Renan Manola

Análise de Desempenho no Uso de Pré-busca para Distribuição de Vídeo sobre Redes P2P

Vitória - ES

5 de setembro de 2011

Renan Manola

Análise de Desempenho no Uso de Pré-busca para Distribuição de Vídeo sobre Redes P2P

Dissertação apresentada para obtenção do Grau de Mestre em Informática pela Universidade Federal do Espírito Santo.

Orientador:
Prof. Dr. Magnos Martinello

Co-orientador:
Profa. Dra. Roberta Lima Gomes

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO - UFES
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA

Vitória - ES

5 de setembro de 2011

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Manola, Renan, 1986-
M285a Análise de desempenho no uso de pré-busca para distribuição de vídeo sobre redes P2P / Renan Manola. – 2011.
83 f. : il.

Orientador: Magnos Martinello.
Coorientadora: Roberta Lima Gomes.
Dissertação (Mestrado em Informática) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Gravações de vídeo. 2. P2P. I. Martinello, Magnos. II.Gomes, Roberta Lima. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 004

Dissertação de Mestrado sob o título *"Análise de Desempenho no Uso de Pré-busca para Distribuição de Vídeo sobre Redes P2P "*, defendida por Renan Manola e aprovada em 5 de setembro de 2011, em Vitória, Estado do Espírito Santo, pela banca examinadora constituída pelos professores:

Prof. Dr. Magnos Martinello
Universidade Federal do Espírito Santo
Orientador

Profa. Dra. Roberta Lima Gomes
Universidade Federal do Espírito Santo
Co-orientadora

Prof. Dr. José Gonçalves Pereira Filho
Universidade Federal do Espírito Santo

Prof. Dr. Cesar Marcondes
Universidade Federal de São Carlos

Prof. Dr. Luis Carlos Erpen de Bona
Universidade Federal do Paraná

Dedicatória

Aos meus Pais, pela compreensão incalculável.
À Darlene, pelo amor constante em minha vida.
Às minhas avós Martha e Maria, pelo enorme afeto.

Agradecimentos

A elaboração de uma dissertação de mestrado é uma tarefa árdua e complexa que não só exige do autor grande esforço e dedicação, mas também o suporte incalculável de várias outras pessoas. Acredito que o mestrado consista em um processo de aprimoramento intelectual e, principalmente, pessoal. Dessa forma, acredito que cresci muito como pessoa neste período de dois anos e meio que compreendeu tantos acontecimentos e decisões importantes em minha vida.

Gostaria de agradecer inicialmente ao meu orientador Magnos por confiar no meu potencial, me aceitar como aluno de mestrado e me ajudar bastante na produção e revisão desta dissertação. À minha co-orientadora Roberta pelas opiniões pertinentes e direcionamento sobre o foco deste trabalho. Sou muito grato ao Cesar pela ajuda fundamental durante o início do mestrado, pelas ideias, pela colaboração no artigo e por ter me ajudado no aprendizado do simulador de rede. Agradeço ao pessoal do LPRM pela companhia durante o tempo em que trabalhei lá, em especial, ao Maycon pelas conversas frutíferas sobre vulnerabilidades e ataques em sistemas Web e ao João por ter me ajudado com o empréstimo de seu cluster e ter me ensinado como instalar um do zero. Sou grato, também, ao pessoal do Suporte pela consideração em ter cedido as máquinas do LabGrad para construção do cluster que rodou as simulações deste trabalho.

Agradeço também aos meus alunos da disciplina que lecionei em 2010/1, pela paciência e compreensão nas aventuras e um professor de primeira viagem. Agradeço ao pessoal da pós-graduação, em especial, Rogelio e Anderson pelos momentos de descontração e seriedade quando era necessário. Também agradeço ao pessoal do Prodest: Calvin, Uanderson, Wagner, Filipe, Zanol, Lívio e demais por proporcionarem àquele lugar um ambiente, ao mesmo tempo, descontraído e produtivo. Trabalhar lá foi um grande prazer e me ensinou bastante.

Por fim, e não menos importante, agradeço aos meus pais por sempre acreditarem em mim e pelo suporte financeiro durante todo este tempo. Agradeço também à minha namorada Darlene pelo suporte emocional, amor incondicional e grande compreensão durante os momentos que exigiram maior dedicação à este trabalho.

Resumo

Sistemas Par-a-Par de Video Sob Demanda consolidaram sua importância nos últimos anos. Tal consolidação é decorrente dos elevados ganhos que se pode ter em termos de economia na banda de transmissão dos servidores. Neste trabalho, avalia-se como esta tecnologia pode potencializar os ganhos de transmissão de mídia contínua na Internet. Para atingir este objetivo, a metodologia adotada apoia-se em um ambiente de simulação a nível de pacotes que possui características que permitem capturar diferentes variáveis referentes a dinâmica da Internet, tais como: simulação completa da pilha TCP/IP, considerações sobre topologia de rede e inserção de tráfego de fundo. Na simulação foi implementado um algoritmo de difusão de vídeo P2P VoD e um algoritmo de pré-busca. As análises dos resultados apontam para uma potencial economia de *upload* no servidor de conteúdo de 43% em um cenário menos idealizado do que a proposta de base que atingiu 73%. Em termos da percepção da qualidade de vídeo dos clientes, o presente trabalho indica que sua experiência pode ser degradada de forma considerável usando UDP quando existe o tráfego de fundo, mesmo com a economia do servidor se mantendo constante. Ainda na percepção dos clientes, percebeu-se também que, embora o uso da pré-busca seja benéfico aos mesmos, o grande fator diferencial na melhor qualidade percebida foi a escolha da camada de transporte apropriada.

Abstract

Peer-to-Peer Video on Demand Systems have well stabilished its importance in recent years. Such consolidation is due to the high gains that may be archieved in terms savings in transmission bandwidth of the servers. In this work, we evaluate how this technology can enhance the gains for the transmission of continuous media over the Internet. To achieve this goal, the methodology relies on a packet-level simulation that has the ability to allow capturing different variables that regard the dynamics of the Internet, such as: full simulation of the TCP/IP stack, considerations on network topology and insertion of background traffic.

In this simulation was implemented a video streaming P2P VoD algorithm and an algorithm that uses pre-fetching. Result analisys point out to a potential economy of the content server's upload bandwidth of up to 43% in a less idealized scenario than the results obtained on the base proposal that has archieved an economy of 73%. In terms of user's perceived video quality, this work indicates that the user's experience can be greatly degraded when using the UDP with background traffic, even with the economy of the server remaining constant. Still on the user's video quality perception, it was noted that, although the use of pre-fetching was proved beneficial to them, the key factor in best quality perceived was the choice of a proper transport layer.

Sumário

Lista de Figuras

Lista de Siglas e Abreviaturas

P2P	Peer-to-Peer
CDN	Content Distribution Network
QoS	Quality of Service
VoD	Video on Demand
P2P VoD	Peer-assisted Video on Demand
DiffServ	Differentiated Services
ISPs	Internet Service Providers
TOS	Type of Service
ECN	Exlpicit Congestion Notification
IntServ	Integrated Services
RSVP	Resource Reservation Protocol
UDP	User Datagram Protocol
RUDP	Reliable User Datagram Protocol
SCTP	Stream Control Transmission Protocol
DCCP	Datagram Congestion Control Protocol
HOL	Head of Line
Voip	Voice over IP
IGMP	Internet Group Message Protocol
ALM	Application-Level Multicast
SSM	Source Specific Multicast
RTT	Round Trip Time
PTTs	Ponto de Troca de Tráfego
DHT	Distributed Hash Tables

VCR	vídeo Cassette Recorder
NVoD	Near vídeo-on-Demand
TvoD	True vídeo-on-Demand
IVoD	Interactive vídeo-on-Demand
ATD	Availability to Demand Ratio
MDC	Multiple Descriptor Coding
DHT	Distributed Hash Tables
NAM	The Network Animator
RTP	Real Time Protocol

1 Introdução

A Internet está assumindo um papel cada vez maior na vida das pessoas. Uma pesquisa recente (BANCO..., 2011) indica que a porcentagem de usuários de Internet no mundo quase dobrou em 5 anos (de 2004 para 2009). As redes par-a-par (P2P) surgem como um modelo promissor na Internet por fazerem uso de recursos (capacidades de rede, armazenamento e processamento) que os próprios clientes possuem. Devido a esta característica, pode-se afirmar que tais redes fizeram com que os clientes também se tornassem servidores. As redes P2P começaram a popularizar-se por meio de aplicações voltadas ao compartilhamento de arquivos. Programas como Napster[1], Audiogalaxy[2] e KaZaA[3] foram os pioneiros a fazerem uso desse modelo em larga escala. Hoje em dia pode-se encontrar serviços P2P aplicados aos mais diversos cenários, tais como: comunicação instantânea (ICQ..., 2011), armazenamento de dados em rede (CLARKE et al., 2001) e processamento distribuído (ANDERSON et al., 2002).

A difusão de vídeo na Internet também tem experimentado uma grande popularidade ao longo dos anos. Portais como Youtube[4] e NetFlix[5] são exemplos de serviços que operam sobre a Internet sendo responsáveis por uma proporção considerável do tráfego nos EUA (HUANG; LI; ROSS, 2007). Em 2006, apenas o Youtube já armazenava cerca de 45 terabytes de vídeo, atraindo aproximadamente 1.73 bilhões de visualizações até aquele ano (LIU; GUO; LIANG, 2008). Esse fato aliado com o rápido crescimento de redes residenciais de banda larga, levam o tráfego de vídeo a ser potencialmente dominante na Internet em um futuro próximo.

Para que um sistema de difusão de vídeo alcance sucesso, é necessário que este consiga entregar aos seus usuários boa qualidade de vídeo, provendo mecanismos que consigam compensar variações grandes nas taxas de recepção, sendo escaláveis a medida que o número de usuários aumenta e evidentemente proporcionando boa relação custo-benefício ao provedor de conteúdo. Tais requisitos são extremamente desafiadores para serem alcançados simultanea-

[1] http://www.napster.com
[2] http://www.audiogalaxy.com
[3] http://www.kazaa.com
[4] http://www.youtube.com
[5] http://www.netflix.com

mente, por esta razão a difusão de vídeo (*Video Streaming*) na Internet em larga escala ainda é fonte de muita pesquisa e investigação (HUANG; LI; ROSS, 2007).

Apesar de não conseguir satisfazer todas as características desejáveis na difusão de vídeo na Internet, o modelo cliente-servidor ainda é a solução predominante nos grandes provedores de conteúdo na Internet. Mecanismos alternativos, tais como as redes de distribuição de conteúdo (CDN) ajudam a tornar o modelo cliente-servidor, ainda, economicamente aceitável. No entanto, com o crescimento tanto do número de acessos aos serviços quanto da demanda por vídeos de maior qualidade (1080P..., 2009), percebe-se que soluções mais eficientes devem ser buscadas em relação a este tradicional modelo. Neste contexto, pode-se prever que o modelo cliente-servidor torna-se cada vez mais incompatível com tais requisitos, uma vez que para garanti-los é necessário um investimento muito alto por parte dos provedores de conteúdo.

