UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

HUGO SCHROTER LAZZARI

Geração de rotas otimizadas e caminhos alternativos considerando métricas de qualidade de conexão

Trabalho de Graduação.

Prof. Dr. Valter Roesler
Orientador

Porto Alegre, dezembro de 2013

“Don’t only practice your art,
but force your way into its secrets,
for it and knowledge can
raise men to the divine. “
— LUDWIG VAN BEETHOVEN

AGRADECIMENTOS

À minha família por todo o apoio, carinho e dedicação em todas as etapas da minha vida. Muito obrigado!

Aos meus amigos por todos os momentos que passamos juntos.

Ao meu orientador Valter Roesler pelo apoio e interesse durante todo o desenvolvimento deste trabalho.

À UFRGS, ao Instituto de Informática e a todos os outros professores pelo ensino extremamente qualificado.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ALM	*Application Layer Multicast*
ICMP	*Internet Control Message Protocol*
IP	*Internet Protocol*
ISP	*Internet Service Provider*
IVS	*INRIA Videoconferencing System*
JSON	*JavaScript Object Notation*
NAT	*Network Address Translation*
P2P	*Peer-to-Peer*
PDV	*Packet Delay Variation*
QOS	*Quality of Service*
RTT	*Round-trip Time*
SNMP	*Simple Network Management Protocol*
TCP	*Transmission Control Protocol*
TTL	*Time-to-Live*
UDP	*User Datagram Protocol*
VOIP	*Voice over Internet Protocol*

LISTA DE FIGURAS

RESUMO

O objetivo deste trabalho é a criação de uma ferramenta que, a partir de um conjunto de servidores distribuídos, gere rotas em nível de aplicação priorizando uma das seguintes métricas: latência, jitter ou vazão. Além disso, a ferramenta gera caminhos alternativos mais disjuntos possíveis em nível de rede visando aumentar a disponibilidade do acesso aos servidores. Dessa forma, caso um caminho falhe, a aplicação pode utilizar outro como alternativa, minimizando a probabilidade de ambos compartilharem o mesmo caminho.

A ferramenta criada visa ser utilizada como apoio à decisão para escolha dos servidores mais adequados para integrar uma videoconferência distribuída, visando resolver dois problemas em videoconferências centralizadas: baixa disponibilidade e baixa escalabilidade.

Para validar a ferramenta, utilizou-se uma rede real distribuída no Brasil, composta por 8 servidores localizados em diversas regiões do País. Efetuaram-se diversas medições de jitter, latência, vazão e análise de caminhos alternativos.

Os resultados foram analisados através da criação de cenários-exemplo de como se comportaria um sistema de videoconferência distribuída utilizando tal rede. Um fato observado é que nem sempre a conexão direta entre dois servidores apresenta a menor latência ou a maior vazão. Cuidados devem ser tomados em relação ao tráfego gerado pela própria ferramenta no sistema.

Comprovou-se que uma ferramenta desse tipo pode ser útil para apoio à decisão do melhor caminho visando aumento de escalabilidade e disponibilidade em sistemas de videoconferência distribuídos.

Palavras-chave: Monitoramento, Desempenho, Rotas, Otimização, Rede de Computadores.

Creating optimized routes and alternative paths considering quality metrics connection

ABSTRACT

The objective of this work is the creation of a tool, that starting from a set of distributed servers, create application-level routes prioritizing one of the following metrics: latency, jitter and throughput. Furthermore, the tool can create alternative paths that are disjoint at the network level, in order to increase the availability of access to servers. Thus, if a path fails, the application can use the other path as alternative, minimizing the likelihood of both sharing the same path.

The tool aims to be used as a decision support for selecting the most appropriate servers to integrate a distributed videoconferencing, aiming to solve two problems in centralized conferencing: low availability and low scalability.

To validate the tool, we used a real distributed network in Brazil, consisting of 8 servers located in various regions of the country. We carried out several measurements of jitter, latency, throughput and analysis of alternatives.

The results were analyzed by setting up scenarios, that behave as a video conferencing system, using a distributed network. A fact, that was observed, is that not always a direct connection between two servers features the lowest latency or the highest throughput. When using the tool, care should be taken in relation to the traffic generated by the tool itself in the system.

It was proved that such a tool can be useful for decision support when the aim is to increase scalability and availability in a distributed conferencing system.

Keywords: Performance, Network, Metric, Routing.

1 INTRODUÇÃO

Sistemas de videoconferência consistem na transmissão em tempo real de voz e imagem entre um conjunto de usuários. Uma videoconferência permite que pessoas, independentemente de sua localização geográfica, possam estabelecer um processo de comunicação, com uma particularidade que a difere das audioconferências e das ligações telefônicas, pois agrega a imagem ao som tornando o processo comunicativo mais eficiente. Um exemplo de videoconferência pode ser visualizado na Figura 1.1, onde é mostrado o sistema do Mconf, que permite conferências centralizadas entre um conjunto de clientes.

Figura 1.1: Exemplo de videoconferência (MCONF, 2013)

Uma videoconferência possui, segundo (TANENBAUM; WETHERALL, 2011), as seguintes características:

- Alta sensibilidade à latência, ao *jitter* e à vazão.
- Baixa sensibilidade à perda de pacotes.

Além disso, ela pode adquirir várias configurações bem características entre as quais:

- (UM PARA UM), como o Facetime (APPLE, 2011), onde um usuário interage diretamente com outro usuário através de áudio e vídeo;
- (UM PARA N), como o Twitch.TV (TWITCH, 2011), onde um usuário envia áudio e vídeo para o servidor e este o reenvia para um grupo de pessoas, sendo que os *n*

usuários não enviam vídeo, mas podem interagir de outras formas como o envio de mensagens de texto.

- (N PARA N), como o Mconf (ROESLER et al., 2013) e (ROESLER et al., 2012), onde um conjunto *n* de usuários pode interagir entre si, enviando áudio e vídeo sem restrições.

Segundo (LE; NGUYEN, 2013), existem duas correntes de arquitetura de videoconferência que são: Centralizada e Distribuída.

- Centralizada - visão clássica de cliente/servidor, onde todos os usuários interagem diretamente com o servidor e este fica responsável por todo o controle e a distribuição do conteúdo. Algumas vantagens dessa configuração são: facilidade de implementação e controle do sistema. Uma das desvantagens é o seu grande consumo de recursos, pois, geralmente, é necessária uma grande banda a fim de suportar uma quantidade razoável de usuários conforme será mostrado na seção 1.1, sendo, portanto, pouco escalável. O Mconf (ROESLER et al., 2013) é um exemplo de videoconferência centralizada.

- Distribuída - Comunicação direta entre os usuários, sendo que eles são clientes e servidores simultaneamente. Algumas das vantagens são a não necessidade de muitos recursos de processamento e de demanda quando comparado à arquitetura centralizada. Algumas desvantagens são a complexidade na sincronização e no controle dos usuários. O Skype (BASET; SCHULZRINNE, 2004) é um exemplo de videoconferência distribuída.

1.1 Motivação

O crescente avanço tecnológico na área de redes gerou um aumento na vazão da Internet. Esse fato levou a uma mudança de paradigma no comportamento dos usuários, que passaram a realizar videoconferências de maior qualidade. Consequentemente, pessoas dos mais variados lugares do mundo puderam se conectar. Com o aumento da capilaridade e do uso da rede alguns locais menos privilegiados evidenciaram problemas de qualidade na conexão. Dentre os problemas encontrados pode-se destacar a alta latência nesses ambientes e a baixa vazão. Evidenciaram-se, também, problemas na utilização de sistemas com um servidor centralizado, entre os quais se têm a baixa escalabilidade e a baixa disponibilidade. As próximas subseções descrevem com maiores detalhes tais problemas.

1.1.1 Escalabilidade

Segundo (COULOURIS et al., 2011), escalabilidade é a habilidade de um sistema conseguir funcionar razoavelmente bem com o aumento da carga ou do número de usuários. Ainda, segundo o autor, um sistema é escalável quando o custo para adicionar um usuário é proporcional ao custo dos recursos que devem ser adicionados.

No exemplo a seguir, será mostrado que videoconferências centralizadas têm problemas em lidar com o crescimento do número de usuários. A exibição de vídeo pelos usuários torna-se muito custosa para o servidor, já que a banda máxima disponível pode ser facilmente ultrapassada.

Para exemplificar esse problema se pode considerar o seguinte cenário, que será utilizado nos próximos exemplos:

1. A banda disponível para *upload* (envio de dados pelo servidor) é de 100 Mbit/s.
2. A banda disponível para *download* (recebimento de dados pelo servidor) é de 100 Mbit/s.
3. O custo para o envio ou recebimento de um fluxo de vídeo é de 250 kbit/s.
4. 6 clientes conectados na videoconferência.

A Figura 1.2 ilustra uma videoconferência com arquitetura centralizada utilizando o cenário descrito anteriormente como base. Na figura, C_x representa o cliente *x*, com (*x* = 1 até 6) e *y*V representa o número *y* de fluxos em um único sentido.

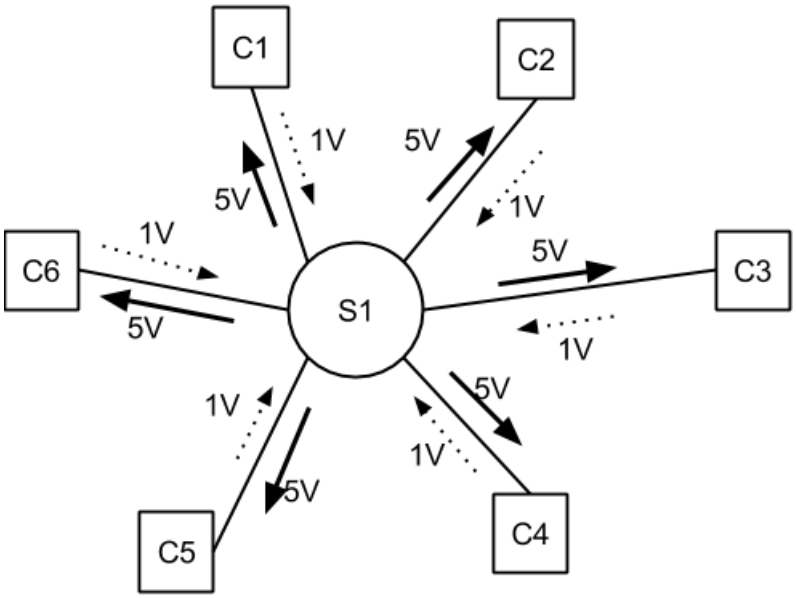


Figura 1.2: Exemplo de uma videoconferência com 1 servidor

Para realizar o cálculo da vazão de saída do servidor deve-se executar a seguinte fórmula:

$$V = F * T \tag{1.1}$$

Na fórmula (1.1), V refere-se à vazão utilizada no momento, F refere-se ao número de fluxos de vídeo sendo enviados pelo servidor e T refere-se à taxa para transferir um fluxo. No exemplo, F é igual a 30, pois têm trinta fluxos saindo do servidor na Figura 1.2 e T é igual a 250 kbit/s, portanto V é igual a 7,5 Mbit/s.

Analisando a Figura 1.2, pode-se deduzir uma fórmula geral para calcular o número de fluxos que partem do servidor central. Considerando que todos os usuários enviam vídeo em um servidor centralizado, a fórmula é:

$$F = C * (C - 1) \tag{1.2}$$

Na fórmula (1.2), C refere-se ao número de clientes no servidor e F ao número de fluxos.

Utilizando as fórmulas (1.1) e (1.2) é possível obter o número máximo de clientes suportados até ultrapassar a banda máxima disponível. Considerando a mesma configuração do exemplo anterior, onde a banda máxima de *download*/*upload* é de 100 Mbit/s e o custo de envio ou recebimento de cada fluxo é de 250 kbit/s, o número máximo de usuários será

de 20. Esse valor é obtido aplicando a fórmula (1.1), que retorna 400 fluxos. Com o número de fluxos pode-se aplicar a fórmula (1.2), cujo valor resultante é, aproximadamente, 20 usuários.

A fórmula geral para descobrir a vazão em um servidor centralizado é:

$$V = C * (C - 1) * T \tag{1.3}$$

Assim, pode-se concluir que suportar muitos usuários ou aumentar a taxa de transferência de vídeo de cada usuário se torna complexo devido à banda necessária no servidor. Uma solução para aumentar a escalabilidade é implementar um sistema distribuído de videoconferência, tentando assim minimizar o custo parcial em cada servidor.

Na Figura 1.3, considera-se outra videoconferência com arquitetura distribuída que possuem dois servidores, S_1 e S_2, baseado no cenário ilustrado anteriormente. Nesse exemplo é possível observar que a quantidade de fluxos saindo de cada servidor é menor que o mostrado na Figura 1.2. O número de fluxos saindo de S_1 ou de S_2 é igual a 18, Assim, aplicando a fórmula (1.1) e, considerando a taxa de transferência de um fluxo igual a 250 kbit/s, tem-se vazão de 4,5 Mbit/s por servidor. Consequentemente, a vantagem de se distribuir os usuários entre os servidores se torna evidente, pois diminui o custo com a banda em um servidor já que diminui o tráfego de saída e a quantidade de processamento no servidor.

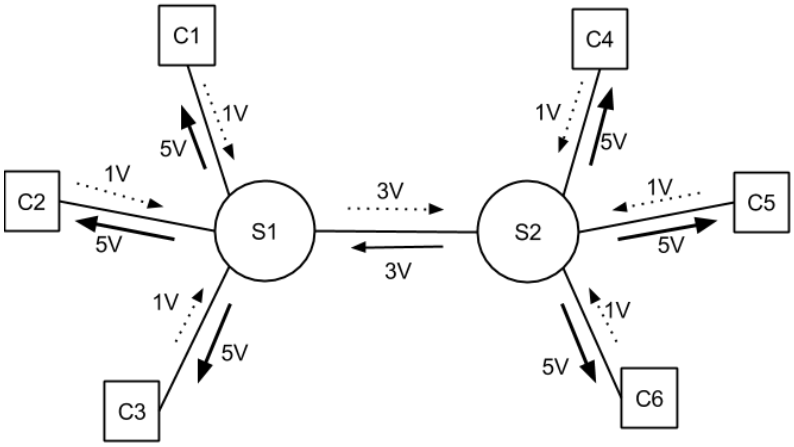


Figura 1.3: Exemplo de uma videoconferência com 2 servidores

A fórmula geral para se calcular o número de fluxos por servidor em uma arquitetura distribuída é:

$$F = \frac{C * (C - 1)}{S} + \frac{C * (S - 1)}{S} \tag{1.4}$$

onde C, S e F são, respectivamente, número de clientes, de servidores e de fluxos saindo do servidor. Na Figura 1.4, o eixo X representa o número de clientes, o eixo Y representa o número de fluxos. O gráfico apresenta, em uma curva pontilhada, um sistema distribuído com dois servidores e, em uma curva cheia, um sistema centralizado. O sistema distribuído levará vantagem não importando o número de clientes, pois o número de fluxos sempre será menor. Porém o número de clientes em cada servidor deve ser maior que o número de servidores no sistema.

