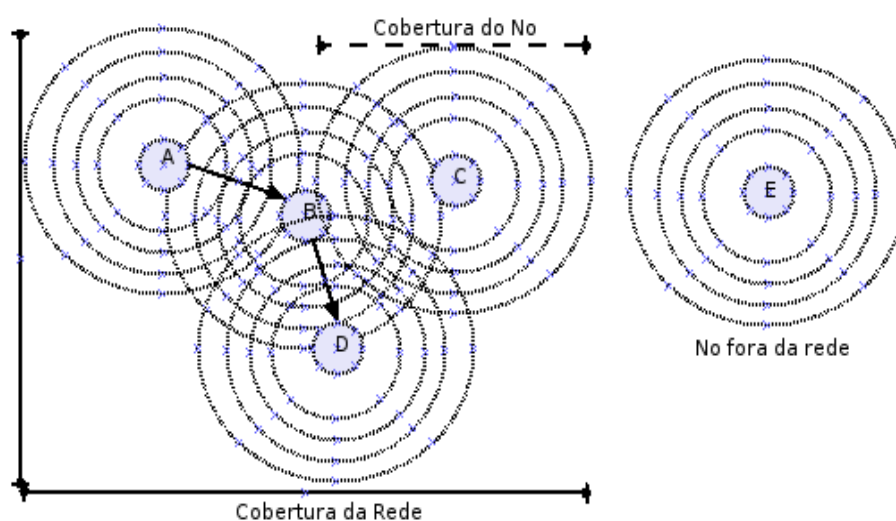


Figura 14: Alcance de uma rede adhoc

A baixa autonomia dos nós também contribui para que o alcance do sinal (*wireless*) seja reduzido, visto que seus transmissores não podem consumir energia demasiadamente sem o risco de esgotarem a bateria do equipamento. A cobertura da rede é dada pela junção das áreas de coberturas de todos os nós e pelos seus espalhamentos geográficos. A figura 14 ilustra a relação entre o alcance dos nós e o alcance total da rede, e mostra como as mensagens são encaminhadas. As redes *ad hoc* precisam ser escaláveis devido à flutuação do número de nós decorrente da movimentação dos mesmos. O protocolo

de roteamento deve ser capaz de encontrar novos caminhos pela rede caso um nó deixe de funcionar ou fique fora de alcance, e independente da estratégia adotada, devem ser facilmente escaláveis, e entradas e saídas de nós devem ser rapidamente detectadas e tratadas.

Segundo nossas pesquisas, o Padrão IEEE 802.11, é o mais difundido para criação de redes *ad hoc*. Ele foi publicado em (50), e é um padrão para redes locais sem fio (*wireless*) que define na camada física 3 modos de transmissão, a FHSS (*Frequency Hopping Spread Spectrum*), a DSSS (*Direct Sequence Spread Spectrum*) e o infravermelho. E na camada de MAC (*Medium Access Control*) dois mecanismos de controle de acesso ao meio, o DCF e o PCF. O primeiro é um mecanismo distribuído, no qual cada elemento da rede deve escutar o meio e transmitir apenas quando o meio estiver vazio e é utilizado nas *MANETs*, enquanto o segundo é um mecanismo centralizado onde o ponto de acesso controla o acesso ao meio(51).

Para atenderem aos requisitos impostos pelas *ad hoc* móveis, as aplicações de voz sobre IP também precisaram evoluir para serem escaláveis e totalmente descentralizadas. Isto foi alcançado primeiramente por Kundan(12) e posteriormente por Bryan (14) cujos estudos são suscitantemente apresentados no Capítulo 3.

2.3.2 Avaliação de performance em redes *ad hoc*

Uma rede *ad hoc* é muito sensível no que se refere a performance e pequenas mudanças podem acarretar ganhos ou perdas significantes. Para que se possa avaliar sua performance é necessário a escolha de métricas que informarão o comportamento da rede enquanto alguns fatores são modificados. No estudo (52), o autor sugere a utilização das seguintes métricas que também serão utilizadas neste trabalho:

- **Vazão** - medida em bits por segundo (bits/seg ou bps). Mede a efetividade da rede em entregar pacotes de dados, de uma fonte até o destino. Trata-se da quantidade de bits recebidos com sucesso em um segundo por uma determinada estação.
- **Número de pacotes de sobrecarga** - número médio de pacotes de controle de roteamento, incluindo replies, requests e erros.
- **Taxa de Entrega** - determina a eficiência da rede. Determina quantos pacotes em média uma estação precisa enviar para que este pacote chegue ao destino com sucesso.

- **Taxa de Perda** - devido a erros na camada física ou em camadas superiores pacotes podem não ser entregues ao destino.

Segundo o autor, tais métricas são principalmente influenciadas por quatro fatores: quantidade de nós, densidade (nós por m^2), mobilidade e quantidade de conexões simultâneas. O estudo foi baseado no padrão 802.11g e utilizou o protocolo de roteamento AODV (53) (ad hoc On-demand Distance Vector) que obriga a todos os nós intermediários a estabelecerem dinamicamente entradas em tabelas de roteamento locais para cada destino ativo onde cada nó sabe qual é o próximo nó para chegar ao destino e a distância em número de saltos.

3 *Problema de Interesse*

No capítulo anterior foram apresentados os fundamentos de aplicações de telefonia IP baseadas no protocolo SIP e a dependência existente ao modelo cliente/servidor que ainda persiste neste tipo de aplicação. Como alternativa ao modelo centralizado foram apresentados os conceitos que regem a tecnologia P2P, assim como algumas de suas implementações. As redes *ad hoc* também foram apresentadas como possível cenário onde a demanda por uma estratégia descentralizada por parte das aplicações P2P-VoIP é mandatória e estudos estão sendo desenvolvidos no sentido de proporcionar escalabilidade e robustez a esse tipo de aplicação. Neste capítulo apresentaremos dois desses estudos e a necessidade de uma investigação mais profunda no que se refere aos impactos não aparentes sobre o ambiente e por conseguinte sobre a aplicação em uso.

3.1 Trabalhos Relacionados

Kundan, em seu estudo (12), propôs a utilização de uma arquitetura baseada na busca com DHT utilizando o algoritmo *Chord* para localização dos nós, tendo sua sinalização totalmente baseada nas requisições previstas no protocolo SIP. Os participantes do anel podem assumir eventualmente o papel de **super-nós** tornando-se responsáveis diretamente por outros nós da rede, agindo como um Servidor *Proxy* e/ou de Registro. A função *hash* é aplicada à URI do usuário para a codificação da chave, enquanto o IP dos nós é utilizado para a geração dos seus identificadores no anel. O processo de entrada de um nó no anel é iniciado através do envio de uma requisição **REGISTER** para o seu **sucessor** ou para um endereço previamente conhecido. O anel é mantido através da renovação periódica do registro por parte dos nós, que são considerados inativos ou fora da rede caso não realizem este procedimento dentro do tempo especificado. Os nós também são capazes de detectar a presença de outros através de uma "Lista de Amigos" (*Friends List*), nesse caso os endereços contidos nesta lista serão os primeiros a serem procurados.

Outro trabalho publicado por David Bryan (14) apresenta uma solução semelhante.

Seu estudo se concentra em combinar VoIP e Mensagens Instantâneas (*Instant Messaging-IM*) em uma aplicação P2P. Para tal, escolheu respectivamente os protocolos SIP(9) e SIMPLE (54) por serem de domínio público e por já possuírem várias implementações disponíveis. A estratégia adotada também utiliza o algoritmo *Chord* para formação da rede e busca de nós, mas adiciona a possibilidade de escolher entre uma chamada VoIP ou uma conversa textual utilizando *Instant Messaging*.

Atualmente, um grupo de desenvolvedores tomaram para si a tarefa de planejar e implementar uma solução SIP totalmente descentralizada através de uma lista de discussão (55) onde problemas, soluções e idéias são debatidos. No entanto, somente rascunhos e propostas foram apresentados sem que um consenso sobre um padrão de fato tenha sido alcançado. Pesquisas sobre o comportamento de aplicações P2P-VoIP em redes *ad hoc* ainda são tímidas, visto que, o principal foco das discussões é sobre a capacidade de travessia de NATs por parte da aplicação.

Os estudos realizados por Kundan e David Bryan foram os pioneiros no que se refere a busca por uma solução VoIP-P2P totalmente descentralizada e utilizando um protocolo aberto e até o presente momento nenhum outro trabalho consistente foi publicado sobre o assunto. Seus resultados mostraram que o preço pela descentralização é o aumento da **latência de sessão** que pode consumir até 6 vezes mais tempo que a observada utilizando-se uma arquitetura cliente/servidor.

3.2 Considerações sobre uma chamada VoIP

Chamaremos neste trabalho de *Latência de Sessão* o tempo que um cliente VoIP leva para o estabelecimento de uma sessão com outro. Este tempo começa a ser contabilizado a partir do envio da requisição INVITE pela origem e é encerrado pelo recebimento da requisição ACK pelo destino.

Estudos já realizados mostraram que há um aumento significativo no tempo de latência de sessão quando o ambiente se torna descentralizado o que pode acarretar um aumento ainda maior quando sobre uma rede *ad hoc*. Segundo (10, 11, 18) um tempo aceitável de latência de sessão seria entre 2 a 5 segundos e a partir daí o usuário tornaria-se impaciente pelo fato de não receber resposta alguma da aplicação.

Como já mencionado, uma chamada VoIP pode ser dividida em duas etapas: Localização e transmissão multimídia. Num ambiente descentralizado a localização é fator determinante para que um usuário possa iniciar uma chamada com outro, visto que sem

saber seu endereço IP não poderá alcançá-lo. A transmissão multimídia pode ocorrer diretamente entre os pontos após ambos terem conhecimento de suas localizações.