A transmissão de vídeo fazendo uso de uma tecnologia denominada *Multicast* tentou resolver tal problema de escalabilidade organizando uma estrutura em árvore de transmissão. Embora funcione bem, esta abordagem não experimentou padronização e implantação global em termos de endereçamento e questões referentes à segurança ficaram a desejar. Os motivos desta não padronização são melhor explicitados no próximo capítulo.

As redes P2P podem ser empregadas para difusão de vídeo uma vez que o vídeo é um tipo específico de arquivo. No entanto, tal emprego do P2P como modelo para difusão de vídeo impõe novos requisitos e múltiplos desafios a tais redes, essencialmente centrados em garantir níveis mínimos de qualidade de serviço (QoS). Os algoritmos concebidos para distribuição de conteúdo estático (não contínuo) não precisam preocupar-se com requisitos temporais, ou seja, não é exigido entregar dados a uma taxa constante ao cliente, uma vez que ele precisará terminar de baixar o conteúdo inteiro antes de visualizá-lo. Por outro lado, a distribuição de vídeo por meio de redes P2P impõe restrições maiores já que é necessário que o usuário receba o vídeo a uma taxa que é, pelo menos, igual a taxa de codificação do mesmo.

Existem, basicamente, duas formas principais de se transmitir vídeo fazendo-se uso de redes tanto par-a-par quanto no modelo-cliente-servidor, a distribuição de Vídeo sob Demanda (VoD) e a difusão de vídeo ao vivo (*Live Stream*). Na difusão ao vivo o sistema preocupa-se em entregar o vídeo de forma a minimizar a disparidade temporal entre o que ocorre na filmagem e quando o espectador recebe esse evento. Além disso, nesta modalidade de transmissão também é importante garantir uma certa garantia de sincronia entre a visualização dos participantes, tal sincronia é interessante pois evita que uns usuários vejam uma cena muito tempo antes de outros. Um exemplo tradicional desse problema é o cenário em que se transmite um jogo de futebol ao vivo pela Internet e um vizinho assiste (e comemora) o gol antes do outro. No

segundo mecanismo de difusão, o de vídeo sob demanda, o cliente escolhe o que deseja assistir. Portanto, nesta modalidade, o sistema deve ser capaz de entregar a transmissão de rede em taxas equivalentes à de reprodução do vídeo. Outra diferença fundamental consiste na possibilidade das operações *avanço* e *retrocesso* no vídeo por parte do vídeo sob demanda, enquanto na transmissão de vídeo ao vivo apenas a operação retrocesso é suportada.

Alguns exemplos de aplicações como PPLive[6], TVU[7] , Joost[8], SOPCast[9] ou CoolStreaming[10] têm oferecido uma gama de canais usando P2P para comunidades de milhares de usuários. Particularmente, a distribuição de vídeo sob demanda auxiliada por pares (P2P VoD), tem se mostrado como uma solução promissora, sendo alvo de um considerável corpo de pesquisa nos últimos anos (HUANG; LI; ROSS, 2007) (GAO et al., 2010) (HE; GUAN, 2009). Um dos focos de investigação encontrado na literatura consiste na redução da banda de transmissão nos servidores de VoD, no qual os pares participantes da rede P2P auxiliam o servidor na entrega de conteúdo. O uso desse P2P VoD pode potencializar uma economia expressiva de banda de envio para os servidores de conteúdo (YIN et al., 2009).

Na difusão de vídeo P2P VoD, podem existir pares que consigam retransmitir o vídeo para outros pares e ainda sobra alguma banda de envio. Com esta banda remanescente pode-se adotar duas abordagens: (i) usá-la para atender as necessidades de recepção de vídeo de outros pares ou (ii) usá-la para enviar partes extra de vídeo aos pares que já recebem a transmissão de vídeo na mesma taxa de codificação. A primeira abordagem tem a vantagem de permitir que o sistema suporte mais usuários simultâneos ao custo de tais usuários possuírem um *buffer* de vídeo pequeno, uma vez que eles recebem o vídeo na mesma taxa que o reproduzem. A segunda abordagem, que é definida pelo nome de pré-busca (*pre-fetch*), preza por garantir aos usuários um crescimento em seus *buffers* pois eles podem receber o vídeo em uma taxa maior do que o reproduzem, ao custo do sistema não ter um crescimento na quantidade de usuários simultâneos. Neste contexto, a pré-busca pode ser definida como um fluxo extra de vídeo que um dado par recebe sendo que nesse fluxo recebe-se partes de vídeo que o par não precisa no momento da recepção. Tal fluxo extra (pré-busca) faz com que o *buffer* de recepção do cliente cresça, o cliente pode fazer uso de tal buffer em (i) momentos de alto congestionamento da rede quando as taxas de transmissão não conseguem ser entregues em sua totalidade ou mesmo (ii) para que o cliente deixe de receber fluxos de vídeo, com vistas a economizar banda de envido do servidor de vídeo (caso seja o caso dele ser auxiliado pelo servidor).

[6] http://www.synacast.com/en/
[7] http://www.tvunetworks.com/
[8] http://www.joost.com/
[9] http://www.sopcast.com
[10] http://www.coolstreaming.us/

O trabalho (HUANG; LI; ROSS, 2007) obteve grande repercussão ao propor um algoritmo de P2P VoD que objetiva reduzir a carga de envio (*upload*) dos servidores de conteúdo. Tal artigo mostrou, por meio de simulação de eventos discretos, o quanto os servidores de vídeo do MSN Vídeos poderiam ter economizado em suas bandas de *upload* caso esta tecnologia fosse usada. Adicionalmente ao algoritmo de P2P VoD, o artigo propôs dois algoritmos de pré-busca que objetivam obter economias maiores ainda para os servidores. É argumentado que a economia potencializada pela pré-busca decorre da possibilidade dos clientes (pares) do sistema poderem construir um *buffer* de vídeo razoável e, em um dado momento, cessarem a recepção do fluxo de vídeo que recebem do servidor para consumir o *buffer* que possuem. A pesquisa mostrou que a diferença de *upload* dispendida no servidor de conteúdo entre usar ou não a pré-busca foi de até 5 vezes. Embora os resultados pareçam muito promissores, é importante ressaltar que as simulações realizadas não consideraram aspectos inerentes de uma rede real, portanto, realizou-se uma simulação muito otimista. É comum perceber em outros trabalhos que também exploram o uso da pré-busca a não-preocupação em simular ambientes de rede com características mais realistas. Essa característica pode ser confirmada na leitura dos trabalhos relacionados desta dissertação.

1.1 Abordagem e Contribuição

Tendo em vista que o trabalho (HUANG; LI; ROSS, 2007) e outros adotam suposições um tanto otimistas em relação à simulação da arquitetura de rede, o presente trabalho distingui-se dos anteriores, ao avaliar o desempenho do uso de um algoritmo de pré-busca proposto em (HUANG; LI; ROSS, 2007) na transmissão de P2P VoD, com o diferencial de simular a pilha completa do TCP/IP. Tal simulação é realizada por meio de um *framework* de execução e avaliação de simulações que também é proposto neste trabalho. Esta abordagem permite compreender diversos compromissos em um projeto num sistema P2P de transmissão de vídeo, considerando não apenas a perspectiva da economia de banda do servidor, mas também quantificando o impacto da técnica de pré-busca na rede e a continuidade de reprodução do vídeo percebida pelos clientes. A abordagem é estruturada por uma topologia composta de nodos e enlaces nos quais ocorrem (i) atrasos e perdas, (ii) distintos protocolos na camada de transporte e (iii) tráfego de fundo representando múltiplas variáveis de rede presentes na Internet.

1.2 Objetivos

Esta dissertação possui dois objetivos gerais principais sendo que um deles pode ser dividido em vários objetivos específicos.

- Desenvolver um *Framework* para análise de desempenho em simulação de redes P2P VoD com as seguintes características:
  - Fazer uso de um simulador de redes bem estabelecido;
  - Considerar a existência de uma topologia de rede realista;
  - Permitir a existência de tráfego de fundo sintético;
  - Ser parametrizável para poder simular facilmente diferentes cenários;
  - Gerar gráficos comparativos de forma automática nas perspectivas: (i) do servidor, (ii) dos clientes e dos (iii) enlaces de rede;
  - Ser compatível para funcionamento em *cluster*;
  - Permitir diferentes configurações de ordem de chegada dos pares ao sistema P2P VoD;
  - Permitir o uso de diferentes camadas de transporte;
- Avaliar os ganhos decorrentes da utilização de pré-busca na distribuição de vídeo sobre P2P para ambos servidor e clientes fazendo-se uso do *framework* desenvolvido;
  - Analisar como as diferentes ordens de chegada dos pares podem impactar a economia na banda de *upload* do servidor;
  - Realizar medições sobre o impacto do tráfego de fundo na difusão de vídeo;
  - Avaliar sob a ótica do servidor, clientes e enlaces qual camada de transporte é mais adequada ao sistema simulado;
  - Analisar a qualidade na recepção de vídeo percebida pelo usuário quando se usa ou não a pré-busca;

1.3 Organização da Dissertação

Esta dissertação está dividida da seguinte forma: No capítulo 2, é descrita a fundamentação teórica sobre redes par-a-par e vídeo sob demanda. No capítulo 3 são abordados os trabalhos

relacionados agrupados em duas categorias: os sistemas de transmissão P2P VoD em geral, e os sistemas que fazem uso explicitamente da pré-busca. No quarto capítulo é apresentado o framework desenvolvido, abordando suas características e quais foram os parâmetros usados na simulação analisada neste trabalho. Os resultados provenientes da simulação são analisados no capítulo 5, juntamente com tais resultados também são descritas algumas conclusões preliminares. Por fim, no capítulo 6 são apresentadas as conclusões deste trabalho, destacando os pontos-chave que puderam ser obtidos e os trabalhos futuros.

2 *Fundamentação Teórica*

Este capítulo aborda conceitos introdutórios sobre as soluções existentes para difusão de vídeo na Internet, demonstrando a evolução das soluções propostas até culminar na difusão por meio do P2P. Posteriormente, são detalhadas as 3 principais topologias de organização em sistemas par-a-par. Em seguida é apresentada a definição de sistema de vídeo sob demanda com uma respectiva classificação e também apresenta-se um mecanismo denominado *peer-assisted VoD*. Adicionalmente, é demonstrado um algoritmo de transmissão de vídeo sob demanda auxiliado por P2P incluindo duas abordagens de pré-busca.

2.1 Mecanismos para Difusão de Vídeo na Internet

O modelo de arquitetura de protocolos da Internet, bem conhecido como modelo TCP/IP, provê um serviço baseado no melhor esforço, ou seja, um serviço sem garantias explícitas de banda, variação máxima de atraso ou ausência de perdas. Para aplicações como difusão de mídias contínuas (e.g. vídeo), atrasos na entrega dos pacotes, variação de atraso (*jitter*) ou perda comprometem significativamente a qualidade de serviço do vídeo percebido pelo usuário.