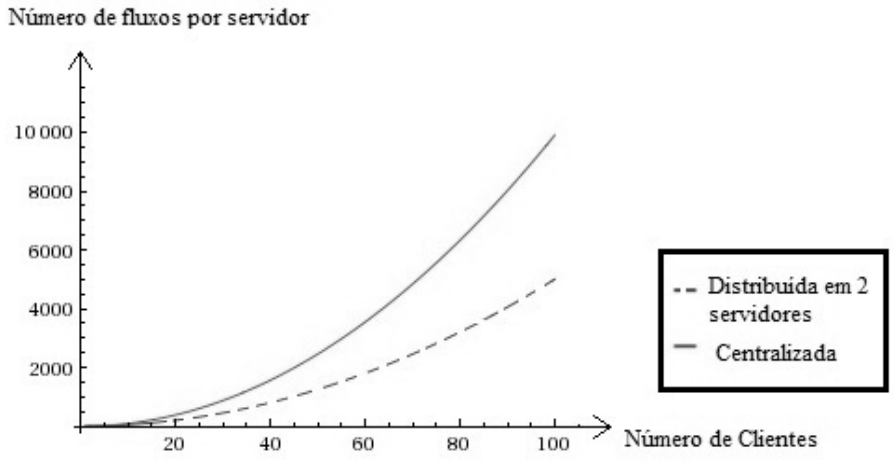


Figura 1.4: Comparação dos fluxos entre sistema centralizado e distribuído com 2 servidores

1.1.2 Disponibilidade

O problema da baixa disponibilidade está relacionado, principalmente, com as falhas que podem ocorrer em um servidor ou na rede. Uma técnica para aumentar a disponibilidade é replicar o conteúdo da videoconferência em mais de um servidor.

As falhas que geram o problema da disponibilidade podem ocorrer pelos mais diversos motivos. Saliente-se que o foco desse trabalho será nas falhas de rede que ocorrem entre os servidores e as falhas que podem ocorrer em um servidor.

Para exemplificar esse problema serão analisadas duas situações de videoconferências: a primeira centralizada e a segunda distribuída.

1.1.2.1 Baixa disponibilidade em videoconferência centralizada

O problema da baixa disponibilidade em uma videoconferência centralizada é mais complexo de ser tratado, pois só existe um servidor com a conferência. Caso esse servidor falhe todos os usuários ficarão sem acesso ao serviço ou caso ocorra uma falha na rede entre o servidor e algum dos clientes pode ocorrer uma interrupção parcial do serviço.

Na Figura 1.5, é mostrada uma videoconferência centralizada onde os servidores são representados por S, os roteadores por R e os clientes por C. Na Figura 1.5a pode-se observar o conjunto de servidores entre os clientes e o servidor. Já na Figura 1.5b ocorre uma falha no ponto sinalizado e, consequentemente, os clientes ficam sem acesso ao serviço.

1.1.2.2 Baixa disponibilidade em videoconferência distribuída

Segundo (COULOURIS et al., 2011), sistemas distribuídos fornecem um alto grau de disponibilidade para falhas de *hardware*. O problema da baixa disponibilidade em uma conferência distribuída, apesar de continuar sendo complexo, pode ser tratado, pois existe mais de um servidor na rede. Na Figura 1.6a é mostrado um conjunto de clientes conectados na rede. Nessa figura têm-se:

1. S_x, representa um servidor presente na rede e *x*, o identificador do servidor;

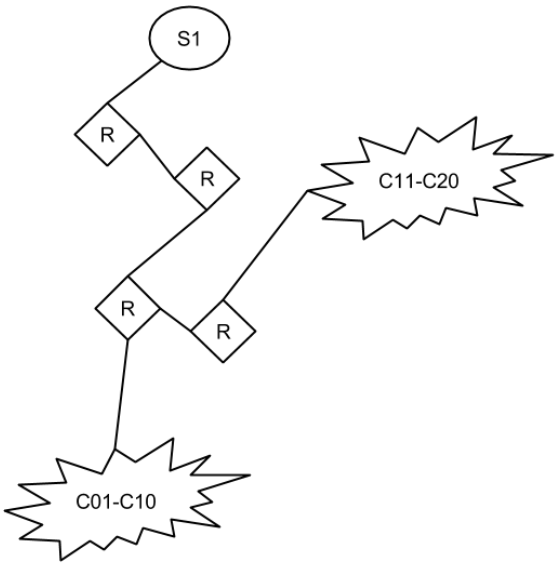
2. R representa um roteador que liga os servidores à rede;

3. C_1 representa um cliente que está enviando um vídeo com qualidade normal para S_1;

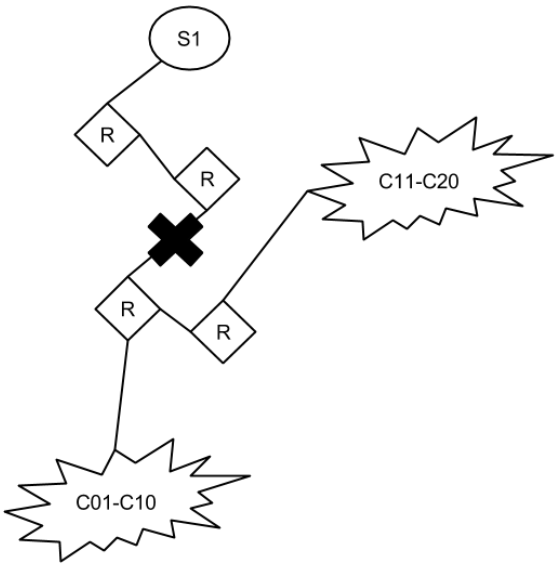
4. Brasil, USA e China representam um conjunto de clientes que estão localizados nesses países;

5. Os fluxos, com setas cheias, representam um sinal de vídeo com qualidade normal;

6. Os fluxos, com setas pontilhadas, representam um sinal de vídeo com baixa qualidade em relação à fonte original (podendo ter, por exemplo, menos quadros por segundo de vídeo, menores resoluções ou outras limitações), para uma economia de banda, pois eles servirão como redundância.



(a) Sem falha de rede



(b) Com falha de Rede

Figura 1.5: Videoconferência entre um conjunto de clientes e um servidor

Assim, nesse exemplo existem duas rotas de distribuição na videoconferência onde a rota principal é S_1->S_2 e S_1->S_3 e a rota alternativa é S_1->S_4->S_5.

Na rota principal é enviado um sinal de vídeo de qualidade normal para os clientes localizados no Brasil, China e USA. Porém, existe uma rota alternativa, que busca ser o mais disjunta possível da rota principal. Nessa rota alternativa, os clientes recebem um sinal de vídeo com menor qualidade, contudo caso ocorra uma falha na rede ou em algum servidor essa rota pode evitar a interrupção completa do serviço. A interrupção parcial do serviço deve-se à existência de um caminho alternativo até os clientes.

Na Figura 1.6b, pode-se observar, nos pontos sinalizados, que caso ocorra uma falha em algum dos pontos, os clientes continuarão a receber sinal na videoconferência. Isso se deve ao fato da rota alternativa evitar se sobrepor à rota original. Portanto, os usuários receberam um vídeo com menor qualidade, mas ainda sim terão um sinal, consequentemente, melhorando a experiência deles no sistema.

A rota alternativa busca resolver, também, o problema particionamento de rede. Segundo (COULOURIS et al., 2011), o particionamento de rede consiste em uma divisão da rede gerando uma falta de comunicação entre nodos na rede. Na figura 1.7 é possível observar um exemplo de particionamento da rede. Com a falha do servidor mostrado na figura, a rede fica dividida em duas redes. Assim, com uma rota alternativa que não se utiliza o servidor em questão, essa situação poderia ser tratada.

A motivação para este trabalho consiste na resposta de duas perguntas que são: a primeira, como escolher os melhores servidores que permitam a minimização do problema de escalabilidade e a segunda, como criar rotas alternativas entre dois servidores. Respondendo essas perguntas é possível minimizar os problemas apresentados anteriormente.

1.2 Objetivo

O objetivo é criar uma ferramenta que gere rotas otimizadas de distribuição de conteúdo entre um conjunto de servidores. Os caminhos que serão fornecidos pela ferramenta devem priorizar:

1. A menor latência, onde o caminho terá a menor latência entre o servidor de origem e o de destino.

2. O menor *jitter*, onde o caminho terá o menor *jitter* entre o servidor de origem e o de destino.

3. A maior vazão, onde o caminho terá a maior vazão entre o servidor de origem e o de destino.

A ferramenta também fornece caminhos alternativos que busquem ser o mais disjunto possível do caminho principal a fim de aumentar a disponibilidade baseando-se no conhecimento das rotas entre os servidores.

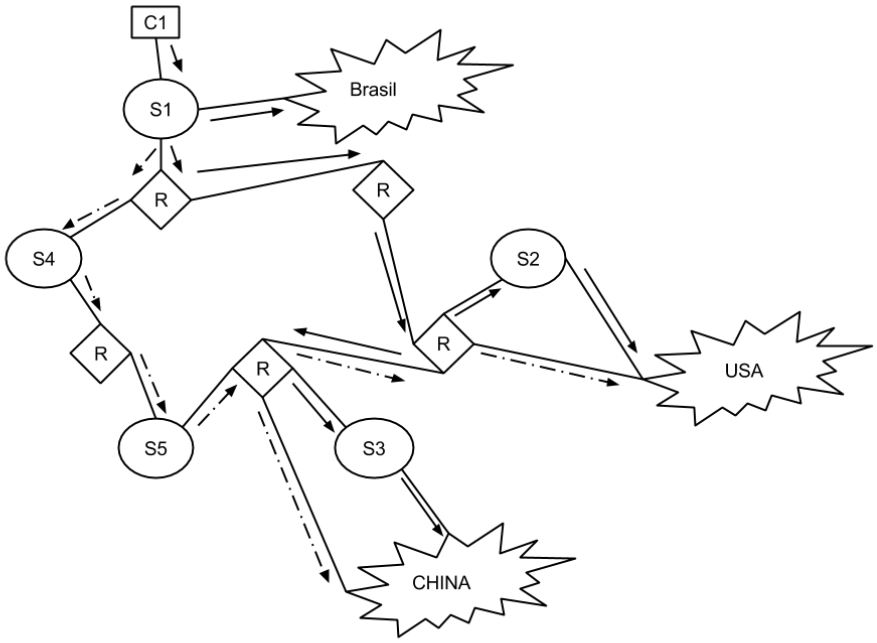
Para atingir esse objetivo, é necessário estudar algumas técnicas que auxiliem no descobrimento de características da rede. Ressalta-se que as métricas estudadas serão a latência, o *jitter*, a vazão e as rotas entre os servidores.

A justificativa para este trabalho é a proposta de técnicas que auxiliem a resolver os problemas apresentados nas subseções 1.1.1 e 1.1.2. Os problemas descritos são a baixa escalabilidade e a baixa disponibilidade. Para auxiliar na resolução do primeiro, a ferramenta fornece rotas entre dois servidores utilizando um conjunto de servidores intermediários. Essas rotas devem priorizar alguma das métricas citadas anteriormente.

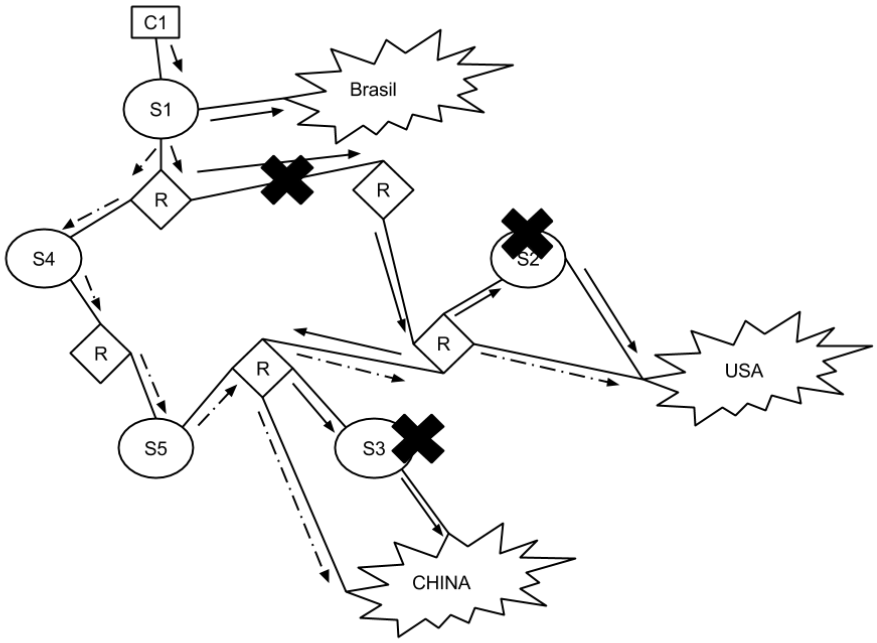
Já para a resolução do segundo, a ferramenta fornece rotas alternativas cuja finalidade é aumentar a disponibilidade, pois as mesmas buscam ser disjuntas das rotas originais.

1.3 Organização do texto

O trabalho encontra-se dividido em cinco capítulos. O capítulo 2 aborda os trabalhos anteriores relacionados ao tema principal além de alguns conceitos necessários para a compreensão do mesmo, particularmente, as seções 2.2 e 2.2.4, que tratam das métricas utilizadas neste trabalho. O capítulo 3 apresenta o desenvolvimento do estudo, abordando a criação de uma ferramenta para cumprir o objetivo proposto na seção 1.2. Também aborda a metodologia dos experimentos utilizando a ferramenta criada e a descrição do ambiente de experimentação. O capítulo 4 demonstra alguns resultados obtidos a partir da medição de qualidade entre um conjunto de servidores, além de otimizações relevantes que ilustram os benefícios deste estudo. O capítulo 5 refere-se às conclusões finais e propostas de trabalhos futuros.