3.3 Definição do foco

Esperamos observar neste trabalho o impacto exercido por algoritmos de busca normalmente utilizados em aplicações P2P sobre aplicações VoIP em uma rede *Wireless adhoc* através de medições direcionadas ao tráfego e utilização do meio físico por parte desta aplicação. Através de mudanças feitas no meio pretende-se identificar os fatores determinantes na performance da rede e do algoritmo de busca utilizado. Para então sugerir indícios para a otimização das estratégias de localização das aplicações P2P-VoIP já disponíveis e as que vierem a ser desenvolvidas.

O número reduzido de aplicações P2P-VoIP disponíveis para a realização dos experimentos e a disponibilização de um cenário suficientemente real para que os dados sejam coletados configuram-se nos maiores desafios. Embora existam diversas opções de algoritmos de busca P2P, poucos são disponibilizados como aplicações e as poucas aplicações disponíveis ainda possuem restrições de licença de uso. Dentre as opções disponíveis apresenta-se o uso de simuladores de redes virtuais tanto do meio físico como do que se concerne a aplicação, no entanto observou-se em pesquisas preliminares que esses tipos de simuladores atuam em diferentes camadas do modelo OSI/ISO o que os torna incompatíveis forçando a busca de alternativas para que seja possível a combinação e observação de resultados através desses ambientes. A adaptação de uma aplicação P2P-VoIP para que suporte algumas mensagens do protocolo SIP também é iminente, e pode configurar-se em um esforço considerável visto que as bibliotecas disponíveis de algoritmos P2P teriam que ser compatíveis com o ambiente.

3.4 Benefícios esperados

A ubiquidade é cada vez mais uma realidade no cenário das redes de computadores, os dispositivos estão melhores, menores e totalmente conectados. Por algum tempo a tecnologia de VoIP ficou encerrada em ambientes fixos, mas logo a necessidade de mobilidade levou a estudos que quebrassem suas algemas desse tipo de ambiente. No entanto para cada novo desafio surgem novos obstáculos a serem superados por meio de pesquisa e experimentação.

Redes móveis *ad hoc* ainda são uma fronteira a vencer para as aplicações VoIP, deve-se isto em grande parte pela imprevisibilidade de suas topologias que tornam a aplicação dos atuais protocolos inviáveis, pedindo assim, que novas alternativas sejam oferecidas. Como em toda batalha, o conhecimento do campo onde ela deve ser travada é fundamental para o seu sucesso. Os dados fornecidos por este estudo podem nortear o desenvolvimento de novas aplicações que levem em consideração as diversas nuances na utilização de P2P-VoIP em ambientes totalmente descentralizados tanto no nível do *software* quanto do *hardware*. Tais aplicações poderiam tornar-se sensíveis ao ambiente, e assim possibilitando sempre a busca da melhor configuração para um determinado ambiente a qual forem expostas.

4 *Abordagem do problema*

O capítulo anterior preocupou-se em oferecer uma visão esclarecedora sobre o problema abordado neste trabalho; assim como visões provenientes de outros estudos já publicados e que o interseccionam oferecendo alternativas para novas abordagens. Também foram apontadas lacunas no que se refere ao estudo do comportamento das aplicações P2P-VoIP sobre redes *adhoc* e que permitiram o fomento deste trabalho. Neste capítulo serão abordadas as diretivas que nortearam a formulação das estratégias de ação pelas quais se almeja alcançar resultados consistentes e esclarecedores sobre o problema proposto.

4.1 Escolhendo uma estratégia P2P-VoIP

Ao iniciar as investigações para a realização deste trabalho procurava-se uma abordagem P2P para a qual o protocolo SIP fosse adaptado e assim realizar estudos do seu comportamento em redes *ad-hocs*. Esta tarefa mostrou-se árdua visto que à época (meados de 2003) existiam apenas duas aplicações que aparentemente atendiam as necessidades: Skype e Microsoft Messenger.

Essas aplicações são proprietárias e seus protocolos não foram abertos para a apreciação da comunidade científica, o que dificultou o entendimento de seus protocolos. No entanto Saikat (56) e Salman (13) demonstraram que existem elementos centralizadores no Skype chamados *Skype Login Servers*, pelo menos no que se trata da validação de seus usuários. O mesmo pode ser verificado em relação ao MS Messenger, que necessariamente solicita que seus usuários se conectem aos seus servidores de acesso, para só então poderem utilizar os serviços disponíveis. A natureza do ambiente *adhoc* não favorece a utilização de aplicações que possuam algum tipo de ponto centralizador, visto que, uma falha neste possa comprometer todos os demais nós da rede.

Investigações posteriores mostraram a existência de duas outras aplicações proprietárias

ditas VoIP-P2P como Avaya-One X (7) e PeeRio Biz (57), mas que, no entanto, não estão disponíveis para utilização sem que sejam compradas.

O trabalho realizado por Kundan leva em consideração redes fixas e cabeadas, e nenhum estudo foi realizado sobre o comportamento de sua solução em ambientes *ad hoc*, mas infelizmente sua aplicação não foi disponibilizada para que fosse submetida a esse ambiente. Bryan deu um passo adiante em seu estudo e propôs uma solução que também englobasse redes totalmente desestruturadas e móveis, no entanto a indisponibilização da sua aplicação para a comunidade em geral não permitiu sua validação e estudo em nossos experimentos.

Na ausência de uma aplicação já consolidada para a realização dos experimentos, restou a alternativa de adaptar protocolos ou aplicações P2P já amplamente disponíveis na Internet. Os exemplos de redes P2P estruturadas e desestruturadas fornecidos no Capítulo 2 foram selecionados e analisados segundo o seguinte critério:

1. Descentralização do Algoritmo
2. Disponibilidade da Aplicação
3. Abertura do código
4. Compatibilidade com o ambiente de experimentação

Dado os critérios acima, as opções ficaram restritas a uma aplicação: o *mutella* (58). Este *software* é uma implementação do GnuTella (22) versão 0.6 (35) que utiliza o conceito de ultra-nó já citado no capítulo anterior. Além de atender aos requisitos já citados, o *mutella* tem seu núcleo P2P em forma de uma biblioteca que pode ser utilizada em outras aplicações com modificações substancialmente menores do que as necessárias em outras bibliotecas existentes.

4.2 Construção do Ambiente

Existem três maneiras disponíveis para estudar o comportamento de uma aplicação em um ambiente *ad hoc*. A primeira é utilizando uma rede real, o que poderia limitar a quantidade de nós envolvidos na experiência visto que dificilmente teria-se um número de nós suficientes para validar os resultados (acima de 100). A segunda é por meio matemático onde criaria-se um modelo para a representação dos cenários envolvidos. Por

fim, a terceira forma é fazendo uso da simulação, que nos oferece liberdade e maior controle sobre as variáveis a serem modificadas sem que para isso devamos criar um novo modelo matemático. Optou-se pela terceira alternativa sobre a qual norteamos nossas investigações de disponibilidades de *hardware* e *softwares* necessários para a realização dos experimentos.

O primeiro obstáculo encontrado na criação do cenário foi o fato de que redes P2P são percebidas na Camada de Aplicação do modelo ISO/OSI sendo apenas virtuais e ignoram o meio no qual estão sendo utilizadas. No entanto, redes *adhoc* são redes fisicamente definidas e utilizam *hardwares* e protocolos específicos para o seu funcionamento. O problema reside no fato de que existem simuladores específicos para cada uma, mas não um simulador (até a finalização deste trabalho) que consiga simular tanto o meio físico quanto as aplicações sobre ele.

Optou-se então por uma abordagem em camadas, onde a aplicação pudesse ser utilizada por sobre um ambiente totalmente simulado de maneira transparente. Este modelo foi inicialmente utilizado por Abinader (59) em seu estudo que faz uso de um simulador de redes muito popular chamado Network Simulador (NS2) (60) combinado com um ambiente de execução virtual chamado UML.

O NS2 é um simulador baseado na Camada de Rede do Modelo OSI/ISO, portanto a simulação de uma rede P2P não seria possível, visto que deveria ser executada em uma camada superior. O que se faz é transferir a responsabilidade de comportar a aplicação para o ambiente UML, deixando apenas o meio por onde as mensagens serão enviadas por conta do NS2. Essa estratégia permite que se tenha controle total sobre o meio a ser utilizado assim como sobre as variáveis a serem modificadas para a observação dos resultados.

A coleta dos dados gerados nos experimentos também deverá observar essas duas camadas sendo necessário monitorar tanto o meio fornecido pelo NS2 quanto o comportamento da aplicação hospedada dentro do UML. Felizmente, o NS2 já fornece meios de coleta de dados em seu ambiente, gerando inclusive arquivos que podem ser utilizados diretamente em um *software* gerador de gráficos. Quanto ao monitoramento da aplicação, resolveu-se pela utilização do *iptraf* (61) e do *Ethereal* (62) considerando sua ampla utilização no ambiente científico para monitoramento de redes.

configuração as buscas terão pouca chance de serem bem sucedidas devido a baixa densidade de nós que leva ao isolamento de alguns e a formação de "redes" independentes.