A camada de rede da Internet é responsável por prover um serviço de conectividade entre hospedeiros (*hosts*) distribuídos globalmente, fornecendo endereçamento entre os computadores participantes de uma dada sessão de comunicação. Os endereços são expressados por meio de palavras de 32 *bits* (IPv4), denominados endereços IP, com uma semântica hierárquica que determina que parte do IP é referente a rede e qual parte refere-se ao endereço do *host*.

Uma forma de garantir qualidade de serviço na Internet apoiando-se no protocolo IP, é utilizar 6 *bits* do cabeçalho que compõe uma solução denominada *DiffServ* (*Differentiated Services*) (RFC2475..., 1998), ou serviço diferenciado. Esta solução estabelece uma forma simples e escalável para provimento de garantia de qualidade de serviço. São definidas classes de serviços nas quais cada pacote é classificado, e a cada uma destas classes são atribuídas prioridades diferentes para tratamento dos dados. Embora seja um mecanismo presente na padronização do

protocolo IP, o mesmo não experimentou grande aceitação nos ISPs (*Internet Service Providers*), principalmente pela falta de uma padronização de como cada classe de serviço deve ser tratada especificamente. Os outros dois bits remanescentes do campo de TOS (*Type of Service*) do IP são usados para o ECN (*Explicit Congestion Notification*). A combinação destes dois *bits* permite codificar quatro tipos de mensagens diferentes sobre o estado de congestionamento experimentado por um pacote na sua trajetória entre uma origem e destino. Determinadas aplicações ou até mesmo protocolos de transporte podem fazer uso desses campos para tomar decisões melhores sabendo da existência de congestionamento na rede.

Um outro mecanismo alternativo, também implementado na camada de rede é o: IntServ (*Integrated Services*) (RFC1633..., 1994), que é orientado a fluxos, ou seja, este realiza a marcação e priorização de fluxos (definidos por vários pacotes) fim-a-fim e não de pacotes isolados como no *DiffServ*. Esta abordagem confere a possibilidade de um ajuste fino permitindo a qualidade de serviço em fluxos específicos de determinadas aplicações. Para que este mecanismo funcione, é necessário um suporte da aplicação e de todos os roteadores entre os dois computadores participantes da comunicação. O protocolo usado para o estabelecimento do fluxo reservado é o RSVP (*Resource Reservation Protocol*) (ZHANG et al., 1993). Este mecanismo também não foi amplamente implantado na Internet devido à complexidade de manter uma tabela de estado em cada roteador para cada fluxo ali reservado, o mesmo não se mostrou escalável para uma quantidade grande de fluxos. Como pode ser notado, com os mecanismos implementados na camada de rede, ainda é um desafio garantir qualidade de serviço para transmissão de mídias contínuas de forma escalável. Esse desafio nos motiva a tentar buscar tal solução em camadas superiores.

A camada de transporte possui grande importância na arquitetura de rede em camadas. Ela desempenha um papel fundamental de fornecer serviços de comunicação aos processos de aplicação que rodam em hospedeiros distintos. Processos da camada de aplicação usam a comunicação lógica provida pela camada de transporte para enviar mensagens entre si, sem a preocupação dos detalhes da infra-estrutura física utilizada para transportar mensagens.

As aplicações tradicionalmente fazem uso do modelo de comunicação cliente-servidor, nem sempre tais aplicações preocupam-se com garantias de qualidade de serviço, portanto, pode-se fazer uso dos mecanismos de QoS elaborados que podem existir na camada de transporte. Dentre os protocolos de transporte que podem oferecer garantias de QoS ou implementarem estratégias mais reativas ao congestionamento da rede e constituem alternativas aos protocolos *vanilla* da Internet (TCP e UDP), estão: UDP Lite (LARZON et al., 1999), RUDP (*Reliable User Datagram Protocol*) (RELIABLE..., 1999), SCTP (*Stream Control Transmission Protocol*) (NA-

TARAJAN et al., 2006) e DCCP (*Datagram Congestion Control Protocol*) (DATAGRAM..., 2004). Os protocolos que têm obtido maior repercussão foram os dois mais recentes: SCTP e DCCP.

O SCTP foi projetado com intuito de suprir algumas deficiências do TCP. Este protocolo permite o serviço de *multistreaming*, possibilitando à aplicação separar logicamente o envio de diferentes objetos (imagens, dados de sinalização, pedaços de vídeo, etc.) em uma mesma conexão SCTP. A vantagem desse mecanismo é a não existência do HOL[1] (*Head of Line*) *Blocking*, pois cada objeto enviado é tratado independentemente dos outros, assim, tais objetos podem ser enviados em paralelo. No caso do TCP, a transmissão de vários objetos em uma mesma conexão ocorre de forma sequencial, sendo que quando perdas acontecem, todos os objetos ainda não enviados necessitam esperar a recuperação de tais perdas. A solução de abrir múltiplas *threads*, uma para cada objeto, possui a desvantagem da necessidade de alocar mais recursos para novas conexões, aumentando o *overhead*.

Além disso, cada nova conexão TCP não possui comunicação com outras na camada de transporte, dessa forma, as decisões para controle de fluxo e congestionamento são locais de cada conexão diferente. Devido a essa característica, é dito que o uso de múltiplas *threads* para múltiplas conexões TCP constitui uma forma de usar mais recursos da rede, tornando a aplicação que o faz menos justa com as outras presentes na Internet. No SCTP, mesmo quando existem diferentes conexões, as mesmas possuem uma tomada de decisão conjunta sobre o controle de fluxo e congestionamento. Em aplicações de difusão de vídeo, a característica da separação de sessões entre uma mesma conexão SCTP pode ser interessante quando se faz uso de codificação de rede. Neste cenário, cada sub-stream pode ser enviada em uma sessão diferente de uma mesma conexão.

Outro aspecto relevante do SCTP é a sua funcionalidade denominada *multihoming*, que permite que ambos emissor e receptor possuam dois IPs diferentes. Neste mecanismo, um par de IPs pode ser usado para a comunicação principal e o outro par pode ser usado no caso de falhas ou perdas excessivas do primeiro. A mudança dos IPs usados no nível de rede ocorre de forma imperceptível para a aplicação.

A terceira principal funcionalidade desse protocolo de transporte consiste na melhor defesa contra ataques do tipo SYN *Flood*. No estabelecimento da conexão do SCTP é usado o mecanismo denominado *four-way handshake*, quando um cliente envia um pacote de SYN para o

[1] O HOL decorre da característica intriseca do TCP enviar objetos de forma serial. Caso se necessite enviar 10 objetos em um dado momento, o TCP os envia um a um, mesmo que os 10 objetos já estejam disponíveis pela aplicação ao mesmo tempo. Caso na hora de enviar o objeto número 4 ocorra algum problema na conexão, ou sejam necessárias retransmissões, todos os outros 6 objetos ainda não enviados são diretamente afetados por tal problema. O HOL não ocorreria se o TCP enviasse objetos em paralelo em uma mesma conexão.

servidor, este retorna ao cliente uma mensagem de ACK juntamente com um *cookie*, que contém algumas informações iniciais para estabelecimento da conexão. Neste momento o servidor não aloca recursos pois ainda não houve finalização do *handshake* devido à necessidade do cliente ter que receber este *cookie* e processá-lo para poder continuar com a requisição, os ataques de SYN *flood* tornam-se bem mais difíceis do ponto de vista do atacante.

Apesar das três principais funcionalidades apresentadas pelo SCTP, este protocolo implementa controle de fluxo e congestionamento similares aos do TCP. Além disso, em seu funcionamento padrão, também oferece o serviço de entrega confiável de dados ordenados. Existe a extensão denominada PR-SCTP que permite a entrega garantindo menos confiabilidade, basicamente, define-se um *timeout* dentro do qual tenta-se realizar as retransmissões quando ocorrem perdas, caso este tempo estoure, o protocolo ignora as perdas e continua. Esta extensão mostra-se mais adequada à difusão de vídeos. Outra característica relevante desse protocolo é sua capacidade de ser configurado para entregar os dados de forma não ordenada, assemelhando-se ao serviço provido pelo UDP. Esta entrega não-ordenada também pode ser mais aplicada às aplicações de transmissão de mídia contínua.

O DCCP foi proposto como alternativa ao UDP para ser usado em aplicações de transmissão de áudio e vídeo. O UDP, embora seja um protocolo mais presente na Internet e em aplicação com requisitos de tempo real, não possui mecanismos para controle de congestionamento e nenhuma garantia de entrega confiável. A proposta do DCCP é implementar um protocolo com controle de congestionamento que possui mecanismos de garantia de confiabilidade seletiva. O DCCP herda do TCP a característica de ser orientado a conexão e fornecer controle de congestionamento. Similarmente ao UDP, o DCCP herda a característica de não garantir a entrega e nem a ordenação dos dados enviados.

No protocolo DCCP existe o conceito de conexão bidirecional que é implementada como sendo duas conexões unidirecionais usando o mesmo *socket*. Cada conexão unidirecional é dita ser *half-connection*, assim, em cada sentido é usada uma half-connecion. Embora estas subconexões sejam logicamente distintas, elas se sobrepõe. Por exemplo, em uma transmissão de A para B um pacote *DCCP-DataAck* contém dados da aplicação gerados por A e informações que confirmam a recepção de dados transmitidos de B para A. Esse mecanismo também é conhecido por *pigbacking*. Embora o DCCP não implemente confiabilidade na entrega dos dados, ele permite que o receptor informe ao emissor quais pacotes foram recebidos. Para isso, este protocolo implementa um mecanismo com ACK de ACK, assim, ele garante confiabilidade na entrega dos ACKs. Essa garantia é necessária pois estes ACKs possuem um papel importante nas decisões do controle de congestionamento.

É interessante também menciocar que o DCCP implementa três tipos de controle de congestionamento: *TCP-Like Congestion Control* (CCID-2), *TCP Friendly Rate Control* (CCID-3) e *TCP Friendly Rate Control for Small Packets* (CCID-4). O DCCP permite que o controle de congestionamento seja implementado de forma modular independendo do resto das características do protocolo. Ainda neste quesito, é possível que a aplicação que faz uso deste protocolo escolha qual controle de congestionamento será usado, podendo modificar o mesmo no meio da conexão. Ainda é possível que seja empregado um controle de congestionamento em uma direção da conexão e seja implementado outro mecanismo na outra direção. Para aplicação de difusão de vídeo o CCID-3 seria mais adequado por implementar um controle de congestionamento que muda a taxa de transmissão de forma menos abrupta que o CCID-2. Já o CCID-4 seria mais aplicável em transmissões que usam o Voip.

Embora ambos SCTP e DCCP apresentem-se como protocolos de transporte com características favoráveis à transmissão de vídeo na Internet, percebe-se que em alguns casos os próprios roteadores domésticos e de menor capacidade não encaminham pacotes com IPs que possuem protocolo de transporte diferente do TCP ou UDP. Dessa forma, essa característica impacta na sua implantação em larga escala impedindo que aplicações façam uso destes protocolos em um sistema de difusão de vídeo.