(a) Sem falha de rede



(b) Com falha de Rede e de Servidores

Figura 1.6: Videoconferência distribuída entre servidores com um conjunto de clientes

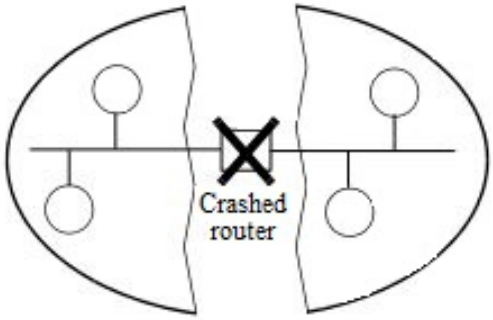


Figura 1.7: Particionamento de rede (COULOURIS et al., 2011)

2 CONCEITOS BÁSICOS

Neste capítulo serão apresentados trabalhos já desenvolvidos na área de videoconferência e que se relacionam com o assunto abordado neste estudo. Também serão apresentadas métricas utilizadas para a geração de rotas pela ferramenta proposta na seção 1.2. A revisão bibliográfica inclui os seguintes tópicos: sistemas de videoconferência, sistemas de videoconferência P2P, métricas e o algoritmo de Dijkstra, que é utilizado na geração das rotas pela ferramenta proposta.

2.1 Sistemas de Videoconferência

Um sistema de videoconferência consiste em um sistema de comunicação entre dois ou mais usuários. De acordo com (ITO, 1998) os requisitos para uma videoconferência são o envio de, no mínimo, som e imagens para um conjunto de usuários conectados na conferência. Constata-se que é muito comum encontrar sistemas de conferência em um arranjo tipo estrela, onde o nodo central fica responsável pela distribuição de todo o conteúdo para os clientes se conectando no nodo central, caracterizando, assim, uma conferência centralizada. Segundo (KESHAV, 1994), sistemas de videoconferência foram por muito tempo considerados *killer applications* (uma aplicação considerada desejável e indispensável), os quais conduziriam o futuro de redes multimídia. Porém, isso não foi observado na prática. Inicialmente, pensava-se que seria só construir um sistema que os usuários migrariam para o uso de videoconferência, mas o que ocorreu foi um relativo sucesso e seguido por um baixo interesse do usuário comum. Logo, videoconferência deixou de ser vista como uma *killer application*, para ser vista como uma aplicação com um objetivo específico. Ainda, de acordo com (KESHAV, 1994), um usuário tolera mais uma perda na qualidade de vídeo do que uma falha na qualidade do áudio. Constata-se que os usuários preferem o som à imagem.

O trabalho de (LUO et al., 2007) apresenta alguns dados importantes que devem ser considerados no uso de videoconferências, como:

- A escassez de endereços de IPv4, que levou à adoção de NAT, onde o endereço de cada cliente pode não ser único. Isso pode tornar complexo o início de uma conexão entre clientes.

- A heterogeneidade da condição da rede. Por exemplo, alguns usuários podem acessar por redes sem fio e outros por rede cabeada. Cada conexão pode ter um comportamento diferenciado.

- O baixo suporte no nível de rede ao *multicast*, que segundo (COULOURIS et al., 2011), é uma comunicação entre um nodo para muitos.

- Poucos usuários têm capacidade de se comportar como cliente e servidor simultaneamente.

- A latência com mais de 300 ms começa a ser reparada pelos usuários.

Muitos sistemas tradicionais de videoconferência ainda enfrentam problemas para lidar com um grande número de usuários. Desde os sistemas mais antigos como o *INRIA Videoconferencing System* (IVS), que é descrito em (TURLETTI; HUITEMA, 1996), até os sistemas mais modernos. A causa desse problema conforme explicado na seção 1.1 é devido à utilização de uma configuração centralizada para a transmissão dos dados para os clientes. O problema da escalabilidade em videoconferências já é abordado há alguns anos na comunidade científica como no trabalho de (LUO et al., 2007). Uma das soluções, para resolver o problema da replicação excessiva no envio dos dados, é criar um sistema de *multicast*, em nível de aplicação. Já que, segundo (BANERJEE; BHATTACHARJEE; KOMMAREDDY, 2002), o suporte a *multicast*, em nível de rede, ainda não é muito difundido pelos ISP.

Segundo (LE; NGUYEN, 2013), em uma videoconferência, geralmente poucos usuários estão ativos falando, portanto, comenta-se que uma técnica para reduzir o gasto de banda seria detectar o usuário que não está falando e cancelar a transmissão até ele recomeçar a falar. Portanto, pode-se diminuir o gasto de banda com a transmissão de áudio com a detecção de fala ou não de um usuário. De acordo com (LIU; GUO; LIANG, 2008), o aumento de *Video-over-IP* cresceu muito nos últimos anos; só em 2006 ocorreu um aumento de mais de 30% no número de *streamings* de vídeo na Internet. Com esse acréscimo, evidenciou-se como é custoso realizar uma conferência centralizada. Assim, buscaram-se algumas alternativas como sistemas de videoconferência P2P.

As próximas subseções descrevem redes de sobreposição (*overlay*) e seu relacionamento com videoconferências, além de videoconferências *peer-to-peer*.

2.1.1 Redes de Sobreposição

Conforme citado anteriormente, segundo (COULOURIS et al., 2011), *multicast* é um importante requisito para aplicações distribuídas e deve ser fornecido mesmo quando o suporte para o IP *multicast* não está disponível. Para fornecê-lo, geralmente, cria-se uma rede de sobreposição (*overlay*) sobre a rede TCP/IP.

Segundo (COULOURIS et al., 2011), redes de *overlay* são redes virtuais, que consistem de conexões e nodos virtuais, as quais residem sobre uma outra rede (como a rede IP). Elas oferecem algo que não era promovido anteriormente, podendo, por exemplo, prover uma rede mais eficiente para algum tipo de aplicação ou funcionalidades adicionais, como *multicast*, além de outros serviços.

Algumas vantagens, segundo (COULOURIS et al., 2011), são a habilidade das redes de sobreposição coexistirem simultaneamente e permitirem a criação de novas redes sem alterar a rede física. Segundo o autor, o Skype (MICROSOFT, 2013) é um exemplo de arquitetura de *overlay*.

Na Figura 2.1 é mostrada um exemplo de uma rede de sobreposição para a distribuição de conteúdo. Nessa figura, o nodo fonte envia dados para os nodos finais, através dos nodos intermediários. Observa-se que a comunicação na rede de sobreposição criada é unidirecional, já na camada de rede, o fluxo é bidirecional.

Com as redes de sobreposição é possível criar uma ALM (*Application Layer Multicast*). Uma ALM busca prover *multicast* em nível de aplicação, criando uma rede em

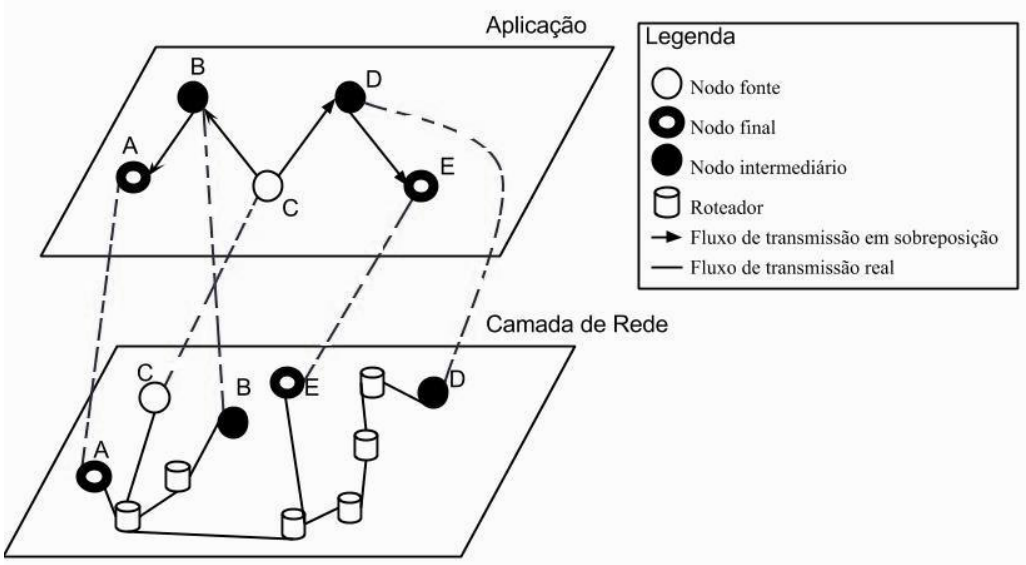


Figura 2.1: Exemplo de uma rede de sobreposição para distribuição de conteúdo

overlay à camada de rede. Em (LUO et al., 2007), é proposto um sistema de videoconferência multiusuários utilizando uma camada de *multicast* no nível de aplicação.

2.1.2 Sistemas de videoconferência *peer-to-peer*

Para realizar a transmissão de vídeo na Internet, com uma qualidade similar a observada nas televisões, o indicado é uma taxa de pelo menos 400 kbit/s para cada usuário, segundo (LIU; GUO; LIANG, 2008). Logo, torna-se complexo sustentar um grande número de usuários em uma conferência com voz e vídeo. Devido ao problema do alto custo para suportá-los, surgiram novas alternativas que tentam solucionar esse problema. Um exemplo são os sistemas de videoconferência em P2P, que ressurgiram como um novo paradigma para a construção de aplicações distribuídas na rede, segundo (LIU; GUO; LIANG, 2008). Consiste na busca de encorajar os usuários a agir como clientes e servidores, sendo chamados de *peers*.

Sistemas *Peer-to-peer* podem ser classificados em duas grandes categorias: árvore e malha.

- Árvore - um ponto de origem envia o vídeo para os seus filhos conforme pode ser visto na Figura 2.2. Essa distribuição apresenta alguns problemas como a falha de um *peer* acarreta na queda de todos os seus descendentes.

- Malha - nos sistemas em árvore um filho somente possui um pai, conforme mostrado na Figura 2.2, já na distribuição em malha um filho pode ter mais de uma pai, removendo assim o ponto único de falha do anterior. A distribuição em malha tende a ser mais dinâmica e torna o sistema um pouco mais robusto contra falhas que ocorrem com a queda de um *peer*. Muitos sistemas modernos de distribuição de vídeo em P2P estão adotando a abordagem em malha como, por exemplo, (ZHANG et al., 2005).

Segundo (LIU; GUO; LIANG, 2008), existem alguns problemas em aberto na área de distribuição de vídeo em P2P, tais como:

- A baixa taxa de *upload* dos usuários comuns.

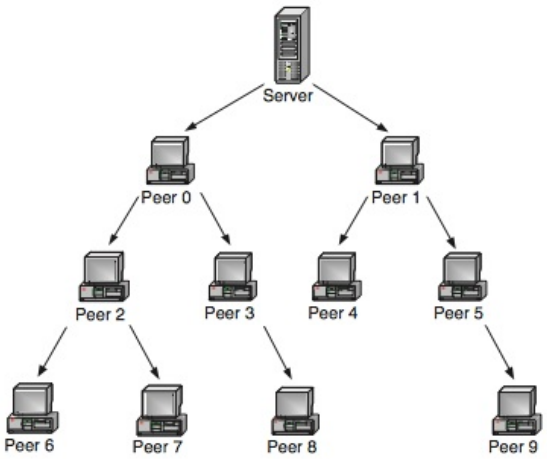


Figura 2.2: Distribuição em árvore de uma transmissão em P2P (LIU; GUO; LIANG, 2008)

- Com um número pequeno de usuários a qualidade da transmissão pode ficar com baixa qualidade e instável.
- Os atrasos para iniciar a transmissão do vídeo.

Uma das soluções para melhorar a qualidade, segundo (LUO et al., 2007), é a inclusão de *super peers*. Estes seriam servidores de mídia que não participariam da conferência e agiriam como retransmissores de conteúdo para a conferência. Assim, a ferramenta proposta na seção 1.2 têm uma relação fundamental com os sistemas de videoconferência em P2P. Com o auxílio da ferramenta, pode-se criar um sistema misto distribuído envolvendo técnicas de videoconferência centralizada com P2P, empregando o uso de vários servidores, que funcionariam como *super peers*. Assim, as rotas fornecidas poderiam ser utilizadas para a criação de árvores ou malhas de distribuição.

2.2 Métricas

Para avaliar uma conexão entre dois nodos de uma rede é necessário definir algumas métricas, que servirão como descritores da qualidade da conexão. Para realizar a avaliação foram escolhidas as seguintes métricas: latência, *jitter* e vazão, bem como a rota entre os nodos. As três primeiras métricas, segundo (TANENBAUM; WETHERALL, 2011), são conhecidas como *QoS* (Qualidade de Serviço), pois fornecem informações sobre a qualidade da conexão. Nas próximas subseções, as métricas essenciais para este trabalho serão brevemente descritas.

2.2.1 Latência

Segundo (TANENBAUM; WETHERALL, 2011), a latência é o intervalo de tempo que um dado leva para ir da origem até o destino. Ela também é conhecida como atraso, podendo ser medida de diversos modos, tais como:

- A medição do tempo decorrido do envio do pacote da origem até sua chegada ao destino, sendo muito complexa de ser obtida devido aos problemas inerentes à medição de tempo em relógios distintos. Segundo (COULOURIS et al., 2011), dois relógios distintos, mesmo que possuam valores iniciais iguais, tendem a sofrer uma

variação no valor do tempo medido, consequentemente, se afastando do tempo perfeito. Um termo muito utilizado para essa variação é *Clock Drift Rate*.

- A medição do tempo de ida e volta do pacote, sendo que esse tempo é chamado de *Round-trip Time* (RTT) e a latência pode ser aproximada dividindo esse tempo por dois. Quando se aproxima o valor da latência, dividindo o RTT por dois, nem sempre se obtém um dado preciso, pois o pacote pode ter utilizado caminhos diferentes ou enfrentado congestionamentos diferentes na ida e na volta.