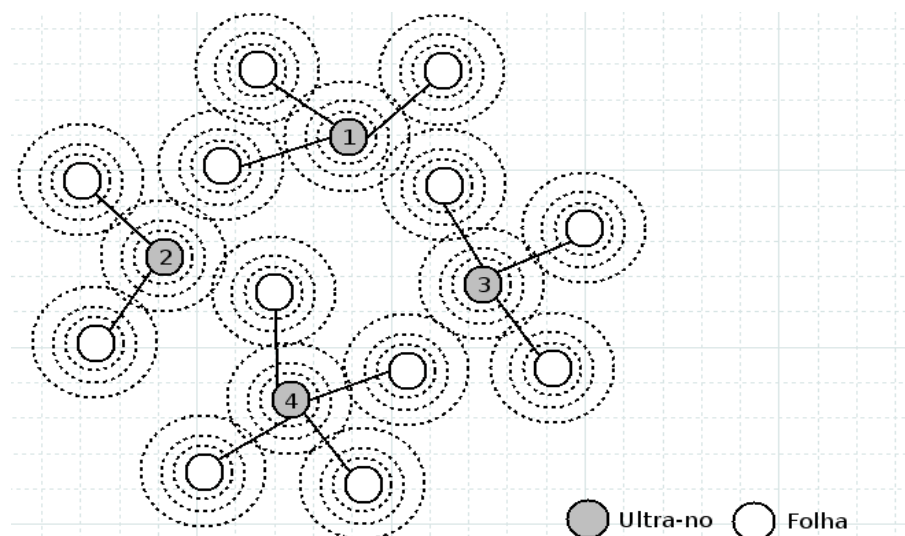


Figura 16: Alta densidade de nós

Na Figura 16 os nós estão posicionados dentro de uma área menor possibilitando que todos façam parte do mesmo grupo ou rede. Neste tipo de configuração de maior densidade de nós, espera-se que as buscas sejam mais bem sucedidas e que o tempo de latência de sessão seja consideravelmente inferior ao percebido em um ambiente de baixa densidade.

4.3.2 Hipótese 2

Caso se comprove a hipótese anterior, a mobilidade dos nós também se tornaria um fator influenciador na performance tanto da rede quanto da aplicação P2P. É previsível que os nós mantenham-se entrando e saindo do alcance um dos outros a medida que se movimentam e dessa maneira as estruturas hierárquicas formadas pelos protocolos P2P estarão em constante mutação levando a um *overhead* causado pelo processo de *join* adotado pelo nó.

Na Figura 17 são mostradas diversas estruturas hierárquicas formadas conforme nós juntam-se a rede, elas são definidas pelo protocolo P2P utilizado e definem que existam nós com maiores responsabilidades, os ultra-nós e outros que atuam apenas como membros ordinários, ou folhas. No ambiente *ad hoc* essas estruturas estão sujeitas a perdas inesperadas de membros, e seus participantes podem ter que atualizar suas conexões, ou mesmo estabelecerem novas conexões para que não fiquem isolados.

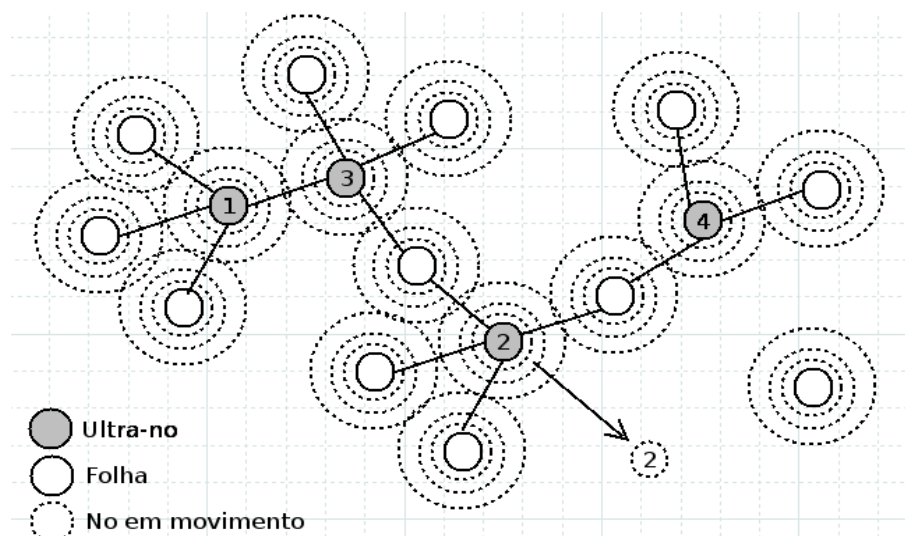


Figura 17: Estruturas hierárquicas da rede P2P

A Figura 17 ilustra uma possível movimentação do nó 2, que é um ultra-nó, de forma que fique fora de alcance dos demais membros da sua estrutura, o que causa uma quebra das conexões que mantinham os nós coesos impossibilitando que realizem buscas por toda a rede. A Figura 18, ilustra a nova configuração da rede após a movimentação do nó.

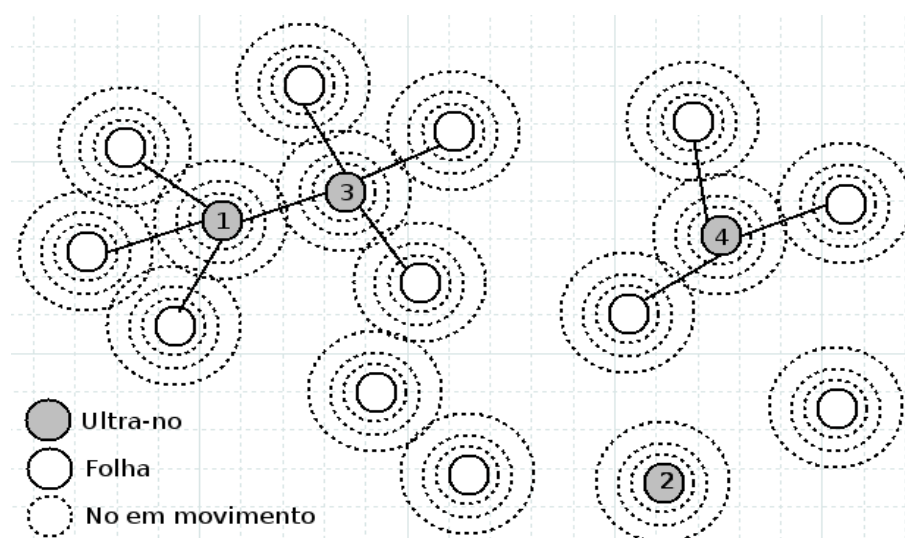


Figura 18: Estruturas hierárquicas da rede P2P após a movimentação de um nó

Acredita-se, pelo conhecimento empírico adquirido pelo autor através da convivência com o ambiente *ad hoc* e aplicações VoIP, que o modelo hierárquico não é indicado para redes onde espera-se um alto grau de mobilidade, visto que o tempo de reconfiguração dessas estruturas quando ocorre uma perda de um membro pode gerar um alto *overhead*. A utilização pura do método de inundação poderia diminuir esse *overhead* já que os nós tornariam-se completamente independentes e suas mensagens seriam propagadas para

toda rede, e não primeiramente para os ultra-nós aos quais estão subordinados. O efeito colateral esperado com a utilização dessa abordagem é um crescimento do tráfego na rede, já que seria inundada com mensagens de todos os nós participantes.

5 *Experimentos Realizados*

O capítulo anterior apresentou hipóteses formuladas a partir de experiências empíricas realizadas durante a experiência profissional e acadêmica do autor as quais espera-se comprovar ou não por meio dos experimentos descritos neste capítulo. Nele será apresentado o cenário criado para que a coleta de dados fosse possível, assim como a descrição e preparação desses experimentos seguidas pelas considerações finais sobre os resultados obtidos.

5.1 Metodologia

O primeiro passo foi definir como a avaliação de desempenho dos algoritmos de busca do mutella poderiam ser realizados dentro de um ambiente real. Esse ambiente real consistiria de várias máquinas com o mutella instalado e dispositivos de comunicação sem fio conectados a elas, afim de prover conectividade entre os nós, no entanto, a realização desse tipo de experimento iria requerer recursos além dos disponíveis.

Partiu-se então para a utilização do NS2(60), simulador de eventos discretos bastante disseminado na comunidade acadêmica de redes, que possuía todos os recursos necessários implementados, como a possibilidade de simulação de uma rede adhoc com inúmeros nós, a implementação das camadas físicas e MAC das redes sem fio 802.11 utilizados nos nossos experimentos, além da possibilidade de variação dos parâmetros da rede, como taxa de erro de bit (BER), área de cobertura e distância dos nós.

Como já foi mencionado anteriormente, o NS2 não simula a camada de aplicação, onde se encontra o mutella, ou seja, não seria facilmente possível realizar os experimentos com o protocolo de busca P2P, ao menos que houvesse uma modelagem matemática desse tipo de tráfego, o que ainda não seria completo, pois desejamos avaliar a interação entre os nós.

Outras soluções de simuladores também foram investigadas, dentre elas a utilização,

o OMNet (63). No entanto, a solução que atendia nossas necessidades foi a emulação. Com a emulação seria possível utilizar uma rede 802.11 com vários nós, tendo o tráfego transmitido nessa rede seria gerado por um cliente P2P real, tornando possível utilizar o mutella e deixar os experimentos mais realísticos.

A emulação de uma rede com nós reais não é trivial, pois exige a interação de um simulador de redes e um sistema operacional (kernel) adaptado. A solução encontrada foi baseada nos trabalhos em (59, 64). Essa solução utiliza o NS2 em conjunto com o UML (User-Mode-Linux)(65). O UML cria máquinas virtuais Linux dentro de uma mesma máquina física e os tráfegos gerados por essas máquinas virtuais são enviados para o NS2 que emula uma rede entre elas.