A escalabilidade pode ser limitada não só pelo uso da camada de transporte, mas também pelo modelo de transmissão dos dados. A difusão de vídeo seguindo o modelo cliente-servidor funciona da seguinte forma: é estabelecida uma conexão ponto-a-ponto entre o servidor e o cliente interessado no vídeo, por meio dessa conexão, o fluxo de vídeo é enviado. Quando vários clientes tentam acessar o vídeo ao mesmo tempo, o servidor necessita abrir uma conexão para cada e enviar os mesmos pedaços de vídeo em paralelo aos vários solicitantes.

Um exemplo que expõe a não-escalabilidade do modelo cliente-servidor é a difusão ao vivo de eventos que atraem milhões de usuários, como algumas transmissões *on-line* do *website* g1.com.br. O problema ocorre pois os servidores possuem uma banda de *upload* máxima limitada por contrato quando se efetuou a terceirização do serviço de hospedagem do *website*. Por exemplo, supondo um vídeo de qualidade 300*Kbps* e a existência de 10 clientes, a banda de *upload* deste servidor será de 3*Mbps* (300*Kbps* vezes 10 clientes). Caso seja usado o P2P e todos os clientes possuam banda de *upload* superior a 300*Kbps*, os mesmos podem se organizar em uma estrutura de difusão em árvore fazendo com que a demanda de tal servidor caia para 300*Kbps*. Ou seja, gerando uma economia no servidor de 90%.

Tendo em vista este exemplo, torna-se claro que o modelo cliente-servidor não se adéqua muito bem a cenários em que o número de clientes é da ordem de milhares, talvez, milhões,

ou seja, confirma-se que o modelo cliente-servidor não é escalável, uma vez que é necessário que o servidor tenha uma banda de envio igual à taxa de codificação do vídeo multiplicada pelo número de clientes para os quais ele suporta enviar simultaneamente.

A próxima sessão pretende abordar as alternativas que foram desenvolvidas para solução do problema de escalabilidade no modelo cliente-servidor quando se pretende fazer transmissão de vídeo na Internet.

2.1.1 Transmissão de Vídeo em Redes com Suporte Multidestinatário

Um mecanismo proposto que busca superar as deficiências do modelo cliente-servidor é a comunicação multidestinatária (*multicast*) no nível IP. Esse tipo de comunicação permite que a informação seja enviada apenas uma vez pelo servidor mesmo que hajam vários clientes (RFC1112..., 1989). A Figura 2.1 exibe de forma mais clara a principal diferença entre o modelo cliente-servidor e *multicast*.

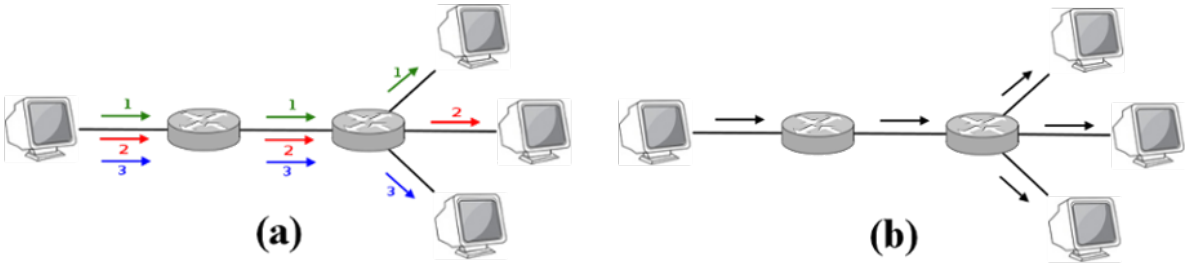

Figura 2.1: Diferença entre o (a) *unicast* e (b) *muiticast*.

Esta solução permite um aumento de escalabilidade, uma vez que transfere a responsabilidade de replicar o envio de vídeo do servidor para os ativos de rede (no nível IP), dessa forma, economizando a banda do servidor. Na Figura 2.1 pode-se ver que quando é usado o *unicast* (a) o servidor acaba gastando uma banda de *upload* de 3 vezes a taxa de reprodução do vídeo, pois necessita criar três conexões diferentes. Quando se usa o *multicast* (b), o servidor envia o fluxo uma única vez e a rede encarrega-se de entregá-lo aos destinos correspondentes sem que o servidor necessite abrir várias conexões como no *unicast*.

Para este tipo de comunicação foram separados faixas de endereços de IPs específicos. Em geral, funciona da seguinte forma: um servidor que deseja disponibilizar um vídeo na rede simplesmente envia tal vídeo para um determinado IP que ele pode escolher, como sendo o endereço do grupo *multicast*. Os clientes que desejam assistir ao vídeo devem saber qual endereço é este e também entrar nesse grupo por meio de um protocolo que foi desenvolvido

especificamente para este propósito IGMP. Desta forma, o servidor envia o vídeo em um único fluxo para o endereço do grupo e os ativos da rede se encarregam de determinar os caminhos e replicações que serão feitos para a transmissão chegar até os clientes que participam do mesmo grupo.

O mecanismo de transmissão multidestinatário foi desenvolvido com vistas a tornar-se padrão de *facto* na Internet. No entanto, os grandes provedores de Internet não se esforçaram para implantar as configurações necessárias em seus ativos de rede objetivando suportar essa tecnologia. Foi afirmado em (HOLBROOK; CHERITON, 1999) que essa lenta adoção do IP *multicast* seja devida aos seguintes fatores:

- O modelo foi criado para permitir a existência de vários emissores (n) para vários receptores (m), sem implementar restrições de autenticação tanto para criação de grupos, participação como emissor e participação como receptor;
- Não pensou-se em como distribuir o endereçamento IP dos grupos de forma global e nem se a faixa de IP alocada para este fim suportaria tal distribuição;
- O *multicast* não prevê mecanismos de segurança contra ataques nas sessões e rotas *multicast*, e nem implementa mecanismos de integridade dos dados;

Em suma, tais fatores demonstram que desde quando este modelo de distribuição foi concebido, não pensou-se em sua implantação global. Posteriormente outras propostas que assemelham-se ao *multicast* surgiram para tentar resolver seu problema, a saber: ALM (*Application-Level Multicast*) (EL-SAYED; ROCA; MATHY, 2003) e o SSM (*Source Specific Multicast*) (HOLBROOK; CHERITON, 1999), no entanto, vale ressaltar que nenhuma delas foi adotada globalmente. Tendo em vista tal dificuldade de se implantar o *multicast* na Internet, os provedores de conteúdo voltaram-se ao paradigma anterior, reavaliando o modelo cliente-servidor e descobrindo como ele poderia ser usado para entregar vídeo aos usuários da Internet de forma prática.

2.1.2 Redes de Distribuição de Conteúdo

As redes de distribuição de conteúdo, também conhecidas como CDNs (*Content Distribution Networks*), constituem uma forma otimizada de utilizar o modelo cliente-servidor para replicar dados na Internet colocando-os mais "próximos"dos clientes. Esse modelo surgiu quando houve a percepção por parte dos provedores de serviços, que não basta ter um grande *datacenter* com muita banda de *upload* e vários servidores. Caso tal *datacenter* esteja muito distante

dos clientes (em diferentes ISPs), a transferência das informações sempre podia sofrer grandes atrasos, prejudicando a qualidade de resposta. No contexto de transmissão de vídeo, esse *delay* (atraso) pode ser ainda mais impactante.

Para mitigar esse problema, as CDNs replicam os dados que encontram-se no servidor de origem para outros servidores que localizam-se nas bordas de vários ISPs. Sempre que um vídeo é requisitado por um cliente, o servidor central transfere a requisição para um servidor replicado que encontra-se mais próximo. Caso o servidor mais próximo não possua o conteúdo solicitado ainda, este é atualizado pelo servidor central, assim o cliente é sempre atendido pelo servidor mais próximo a ele. Em geral, as CDNs adotam algumas métricas para definir qual servidor de replicação será escolhido como mais próximo para atender a uma requisição. Dentre elas pode-se citar: número de saltos, tempo de ida e volta (*Round Trip Time* - RTT) e cargas de processamento e rede dos servidores disponíveis (ALBUQUERQUE; PROENçA; OLIVEIRA, 2006).

A Figura 2.2 explica melhor o funcionamento de uma CDN, primeiramente o cliente faz uma requisição para o servidor principal (1), posteriormente o servidor principal encaminha esta requisição para um servidor de replicação escolhido, neste caso, o mais próximo (2), por fim, o servidor de replicação serve o cliente com o vídeo (3).

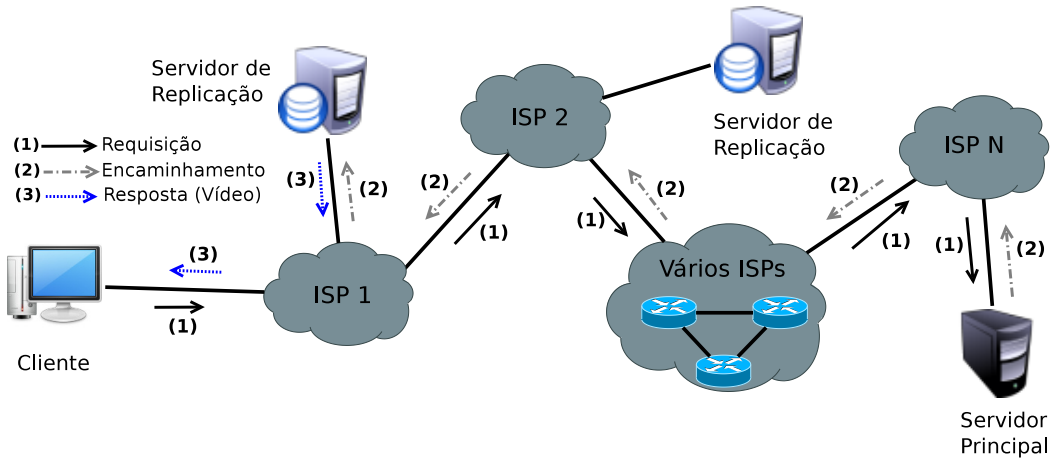

Figura 2.2: Mecanismo tipico de funcionamento de uma CDN.

Atualmente, vários *websites* que hospedam vídeo na Internet fazem uso das CDNs. Por exemplo: YouTube[2], Google vídeo[3], Yahoo vídeo[4], DailyMotion[5], etc. Existem também empresas que vendem serviços que auxiliam na implantação desta tecnologia, dentre as mais co-

[2]http://www.youtube.com
[3]http://vídeo.google.com
[4]http://vídeo.yahoo.com
[5]http://dailymotion.com

nhecidas pode-se citar: Akamai[6], Amazon CloudFront[7], CDNetworks[8] e Cotendo[9].