A latência pode ser influenciada por vários fatores, segundo (PADMANABHAN; MOGUL, 1995), tais como:

- Distância entre os dois pontos, pois quanto maior a distância maior será o tempo que a informação deve trafegar na rede;
- Meio de transferência, que liga dois nodos na rede interfere no valor da latência, por exemplo, rede cabeada e sem fio;
- Congestionamento que possa ocorrer na rede, devido a muitos usuários ou a uma pequena vazão;
- Falta de recursos, como capacidade de processamento, no ponto de origem ou de destino, que atrasa o processamento do dado;
- Filas nos roteadores intermediários presentes na conexão.

2.2.2 Jitter

Segundo (TANENBAUM; WETHERALL, 2011), o *jitter* é a variação no tempo médio do atraso (latência). Ainda, segundo o autor, o vídeo, e mais particularmente, o áudio são extremamente sensíveis ao *jitter*. É recomendado chamar o *jitter* de variação do atraso do pacote ou PDV (*Packet Delay Variation*), segundo (ITU, 2011). Mesmo pequena quantidade de *jitter* pode causar problemas na reprodução de áudio e vídeo. Segundo (TANENBAUM; WETHERALL, 2011), algumas causas são:

- Congestionamento na rede, fazendo com que um pacote entre em uma fila em algum roteador intermediário e, consequentemente, demore um tempo maior que o normal para ser enviado;
- Ruído térmico, que ocorre principalmente em circuitos elétricos, assim um pacote pode sofrer um atraso eventual;
- Erros de configuração que pode gerar o descarte eventual de alguns pacotes sendo necessário o reenvio do mesmo e, consequentemente, uma variação no tempo médio da latência, que é o *jitter*;
- Dependendo do conteúdo que será enviado, por exemplo, um pacote de voz, pode ocorrer um atraso eventual na sua codificação, gerando assim um atraso diferente na hora do seu envio.

Segundo (TANENBAUM; WETHERALL, 2011), existem diversas técnicas para tratar o problema do *jitter*, tais como: a criação de um buffer de *jitter* ou a transmissão de conteúdos não são sensíveis ao *jitter*.

2.2.3 Vazão

A vazão é também chamada de taxa de transferência ou *throughput*. De acordo com (TANENBAUM; WETHERALL, 2011), ela é a taxa de tempo para a transferência de certa quantidade de dados entre dois nodos na rede. As principais unidades de representação para a vazão são: kbit/s (quilobit por segundo), Mbit/s (Megabits por segundo) e, recentemente, Gbit/s (Gigabits por segundo). Segundo (JIN; TIERNEY, 2003), a forma mais conveniente para a medição da vazão consiste na medição ativa, pois não necessita de nenhum privilégio especial para executar essa ação. A medição ativa da vazão consiste no envio de pacotes na rede com a finalidade de estimar o valor da vazão. Ainda, segundo (JIN; TIERNEY, 2003), existem medições passivas, como SNMP (*Simple Network Management Protocol*), que é menos intrusivo, porém elas necessitam de privilégios especiais. A vazão pela medição ativa é obtida através do envio máximo de dados pelo enlace até saturá-lo, medindo assim o tempo que levou para completar o envio de todos os dados. Porém, segundo (JIN; TIERNEY, 2003), essa técnica de medição pode causar alguns problemas como esgotamento de *buffer* nas filas dos roteadores envolvidos no enlace. Ainda, por ser tão intrusiva, pode afetar outras comunicações que estão ocorrendo na mesma conexão, portanto não deve ser executada seguidamente.

Segundo (GUOJUN, 2003), existem diversas vantagens em conhecer previamente a vazão em uma conexão. Uma delas é a melhor distribuição de conteúdo em uma rede, pois, quando um sistema sabe sua banda máxima disponível em seu enlace se pode filtrar quais dados serão enviados.

2.2.4 Rota

A rota em uma rede é, segundo (TANENBAUM; WETHERALL, 2011), o caminho que o pacote de dados percorre desde a origem até o destino. Segundo (WANG; CROWCROFT, 1992), um caminho pode ser estático, semi-estático ou dinâmico. Um algoritmo de roteamento estático possui uma implementação menos complexa, porém é mais suscetível a falhas de recursos e a mudanças na topologia da rede. Por outro lado, algoritmos semi-estáticos, como o algoritmo do menor caminho (*Shortest Path Algorithm*), que é muito utilizado na Internet, permanece constante por pequenos períodos de tempo, porém é atualizado quando ocorre mudanças significativas na rede. Contrastando com os anteriores, existem algoritmos de roteamento dinâmico, que alteram continuamente as decisões de rota para refletir mudanças na topologia e no fluxo da rede; contudo, eles são mais complexos e requerem grande quantidade de troca de dados para funcionar.

Segundo (TANENBAUM; WETHERALL, 2011), é natural que a rede faça a escolha automática de qual é a melhor rota para um pacote. Segundo (GOLDBERG; HARRELSON, 2005), encontrar o caminho com o menor custo é um problema fundamental para muitas aplicações. Para a criação de rotas existem diversos algoritmos. Neste trabalho, para atingir o objetivo proposto na seção 1.2, será utilizado o algoritmo de Dijkstra na geração de rotas que priorizem as métricas descritas anteriormente. O funcionamento do algoritmo de Dijkstra (DIJKSTRA, 1959) consiste na busca do caminho com o menor custo entre um par de nodos, considerando-se um peso para cada aresta e descobrindo, por fim, o menor caminho entre os nodos.

3 DESENVOLVIMENTO E METODOLOGIA DOS EXPERIMENTOS

Este capítulo detalha a criação da ferramenta cuja finalidade é a geração de rotas entre um conjunto de servidores. Essas rotas devem priorizar alguma das métricas citadas na seção 2.2 e, também, possibilitar a criação de rotas alternativas, que tem o intuito de aumentar a disponibilidade a fim de resolver os problemas citados na seção 1.1. O sistema criado é composto de 3 módulos independentes, sendo eles: o *Medidor*, o *Visualizador* e o *Otimizador*. O primeiro e o último módulos possuem funcionalidades essenciais para a geração dos resultados e conclusões apresentados no capítulo 4 e para cumprir o objetivo proposto na seção 1.2.

Este capítulo está dividido em duas seções: a primeira, consiste na implementação da ferramenta e a segunda, em uma metodologia de experimentação e na especificação dos experimentos realizados.

3.1 Implementação

A implementação da ferramenta proposta consiste na criação de três módulos:

- ***Medidor***, que é responsável pela coleta de dados sobre a qualidade da conexão entre um conjunto de servidores. Esses dados são fornecidos aplicando as métricas de latência, *jitter*, vazão e rota.

- ***Visualizador***, que é responsável pela visualização dos dados fornecidos pelo *Medidor* em um grafo interativo.

- ***Otimizador***, que é responsável pela criação de rotas com os dados fornecidos pelo *Medidor*.

Nas próximas três subseções serão descritas as funcionalidades e o funcionamento de cada módulo.

3.1.1 Medidor

O *Medidor* é uma aplicação criada em *Shell Script*, que executa em um terminal no sistema operacional Linux. A função principal do *Medidor* é a realização da medição da qualidade da conexão de acordo com as métricas de latência, de *jitter*, da vazão e da rota. Estas métricas foram descritas no capítulo 2. Para realizar a medição este módulo utiliza duas ferramentas auxiliares, que são o Nuttcp (FINK, 2007) e o Traceroute (LINUX, 2011). Essas ferramentas serão descritas nas próximas subseções, após ser feita

a explicação sobre o *Medidor*. O funcionamento do servidor pode ser observado na Figura 3.1, onde S_1 até S_n representam os servidores, que serão avaliados, e *M* representa o nodo, que requisita a medição das métricas do servidor. Pode-se observar, pela seta cheia, que a instância do *Medidor* na máquina *M* realiza a requisição da medição para o servidor S_1. Após receber os resultados, requisita-se a medição para os outros servidores até, finalmente, a última requisição para S_x, que é representada pela seta pontilhada. O funcionamento passo-a-passo do fluxo do *Medidor* é:

1. O nodo *M* recebe como entrada uma lista com os servidores que serão avaliados.
2. *M* requisita a medição a partir de um servidor, que ainda não foi avaliado, ou seja, ainda não foi utilizado como origem da medição, para todos os outros servidores.
3. *M* armazena o resultado em um JSON, devido à facilidade de processá-lo posteriormente. O JSON conterá todas as medições.
4. Após armazenar a medição, retorna-se ao passo 2, se ainda existe um servidor não avaliado.

A saída do programa é um JSON, esse consiste em uma lista de pares (S_x,S_y), onde cada par representa a conexão entre o servidor *x* e o servidor *y*. Cada par possui um valor para a latência, *jitter*, vazão e uma rota. A saída gerada será utilizado pelo *Visualizador*, para exibir graficamente os dados obtidos, e pelo *Otimizador*, para gerar rotas otimizadas de acordo com alguma das métricas.

Os comandos utilizados para obter as métricas de latência, vazão e *jitter* entre os servidores foram respectivamente:

- nuttcp -T5 servidorOrigem servidorDestino
- nuttcp -T5 -j -v servidorOrigem servidorDestino

Nesse comando é possível observar que o intervalo de tempo de cada medição é de 5 segundos, através do parâmetro (-T5), sendo que foi escolhido esse valor, pois não foram observadas mudanças significativas nos valores obtidos com tempos maiores de medições. Nota-se que um tempo maior ajuda a mascarar interferências que possam ocorrer na obtenção dos valores, porém quanto maior o tempo, maior será o impacto da medição na rede. Observa-se que deve existir uma instância ativa do *Medidor* em cada servidor a fim de realizar a avaliação das conexões. Portanto, é necessário o acesso aos servidores.

O comando utilizado para obter a rota entre a origem até algum servidor de destino é:

- traceroute -I servidorDestino

Nesse comando para obter a rota o parâmetro (-I) indica que será utilizado o protocolo ICMP na medição.

As duas ferramentas, que foram utilizadas pelo *Medidor* para obter as informações sobre as conexões de acordo com as métricas descritas anteriormente, são:

- Nuttcp, que realiza a medição de desempenho de uma rede e fornece as informações de latência, *jitter* e vazão;
- Traceroute, que exibe a rota entre dois pontos em uma rede.

As próximas subseções descrevem com maiores detalhes as duas ferramentas.

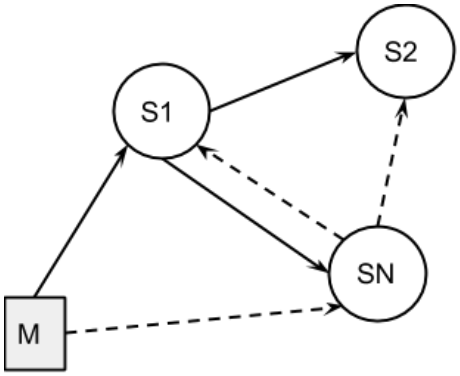


Figura 3.1: Fluxo de funcionamento do *Medidor*

3.1.1.1 Nuttcp

Nuttcp é uma ferramenta de medição do desempenho de rede. Uma de suas funcionalidades é determinar a vazão em TCP ou UDP. A ferramenta possibilita o gerenciamento remoto das medições, portanto permite um maior controle sobre as medições.

O Nuttcp, segundo (FINK, 2007), é baseado no Ttcp, que foi escrito por Mike Muss em 1984. Portanto, como ele tem um longo período de desenvolvimento, é factível presumir que o mesmo possui uma boa fundamentação teórica e uma boa confiabilidade. De acordo com (TIERNEY et al., 2012), a ferramenta possui mais confiabilidade quando comparada ao Iperf, este também muito utilizado para testes de desempenho de rede.

3.1.1.2 Traceroute

O Traceroute é uma ferramenta de diagnóstico de rede cuja funcionalidade é exibir o caminho que um pacote de dados percorre desde o servidor origem até o servidor destino. O funcionamento da ferramenta, segundo (LINUX, 2011), consiste no envio de pacotes *probes* com um número de *time-to-live* (TTL) incremental de 1 até *n*, onde *n* representa o número de roteadores intermediários até o destino final. O TTL é o número de saltos (*hops*) que o pacote continua existindo na rede até ele ser descartado. O Traceroute aguarda o recebimento de uma mensagem avisando que o tempo limite (número de saltos máximo) foi excedido, pois a cada roteador o TTL é decrementado (TTL=TTL-1), e quando ele atinge o valor "zero", o roteador envia uma mensagem de *TTL Exceeded* à origem.

Pode-se escolher o uso dos protocolos UDP, TCP ou ICMP para realizar o descobrimento da rota. Segundo (BLANCHER, 2003), atualmente, alguns roteadores podem não responder as requisições do Traceroute devido ao risco de ataques ou de alguns *firewalls*, que podem bloquear essas requisições. Nesse caso, o Traceroute normalmente detecta isso por *timeout* e coloca um "*"no lugar do roteador, não obtendo a informação do mesmo. A Figura 3.2 apresenta um resultado do comando "traceroute allspice.lcs.mit.edu"mostrando os roteadores intermediários. Observa-se que o roteador número 13 não retornou a mensagem.