As máquinas virtuais possuem um sistema de arquivo onde foi instalado o mutella. Como o UML é uma outra arquitetura, diferente da Intel x86 comumente utilizada, foi necessário realizar uma adaptação no código do mutella para que ele fosse compilado e o compilador gerasse código executável para esse tipo de arquitetura.

A integração entre NS2 e UML mostrou-se complexa, visto que foi necessário adaptar tanto o simulador quanto o kernel do Linux para que trabalhassem em conjunto. Além disso, a preparação do sistema de arquivos foi importante para poder executar o mutella dentro da nova arquitetura.

A solução mostrou-se eficiente visto que permitiu que o simulador captasse dados de uma aplicação P2P real e oferecendo o ambiente sem fio desejado. Também possibilitou a realização experimentos que demandariam muitas máquinas em apenas uma com poder computacional suficiente, gerando tráfego real em uma rede totalmente manipulável. Após esta fase, houve a preparação dos arquivos que descreviam a rede a ser simulada pelo UML e a preparação das ferramentas de coleta dos dados. A configuração e preparação do ambiente e das soluções geradas será apresentada nas próximas seções.

5.2 Preparação do Ambiente

Como dito anteriormente é necessário que o hardware utilizado no ambiente atenda as especificações mínimas descritas na documentação do UML que reserva 128 Mb de memória RAM e 1 Gb de disco para cada máquina virtual. Foi utilizado um computador desktop com processador Intel Pentium 4 de 3.0 Ghz, 1 GB de memória RAM e 120 GB de capacidade de disco rígido. Experimentos iniciais mostraram que máquinas de menor capacidade computacional mostraram resultados insatisfatórios, visto que tinham

seu processamento congelado na maioria das simulações. Além disso, existe o consumo de processamento equivalente de três kernels do Linux sendo executados simultaneamente, mais o NS2.

Primeiramente foi necessário a escolha de uma distribuição Linux que desse suporte ao UML e por conseguinte permitisse ser configurada adequadamente. A distribuição Linux utilizada na máquina host (que hospeda as máquinas virtuais) foi o **Debian Sarge** por ser uma distribuição reconhecidamente estável e facilmente adaptável. A versão do kernel utilizada foi a 2.6.16, pois resolve uma falha na operação das interfaces TUN/TAP evitando assim um *kernel panic*.

Uma importante configuração foi a inclusão de suporte ao NAT, pois posteriormente as máquinas virtuais deverão se comunicar com a Internet para atualização de pacotes da distribuição. Esse processo é feito configurando o *iptables* no kernel do Linux e recompilando-o.

O UML é um modo seguro de executar programas experimentais em um sistema Linux, onde é possível executar aplicações distintas sem que o sistema hospedeiro (*host*) seja comprometido. Portanto, todos os recursos do UML são inferiores aos recursos da máquina hospedeira, uma vez que é um simples arquivo que contém um outro sistema de arquivos e de um kernel compilado com suporte para UML. Todos os kernels da versão 2.6 possuem suporte a UML no código fonte, basta que para isso seja habilitado no menu de configuração para depois compilá-lo.

O sistema de arquivos pode ser construído ou obtido gratuitamente em (65). A opção mais comum, e utilizada neste trabalho, é a segunda, pois satisfaz as necessidades para execução da aplicação. Na Internet, é possível encontrar sistemas de arquivos que imitam as a maioria das distribuições Linux. Neste trabalho, foi utilizado um sistema de arquivos baseado na distribuição Debian, já mencionada nos parágrafos anteriores.

Tendo o ambiente de emulação pronto, partiu-se para a instalação e configuração do simulador NS2. A versão utilizada nestes experimentos foi a 2.27 disponível em (60). Depois de instalado, o próximo passo foi instalar o módulo NSE, que permite emular um tráfego real dentro do NS2, como explicado anteriormente. O módulo NSE pode ser encontrado em patches e aplicado no diretório raiz do NS2.

O módulo NSE requer que muitas outras ferramentas externas ao NS2 já estejam funcionando perfeitamente. Uma dessas ferramentas é a emulação de terminal UML com suporte às interfaces de rede TUN/TAP. Este suporte deve ser habilitado no menu

de configuração do kernel e, para que as configurações surtam efeito, o kernel deve ser recompilado.

Foi necessário ainda uma adaptação importante no ambiente de emulação, pois precisou-se modificar o valor do tamanho máximo de uma unidade de transmissão (Maximum Transfer Unit-MTU). Essa modificação é devido ao fato desse valor ser diferente na Ethernet, 1500 bytes, enquanto que no padrão 802.11 é 2312 bytes. Essa modificação é feita através de um *patch* para o kernel do Linux já que as interfaces TUN/TAP foram nativamente implementadas para redes Ethernet. Portanto, é necessário uma modificação desse valor no código fonte do kernel.

Além disso, foi utilizado o software *tunctl* (66) que gerencia as interfaces TUN/TAP. Este programa pode ser encontrado em (66) para qualquer distribuição Linux. Com isso, emulou-se os terminais UML, com rede sem fio simulada pelo NS2.

A terceira parte foi a configuração do *mutella* nas máquinas virtuais e sua adaptação para o envio de uma requisição **INVITE** e reconhecimento da resposta. A execução dessa aplicação mostrou-se um difícil obstáculo uma vez que detectou-se que não era compatível com o ambiente emulado oferecido pelo UML, e suas bibliotecas eram projetadas para operarem no ambiente x86. A superação veio através da alteração do código fonte para que suportasse o ambiente emulado.

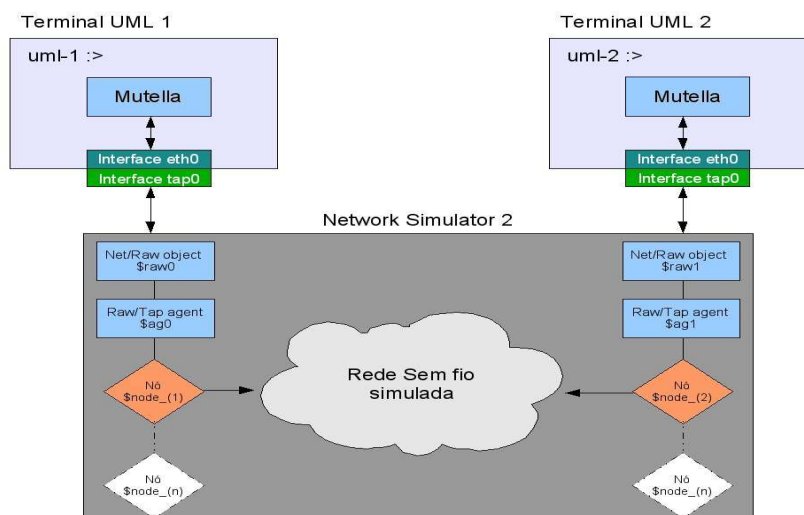


Figura 19: Esquema de utilização UML e NS2

A Figura 19 apresenta uma descrição do ambiente que foi utilizado. O *mutella* foi configurado nos Terminais UML 1 e 2 na parte superior da figura. Cada terminal UML possui uma interface *eth0* por onde os pacotes de dados são enviados para a rede onde estão conectados, seja a Internet ou o NS2. Para o caso que essa rede seja provida pelo

NS2, essas interfaces são conectadas à uma ou mais interfaces virtuais conhecidas como **tap0**. É através dessas interfaces virtuais que os pacotes reais são introduzidos numa rede emulada. A parte inferior da figura apresenta essa rede emulada. Um agente do NS2 recebe esses pacotes vindos do mutella e da interface tap0 e os envia através da rede adhoc para o destino.

Como se pode observar, cada nó que participa da simulação está conectado a uma instância de um agente Raw/Tap que foi configurado para capturar pacotes de uma determinada interface de rede, neste caso, as interfaces de rede que o UML utiliza. E cada um desses agentes está conectado a um objeto NS2 Network, que é quem de fato captura os pacotes através da biblioteca tcpdump. Como no cenário vários nós foram criados, mas apenas dois enviavam dados reais, apenas precisamos adicionar de 2 a 50 nós extras na rede para o roteamento dos pacotes (*node(1...n)*).

5.3 Cenário dos experimentos dentro do NS2

O cenário utilizado nos experimentos envolve a transmissão de pacotes sobre uma rede que não possui um elemento centralizador, ou seja, uma rede que não é infra-estruturada. Todas as estações utilizam o protocolo 802.11 configurado através de um script em linguagem Tcl para o NS2. Vários parâmetros foram definidos dentro deste arquivo, afim de prover uma rede bem realística para os experimentos. A Tabela 1 apresenta um resumo destes parâmetros:

Parâmetros	Valores
Camada Física e MAC	802.11b
Espaço Geográfico	200m ²
Taxa de Erro no Canal	BER = 10 ⁻⁹
Número de Nós	2,10,25,50
Tempo total de uma simulação	300 s
Protocolo de Roteamento	AODV

Tabela 1: Lista de parâmetros utilizados na construção do cenário de experimentos

Portanto, tem-se um cenário com dois terminais UML que geram tráfego real e que desejam se comunicar. Mas, em uma aplicação real, não existiriam apenas dois nós em uma rede P2P. Por conta disso, o número de nós é aumentado para 10, 25, e 50 nós em cada cenário, mas o objetivo final, continua sendo os dois terminais UML se comunicarem um com o outro.