Embora seja uma tecnologia que ainda é amplamente usada na Internet, as CDNs apresentam problemas de custo/benefício elevado, uma vez que para atender uma demanda cada vez maior de usuários é necessário que sejam implantados mais servidores de replicação, o que implica em aumento de despesas para suprir esta necessidade. Por outro lado, os ISPs possuem grandes interesses em colocar servidores de CDNs dentro de suas redes como forma de economizar sua banda em PTTs (Ponto de Troca de Tráfego). Desta forma, os ISPs chegam a pagar aos provedores de conteúdo para que se consiga implantação de tais CDNs. Esta característica ajuda a aliviar os custos na implantação e manutenção das CDNs por parte dos provedores de conteúdo.

2.1.3 Redes Par-a-Par (P2P)

As redes par-a-par surgiram com objetivo principal para compartilhamento de recursos na Internet. Tais redes funcionam de forma descentralizada, em geral, para um cliente fazer parte de uma rede P2P, basta que o mesmo execute um aplicativo que é responsável por inseri-lo na mesma. A escalabilidade provida pelo P2P decorre do compartilhamento de recursos que este promove, em redes deste tipo, os pares integrantes participam oferecendo suas capacidades de processamento, armazenamento e rede (CLARKE et al., 2001) (ANDERSON et al., 2002). Portanto, pela sua arquitetura descentralizada, a rede P2P não impõe limites de crescimento em termos de número de participantes (propriedade de ser auto-escalável). Essa vantagem potencial possibilita seu uso na transferência de vídeo, sendo uma alternativa bem mais barata às tradicionais CDNs que ainda são predominantes na distribuição de vídeo de grandes provedores de conteúdo.

Em uma rede P2P as vizinhanças são construídas, em princípio, sem que se leve em consideração a posição física dos pares, dessa forma, cria-se o conceito de uma rede com topologia completamente diferente da existente fisicamente. Essa é uma característica típica das redes P2P, essas redes também são chamadas de redes sobrepostas (*overlay*). A Figura 2.3 apresenta graficamente um exemplo.

[6]http://akamai.com
[7]http://aws.amazon.com/cloudfront
[8]http://www.us.cdnetworks.com
[9]http://cotendo.com

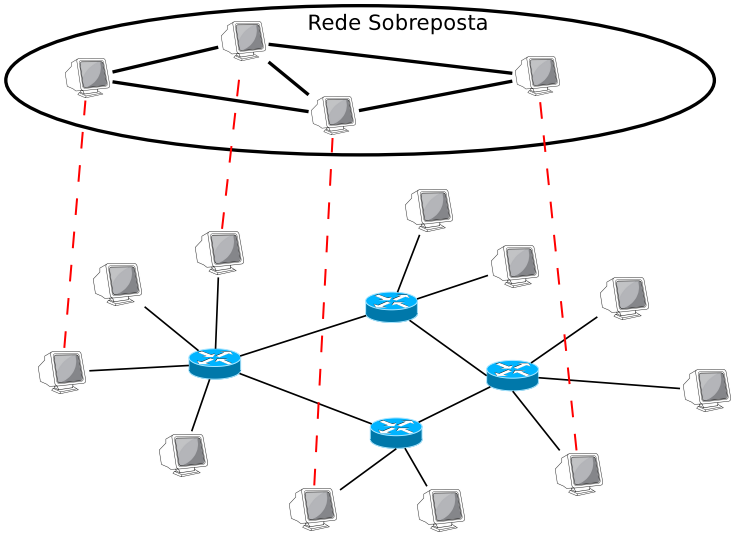

Figura 2.3: Desenho esquemático de uma rede sobreposta (*overlay*)

Como foi citado anteriormente, mesmo que a arquitetura P2P não imponha limites para seu crescimento, pode existir um limite para seu desempenho ótimo dependendo da forma como a mesma é construída. Neste contexto, os algoritmos de construção de rede P2P tornam-se importantes, pois eles implementam uma lógica descentralizada que, ao mesmo tempo, permite a aplicação atingir seu objetivo e não sofrer perda de desempenho quando ocorrem eventos adversos, tais como: saídas de pares, congestionamento, indisponibilidade de enlace, etc.

Inserido no contexto da criação da topologia também está a gerência dos recursos, o algoritmo P2P deve ser capaz de gerenciar os recursos de cada participante de forma que as capacidades do mesmo sejam avaliadas e façam parte do processo de tomada de decisão global do sistema. Deve-se implementar mecanismos que previnam a rede sobreposta de ter seu desempenho prejudicado quando um novo participante entra com capacidades muito limitadas, por exemplo.

Em geral, as redes P2P fazem uso de servidores denominados "*trackers*"os quais se encarregam de manter uma lista dos pares participantes de uma sessão. Em sessões de transmissão P2P mais populares, tais servidores *trackers* podem experimentar sobrecarga por estes serem um ponto único de contato inicial entre todos os pares. Com objetivo de resolver este problema foram propostas as tabelas *hash* distribuídas DHT (*Distributed Hash Tables*). Este mecanismo faz com que os próprios clientes do sistema P2P atuem como *trackers* em um dado momento. De modo geral, de posse do nome do arquivo que se busca, pode-se realizar consultas de forma iterativa aos clientes do sistema P2P até que se consiga descobrir qual cliente possui a informação dos clientes que possuem o arquivo desejado.

2.2 Arquiteturas de Difusão de Vídeo Sobre Redes Par-a-Par

A arquitetura de distribuição pode impactar diretamente no desempenho do sistema P2P como um todo. Decisões do tipo: com quais pares um novo participante irá se conectar, o que fazer quando um par sair do sistema subitamente (*peer-churn*), qual pedaço de informação enviar para qual par, etc; podem ser cruciais para determinar sucesso ou fracasso de sistemas deste tipo. Esta subseção objetiva apresentar as três principais arquiteturas de distribuição conhecidas e aplicadas a redes P2P. Duas delas podem ser exemplificadas conforme exibido na Figura 2.4.

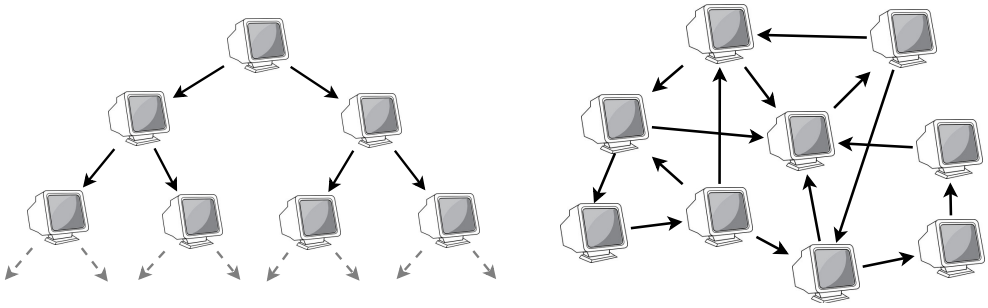

Figura 2.4: Arquiteturas de distribuição P2P, árvore e malha.

2.2.1 Arquitetura em Árvore

A topologia em árvore possui a característica de se assemelhar com a arquitetura de distribuição *multicast*, no entanto, implementada em camada de aplicação. Nesta organização pares se organizam como uma árvore na qual são construídas relações estritas de pai e filho, o servidor de vídeo situa-se no topo da árvore, constituindo sua raiz. A maioria das implementações desta topologia podem ser caracterizadas como *push-based*, ou seja, a maior parte do conteúdo é entregue aos clientes sem que estes necessitem requisitá-lo explicitamente. A grande vantagem de aliar o escalonamento *push-based* com a topologia em árvore está na obtenção de latências menores, uma vez que existe menos sobrecarga de controle.

Como existem relações estritas de pai-filho entre os pares, à baixo de cada par são constituídas novas sub-arvores , caso um par intermediário sofra de perda de conectividade, ou seja, saída do sistema sem avisar, todos os pares abaixo dele podem ser impactados negativamente. Devido a esta característica, a topologia em árvore não é muito resiliente a sistemas cujos pares possuem uma alta dinamicidade de entrada e saída (*peer-churn*). Além disso, a organização deve ser realizada de forma que um par de baixa capacidade de *upload* fique mais próximo das folhas para que exista um número pequeno de pares que dependam do mesmo.

Outro aspecto a ser tratado nesta topologia é a sua relação de tamanho vertical e horizontal,

quanto mais horizontal for a árvore mais pares-folha ela terá e as capacidades de *upload* destes serão desperdiçadas. Por outro lado, quanto mais vertical for a árvore, a chance de falha em um par prejudicar uma grande quantidade de outros pares é menor. Como forma de mitigar o problema do *peer-churn* algumas propostas de sistemas organizados em árvore quebram uma árvore de distribuição grande em árvores diferentes menores.

2.2.2 Arquitetura em Malha

Como foi citado anteriormente, a abordagem em árvore tem a desvantagem principal de introduzir um único ponto de falha em decorrência de cada par ter apenas um emissor de vídeo. Em sistemas P2P de topologia em malha, não existe uma organização topológica estática, os pares estabelecem e terminam conexões dinamicamente. Em um instante qualquer, um par pode manter relações de vizinhança com vários outros pares, tanto para enviar conteúdo aos mesmos quanto para receber, simultaneamente. A vantagem desta abordagem é a resiliência maior ao *peer-churn*, pois uma vez que um par vizinho sai, o par pode continuar recebendo o vídeo de outros pares vizinhos. É possível configurar os pares para que eles sempre mantenham um número fixo de vizinhos, assim, caso um vizinho saia do sistema, este par irá procurar por novos pares para manter o seu número de vizinhos contante.

Nesta arquitetura o vídeo é separado em pedaços menores, denominados blocos (*chunks*), tais blocos podem variar de 16Kb a 256Kb (LIU et al., 2010). Os pares trocam constantemente mapas de disponibilidade (*buffer maps*) sobre os pedaços de vídeo que eles possuem para oferecer o que precisam. A forma de como tais *buffer maps* são espalhados pela rede, em termos de frequência, tamanho e prioridade é decisiva para determinar se o sistema conseguirá obter um bom desempenho. Como os pares comparam os *buffer maps* de outros para determinar quais pedaços do vídeo eles irão solicitar, diz-se que os sistemas em malha são, em sua maioria, do tipo *pull-based*, ou seja, os pares realizam solicitações explícitas.

Similarmente aos sistemas de P2P para compartilhamento do arquivo, a arquitetura em malha de distribuição de vídeo depende da existência de um servidor de rastreamento de informações (servidor *tracker*). Tal servidor é responsável por determinar quais pares integram uma sessão de transmissão de vídeo, armazenando informações dos participantes, tais como: endereço IP, número das portas envolvidas, capacidades de *upload*/*download*, etc. Quando um novo par deseja participar de uma sessão, ele deve contatar o *tracker*, este envia ao par um conjunto aleatório de pares que estão participando ativamente na sessão. Tal listagem de participantes pode variar de dezenas a milhares. Assim que recebe tal listagem, o par tenta estabelecer conexão com um subconjunto desta, quando a conexão é aceita, ele adiciona esse novo par à sua

lista de vizinhos. Para lidar com a constante variação de pares que entram e saem do sistema, cada par atualiza constantemente sua listagem local de vizinhos (LIU et al., 2010).