3.1.2 Visualizador

O *Visualizador* é uma aplicação criada em Javascript. Sua funcionalidade básica é a exibição, em forma de grafo, dos dados fornecidos pelo *Medidor*. Para a construção deste módulo foi utilizado a biblioteca livre *Arbor.JS* (ARBOR.JS, 2011). Ela fornece um algoritmo de ajustes de forças, que resolve o problema da localização dos nodos e evita a

```
traceroute to mercury.lcs.mit.edu (18.26.0.122), 64 hops max, 52 byte packets
 1  0.0.0.0 (0.0.0.0)  8.426 ms  8.867 ms  9.079 ms
 2  c915c002.virtua.com.br (201.21.192.2)  9.775 ms  8.835 ms  27.346 ms
 3  embratel-t0-1-0-1-uacc01.ctamc.embratel.net.br (200.167.43.9)  32.281 ms  34.156 ms  29.464 ms
 4  ebt-t0-6-2-0-tcore01.ctamr.embratel.net.br (200.244.140.194)  33.528 ms
    ebt-t0-15-0-12-tcore01.ctamc.embratel.net.br (200.244.140.218)  42.784 ms
    ebt-t0-15-0-11-tcore01.ctamc.embratel.net.br (200.244.140.214)  36.084 ms
 5  ebt-bp1451-intl01.nyk.embratel.net.br (200.230.220.110)  147.253 ms
    ebt-bp1421-intl01.nyk.embratel.net.br (200.230.220.130)  146.301 ms  145.521 ms
 6  ae59.edge2.newyork1.level3.net (4.71.230.241)  142.357 ms  143.720 ms  142.513 ms
 7  vlan60.csw1.newyork1.level3.net (4.69.155.62)  149.625 ms  151.181 ms
    vlan80.csw3.newyork1.level3.net (4.69.155.190)  153.597 ms
 8  ae-91-91.ebr1.newyork1.level3.net (4.69.134.77)  146.961 ms
    ae-61-61.ebr1.newyork1.level3.net (4.69.134.65)  147.792 ms
    ae-81-81.ebr1.newyork1.level3.net (4.69.134.73)  151.844 ms
 9  ae-1-8.bar2.boston1.level3.net (4.69.140.97)  152.152 ms  150.359 ms  155.136 ms
10  ae-0-11.bar1.boston1.level3.net (4.69.140.89)  146.777 ms  147.183 ms  146.355 ms
11  ae-7-7.car1.boston1.level3.net (4.69.132.241)  155.346 ms  148.546 ms  152.692 ms
12  massachuset.bar1.boston1.level3.net (4.53.48.98)  149.366 ms  168.610 ms  159.077 ms
13  * * *
14  backbone-rtr-1-dmz-rtr-1.mit.edu (18.168.5.1)  149.397 ms  147.314 ms  147.392 ms
15  * * *
16  mitnet.trantor.csail.mit.edu (18.4.7.65)  148.523 ms  147.867 ms  166.170 ms
17  trantor.helicon.csail.mit.edu (128.30.0.246)  151.555 ms  155.182 ms  150.458 ms
18  * * *
```

Figura 3.2: Exemplo de execução do traceroute com situação de bloqueio

sobreposição dos mesmos. O fluxo de execução do módulo é:

- Carregar como entrada o arquivo JSON fornecido pelo *Medidor*;
- Converter os pares presentes na entrada para um grafo;
- Escolher o servidor de origem e a métrica a ser exibida;
- Opcionalmente, pode-se escolher qual dos nodos terá o seu caminho expandido.

Na Figura 3.3 é mostrada a interface do *Visualizador*, onde os nodos mais claros são os servidores. Na imagem é mostrada a latência entre o servidor *mconf-live-rn01.rnp.br* e todos os outros servidores. Ainda, pode ser mostrado o *jitter* e a vazão. Permite-se a alteração do nodo de origem para ver outras informações.

Para facilitar a legibilidade foram feitas algumas escolhas na construção do módulo, como a exibição parcial do grafo gerado a partir da saída do *Medidor* e a expansão das rotas somente a partir da origem. Um dos benefícios proporcionados por esse módulo é a facilidade na visualização dos pontos conectados entre as rotas dos servidores.

3.1.3 Otimizador

O *Otimizador* é uma aplicação criada em Java. A aplicação recebe como entrada um JSON fornecido pelo *Medidor* e gera como saída rotas, que priorizam algumas das métricas. O fluxo de funcionamento do *Otimizador* pode ser observado na Figura 3.4. Para utilizar o *Otimizador* é necessário carregar a saída do *Medidor* e selecionar se a saída deve ser uma rota otimizada ou uma rota alternativa. Se for uma rota otimizada deve-se fornecer o servidor de origem e o servidor de destino. Se for uma rota alternativa deve-se fornecer a rota original e o tipo de otimização por rota descrito na seção 3.1.3.3. Finalmente, deve-se fornecer a métrica que o caminho gerado deve priorizar.

Para a geração das rotas é utilizado o algoritmo de Dijkstra (DIJKSTRA, 1959). Este é utilizado na busca do menor caminho entre dois nodos no grafo criado a partir dos dados fornecidos pela ferramenta de medição. Para a busca da rota com a maior vazão é feita uma modificação no algoritmo de Dijkstra, conforme (MALPANI; CHEN, 2002). Com

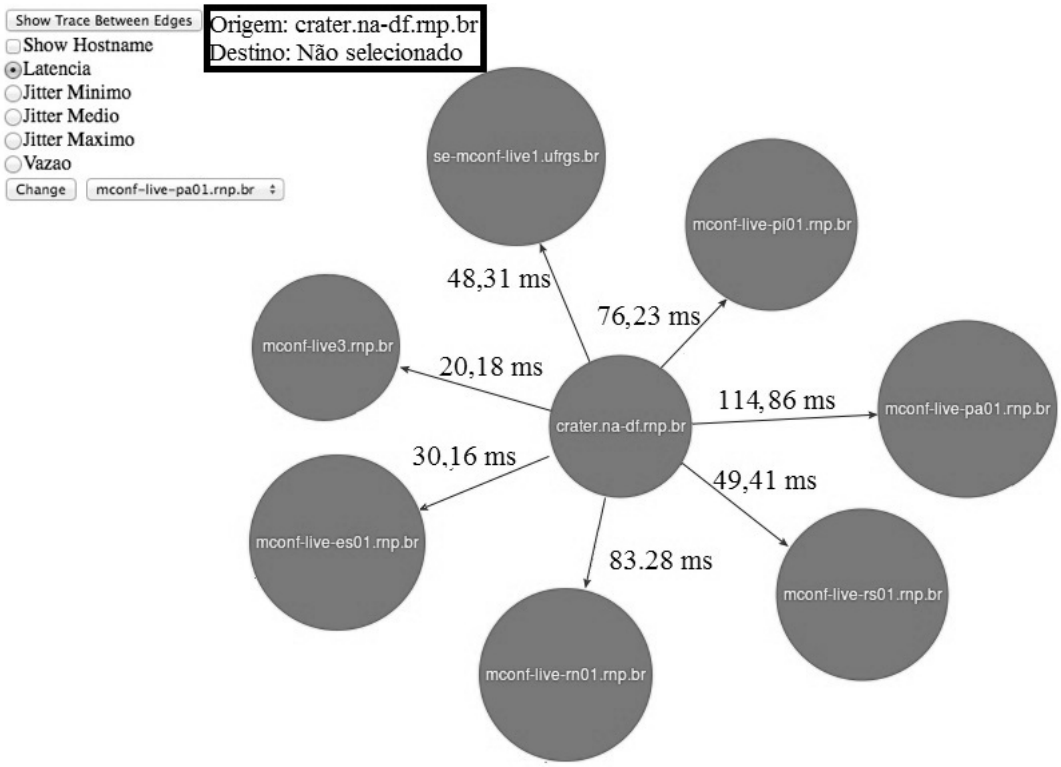


Figura 3.3: Interface inicial do *Visualizador*

essa modificação, o algoritmo passa a fornecer a rota com a maior vazão. Utilizando *Otimizador* é possível solicitar a rota entre dois servidores que possua a menor latência ou o menor *jitter* ou a maior vazão. Ainda se podem requisitar rotas alternativas que auxiliem no aumento da disponibilidade do serviço, conforme descrito na seção 1.1.

A interface original do módulo pode ser vista na Figura 3.5, onde é mostrado o caminho com a menor latência entre o servidor *ES*, localizado no Espírito Santo, e o servidor *RN*, localizado no Rio Grande do Norte.

O módulo possui 4 formas de otimização sendo elas: menor latência, menor *jitter*, maior vazão e rota. Tais formas serão detalhadas nas próximas subseções, com a exceção do menor *jitter*, pois ele é análogo a menor latência.

3.1.3.1 Menor latência

A otimização pela menor latência recebe como entrada dois servidores: origem e destino. A saída esperada é o caminho com a menor latência. Por exemplo, em uma rede com 3 servidores (S_1, S_2, S_3), a latência entre S_1 e S_2 é 10 ms, entre S_2 e S_3 é 8 ms e entre S_1 e S_3 é 30 ms. Logo, a menor latência entre S_1 e S_3 será a rota S_1->S_2->S_3 com 18 ms ao invés do caminho direto de 30 ms. Assim, a rota fornecida será S_1->S_2->S_3. O servidor S_2 atuará como retransmissor nesse caminho, sendo ele um nodo intermediário. Para encontrar essa rota o módulo utilizou o algoritmo de Dijkstra.

3.1.3.2 Maior vazão

A otimização pela maior vazão recebe como entrada dois servidores: origem e destino. A saída esperada é o caminho com a maior vazão, sendo que a vazão de um caminho é a conexão com menor vazão que pertence à rota. Para encontrar a rota com maior vazão é utilizado o algoritmo de Dijkstra modificado, já citado anteriormente. Portanto, em uma

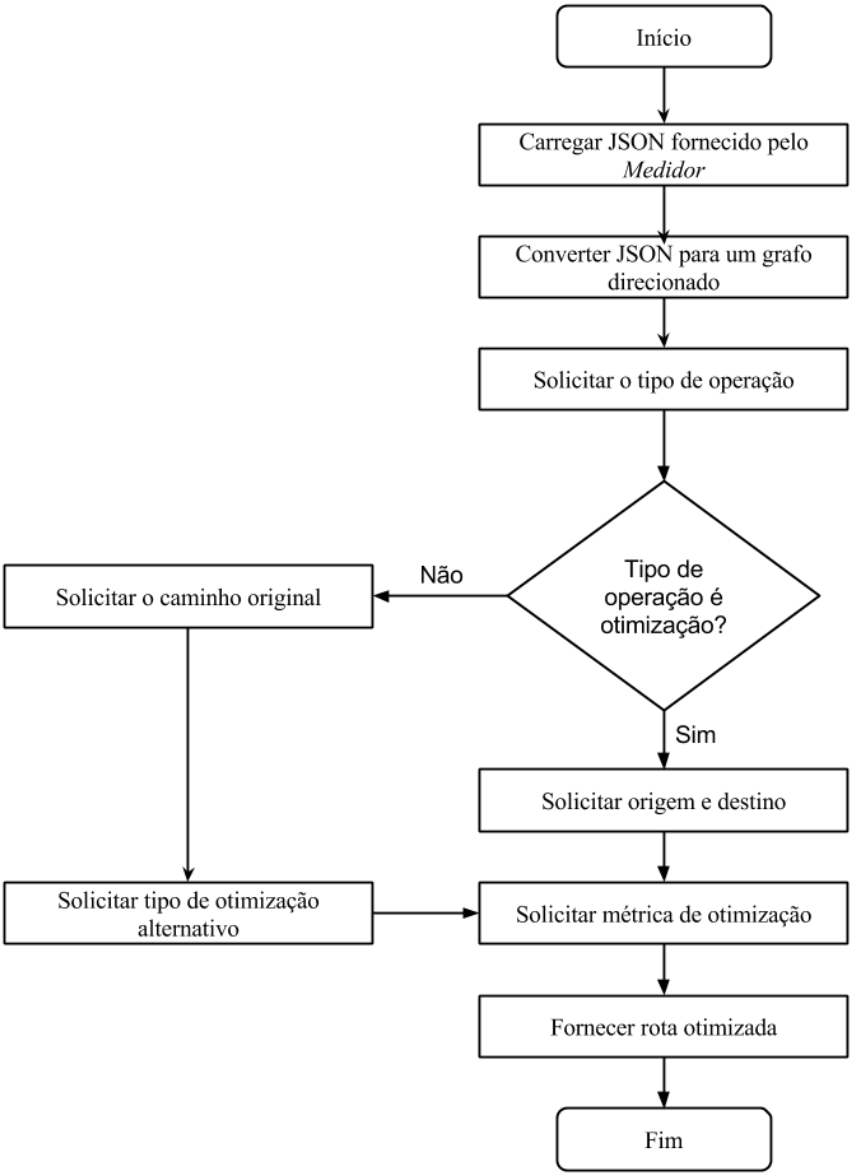


Figura 3.4: Fluxo de execução do *Otimizador*

rede com 3 servidores (S_1,S_2,S_3), a vazão entre S_1 e S_2 é 10 Mbit/s, entre S_2 e S_3 é 8 Mbit/s e entre S_1 e S_3 é 5 Mbit/s. Logo, a maior vazão entre S_1 e S_3 será de 8Mbps, pelo caminho S_1->S_2->S_3, sendo que o caminho é limitado pela conexão entre S_2 e S_3.

3.1.3.3 Rota

A otimização pela rota busca aumentar a disponibilidade dos serviços fornecidos aos clientes. Essa otimização recebe como entrada um caminho e gera como saída um caminho alternativo à rota original. Para encontrar uma rota alternativa foram empregadas duas técnicas: aumento dos pesos do caminho original e remoção de conexões.

Para melhor exemplificar as técnicas foi criada uma rede com quatro servidores de exemplo. Na Figura 3.6 é mostrado a latência e a rota entre os 4 servidores.

A primeira técnica empregada é o aumento dos pesos, que eleva todos os valores das conexões presentes no caminho fornecido a fim de excluir a rota original da melhor

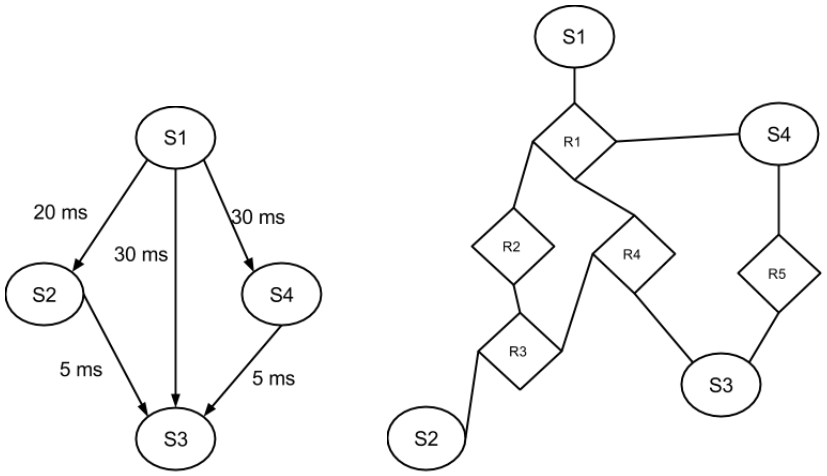
```
Servidores:
0 - crater.na-df.rnp.br
1 - mconf-live-pa01.rnp.br
2 - mconf-live-es01.rnp.br
3 - mconf-live-rn01.rnp.br
4 - mconf-live3.rnp.br
5 - mconf-live-rs01.rnp.br
6 - mconf-live-pi01.rnp.br
7 - se-mconf-live1.ufrgs.br
Escolha servidor de origem:
2
Escolha servidor de destino:
3

Escolha a metrica:
1: VAZAO
2: LATENCIA
3: JITTER
4: MISTA
2

Escolha o algoritmo:
1: DIJKSTRA
2: DIJKSTRA MODIFICADO

1
Distance to mconf-live-rn01.rnp.br: 71.78
Path: [mconf-live-es01.rnp.br, mconf-live-pi01.rnp.br, mconf-live-rn01.rnp.br]
```

Figura 3.5: Interface do *Otimizador*



(a) latência entre os servidores (b) Rota entre os servidores

Figura 3.6: Rede com 4 servidores

escolha. A rota fornecida entre os servidores S_1 e S_3 mostrados na Figura 3.6a foi S_1->S_2->S_3 com latência de 25 ms. Nessa técnica é aumentada a latência entre S_1 e S_2 para um valor suficientemente grande e, também, é aumentada a latência entre S_2 e S_3. A rota alternativa retornada é S_1->S_3. Assim, forçou-se o retorno de uma rota alternativa que pode ser utilizada caso ocorra uma falha no servidor S_2. Originalmente, S_2 era utilizado como um retransmissor de dados entre S_1 e S_3, mas na rota alternativa isso não acontece.