A Figura 20 apresenta uma ilustração simplificada do cenário dos experimentos, com

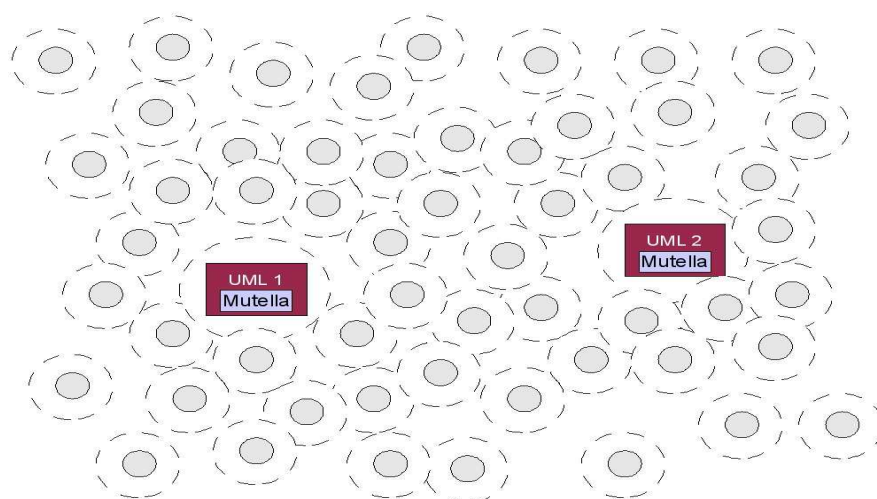


Figura 20: Ilustração de uma rede adhoc e os terminais UML

os vários nós se comunicando e os terminais UML trocando informação entre si dentro de uma rede *adhoc*, utilizando o algoritmo AODV como protocolo de roteamento na camada MAC e um protocolo P2P de inundação para localização de usuários na camada de aplicação.

Dentre os parâmetros selecionados como medida de desempenho, escolheu-se dois desses para fazer medições mais profundas. O primeiro é o atraso de ida e volta (RTT), que no contexto de uma aplicação VoIP foi batizado como **Tempo de Latência de Sessão**, para um nó solicitar informações sobre um outro nó e receber a resposta. Essa informação é importante, pois é uma medida de quanto tempo um usuário poderia vir a encontrar outro usuário numa rede VoIP descentralizada. Um valor muito elevado poderia resultar em insatisfação do usuário antes mesmo de conseguir realizar a chamada.

O outro parâmetro escolhido foi a vazão na rede durante uma busca. Esta valor indica o consumo do canal durante uma busca dentro da rede *adhoc*. Se muitos pacotes forem utilizados e cada nó na rede solicitar ao mesmo tempo informações, isso pode acarretar gargalos na rede *adhoc*.

Através deste cenários e suas variações foi possível avaliar qual o desempenho de uma aplicação VoIP em um ambiente P2P, em relação a atraso e, por conseguinte, qualidade de serviço da localização.

5.4 Execução dos experimentos

Após a configuração dos scripts e programas a serem executados, iniciamos os experimentos fazendo o Mutella procurar um determinado nó na rede configurada. Cada execução demorou 300 s para ser concluída e coletamos as informações referentes ao TLS (Tempo de Latência de Sessão) e vazão utilizando o Ethereal(62), que é um programa que captura pacotes em uma determinada interface na rede. Coletamos as amostras em arquivos de log para posterior análise. Essas amostras consistiam em coletas dos pacotes do protocolo gnutella e possuíam o tempo de início e término de uma busca por um nó na rede.

Para cada cenário avaliado (10, 25, 50 nós), coletou-se um número variável de amostras (apresentados na seção de resultados) e calculou-se o tempo de busca por um nó na rede em duas diferentes situações: com mobilidade e sem mobilidade dos nós. Isso é feito calculando-se a diferença entre o momento em que uma busca foi realizada e o momento em que um resultado positivo foi recebido de uma rede P2P.

Para análise da vazão, utilizamos tanto a ferramenta Iptraf (61) quanto uma opção do Ethereal para cálculo e análise da vazão. Essas ferramentas armazenam a vazão durante a execução do experimentos e depois fazem cálculos que incluem o tamanho das mensagens e a taxa de geração de pacotes.

Um dificuldade encontrada foi a de coletar informações relacionadas com o Mutella na camada de aplicação e configurá-lo para sempre procurar um nó na rede. Foi necessário observar a maneira como recebia parâmetros de configuração do sistema. Isso foi feito configurando parâmetros num arquivo de configuração `.mutella/termrc`.

Um outro problema extra que também foi solucionado foi a configuração do tipo de arquivos que o mutella deveria procurar na rede. Isso porque a maioria dos softwares P2P está otimizada para procurar arquivos de tamanho maior que 3000 kbytes, como arquivos de música e vídeo. Como a busca por um nome na rede VoIP não deve retornar um arquivo maior que 100 kb, tivemos que otimizar o Mutella para que também procurasse por arquivos ou nomes com esse tamanho.

5.5 Considerações sobre os resultados

Os resultados obtidos confirmaram, em parte, o comportamento esperado pelas hipóteses apresentadas no capítulo anterior com relação ao aumento dos tempos de latência e a

quantidade de nós presentes na rede, no entanto, os valores obtidos nas medições foram inferiores aos esperados e se mostraram despresíveis no que se referem ao tempo máximo esperado por um usuário ao realizar uma chamada VoIP. O número de nós e a mobilidade mostraram-se fatores determinantes no que se refere ao **Tempo de Latência de Sessão**, na vazão no meio físico da rede e na taxa de pacotes perdidos.

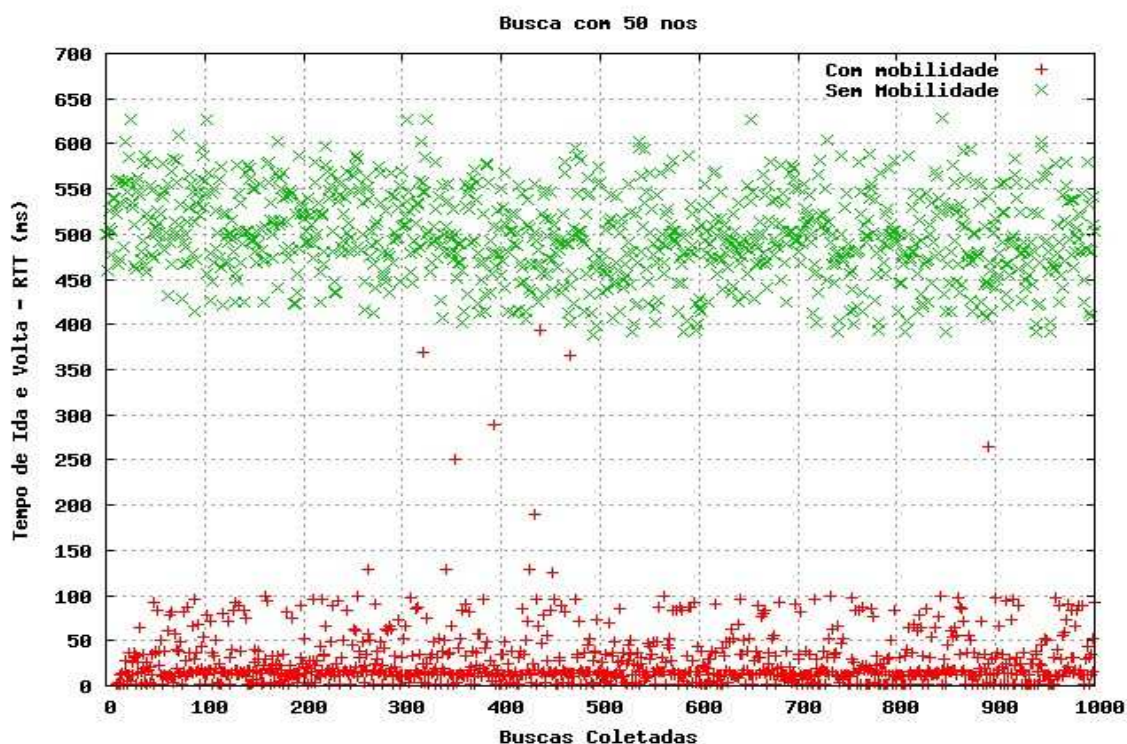


Figura 21: Latência de Sessão para 50 nós

A Figura 21 mostra a latência para um grupo de 50 nós inicialmente sem mobilidade e posteriormente aplicada mobilidade aos nós. Na primeira situação observou-se que a latência estaria na ordem de $400 < T < 600$ milissegundos (T =Tempo de Latência), o que ainda está dentro do tempo aceitável para uma aplicação VoIP. Na segunda situação esperava-se uma latência ainda maior devido a mobilidade dos nós, no entanto, o valor obtido foi menor em todas as repetições realizadas ficando na faixa de $0 < T < 100$.

A medida que diminui-se a quantidade de nós a faixa de valores dos tempos de latência não utilizando mobilidade continuou a seguir a tendência mostrada pelo primeiro experimento utilizando 50 nós. Esta faixa foi descendo gradualmente, mas seus valores mostraram-se sempre maiores que os observados nos experimentos com nós com mobilidade. Nas Figuras 22 e 23 pode ver claramente que a faixa de tempos de latência dos nós sem mobilidade vem gradualmente decaindo sobre a faixa dos nós com mobilidade que vai tornando-se-se mais compacta.