Em decorrência da constante troca de *buffer maps*, informações sobre vizinhos e de ter que solicitar o que será recebido, a arquitetura em malha pode sofrer de problemas como grande sobrecarga de controle e elevada latência.

2.2.3 Arquitetura Híbrida

A arquitetura de distribuição híbrida tenta obter o benefício da baixa latência e resiliência ao *peer-churn* ao mesmo tempo. Para atingir esse objetivo, esta arquitetura faz uso de características de ambas topologias citadas anteriormente. No que tange o escalonamento de dados, sistemas *push-based* são mais apropriados para sistemas limitados em upload, (*upload-constrained*), uma vez que o par emissor pode usar sua capacidade de *upload* completamente para enviar o vídeo aos seus receptores, sem gastar esta banda escassa com dados de controle. Por outro lado, os sistemas *pull-based* são mais indicados para sistemas limitados em *download* (*download-constrained*), uma vez que o par que recebe o conteúdo pode ajustar a taxa de recepção juntamente com o emissor baseado em sua capacidade de download.

Um sistema pode ser híbrido por combinar escalonamento de pedaços *push-based* e *pull-based*. Por exemplo, um sistema pode fazer uso de topologia em malha, com predominância igual de escalonamento *pull-based* para algumas informações e *push-based* para outras. Tratando-se especificamente do escalonamento dos dados, um mecanismo de distribuição híbrido bem conhecido é o *push-pull*. Toda a vez que um par realiza uma operação de *pull* (solicitação) à outro par, o emissor passa a enviar o conteúdo para o solicitante por meio de uma operação de *push*, sem que seja necessário posteriores solicitações (CIGNO; RUSSO; CARRA, 2008). Por outro lado, um sistema pode ser híbrido por possuir uma arquitetura predominantemente árvore e permitir que alguns pares-folha estabeleçam conexões com outros pares dentro da árvore sem obedecer hierarquias diretas.

Até o momento foram explicitados conceitos fundamentais sobre P2P e redes. Como este trabalho objetiva investigar a implementação de um algoritmo P2P para transmissão de vídeo sob demanda, é necessário que também seja definido o que é vídeo sob demanda e bem como quais são as classificações pertinentes. A próxima sessão visa abordar tal necessidade.

2.3 Sistemas de Vídeo Sob Demanda (VoD)

Sistemas de Vídeo Sob Demanda são caracterizados por permitir que os clientes escolham o conteúdo que desejam consumir. Idealmente, sistemas desse tipo também devem permitir operações de VCR (*vídeo Cassette Recorder*). Os exemplos mais próximos tratam-se dos sites de Internet que permitem a visualização de vídeos, como YouTube, Google vídeo, Yahoo vídeo, Vimeo, etc. Esta sessão objetiva tanto descrever uma classificação para sistemas VoD quanto conceituar os sistemas VoD assistidos por pares.

2.3.1 Classificação de VoD

Existem, basicamente, três categorias (51VOD..., 2006) nas quais podem estar inseridos os sistemas de vídeo sob demanda, são elas:

- *Near VoD* (NVoD): Nesta classificação estão os sistemas que assemelham-se muito com o funcionamento da transmissão de televisão convencional. Existem vários canais diferentes para cada conteúdo, ou seja, caso deseje-se assistir a um filme de X horas, o usuário deve escolher entre uma quantidade N de canais disponibilizados para tal filme. Embora os canais transmitam o mesmo conteúdo, existe diferença entre qual parte do filme está sendo exibida, em geral, essa diferença de tempo entre um canal e outro é de X/N horas. Assim, um novo filme começa a ser exibido em intervalos regulares de tempo. Caso o usuário chegue ao sistema em um determinado tempo, para assistir o conteúdo desde o começo, ele necessita esperar o início em um dos canais disponíveis. Por esse motivo que esta classificação chama-se "*near VoD*"(quase VoD), o usuário não interage com o provedor de conteúdo, no entanto, implementa-se uma ilusão de que ele assistirá o conteúdo logo depois que ele deseja. Em geral, a grande maioria das Tvs por assinatura implementam esse mecanismo por meio do *Pay-Per-View*;

- *True VoD* (TvoD): Os vídeos distribuídos nesta categoria são explicitamente requisitados pelo usuário no momento em que ele deseja assisti-los. A *priori*, não existe tempo de espera entre o usuário entrar no sistema e começar a assistir, pois ele faz uma requisição ao servidor e é prontamente atendido. Uma vez que o cliente começa a assistir o conteúdo, a única operação que ele pode realizar de interatividade é a de congelar (pause) e retornar (resume) a reprodução. Esta abordagem foi uma das primeiras formas de entregar VoD aos usuários na Internet, também é implantada em algumas Tvs por assinatura de melhor qualidade;

- *Interactive VoD* (IVoD): O VoD interativo suporta todas operações de VCR, ou seja, permite congelar, retornar, retroceder avançar no vídeo. Esta abordagem é a mais difícil de ser entregue por permitir ao usuário um controle maior sobre o que será exibido. Os *websites* de vídeo presentes na Internet atualmente, em sua maioria, estão enquadrados nesta classificação (Youtube, Googlevídeo, Vimeo, etc.).

2.3.2 Peer-Assisted VoD

O VoD auxiliado por P2P ganhou maior foco em decorrência da crescente demanda por serviços VoD da Internet. Esta abordagem é uma alternativa às CDNs, buscando economizar os recursos dos provedores fazendo uso das capacidades de *upload* dos próprios clientes do sistema para que estes se ajudem de forma autossuficiente. Nesta abordagem, ainda existe um servidor principal de vídeo que armazena todos os vídeos publicados e garante a taxa de transmissão para os clientes sem degradação de qualidade. Contudo, no VoD auxiliado por P2P os pares que estão assistindo a um vídeo do sistema, também ajudam redistribuindo os fluxos. Uma vez que o VoD auxiliado por P2P pode transferir uma fração significativa da carga de *upload* do servidor para os pares, esta tecnologia tem a potencialidade de reduzir dramaticamente os custos de banda de *upload* dos servidores (HUANG; LI; ROSS, 2007).

Existem, basicamente, duas abordagens para prover VoD assistido por P2P. Na primeira, que possui a denominação de abordagem de vídeo único (*single-vídeo approach*), o par apenas redistribui o vídeo que está assistindo atualmente, ele não participa na redistribuição de vídeos que assistiu no passado e encontram-se armazenados na sua máquina. Na segunda abordagem, denominada de abordagem de múltiplos vídeos (*multiple-vídeo approach*), o par pode tanto redistribuir os vídeos que já assistiu quanto aquele que está assistindo atualmente. Comparada com a abordagem de múltiplos vídeos, a abordagem de vídeo simples é consideravelmente mais simples na implementação dos clientes e do *tracker* (HUANG; LI; ROSS, 2007). A seguir são enumeradas as principais características do VoD auxiliado por P2P:

1. Quando os pares não conseguem retransmitir o vídeo entre eles mesmos, o servidor os auxilia enviando a diferença, dessa forma, cada par recebe o vídeo na taxa em que foi codificado, sem perda de qualidade;

2. O servidor só envia vídeo aos pares o necessário para suprir suas necessidades, este não preocupa-se em construir um *buffer* grande de vídeo nos mesmos;

3. Quando os pares conseguem transmitir o vídeo de forma autossuficiente, o servidor cessa

seu envio. Além disso, os pares conseguem enviar vídeo adicional para eles mesmos (pré-busca) com objetivo de construir *buffers* maiores;

De modo geral, existem três estados possíveis no que tange a capacidade agregada do sistema P2P VoD de se autossustentar. O modo *surplus* ocorre quando a somatória de todas as taxas de *upload* (U) dos pares supera a somatória de todas as taxas de *download* (D), ou seja, U/D > 1. Já no modo *deficit*, o sistema não possui *upload* suficiente para seu autossustento, assim U/D < 1. O modo balanceado indica que tais taxas são iguais.

A pré-busca tem chance maior de ocorrer quando o sistema encontra-se no estado *surplus*. Portanto, é importante não só a forma como o sistema P2P se constrói para auxiliar o VoD, mas também como tal banda de *upload* remanescente é alocada para que a pré-busca possa influir em economia no servidor e manutenção da qualidade percebida pelos clientes.

No decorrer do trabalho serão adotados termos mais técnicos para denominar diferentes variáveis na distribuição de vídeo P2P com pré-busca. Em geral, na transmissão de vídeo existem três variáveis principais que controlam o estado de armazenamento e reprodução do vídeo, são elas: *buffer pointer*, *playback pointer* e *buffer level*. De forma a melhor explicar estes termos, as variáveis podem ser melhor visualizadas na Figura 2.5.

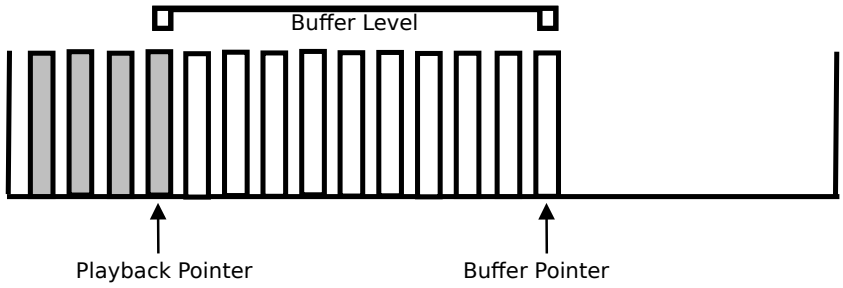

Figura 2.5: Figura ilustrativa sobre a diferença entre *playback pointer*, *buffer pointer* e *buffer level*.

Os clientes assistem o vídeo enquanto o recebem, desta forma, é necessário diferenciar as variáveis envolvidas neste processo. O *playback pointer* denota o ponto no *buffer* em que o cliente está assistindo ao vídeo, ou seja, seu ponto de reprodução do vídeo. O *buffer pointer*, refere-se à parte mais recente do vídeo que o cliente já possui no seu *buffer*, o que chegou por último. Por fim, o *buffer level* indica a diferença entre o *buffer pointer* e o *playback pointer*, ou seja, o quanto de *buffer* o cliente possui. A medida em que o *playback pointer* chega cada vez mais perto do *buffer pointer*, o *buffer level* diminui gradativamente.

Em termos de crescimento, *playback pointer* costuma avançar a uma taxa constante, em

geral, na taxa de codificação do vídeo. Em contrapartida, o *buffer pointer* avança na taxa em que o cliente consegue receber o vídeo, ou seja, na taxa em que os dados lhe são entregues. Devido a esta disparidade entre a taxa da reprodução e a taxa de recebimento do vídeo, é possível que o *playback pointer* encontre-se com o mesmo valor que o *buffer pointer*. Isso indica que a reprodução do vídeo deve ser interrompida até que se consiga obter mais vídeo para tocar. Neste cenário, o *buffer level* torna-se zero, portanto, o sistema de distribuição de vídeo deve sempre se atentar ao *buffer level* dos clientes, evitando ao máximo que esta variável seja zero, pois caso contrário, haverá interrupção na reprodução de vídeo do cliente.