A segunda técnica busca utilizar o conhecimento obtido com os dados fornecidos pelo *Medidor* e, com isso, remover as conexões que possuem roteadores intermediários presentes no caminho original quando for possível. Para ficar mais claro, volta-se a analisar o exemplo anterior. Na Figura 3.6b é possível observar as rotas entre os servidores, que são:

- Entre S_1 e S_2: R_1->R_2->R_3;
- Entre S_1 e S_3: R_1->R_4;

- Entre S_1 e S_4: R_1;
- Entre S_2 e S_3: R_3->R_4;
- Entre S_4 e S_3: R_5;

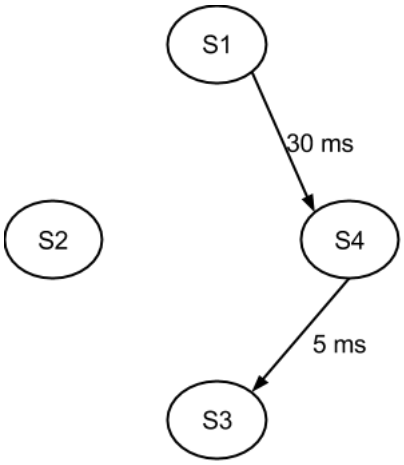


Figura 3.7: Rede com a remoção de rotas

Assim, a rota original era S_1->S_2->S_3, ou seja, ela contém os roteadores: R_1, R_2, R_3 e R_4. A técnica tentará remover todas as conexões (arestas) que contenham um desses roteadores. Contudo, ainda deverá existir uma rota entre S_1 e S_3. Na Figura 3.7 pode-se observar o grafo resultante com a remoção das conexões. Agora, reaplica-se o *Otimizador* para gerar a rota de menor latência cujo resultado será rota S_1->S_4->S_3. Esta rota possui uma maior latência quando comparada a rota obtida pela técnica anterior. Porém, existe uma maior disjunção em nível de rede entre a rota original e a rota obtida por esta técnica quando comparada com a rota obtida pela técnica de aumento de pesos. A rota alternativa obtida contém os roteadores R_1 e R_5, enquanto que a rota obtida com o aumento dos pesos continha os roteadores R_1 e R_4, ambos estão incluídos no caminho original. Com a remoção de conexões o caminho só compartilhará o roteador R_1.

3.2 Metodologia dos experimentos

A elaboração de rotas entre os servidores de acordo com a latência, *jitter* e vazão requer os seguintes passos:

1. Reunir um conjunto com *n* servidores que estejam em redes distintas, de preferência que eles não estejam concentrados em uma região geográfica específica.

2. Avaliar a qualidade de conexão entre cada servidor com o *Medidor*, cuja funcionalidade e funcionamento estão descritos na seção 3.1.

3. Utilizar o *Otimizador*, que deverá criar rotas específicas entre os servidores, a fim de otimizar alguma das métricas apresentadas na seção 2.2.

Seguindo esses passos será possível, também, gerar rotas alternativas entre quaisquer pares de servidores pertencentes à rede.

Para a realização dos experimentos, foi utilizada a ferramenta criada e descrita na seção 3.1, a fim de obter informações sobre a qualidade da conexão de um conjunto real de servidores. Além disso, buscou-se gerar rotas que priorizassem as métricas de latência, *jitter*, vazão e também a geração de algumas rotas alternativas. Os resultados dos experimentos estão descritos no capítulo 4.

Os experimentos foram realizados cinco vezes, sendo que os dados sempre foram medidos no mesmo horário (06:00), entre os dias 12 e 16 de novembro de 2013. A escolha do horário de medição foi proposital, pois buscou-se realizar as medições quando os servidores estavam ociosos, já que eles são servidores de produção. Foram realizados somente cinco repetições, pois não se observou variação maior que 10% quando comparando o resultado a média aritmética das cinco medições. Ainda é necessário um estudo mais aprofundado para a escolha do número de repetições agregando, por exemplo, métodos estatísticos, a fim de validor os resultados.

Na próxima subseção será mostra uma breve descrição do ambiente de experimentação utilizado na geração dos experimentos.

3.2.1 Ambiente de experimentação

O ambiente de experimentação consiste em uma rede composta de 8 servidores conectados a Internet. Todos os oito servidores utilizados estão distribuídos em vários estados do Brasil, conforme Figura 3.8. O endereço de cada servidor é:

1. *crater.na-df.rnp.br* (*DF*), localizado no Distrito Federal;
2. *mconf-live-pa01.rnp.br* (*PA*), localizado no Pará;
3. *mconf-live-es01.rnp.br* (*ES*), localizado no Espírito Santo;
4. *mconf-live-rn01.rnp.br* (*RN*), localizado no Rio Grande do Norte;
5. *mconf-live3.rnp.br* (*RJ*), localizado no Rio de Janeiro;
6. *mconf-live-rs01.rnp.br* (*RS1*), localizado no Rio Grande do Sul;
7. *mconf-live-pi01.rnp.br* (*PI*), localizado no Piauí;
8. *mconf-live2.rnp.br* (*RS2*), localizado no Rio Grande do Sul, na UFRGS;

Todos os servidores descritos são máquinas virtuais, que utilizam a versão do Ubuntu 10.04 como sistema operacional. Na Tabela 3.1 é possível observar a configuração detalhada e individual de cada máquina. A única máquina com reserva de CPU é o servidor *DF*, localizado no Distrito Federal.

Tabela 3.1: Configuração dos servidores

	Total de CPU's	Modelo	Memória RAM (GB)
crater.na-df.rnp.br	8	Intel(R) Xeon(R) CPU E7-4807 @ 1.87GHz	4
mconf-live-pa01,rnp.br	4	Intel(R) Xeon(R) CPU E5620 @ 2.40GHz	4
mconf-live-es01,rnp.br	4	Intel(R) Xeon(R) CPU E5620 @ 2.40GHz	4
mconf-live-rn01,rnp.br	4	Intel(R) Xeon(R) CPU E5620 @ 2.40GHz	4
mconf-live3.rnp.br	4	Intel(R) Xeon(R) CPU E5410 @ 2.33GHz	4
mconf-live-rs01,rnp.br	4	Intel(R) Xeon(R) CPU E5410 @ 2.33GHz	4
mconf-live-pi01,rnp.br	4	Intel(R) Xeon(R) CPU E5620 @ 2.40GHz	4
mconf-live2.rnp.br	4	AMD Opteron(tm) Processor 6136	4

Figura 3.8: Distribuição aproximada dos servidores pelo Brasil

4 RESULTADOS

Nesse capítulo serão apresentados alguns resultados e conclusões sobre os dados obtidos. Estes dados foram adquiridos aplicando a metodologia de experimentação no ambiente descrito no capítulo 3. O capítulo está dividido em duas seções que abordarão brevemente os resultados obtidos. Na primeira, serão mostrados os dados gerados pelo *Medidor* através das métricas comentadas em seções anteriores e, na segunda, serão apresentados os resultados obtidos pelo *Otimizador* citado na subseção 3.1.3.

4.1 Métricas

Nesta subseção serão mostrados os dados obtidos pelo *Medidor* com as métricas citadas na seção 2.2 aplicadas em um conjunto de servidores apresentados na seção 3.2.1. A seguir serão apresentados os resultados obtidos com as métricas de latência, *jitter*, vazão e rota. Os valores apresentados nessa seção são uma média aritmética sobre os valores obtidos com as cinco repetições dos experimentos citados na seção 3.2.

4.1.1 Latência

Nesta subseção será mostrada a latência nas conexões entre os servidores. Na Tabela 4.1, pode-se observar os dados de latência criados pelo *Medidor*. A latência pode ser influenciada por várias variáveis como o uso do servidor e as filas nos roteadores intermediários, por exemplo. Portanto, procurou-se coletar esses dados enquanto não existia nenhum usuário utilizando os servidores. Apesar dessa situação não condizer com a situação real.

Tabela 4.1: latência entre os servidores (ms)

	DF	*PA*	*ES*	*RN*	*RJ*	*RS1*	*PI*	*RS2*
DF	X	114,9	30,2	83,3	20,2	49,4	76,3	48,3
PA	102,8	X	101,1	32,1	110,6	93,9	15,9	89,3
ES	31,2	100,9	X	84,8	9,9	38,5	45,8	29,4
RN	84,2	32,2	84,8	X	94,2	77,9	26,3	77,6
RJ	19,1	95,4	9,9	63,8	X	29,8	55,6	34,4
RS1	47,5	93,1	39,6	76,6	28,7	X	81,4	7,1
PI	77,3	15,9	45,8	25,9	55,5	78,7	X	82,8
RS2	49,3	92,1	30,2	80,2	32,4	10,1	83,2	X

Na Tabela 4.1 a linha i e coluna j, representa a latência na conexão enviando dados do servidor i até o servidor j. Os valores apresentados na tabela correspondem à média

aritmética entre as cinco medições realizadas. Na Figura 4.1, pode-se observar a latência entre o servidor *DF* e todos os outros servidores. Os dados de latência possuem uma relativa variação de um servidor para outro, por exemplo, o servidor *ES*, localizado no Espírito Santo (*mconf-live-es01.rnp.br*), e *RJ*, localizado no Rio de Janeiro (*mconf-live3.rnp.br*), possuem uma latência de 9,9 ms. Em contraste com o exemplo anterior a latência entre *PA* e *RJ* é de 110 ms.

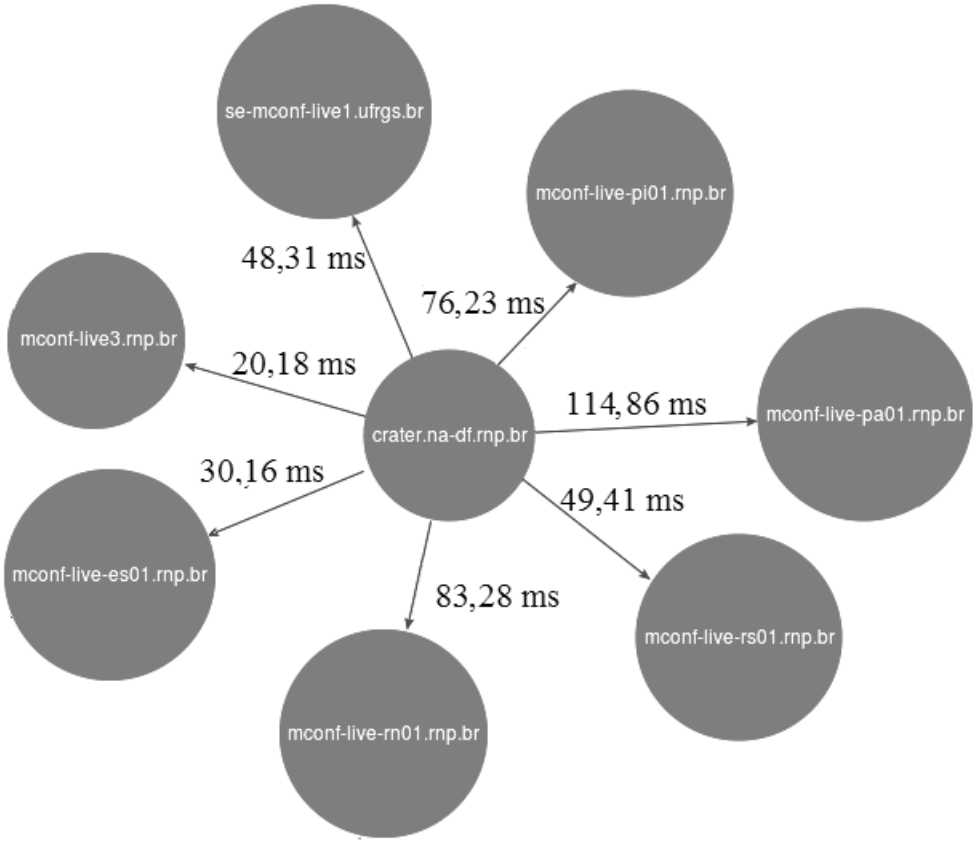


Figura 4.1: Dados de latência entre *DF* até os outros servidores

4.1.2 Jitter

Nesta subseção será mostrado o *jitter* nas conexões entre os servidores. Na Tabela 4.2 se pode observar os dados de *jitter* obtidos pelo *Medidor*. O *jitter*, assim como a latência, pode ser influenciado por vários fatores, sendo um deles o uso do servidor, conforme citado anteriormente. Portanto, procurou-se realizar as medições enquanto não existia nenhum usuário utilizando o servidor. Consequentemente, com essa ação se buscou diminuir os erros na medição e aumentar a precisão dos dados obtidos.

Na Tabela 4.2 a linha i e coluna j, representa o *jitter* médio na conexão envolvendo o envio de dados do servidor i até o servidor j. Os valores apresentados na tabela correspondem à média aritmética entre as cinco medições realizadas. A visualização completa dos dados pode ser feita através do módulo *Visualizador* citado na subseção 3.1.2.