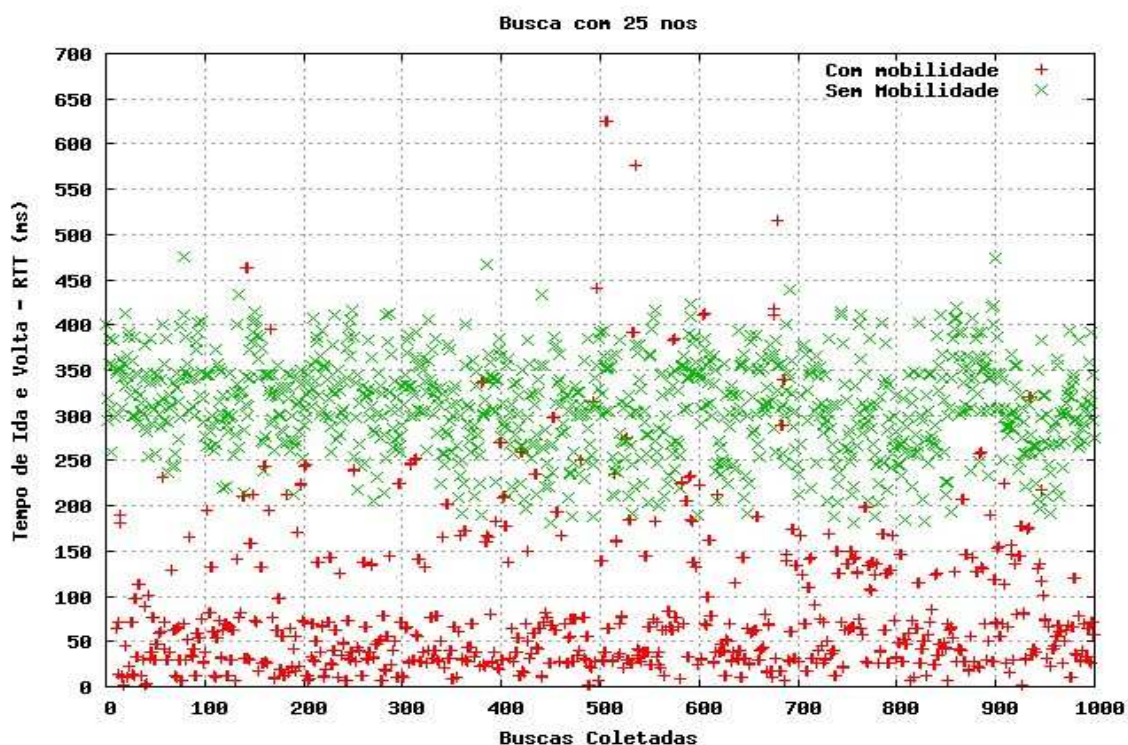


Figura 22: Latência para 25 nós

Este comportamento não era esperado, visto que pensou-se a princípio que a mobilidade causaria um impacto negativo sobre os algoritmos de roteamento tanto da camada de rede quando da camada de aplicação (P2P). Várias repetições dos mesmos experimentos mostraram comportamento idênticos, com pequenas variações nos valores absolutos dos gráficos. Esperava-se que a estabilidade nas rotas levassem a uma melhor desempenho nos tempo de latência, o que não foi observado.

Outro comportamento interessante mostrado pelos gráficos foi a dispersão nos valores dos tempos de latência nos experimentos com mobilidade, a medida que a quantidade de nós é aumentada os tempos de latência tentem a se agruparem em uma faixa cada vez menor de valores e a diferença dessas dispersão entre 10 e 50 nós é claramente confirmada nos gráficos apresentados. Esse comportamento não é observado nos experimentos sem mobilidade, onde a faixa de valores decai de maneira compacta com uma média de diferença entre seus máximos e mínimos de 150 milissegundos.

Num primeiro momento não formou-se nenhuma teoria sobre o comportamento não esperado mostrado pelos experimentos, o que levou a uma investigação tomada por outra métrica, a taxa de perda de pacotes. Neste gráfico, mostrado na Figura 25 observou-se que as perdas verificadas nos experimentos sem mobilidade eram consideravelmente menores do que os com mobilidade. Este fato levou a suposição de que muito mais pacotes eram

enviados quando os nós estavam em movimento a medida que os nós consideram que seus pacotes foram perdidos, isto pode ser comprovado observando o gráfico que mostra a vazão na rede mostrado na Figura 24. Com mais pacotes sendo enviados, mais deles chegam em menos tempo mesmo que para isso a vazão da rede atinja valores maiores, o que pode comprometer a qualidade na transmissão de voz.

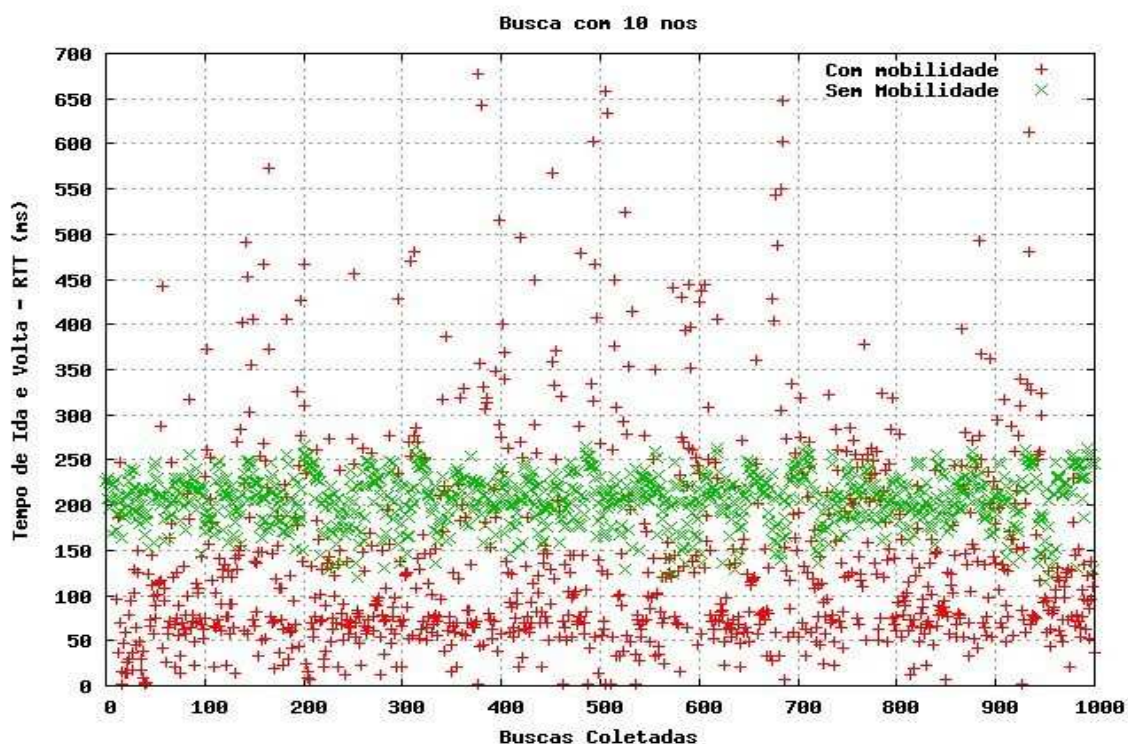


Figura 23: Latência para 10 nós

A vazão também mostrou-se muito sensível à quantidade de nós pertencentes a rede e as mensagens enviadas pelo protocolo de busca P2P. Os valores observados para a vazão são inversamente proporcionais aos observados para a latência, isto é, quanto menor a latência maior a vazão na rede. A Figura 24 mostra a combinação de dois gráficos relativos a vazão onde duas faixas se destacam, a faixa superior representa a vazão para uma rede de 10 nós, enquanto que a inferior a vazão para redes de 50 nós. A razão desse comportamento é facilmente explicada pelo fato de que quanto mais nós, mais mensagens são enviadas como já mencionado anteriormente.

Ao observar-se que a perda de pacotes no ambiente vai em sentido contrário ao tempo de latência, como pode ser constatado no gráfico mostrado pela Figura 25, e que seus valores são maiores em redes com mobilidade mostra que pode-se perder a qualidade na transmissão de voz pela aplicação VoIP onde cada pacote perdido representa um pedaço da mensagem sendo transmitida e que retransmissões não são uma opção preferível neste

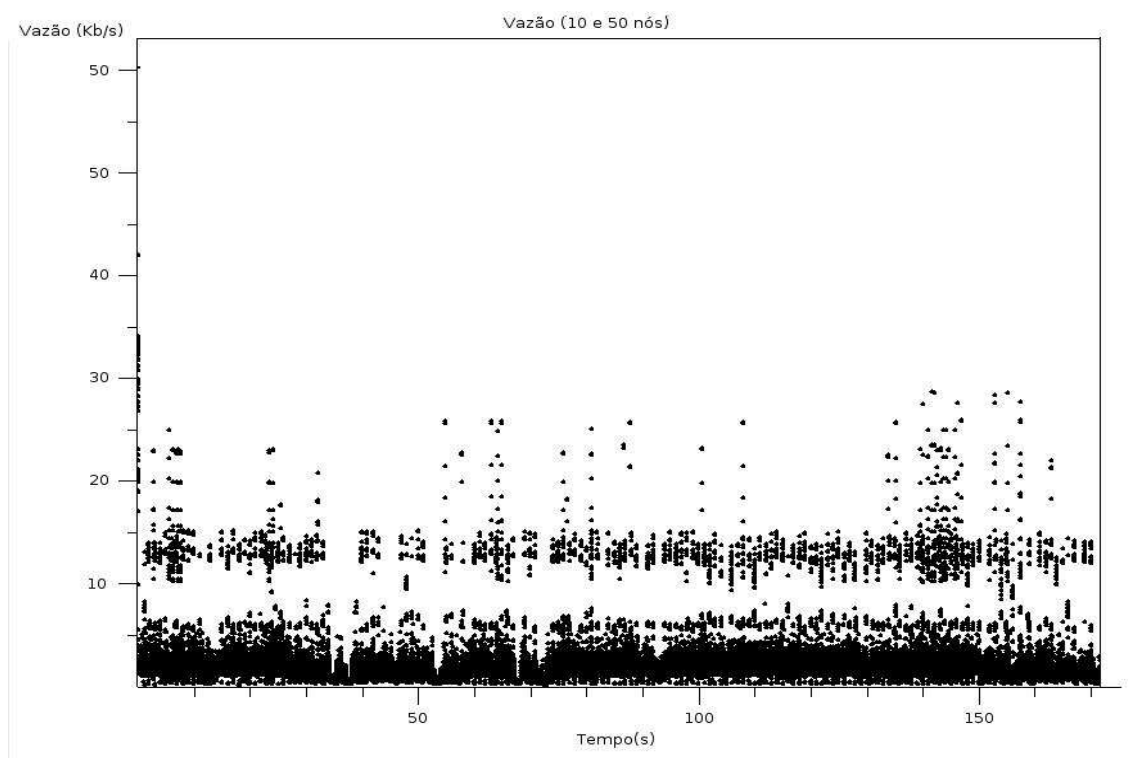


Figura 24: Vazão para 10 e 50 nós

tipo de aplicação.