Como o escopo deste trabalho está concentrado em sistemas de transmissão de vídeo sob demanda (VoD) auxiliado por P2P, a próxima seção é dedicada a apresentar algoritmos de P2P e técnicas de pré-busca de modo detalhado. É preciso ressaltar que estes algoritmos foram implementados no ambiente de simulação escolhido e serviram de base para a metodologia de avaliação de desempenho adotada no trabalho.

2.4 Algoritmos de P2P e Pré-busca

Tendo em vista o potencial do VoD assistido pelo P2P e por ser o foco central do presente trabalho, é necessário que se entenda o funcionamento da criação da topologia P2P e como trabalham as políticas de pré-busca. Nesta seção, serão abordados detalhes do algoritmo P2P empregado nesta dissertação proposto em (HUANG; LI; ROSS, 2007), bem como o detalhamento das políticas de pré-busca.

2.4.1 Algoritmo P2P: No-Prefetching

A arquitetura P2P VoD utilizada neste trabalho é descrita pelo nome de *No-Prefetching*. Nessa arquitetura, o último par (n) a chegar é servido pelo par que chegou imediatamente antes dele (n-1). Caso, não exista capacidade de *upload* suficiente nesse par (n-1) disponível para servir o vídeo, o servidor auxilia enviando a diferença. A Figura 2.6 considera que a qualidade do vídeo é de 300Kbps e as capacidades de *upload* dos pares que vão chegando, de 1 a 7, são respectivamente: 100Kbps, 200Kbps, 400Kbps, 500Kbps, 100Kbps, 400Kbps, 100Kbps.

É evidente o ganho de escalabilidade no sistema de distribuição de vídeo que se consegue obter com esta abordagem em relação ao modelo cliente-servidor. Na situação final, o servidor acaba precisando servir apenas 800Kbps (somatório das setas em vermelho na situação final: 200Kbps + 100Kbps + 200Kbps + 300Kbps), correspondendo a 2,6 vezes o vídeo original

$(800Kbps/300Kbps = 2,667Kbps)$ mesmo com um total de 7 clientes simultâneos.

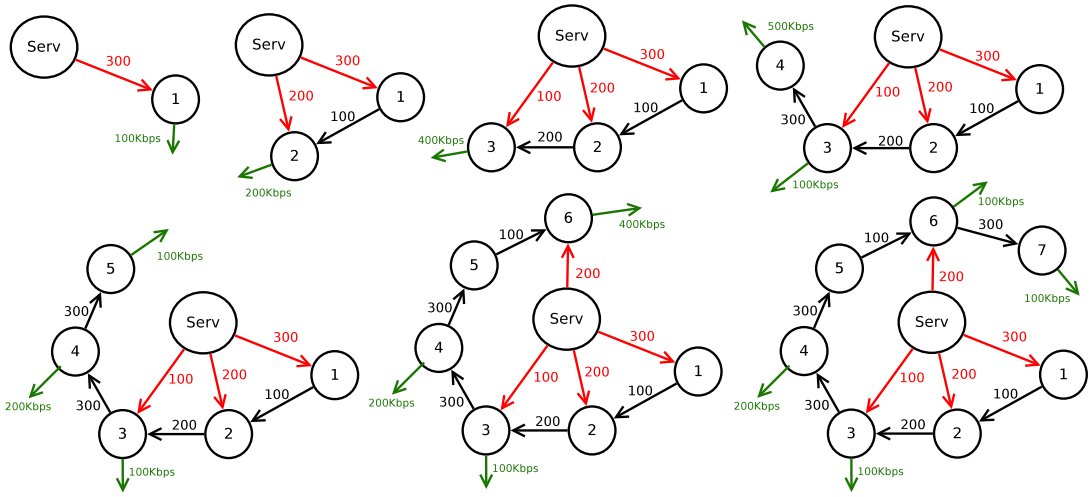

Figura 2.6: Rede sobreposta construída na arquitetura *No-Prefetching*.

As setas em verde, apontando para fora da figura, indicam banda de *upload* remanescente que os pares possuem e que não estão sendo usadas. Essas capacidades podem ser usadas para enviar conteúdo que será reproduzido pelos clientes no futuro. Assim, os nós 3 e 4 poderiam usar suas capacidades de *upload* para enviar conteúdo extra ao nó 6. Com isso, o servidor não precisaria enviar conteúdo para o nó 6 quando este possuir um *buffer* de vídeo razoável. Essas capacidades remanescentes também poderiam ser utilizadas quando houvesse algum momento de intenso congestionamento e as taxas não pudessem ser entregues como deveriam.

Em termos de classificação, esta topologia P2P (*No-prefetching*) enquadra-se na distribuição em árvore, uma vez que existe uma relação estrita de pares pai e filho. Nesta arquitetura, o escalonamento de dados é do tipo *push-based*. Assim, conclui-se que ele pode realizar apenas operações locais de congelar e reproduzir o vídeo, consistindo no provimento de um serviço *TVoD*. Devido a esta característica, essa topologia P2P pode ser usada tanto para VoD quanto para transmissão ao vivo. Como foi mencionado anteriormente, uma característica dessa topologia P2P consiste em ela não fazer uso da banda de *upload* remanescente, assim, deve-se analisar as opções para que se faça uso de tal banda para prover um serviço de pré-busca. Vale a pena ressaltar que a possível pré-busca realizada em cima desta topologia, permite apenas que os pares mais velhos enviem para os mais novos, uma vez que apenas aqueles que possuem conteúdo adicional podem enviar aos recém-chegados.

2.4.2 Pré-Busca: *Greedy*

Esta política adota uma abordagem simplificada na alocação da banda remanescente dos pares, consiste em dedicar toda banda de *upload* que sobra para enviar pré-busca ao par imediatamente posterior. Tal abordagem tem a vantagem de não necessitar de processamento por parte do par que possui banda de *upload* remanescente para decidir qual será seu receptor. A Figura 2.7 exemplifica como as bandas de *upload* são alocadas seguindo essa política.

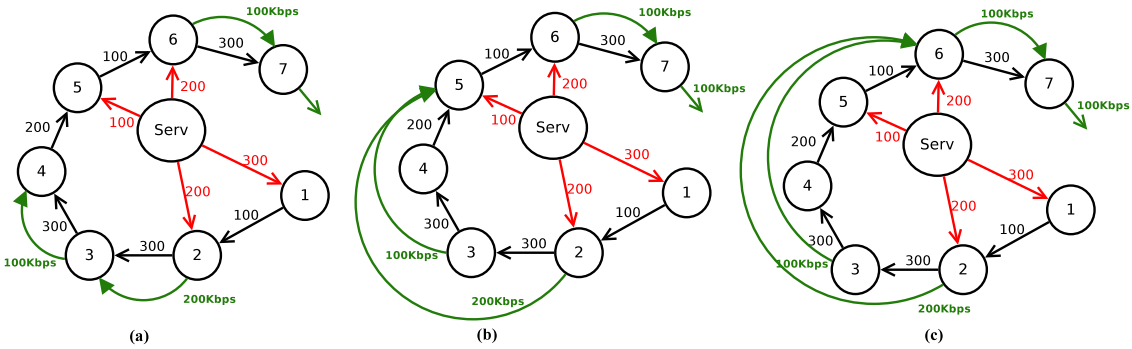

Figura 2.7: Funcionamento da pré-busca *greedy*, com 3 situações possíveis.

Sejam os pares "n", aqueles que chegaram em um momento anterior, e os pares "n+1", os que chegaram em um momento posterior. Como pode ser visto, essa política inicia (Figura 2.7a) com os pares n alocando toda sua banda aos imediatamente posteriores n+1, no entanto, quando ambos n e n+1 alcançam uma sincronia nos seus respectivos *buffers* de recebimento (*buffer point*), a pré-busca *greedy* determina que eles devem enviar o fluxo extra aos pares mais recentes. A Figura 2.7b ilustra esta situação, sendo que os pares 2, 3 e 4 já alcançaram uma sincronia nos seus *buffer points*, portanto, agora todos mandam a pré-busca para o par 5. A partir daí, o funcionamento do algoritmo passa a ser este, a Figura 5c mostra quando os *buffer points* dos pares 2, 3, 4 e 5 são iguais, portanto, agora eles mandam a pré-busca inteira para o par 6. Eventualmente, a banda remanescente do par 6 irá se juntar ao fluxo conjunto de pré-busca dos pares 2 e 3 para ajudar pares que chegam mais tarde. Tendo em vista que neste exemplo existe apenas o par 7 depois do par 6, a pré-busca deste sempre é alocada para aquele.

Como o nome em inglês sugere, a política *greedy* é gananciosa por combinar os esforços dos pares que possuem banda de pré-busca para enviá-la aos que estão precisando momentaneamente. Quando a pré-busca chega ao ultimo par, o algoritmo se reinicia, com as alocações de pré-busca sendo organizadas segundo a Figura 2.7a. O objetivo principal desta política é fazer com que seja mantida, a todo custo, a sincronia dos *buffer points* dos participantes. Esta política não se preocupa em analisar a quantidade de *buffer* que um par possui (*buffer level*).

Em sistemas de transmissão ao vivo, quando a sincronia é crucial, esta política de pré-busca pode ser mais apropriada. Por outro lado, nos sistemas de VoD tradicionais, talvez seja mais interessante haver um foco maior em manter um *buffer level* alto, uma vez que não existem restrições de sincronismo tão estritas quando nos sistemas de transmissão ao vivo. Essa percepção nos faz querer investigar um outro mecanismo de pré-busca que possa atender a este propósito.

2.4.3 Pré-Busca: *Water-leveling*

A política de pré-busca *water-leveling*, objetiva igualar o *buffer levels* de todos os pares, realizando uma alusão ao que ocorre quando recipientes de água são ligados pelo fundo e o nível de água de todos cresce igualmente. O parâmetro principal de tomada de decisão desta política é o *buffer level*. Antes de detalhar o funcionamento deste algoritmo com um pseudo-código, é necessário apresentar algumas definições. Considera-se que os pares que chegam ao sistema possam ser numerados de 1 a n, sendo o par 1 o primeiro a entrar no sistema e o par n o mais recente (ultimo a chegar). As seguintes variáveis são todas relacionadas a um dado par n: o *buffer-level*, a taxa de crescimento (pré-busca total que este par recebe de outros pares) e a banda de *upload* remanescente que pode ser usada para realizar pré-busca.

Pelo pseudo-código a seguir percebe-se que o par n envia a pré-busca de sua capacidade de *upload* inteira ao par que possui menor *buffer level* dentre os dois que chegaram imediatamente posteriores a ele (n+1 e n+2, respectivamente). Caso os *buffer levels* sejam iguais , o par n aloca a pré-busca para cada um dos pares n+1 e n+2 conforme suas taxas de pré-busca que esses já recebem atualmente. Neste caso, objetiva-se fazer com que a taxa(TC) de ambos cresça de forma que após a adição, os dois pares recebam taxas de pré-busca iguais.