Com base nos dados apresentados na tabela é possível concluir que o valor do *jitter* possui uma grande variação entre os servidores.

Tabela 4.2: *jitter* entre os servidores (ms)

	DF	*PA*	*ES*	*RN*	*RJ*	*RS1*	*PI*	*RS2*
DF	X	0,68	0,69	0,67	0,68	0,70	0,68	0,67
PA	0,68	X	0,12	0,48	0,27	0,14	0,13	0,13
ES	0,72	0,13	X	0,36	0,26	0,17	0,16	0,21
RN	0,66	0,43	0,37	X	0,45	0,40	0,37	0,42
RJ	0,63	0,11	0,12	0,35	X	0,24	0,11	0,32
RS1	0,67	0,14	0,11	0,39	0,12	X	0,12	0,12
PI	0,64	0,08	0,15	0,33	0,26	0,17	X	0,13
RS2	0,68	0,13	0,21	0,41	0,32	0,12	0,13	X

4.1.3 Vazão

Nesta subseção será mostrada a vazão nas conexões entre os servidores. Na Tabela 4.3 se pode observar os dados da vazão que foram fornecidos pelo *Medidor*. A técnica empregada para calcular a vazão pode sobrecarregar a rede, conforme explicado na seção 2.2, portanto a medição deve ser feita esparsamente e, preferencialmente, quando o servidor não estiver em uso, caso contrário ocorrerá uma queda na velocidade e na estabilidade da conexão dos outros usuários conectados ao servidor.

Na Tabela 4.3, a linha i e coluna j, representa a vazão média da conexão envolvendo o envio de dados do servidor i até o servidor j. Os valores apresentados na tabela correspondem à média aritmética entre as cinco medições realizadas. Pode-se observar que a vazão entre os servidores é bem variada, levando a fatos interessantes. Uma situação diferente observada, envolvendo os servidores *RJ*, *RS1* e *PI*, foi que a vazão direta entre *RS1* e *PI* era inferior à vazão indireta, consistindo no envio de dados de *RS1* até *RJ*, que eram então retransmitidos até *PI*. A vazão entre os servidores é:

- *RS1* até *PI* é 40,35 Mbit/s
- *RS1* até *RJ* é 414,51 Mbit/s
- *RJ* até *PI* é 87,83 Mbit/s

Tabela 4.3: vazão entre os servidores (Mbit/s)

	DF	*PA*	*ES*	*RN*	*RJ*	*RS1*	*PI*	*RS2*
DF	X	58,18	45,17	53,92	17,74	399,69	84,28	395,65
PA	57,13	X	123,66	10,97	87,85	133,79	18,95	90,19
ES	47,13	90,52	X	44,73	35,42	12,53	140,24	20,68
RN	52,93	96,87	205,67	X	117,74	96,26	21,42	54,70
RJ	18,73	53,38	30,87	297,82	X	31,68	87,83	412,29
RS1	395,67	90,13	18,68	57,57	414,51	X	40,35	421,24
PI	83,27	18,37	93,76	47,96	187,58	224,89	X	40,58
RS2	397,64	91,13	19,67	55,73	413,29	439,30	39,29	X

A transferência direta de dados entre *RS1* até *PI* é limitada a 40 Mbit/s. Porém, utilizando o servidor *RJ* como retransmissor a vazão é ampliada para, aproximadamente, 87 Mbit/s. Consequentemente, deve-se escolher entre dois caminhos:

1. Caminho direto entre *RS1* e *PI*, caminho representado pela seta pontilhada na Figura 4.2, com vazão de 40 Mbit/s;

2. Caminho indireto entre *RS1* e *PI*, através de *RJ*, representado pelas setas cheias na Figura 4.2, com vazão de 80 Mbit/s.

Considerando somente a vazão, fica claro que o caminho indireto é melhor neste caso. Isso é uma das vantagens fornecida pelo *Otimizador*.

Uma situação hipotética pode ser criada com as informações apresentadas anteriormente. O contexto da situação criada é:

1. O sistema representa uma videoconferência onde somente um usuário no servidor *RS1* envia um fluxo de vídeo do *webcam* e um fluxo de áudio.

2. Cada usuário utiliza 1 Mbit/s de banda para receber os dados de controle e de vídeo.

3. Os usuários não enviam dados para o servidor, exceto o apresentador citado no item 1.

4. A latência pode ser desconsiderada.

5. O sistema não possui nenhuma otimização.

6. Inicialmente existem 40 usuários no Piauí e 5 usuários no Rio Grande do Sul.

7. A capacidade de processamento de cada servidor suporta 100 usuários simultaneamente.

Com essas informações pode-se ilustrar uma situação hipotética onde todos os usuários estão conectados no servidor *RS1* conforme mostrado na Figura 4.3a. Em um segundo momento, a qualidade do serviço provido aos usuários no Piauí começa a degradar, gerando falhas na exibição do vídeo. Assim, o sistema deduz que, para essa situação, a melhor solução é compartilhar a conferência com o servidor *PI* conforme mostrado na Figura 4.3b.

Em um terceiro momento, ocorre a entrada de mais 10 usuários no Piauí e, consequentemente, tem-se uma nova degradação no serviço, pois a banda total dos usuários excede o limite de 40 Mbit/s. A melhor solução é adotar a configuração apresentada na Figura 4.3c. Agora o servidor *RJ* será um retransmissor de dados entre *RS1* e *PI*.

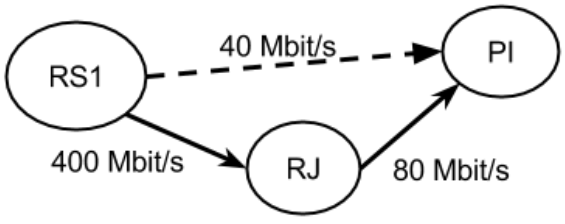
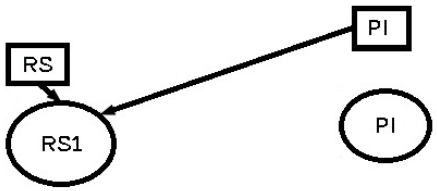
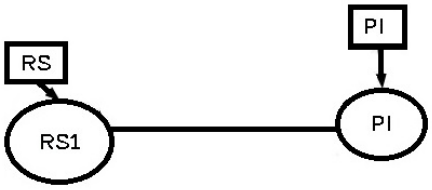


Figura 4.2: Exemplo com servidor intermediário entre *RS1* e *PI*

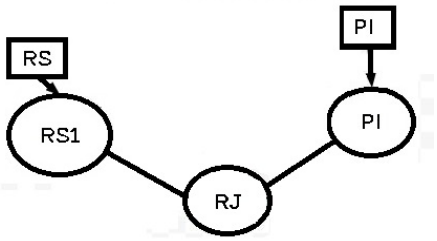
No exemplo citado, mostra-se a importância de conhecer a vazão entre servidores. Essas mudanças podem ser automatizadas e transparentes ao usuário. Elas ainda agregam mais qualidade ao serviço.



(a) Todos os clientes conectados no servidor *RS1*



(b) Clientes conectados em *RS1* e *PI*, com web conferência compartilhada entre os servidores



(c) Clientes conectados em *RS1* e *PI*, com web conferência compartilhada entre os servidores porém utilizando *RJ* como retransmissor

Figura 4.3: Exemplo com diversos clientes conectados em diversos servidores

4.1.4 Rotas

Nesta subseção é feita a descoberta das rotas entre os servidores citados na seção 3.2.1 utilizando o *Medidor*. Os roteadores intermediários, presentes nas rotas, foram abreviados, quando possível. Isso foi feito com o intuito de gerar uma tabela mais compreensível por agregar a localização espacial dos mesmos. Nem todos os nomes dos roteadores foram alterados, sendo assim, nem sempre o nome do roteador envolverá, somente, os estados onde eles estão localizados.

Na Tabela 4.4 são apresentadas as rotas a partir do servidor localizado no Pará (servidor *PA*) para os outros servidores. Essa tabela foi gerada com base nos dados fornecidos pelo *Medidor*. Na Figura 4.4, é possível observar a rota a partir do servidor *PA* até *PI*. A figura pertence ao *Visualizador*, sendo que os servidores são representados por círculos preenchidos e a outra forma representa os roteadores. Foram omitidos alguns roteadores na rota, sendo que a parte omitida está representada por uma seta pontilhada.

Uma vantagem do conhecimento das rotas entre os servidores é a possibilidade de criação de rotas alternativas entre servidores, que aumentam a sua disponibilidade conforme foi citado na seção 1.1.2.

A fim de exemplificar essa vantagem será estuda uma situação onde existe uma videoconferência compartilhada entre dois servidores *RN* e *PA*. O sistema possui duas rotas: a principal, que é a comunicação direta entre os servidores, e uma secundária, que usa o servidor *RJ* como um retransmissor de informações. Na Tabela 4.5 é mostrada a rota entre *PA* e *RN*, que está ilustrada na Figura 4.5.

Em um segundo momento, ocorre uma falha entre o roteador da Paraíba e o do Rio Grande do Norte. Contudo, é conhecida uma rota alternativa que não passa pelo ponto de falha, portanto a falha não afetará o usuário, já que o sistema poderá utilizar a rota alternativa.

Tabela 4.4: Rota entre o servidor *PA* para outros servidores

DF	*ES*	*RN*	*RJ*	*RS1*	*PI*	*RS2*
POP-PA	POP-PA	POP-PA	POP-PA	POP-PA	POP-PA	POP-PA
mxpa-lanpa	mxpa-lanpa	mxpa-lanpa	mxpa-lanpa	mxpa-lanpa	mxpa-lanpa	mxpa-lanpa
PA-PI	PA-PI	PA-PI	PA-PI	PA-PI	PA-PI	PA-PI
PI-PE	PI-PE	PI-PE	PI-PE	PI-PE	PI-PE	PI-PE
PE-AL	PE-AL	PE-PBCGE	PE-AL	PE-AL	lanpi-mxpi	PE-AL
AL-SE	Al-SE	PBCGE-PBJPA	AL-SE	AL-SE		AL-SE
SE-BA	SE-BA	PBJPA-RN	SE-BA	SE-BA		SE-BA
BA-MG	BA-ES	lanrn-mxrn	BA-ES	BA-MG		BA-ES
MG-DF	lanes-mxes	rt1-pop-rn	ES-RJ	MG-SP		ES-RJ
landf-mxdf			lanrj-mxrj	SP-PR		RJ-SP
rt-pop-df				PR-RS		SP-PR
				200,143,255,162		PR-RS
						200,143,255,162
						ufrgs-ve
						lfs-in-ufrgs

Tabela 4.5: Rota entre *PA* e *RN*

POP-PA
mxpa-lanpa
PA-PI
PI-PE
PE-PBCGE
PBCGE-PBJPA
PBJPA-RN
lanrn-mxrn

Nas tabelas 4.6 e 4.7 são apresentadas as rotas entre *PA* até *RJ* e *RJ* até *RN*, respectivamente.

Tabela 4.6: Rota entre *PA* e *RJ*

POP-PA
mxpa-lanpa
PA-PI
PI-PE
PE-AL
AL-SE
SE-BA
BA-ES
ES-RJ
lanrj-mxrj

Assim, mostra-se uma vantagem de considerar as rotas entre os servidores para geração de caminhos alternativos a fim de aumentar a disponibilidade do serviço. Outro fator que deve ser considerado é o grau de disjunção das rotas, isto é, o quanto elas diferem uma da outra com relação aos roteadores envolvidos nas conexões.

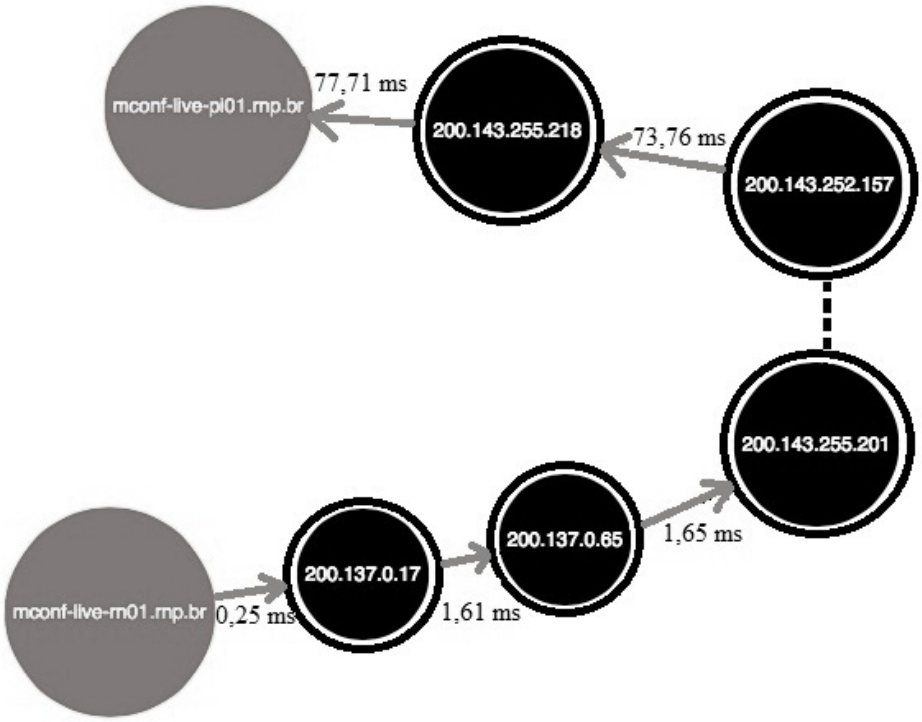


Figura 4.4: Rota entre o servidor *PA* até *PI*

Tabela 4.7: Rota entre *RJ* e *RN*

POP-RJ
mxrj-lanrj
RJ-DF
DF-MG
MG-CE
CE-RN
lanrn-mxrn
POP-RN

4.2 Otimizador

Nesta seção serão apresentados os resultados fornecidos pelo *Otimizador*, criado para analisar os dados fornecidos pelo *Medidor* e assim gerar rotas que priorizem alguma métrica. Um fato observado é que nem sempre a conexão direta com um servidor fornecerá a menor latência ou a maior vazão. Por isso, é necessário estudar a rede a fim de melhor otimizá-la e fornecer um serviço de melhor qualidade.