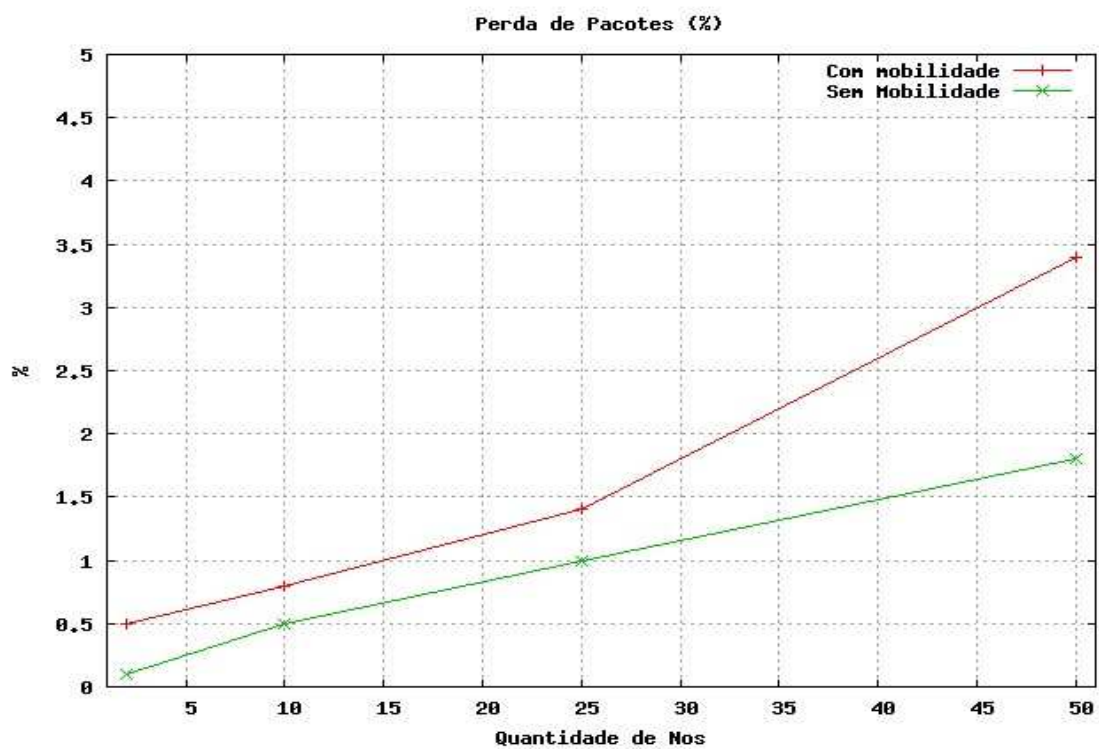


Figura 25: Taxa de Perda para 2 a 50 nós

Embora tenha-se observado um comportamento adverso do esperado inicialmente pe-

las hipóteses apresentadas no capítulo anterior, os experimentos realizados desanuviaram a visão dos fatores que realmente influenciam na performance de algoritmos de localização de uma aplicação VoIP-P2P em redes *adhoc*.

6 *Considerações Finais*

Durante este trabalho procurou-se o caminho da investigação científica aplicada, que por muitas vezes, leva a tantas mais perguntas do que respostas. Nos capítulos anteriores procurou-se oferecer uma visão esclarecedora do que estava sendo investigado, o que acreditava-se poder encontrar e os meios pelos quais a busca seria feita. Este capítulo é dedicado não mais a investigação como um fim, mas a reflexão dos seus resultados, os benefícios que trouxeram e principalmente os novos caminhos abertos por eles.

6.1 Contribuição

Neste estudo fomos capazes de identificar alguns fatores que influenciam direta e indiretamente a performance de aplicações P2P-VoIP através de experimentos realizados em seus algoritmos de busca, o que nos permitiu estabelecer parâmetros pelos quais o desenvolvimento de aplicações desse gênero podem ser direcionados. Por vezes o destino da viagem se torna insignificante perante o percurso que se seguiu para alcançá-lo. Durante a realização deste estudo foram investigadas e aprimoradas idéias e conceitos que poderão ser utilizados por outros na sua continuação ou mesmo para outros objetivos. Entre as contribuições mais significantes destacam-se a modificação do software **mutella** para que se comportasse como uma aplicação VoIP-SIP, a criação do ambiente de simulação para aplicações P2P-VoIP e a identificação de fatores que influenciam na performance de uma aplicação P2P-VoIP e que devem ser levados em consideração quando do desenvolvimento de aplicações semelhantes.

6.2 Considerações sobre a estratégia

Algumas decisões precisaram ser tomadas durante o percurso das investigações, muitas delas levaram ao abandono de algumas alternativas que, em um primeiro momento, foram consideradas viáveis mas que mostraram-se desanimadoras por diversos fatores descritos

nesta sessão.

Como já mencionado, redes P2P atuam na Camada de Aplicação do modelo OSI/ISO enquanto os simuladores de meio físico atuam até a Camada de Rede do modelo OSI/ISO. Esta característica acabou por guiar muitas das escolhas durante a fase de implementação e criação do ambiente de experimentação que se mostrou mais complexo do que previamente esperado pois precisou-se de uma solução que combinasse e monitorasse a operacionalização em duas camadas diferentes do Modelo ISO/OSI.

Espera-Esperava-se poder realizar experimentos utilizando os dois tipos disponíveis de busca P2P, DHT e inundação, e assim além de verificar o comportamento da aplicação poderia-se comparar qual das buscas seria mais eficiente ou pelo menos quais suas vulnerabilidades no ambiente *ad hoc*. Infelizmente isso não foi possível já que não se foi capaz de encontrar nenhuma aplicação compatível com o cenário escolhido utilizando-se o UML e embora muito esforço tenha sido dedicado para torna-la compatível a proximidade do prazo para a conclusão levou ao abandono desse caminho. Pensou-se como alternativa a constatação real dos experimentos por meio da utilização de notebooks para executar as aplicações, mas esta opção mostrou-se economicamente impraticável. Acredita-se que com a combinação dos resultados obtidos com a investigação desses dois métodos de busca poderia fornecer maneiras de combinar as vantagens de um e outro para alcançar uma melhor performance no ambiente *ad hoc*.

6.3 Trabalhos futuros

O esforço de prover uma aplicação VoIP totalmente descentralizada e aberta, ainda é tímido e poucos se aventuram nessa cruzada. No entanto, grandes avanços tem sido alcançados por esses que acreditam no potencial desse tipo de aplicação em um futuro muito próximo onde a mobilidade estará tão intrincada com nosso estilo de vida quanto é um aparelho de TV.

Neste trabalho procurou-se identificar os fatores que direta ou indiretamente contribuem para a eficiência ou deficiência de uma aplicação VoIP descentralizada no ambiente *ad hoc*. Neste aspecto, apenas a primeira fase de uma chamada VoIP foi abordada, a **localização do destino** da mesma e em nenhum momento investigamos o comportamento após esta etapa, que seria a transmissão da voz propriamente dita, e mais, o provimento de serviços já consolidados nas redes fixas como **chamada em espera, detecção de presença, desvio de chamadas** e mesmo **controle de tarifação/bilhetagem**.

A segurança nessas aplicações também é um importante aspecto a ser investigado já que tanto VoIP quanto P2P ainda enfrentam dificuldades para consolidação de suas políticas de segurança mesmo em redes fixas com um maior arsenal contra acessos maliciosos. O fato de ser totalmente descentralizada dificulta a adoção de estratégias convencionais de segurança como validação e controle dos usuários e suas ações enquanto estão na rede o que pode abrir diversos buracos na segurança e integridade dos nós participantes.

Grande parte dos obstáculos enfrentados na realização deste trabalho originaram-se do fato de não se dispor de um simulador capaz de suportar camadas superiores a Camada de Rede do modelo OSI/ISO, tornando difícil o estudo do comportamento da rede durante a utilização de uma aplicação específica. Um estudo valoroso no sentido de possibilitar um simulador que além da Camada de Rede, também possibilitasse o estudo de camadas superiores do Modelo OSI/ISO seria de grande ajuda para a coleta de resultados no futuro.