Para manter a seção sucinta não serão mostrados todos os resultados para todos os servidores, mas um subconjunto de exemplos relevantes que exemplificam os benefícios deste trabalho.

4.2.1 Maior vazão

A maior vazão nem sempre estará em uma conexão direta entre os servidores, já que os enlaces entre os servidores não são simétricos. Por exemplo, na Tabela 4.8 é possível observar a vazão a partir do servidor *ES* até todos os outros servidores, passando por alguns servidores intermediários, cuja função é retransmitir os pacotes recebidos.

Figura 4.5: Rota entre o servidor *PA* até *RN*

Tabela 4.8: Maior vazão entre o servidor *ES* para todos os outros servidores da rede

Destino	vazão	Retransmissores
DF	83,27 Mbit/s	*PI*
PA	90,51 Mbit/s	Nenhum
RN	140,24 Mbit/s	*PI -> RJ*
RJ	140,24 Mbit/s	*PI*
RS	140,24 Mbit/s	*PI*
PI	140,24 Mbit/s	Nenhum

Analisando o caminho na Tabela 4.8 com o maior número de retransmissores *ES* até *RN* é possível observar que a vazão fica limitada pela conexão com menor vazão. Ainda se sabe através da Tabela 4.3 presente na subseção 4.1.3 que a vazão direta entre *ES* até *RN* era de 44,73 Mbit/s. Assim, fica claro o benefício de conhecer a vazão entre os servidores.

Pode-se, assim, através do *Otimizador* obter a rota com maior vazão entre quaisquer dois servidores.

4.2.2 Menor latência

A menor latência, assim como a menor vazão, nem sempre estará em uma conexão direta entre os servidores. É possível observar que a camada de rede nem sempre fornecerá o melhor caminho para uma determinada localização e, tampouco, que qualidade da conexão fornecida será a melhor possível. Por exemplo, na Tabela 4.9 pode-se observar que a latência a partir do servidor *ES* até todos os outros servidores.

Assim, na Tabela 4.9 o caminho entre *ES* até *RN* diretamente possui latência de 84,75 ms (obtido a partir da Tabela 4.1). Porém, com o auxílio do *Otimizador*, o melhor caminho gerado, priorizando a latência, é utilizando o servidor *PI* como retransmissor, já que ele apresenta latência de 71,78 ms.

Figura 4.6: Rota alternativa entre o servidor *PA* até *RN* utilizando servidor *RJ* como retransmissor

Tabela 4.9: Menor latência entre o servidor *ES* para todos os outros servidores da rede

Destino	latência	Retransmissores
DF	49 ms	*RS2*
PA	93,69 ms	Nenhum
RN	76,16 ms	Nenhum
RJ	28,87 ms	*RS2*
RS1	1,16 ms	Nenhum
PI	78,99 ms	*RS2*

Usando essa rota, como alternativa à direta, ocorre uma diminuição em 10 ms na latência.

4.2.3 Menor *jitter*

Assim como a latência, é possível observar a característica de que o menor *jitter* nem sempre estará na conexão direta entre os servidores. Portanto, é imprescindível o estudo das métricas para gerar as melhores rotas.

4.2.4 Rota alternativa

Essa otimização é calculada de duas maneiras. Primeiro, será mostrado o resultado utilizando o aumento dos pesos do caminho anterior e, depois, a rota calculada usando a segunda técnica que envolve a remoção das conexões. Para isso, será adotado o caminho inicial entre os servidores *ES* até *PI*.

4.2.4.1 Rota com aumento dos pesos

Nessa otimização foi fornecida a rota com a menor latência entre *ES* até *PI*, sendo que essa rota não possuía nenhum servidor intermediário. Inicialmente, ela possuía uma latência de 45,8 ms. Para obter uma rota alternativa, foi aumentado o valor da latência entre *ES* até *PI* para o maior valor possível. Consequentemente, este caminho será ignorado. Logo, o *Otimizador* retornará o segundo melhor caminho, já que o primeiro foi excluído do cálculo com a manipulação do peso das conexões do mesmo. O caminho alternativo criado é *ES->RJ->PI* com latência de 65,7 ms. Portanto, os dados de *ES* serão transmitidos até *RJ*, que os retransmitirá até o destino final (*PI*).

4.2.4.2 Rota com redução de conexões e servidores

Nessa otimização, assim como a da subseção anterior, foi fornecida a rota com menor latência entre *ES* até *PI*. Inicialmente, a rota possuía latência de 45,8 ms. São removidas todas as conexões que possuem alguns dos roteadores presentes nessa rota, desde que a remoção deles ainda permita um caminho entre *ES* até *PI*. Então, o caminho retornado será: *ES->RJ->PI* com uma latência de 65,7 ms. O fato de ser o mesmo resultado é devido à pouca quantidade de servidores na rede, fazendo com que existam poucos caminhos alternativos e, portanto, muitos compartilhamentos de roteadores entre os servidores.

5 CONCLUSÃO

Este trabalho teve por objetivo a criação de uma ferramenta para geração de rotas que priorizem as métricas de latência, *jitter*, vazão. Ela também fornece rotas alternativas que buscam ser o mais disjuntas possível com relação às rotas principais em nível de rede. Essa ferramenta procura facilitar a escolha dos servidores que devem ser utilizados para distribuição de uma conferência e criar rotas alternativas entre os servidores. Os caminhos criados pela ferramenta tentam ser superiores às rotas diretas entre os servidores.

Para minimizar o problema da escalabilidade é necessário distribuir as conferências em mais de um servidor, conforme comentado no capítulo 1, observa-se que as conferências distribuídas são mais escaláveis quando comparadas as conferências centralizadas. Portanto, torna-se necessário escolher qual dos servidores faram parte da conferência distribuída. A ferramenta proposta neste trabalho oferece rotas otimizadas de acordo com uma das métricas, para videoconferências, por exemplo, pode-se buscar os servidores, cujas rotas apresentem a menor latência a partir do servidor central da conferência. Com isso, a ferramenta auxilia na resolução do problema da escolha de quais servidores integraram a conferência. Portanto, responde-se a primeira pergunta ("Como escolher os melhores servidores para minimizar o problema da escalabilidade?"), sendo que ela foi explicitada na seção 1.1.

Para minimizar o problema da disponibilidade, uma solução proposta por este trabalho foi a distribuição da videoconferência em mais de um servidor e a criação de rotas alternativas entre os servidores, sendo que essa solução foi comentada no capítulo 1. A ferramenta criada serve como apoio à na minimização do problema da disponibilidade, devido à capacidade da ferramenta de prover a criação de rotas alternativas, que buscam utilizar diferentes caminhos de rede. Com as rotas criadas é possível, por exemplo, aumentar a disponibilidade em uma videoconferência. Com isso, responde-se a segunda pergunta ("Como criar rotas alternativas entre dois servidores?"), que foi explicitada na seção 1.1.

No capítulo 4, que apresenta os resultados dos experimentos, foi observado que as transmissões diretas entre os servidores nem sempre apresentarão a menor latência e nem a maior vazão. Um caso específico tratado, que é apresentado na subseção 4.1.3, consiste na transmissão de dados entre *RS1* e *PI*, que apresenta vazão direta de 40 Mbit/s. Já, se for utilizado o servidor *RJ* como retransmissor dos dados entre os servidores, a vazão será de, aproximadamente, 80 Mbit/s. A ferramenta forneceu o caminho indireto entre os servidores, utilizando *RJ* como retransmissor, quando foi requisitada a rota com maior vazão entre os servidores *RS1* e *PI*. Pode-se observar outros resultados similares quando são requisitadas rotas que priorizem a menor latência ou o menor *jitter* no capítulo 4.

Foi observado, no capítulo 4, que para a melhor utilização das rotas alternativas geradas é necessário que os servidores estejam localizados em redes diferentes, a fim de

diminuir o compartilhamento de roteadores entre os servidores. Um resultado observado, foi que a geração de rotas alternativas não garantirá cobertura total as falhas de redes, mas sim, minimizará o número de falhas observadas pelos usuários.

Analisando os resultados obtidos ficou-se clara a importância da ferramenta, já que ela possibilita a criação de caminhos com maior apoio à decisão, podendo auxiliar na minimização dos problemas de baixa disponibilidade e baixa escalabilidade.

Existe espaço para expansão dos assuntos tratados neste trabalho. Uma das melhorias seria avaliar um conjunto maior de servidores na realização dos experimentos, além de buscar métricas para substituir a técnica de medição ativa da vazão. Uma técnica menos intrusiva, que poderia substituir a medição ativa, seria a criação de um *bot*, que simula o comportamento de um usuário comum. Com ele, seria possível calcular o custo de um cliente e, também, avaliar a qualidade em tempo real observada pelo usuário.

Constatou-se a possibilidade de trabalhos futuros baseados neste estudo, sendo um deles a criação de uma rede *overlay* otimizada para videoconferência. Essa rede utilizaria a inteligência fornecida pelo *Otimizador*, cuja funcionalidade seria a criação das rotas para essa rede.

REFERÊNCIAS

APPLE. **Facetime**. [Online; acessado em 2-Novembro-2013], http://www.apple.com/mac/facetime/.

ARBOR.JS. **Arbor.js**. [Online; acessado em 15-Novembro-2013], http://arborjs.org.

BANERJEE, S.; BHATTACHARJEE, B.; KOMMAREDDY, C. Scalable Application Layer Multicast. **SIGCOMM Comput. Commun. Rev.**, New York, NY, USA, v.32, n.4, p.205–217, Aug. 2002.

BASET, S. A.; SCHULZRINNE, H. An analysis of the skype peer-to-peer internet telephony protocol. **arXiv preprint cs/0412017**, [S.l.], 2004.

BLANCHER, C. About good usage of traceroute. In: MULTISYSTEM AND INTERNET SECURITY COOKBOOK. **Anais...** [S.l.: s.n.], 2003. p.62–65.

COULOURIS, G. et al. **Distributed Systems - Concepts and Design**. 5th.ed. [S.l.]: Addison-Wesley, 2011.

DIJKSTRA, E. A note on two problems in connexion with graphs. **Numerische Mathematik**, [S.l.], v.1, n.1, p.269–271, 1959.

FINK, B. **Nuttcp**. [Online; acessado em 2-Novembro-2013], http://www.lcp.nrl.navy.mil/nuttcp/nuttcp.html.

GOLDBERG, A. V.; HARRELSON, C. Computing the shortest path: a search meets graph theory. In: ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS. **Proceedings...** [S.l.: s.n.], 2005. p.156–165.

GUOJUN, J. **Available Bandwidth Measurement and Sampling**. 2003.

ITO, K. **Video conference system**. US Patent 5,745,161.

ITU. **Y.1540** : internet protocol data communication service â ip packet transfer and availability performance parameters. [Online; acessado em 2-Novembro-2013], http://www.itu.int/rec/T-REC-Y.1540-201103-I/en.

JIN, G.; TIERNEY, B. L. System Capability Effects on Algorithms for Network Bandwidth Measurement. In: ACM SIGCOMM CONFERENCE ON INTERNET MEASUREMENT, 3., New York, NY, USA. **Proceedings...** ACM, 2003. p.27–38. (IMC '03).

KESHAV, S. Experiences with Large Videoconferences on XUNET II. In: ACM. **Anais...** [S.l.: s.n.], 1994.

LE, T.; NGUYEN, H. Human perception-based distributed architecture for scalable video conferencing services: theoretical models and performance. **annals of telecommunications - annales des telecommunications**, [S.l.], p.1–11, 2013.

LINUX. **Traceroute - Linux Man Page**. [Online; acessado em 1-Novembro-2013], `http://linux.die.net/man/8/traceroute`.

LIU, Y.; GUO, Y.; LIANG, C. A survey on peer-to-peer video streaming systems. **Peer-to-Peer Networking and Applications**, [S.l.], v.1, n.1, p.18–28, 2008.

LUO, C. et al. A Multiparty Videoconferencing System Over an Application-Level Multicast Protocol. **Multimedia, IEEE Transactions on**, [S.l.], v.9, n.8, p.1621–1632, 2007.

MALPANI, N.; CHEN, J. A Note on Practical Construction of Maximum Bandwidth Paths. **Inf. Process. Lett.**, Amsterdam, The Netherlands, The Netherlands, v.83, n.3, p.175–180, Aug. 2002.

MCONF. **Mconf**. [Online; acessado em 1-Novembro-2013], `http://mconf.org`.

MICROSOFT. **What is Skype?** [Online; acessado em 16-Dezembro-2013], `http://www.skype.com/en/what-is-skype/`.

PADMANABHAN, V. N.; MOGUL, J. C. Improving HTTP latency. **Computer Networks and ISDN Systems**, [S.l.], v.28, n.1, p.25–35, 1995.

ROESLER, V. et al. Mconf: an open source multiconference system for web and mobile devices. In: **Multimedia / Book 2**. [S.l.]: Intech: Open Access Publisher, 2012.

ROESLER, V. et al. Mconf: collaboration proposal to form a global infrastructure for web conferencing based on open source. In: ASIA-PACIFIC ADVANCED NETWORK. **Anais...** [S.l.: s.n.], 2013. v.35, p.28–32.

TANENBAUM, A. S.; WETHERALL, D. J. **Computer Networks**. 5th.ed. [S.l.]: Prentice Hall, 2011.

TIERNEY, B. et al. Efficient data transfer protocols for big data. In: E-SCIENCE (E-SCIENCE), 2012 IEEE 8TH INTERNATIONAL CONFERENCE ON. **Anais...** [S.l.: s.n.], 2012. p.1–9.

TURLETTI, T.; HUITEMA, C. Videoconferencing on the Internet. **IEEE/ACM Trans. Netw.**, Piscataway, NJ, USA, v.4, n.3, p.340–351, June 1996.

TWITCH. **Twitch TV for Gamers**. [Online; acessado em 1-Novembro-2013], `http://www.twitch.tv/p/about`.

WANG, Z.; CROWCROFT, J. Analysis of shortest-path routing algorithms in a dynamic network environment. **ACM SIGCOMM Computer Communication Review**, [S.l.], v.22, n.2, p.63–71, 1992.

ZHANG, X. et al. CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming. In: INFOCOM 2005. 24TH ANNUAL JOINT CONFERENCE OF THE IEEE COMPUTER AND COMMUNICATIONS SOCIETIES. PROCEEDINGS IEEE. **Anais...** [S.l.: s.n.], 2005. v.3, p.2102–2111 vol. 3.