Referências

- 1 GIGANET. Estatísticas sobre a internet. [Http://www.gnet-rj.com.br/noticias/inet1.asp](http://www.gnet-rj.com.br/noticias/inet1.asp).
- 2 GUIALOG. Estatísticas sobre a internet. [Http://www.guialog.com.br/estatistica-autom.htm](http://www.guialog.com.br/estatistica-autom.htm).
- 3 INFORMÁTICA, S. *Seminário: Impacto na redução de custos com telefonia usando VoIP*. [Http://sisnema.com.br/Materias/idmat016557.htm](http://sisnema.com.br/Materias/idmat016557.htm).
- 4 SKYPE. [Http://www.skype.com](http://www.skype.com).
- 5 GOOGLE. *Google Talk*. [Http://www.google.com/talk/](http://www.google.com/talk/).
- 6 UOL, U. O.
- 7 AVAYA. *Avaya-One X*. [Http://www.avaya.com](http://www.avaya.com).
- 8 ITU-T. Itu-t recommendation h.323. packet-based multimedia communications systems. 2000.
- 9 GROUP, I. S. W. Session initiation protocol. [Http://www.faqs.org/rfcs/rfc3261.html](http://www.faqs.org/rfcs/rfc3261.html).
- 10 NACIONAL, T. Níveis de satisfação dos usuários seafi. [Http://www.tesouro.fazenda.gov.br](http://www.tesouro.fazenda.gov.br).
- 11 WEB, E. for the. Loading time. [Http://www.towson.edu/lieb/editing/loading.html](http://www.towson.edu/lieb/editing/loading.html).
- 12 SINGH, K.; SCHULZRINNE, H. Peer-to-peer internet telephony using sip. *Department of Computer Science, Columbia University*, 2004.
- 13 BASET, S. A.; SCHULZRINNE, H. An analisys of the skype peer-to-peer internet telephony protocol. *Departament of Computer Science, Columbia University*, 2004.
- 14 BRYAN, D. A.; LOWEKAMP, B. B.; JENNINGS, C. Sosimple: A servless, standards-based, p2p sip communication system. 2005.
- 15 CARVALHO, L. S. G. de. *Implementação do Modelo E para Avaliação Objetiva da Qualidade de Fala em Redes de Comunicação VoIP*, school = *Universidade Federal do Amazonas*, year = 2004. Dissertação (Mestrado).
- 16 BRYAN, D. A.; LOWEKAMP, B. B. Sosimple: A sip/simple based p2p voip and im system. 2003.
- 17 GROUP, N. W. Hyper text transport protocol 1.1. [Http://www.faqs.org/rfcs/rfc2616.html](http://www.faqs.org/rfcs/rfc2616.html).

- 18 SINNREICH, H.; JOHNSTON, B. A. *Internet Communications Using SIP*. [S.l.: s.n.], 2001.
- 19 JOHNSTON, B. A. *SIP: Understanding the Session Initiation Protocol - 2nd ed.* [S.l.: s.n.], 2004.
- 20 SCHOLLMEIER, R. A definition of peer-to-peer networking towards a delimitation against classical client-server concepts. 2002.
- 21 NAPSTER Home Page. [Http://www.napster.com](http://www.napster.com).
- 22 CLIP2. The gnutella protocol specification v0.4 (document revision 1.2). [Http://www.clip2.com/GnutellaProtocol04.pdf](http://www.clip2.com/GnutellaProtocol04.pdf).
- 23 PROJECT, P. A. Ensuring dependability of p2p applications at architectural level. [Http://www.atc.gr/p2parchitect/results/0101F05P2PSurvey.pdf](http://www.atc.gr/p2parchitect/results/0101F05P2PSurvey.pdf).
- 24 RATNASAMY, S. et al. A scalable content-addressable network. 2001.
- 25 ZHAO, B. Y. et al. Tapestry: A resilient global-scale overlay for service deployment. 1995.
- 26 ROWSTRON, A.; DRUSCHEL, P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. 2001.
- 27 STOICA, I. et al. Chord: A scalable peer-to-peer lookup service for internet applications.
- 28 MAYMOUNKOV, P.; MAZIERES, D. Kademlia: A peer-to-peer information system based on the xor metric. 1998.
- 29 MALKHI, D.; NAOR, M.; RATAJACZAK, D. Viceroy: a scalable and dynamic emulation of the butterfly. 2002.
- 30 CLARKE, I. et al. Freenet: A distributed anonymous information storage and retrieval system. 1999. [Http://freenetproject.org/freenet.pdf](http://freenetproject.org/freenet.pdf).
- 31 COMPANY, F. peer-to-peer technology. [Http://www.fasttrack.nu](http://www.fasttrack.nu).
- 32 BITTORRENT. [Http://bitconjurer.org/bittorrent](http://bitconjurer.org/bittorrent).
- 33 OVERNET/EDONKEY2000. [Http://www.overnet.com](http://www.overnet.com).
- 34 TOWSLEY, D. Peer-to-peer networking. *Tutorial SBRC2003*, 2003. [Http://www.cin.ufpe.br/gprt/gtp2p/tutoriais/p2p03-tutorialtowsley.pdf](http://www.cin.ufpe.br/gprt/gtp2p/tutoriais/p2p03-tutorialtowsley.pdf).
- 35 [Http://rfc-gnutella.sourceforge.net/developer/testing/index.html](http://rfc-gnutella.sourceforge.net/developer/testing/index.html).
- 36 ROCHA, R. R. *Redes Peer-to-Peer para compartilhamento de arquivos na Internet*. [S.l.], 2003.
- 37 SINGH, K.; SCHULZRINNE, H. Peer-to-peer internet telephony using sip. 2003.
- 38 [Http://www.kazaa.com](http://www.kazaa.com).

- 39 BASET, S. A.; SCHULZRINNE, H. An analysis of the skype peer-to-peer internet telephony protocol. 2004.
- 40 HOUCK, J. B. Big blast in mobile computing growth. *Mobile Tech Today*, 2005. [Http://www.wirelessnewsfactor.com/perl/story/11045.html](http://www.wirelessnewsfactor.com/perl/story/11045.html).
- 41 SPANGLER, T. Baseline survey: The mobile motive. *BaseLine: The project Management Center*, 2006. [Http://www.baselinemag.com/article2/0,1540,1956565,00.asp](http://www.baselinemag.com/article2/0,1540,1956565,00.asp).
- 42 CORSON, S.; BATSELL, S.; MACKER, J. Architectural considerations for mobile mesh networking. Internet draft RPC, version 2, May 1996.
- 43 CORSON, S.; EPHREMIDES, A. A distributing routing algorithm for mobile wireless networks. *Journal on Wireless Networks*, 1(1):61:-81, 1995.
- 44 DAS, B.; SIVAKUMAR, E.; BHARGAVAN, V. Routing in ad-hoc networks using a spine. *IEEE International Conference on Computer Communications and Networks*, 1997.
- 45 DUBE, R. et al. Signal stability based adaptive routing (ssa) for ad-hoc mobile networks. *IEEE Personal Communications*, 1996.
- 46 HAAS, Z.; PEARLMAN, M. The zone routing protocol (zrp) for ad-hoc networks. 1997. Internet Draft.
- 47 JIANG, M.; LI, J.; TAY, Y. Cluster based routing protocol (cbrp) functional specification. 1998. Internet Draft.
- 48 JOHNSON, D.; MALTZ, D.; BROCH, J. The dynamic source routing protocol for mobile ad hoc networks. 1998. Internet Draft.
- 49 ROYER, E. M.; TOH, C.-K. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications*, 1999.
- 50 802.11, I. S. Wireless lan medium access control (mac) and physical layer (phy) specifications. 1999.
- 51 VELOSO, P. B.; RUBINSTEIN, M. G.; DUARTE, O. C. Transmissão de voz em redes ad hoc. 2003.
- 52 NUNES, B. A. A. *Fatores Impactantes na performance de redes Ad hoc sem fio*. Dissertação (Mestrado), 2003.
- 53 BELDING-ROYER, E. M.; DAS, S. R. Ad hoc on-demand distance vector routing. 2002. Internet Draft, draft-ietf-manet-aodv.txt.
- 54 IETF, I. E. T. F. Sip for instant messaging and presence leveraging extensions. [Http://www.ietf.org/html.charters/simple-charter.html](http://www.ietf.org/html.charters/simple-charter.html).
- 55 [Https://lists.cs.columbia.edu/cucslists/listinfo/p2p-sip](https://lists.cs.columbia.edu/cucslists/listinfo/p2p-sip).
- 56 GUHA, S.; DASWANI, N.; JAIN, R. An experimental study of the skype peer-to-peer voip system. 2006.

-
- 57 (TM), P. *PeerioBiz*. [Http://www.peerio.com/index.php](http://www.peerio.com/index.php).
- 58 [Http://mutella.sourceforge.net/](http://mutella.sourceforge.net/).
- 59 ABINADER, F. Avaliação de desempenho de algoritmos de compressão de cabeçalho cooperativos para aplicação voip em redes sem fio. Tese de Mestrado, UFAM. 2006.
- 60 NETWORK Simulator. [Http://www.isi.edu/nsnam/ns](http://www.isi.edu/nsnam/ns).
- 61 [Http://iptraf.seul.org/](http://iptraf.seul.org/).
- 62 [Http://www.ethereal.com/](http://www.ethereal.com/).
- 63 OMNET Simulator. [Http://www.omnetpp.org](http://www.omnetpp.org).
- 64 OVG - University of Magdeburg. [Http://www-ivs.cs.uni-magdeburg.de/EuK/forschung/projekte/nse/index.shtml](http://www-ivs.cs.uni-magdeburg.de/EuK/forschung/projekte/nse/index.shtml).
- 65 USER-MODE-LINUX. [Http://www.user-mode-linux.org](http://www.user-mode-linux.org).
- 66 TUNCTL. [Http://www.user-mode-linux.org/cvs/tools/tunctl/](http://www.user-mode-linux.org/cvs/tools/tunctl/).