Algoritmo 1 Implementação do *water-leveling*

1: **if** $B_{n+1} < B_{n+2}$ **then**
2: $\quad TC_{n+1} \leftarrow TC_{n+1} + Up_n$
3: **else if** $B_{n+1} > B_{n+2}$ **then**
4: $\quad TC_{n+2} \leftarrow TC_{n+2} + Up_n$
5: **else**
6: $\quad TC_{n+1} \leftarrow TC_{n+1} + \frac{Up_n + TC_{n+1} - TC_{n+2}}{2}$
7: $\quad TC_{n+2} \leftarrow TC_{n+1} + \frac{Up_n + TC_{n+2} - TC_{n+1}}{2}$
8: **end if**

Na Figura 2.8 são ilustradas 3 situações possíveis decorrentes do uso desta política. Em (a) visualiza-se que o par 2 verificou que $B_3 < B_4$, devido a isso alocou toda sua pré-busca para o

par 3. Fazendo uso do algoritmo *water-leveling*, o par 3 também alocou sua pré-busca para o par 5 em decorrência de $B_5 < B_4$. Em (b) o par 2 faz uma nova verificação em constata que $B_3 = B_4$, além disso, o par 3 já estava mandando pré-busca ao par 4, nesse caso, o par 2 envia pré-busca aos pares 3 e 4 de forma que a somatória de suas taxas de crescimento individuais sejam iguais (150Kbps). Finalmente, em (c) é possível verificar que após o par 3 mudar de novo o receptor de sua pré-busca, o par 2 recalcula as taxas de pré-busca que envia aos pares 3 e 4 para manter a somatória das taxas de crescimento destes iguais.

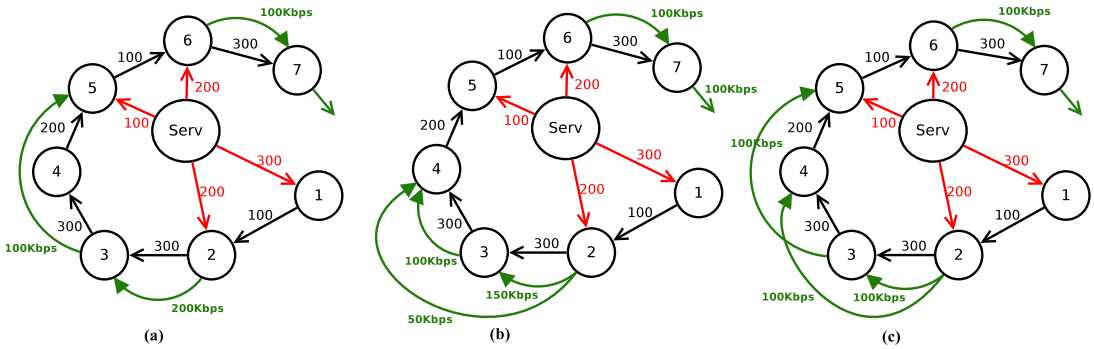

Figura 2.8: Funcionamento da pré-busca *water-leveling*, com 3 situações possíveis.

Repare em (c) que quando o par 5 possuir uma quantidade razoável de *buffer level*, ele poderá interromper momentaneamente a recepção dos 100Kbps de vídeo provenientes do servidor, assim, consumindo seu *buffer* e economizando banda no servidor.

2.5 Considerações Sobre o Capítulo

Tratando-se da comparação entre as políticas de pré-busca, conclui-se que a política *water-leveling* é mais interessante de ser submetida a um estudo mais aprofundado por possuir uma maior justiça coletiva ao comparar *buffer-levels* de dois pares e decidir para quem enviar, ao invés de enviar tudo que se tem sempre para o próximo (política *greedy*). Devido a esse motivo, o *water-leveling* será a política de pré-busca implementada no *framework* de simulação desse trabalho.

Tendo os conceitos teóricos principais sido apresentados, é pertinente investigar quais trabalhos possuem relação com o presente trabalho, ou seja, quais usam sistemas de transmissão de vídeo P2P VoD e quais já investigaram alguns aspectos relacionados à aplicação de pré-busca. O próximo capítulo visa realizar tal investigação.

3 *Trabalhos Relacionados*

Este capítulo enumera as principais pesquisas que foram realizadas na área de P2P VoD e como estas se relacionam com o presente trabalho. Com esse propósito, optou-se por separar a apresentação dos mesmos em duas seções específicas, uma que aborda os trabalhos e sistemas de P2P para transmissão de vídeo em geral e os que adotam explicitamente o uso de pré-busca. O capítulo é concluído apontando as lacunas encontradas nos trabalhos relacionados e como esta pesquisa se propõe a preenche-las.

3.1 Sistemas de Transmissão de Vídeo P2P

O PPStream é uma aplicação de bastante popularidade na China, permitindo acesso a vários canais livres que são transmitidos ao vivo, bem como filmes de várias nacionalidades diferentes por meio de P2P VoD. Um estudo recente afirma que o PPStream mudou sua camada de transporte de TCP para UDP. Foi constatado também que este programa apresentou uma perda média entre a comunicação com outros pares de 11%. Esse sistema também apresenta um bom grau de justiça, uma vez que a razão entre a quantidade de vídeo enviada a um par sobre a quantidade recebida é de 0,83 (LI; CHEN, 2009).

Esse sistema possui basicamente três tipos de mensagem de controle: mensagem de lista de pares, mensagem de mapa de *buffer* e mensagem de requisição de dados. Percebeu-se que alguns pares deste sistema sofreram atrasos de reprodução de até 210 segundos, ou seja, esta foi a diferença de tempo entre o que estava sendo enviado pelo servidor originário do vídeo principal e o que estava sendo reproduzido em alguns pares. Vale a pena ressaltar que tal atraso de reprodução foi verificado no ano de 2008, durante os testes nas Olimpíadas, quando o mesmo ainda usava TCP como protocolo base de transporte (LIANG et al., 2009).

Outro sistema também muito popular é o SopCast, ele se tornou um dos principais sistemas de vídeo P2P na Internet, o seu diferencial em relação a outros sistemas é a sua capacidade de permitir que os próprios usuários insiram seus vídeos na rede para serem distribuídos a todos.

Uma pesquisa recente mostrou que o SopCast faz uso quase que exclusivamente da camada de transporte UDP para, tanto envio de vídeo, quanto mensagens de controle. Nos experimentos realizados, o SopCast apresentou uma estabilidade considerável nas taxas de transmissão de vídeo, gerando praticamente nenhuma interrupção na reprodução do vídeo na perspectiva do cliente (MENDES; SALVADOR; NOGUEIRA, 2010).

Em geral, a qualidade dos vídeos disponibilizados variam entre 250Kbps a 400 Kbps, sendo que alguns canais podem chegar até 800Kbps. Como a maioria dos sistemas deste tipo, no SopCast, do número total de pacotes transmitidos, percebeu-se que em média 60% são pacotes de controle e 40% são pacotes de dados (vídeo), no entanto, vale a pena lembrar que os pacotes de controle possuem um tamanho máximo de 100 *bytes*, e o pacotes de vídeo possuem um tamanho médio de 1400 *bytes* (LU et al., 2009). Um outro estudo mostrou que este sistema faz uso de *buffer* hierárquico. Ou seja, o SopCast possui um *buffer* interno que ele gerencia para saber quais partes do vídeo precisam ser obtidas, assim que ele percebe que existe uma porção de vídeo contígua, ele disponibiliza para um *player* de vídeo reproduzir. Desta forma, o *player* de vídeo faz *buffer* do que o SopCast disponibiliza e somente após um limiar configurado no *player* que se inicia a reprodução do vídeo (SENTINELLI et al., 2007).

O UUSee foi desenvolvido principalmente para transmissão de vídeo ao vivo, portanto, adaptaram o algoritmo para as restrições de tempo mais relaxadas do P2P VoD. Esse sistema implementa a funcionalidade de cache duplo, que permite a presença de dois *buffers* de granularidade diferentes: o primeiro, de maior granularidade, preocupa-se em indicar se um grupo de blocos está quase completo ou não, geralmente é atribuído um *bit* para o valor 1 quando tal grupo de segmento está 80% completo. Tratando-se da granularidade menor, é estabelecido se um segmento dentro de um grupo foi ou não completamente recebido, os mapas de segmentos possuem um tamanho médio de 10 a 30 *bytes*, quando um segmento é completamente recebido e decodificado, em tal mapa é setado um bit para 1. O UUSee faz uso de mapas de *buffers* relativamente pequenos pois usa um tamanho de segmento grande (500KB), o que é muito em relação a outros sistemas comerciais como o PPLive (14KB), por exemplo. Tal decisão foi tomada para diminuir a sobrecarga de transferência de mapas de buffer, além disso, tal tamanho de segmento é também devido à codificação de rede empregada (LIU; GUO; LIANG, 2008).

Também vale a pena ressaltar que o UUSee faz uso, unicamente, do UDP na camada de transporte, o que é mais apropriado em sistemas que fazem muito uso de codificação de rede[1]. Esse é um dos poucos sistemas que suportam vídeos de alta qualidade, as taxas de codificação

[1] A codificação de rede especifica padrões para dividir fluxos de vídeo em vários outros de menor qualidade sendo que a aplicação que os recebe seja capaz de os compor e obter qualidades gradativamente maiores a medida em que se compõe os sub-fluxos existentes.

dos vídeos variam de 264Kbps à 1.3Mbps (LIU et al., 2010).

Já o PPLive afirma ter sido o primeiro sistema de P2P VoD que introduziu o conceito de *buffer* hierárquico e aplicou em sua implementação. Em uma medição realizada em 2006 foi constatado que o PPLive obteve uma quantidade de usuários simultâneos próximo aos 200 mil (SENTINELLI et al., 2007). Sabe-se também que as taxas de recebimento dos vídeo pelos usuários variaram entre 400Kbps e 800Kbps, sendo que a taxa agregada foi em torno de 100Gbps (SENTINELLI et al., 2007).

O PPLive faz uso de um mecanismo denominado de *taxa de disponibilidade por demanda* (*Availability to Demand Ratio* - ATD), que representa a razão entre os pares que possuem pedaços de um vídeo e os que estão assistindo este vídeo. É definido um limiar para essa taxa baseado na capacidade de *upload* dos pares. Baseado no ATD, o servidor de vídeo sabe de quantos pares são necessários para que a sua participação no auxílio dos mesmos pode ser fortemente diminuída. Um diferencial que o PPLive possui em relação aos outros sistemas consiste em implementar mecanismos de autenticação de pedaços, desta forma, ele é resiliente em casos de ataques de poluição de conteúdo (SENTINELLI et al., 2007).

O Zatoo é um sistema de vídeo P2P que foi concebido com o objetivo de disponibilizar a transmissão dos jogos da copa do mundo de 2006. Deste então a implementação foi sendo melhorada e os desenvolvedores decidiram torná-lo um sistema comercial disponível na Internet. Diferentemente dos outros sistemas do tipo, o Zatoo possibilita apenas que usuários de um país específico acessem conteúdos de canais específicos, ou seja, ele se propõe a ser usado somente em ocasiões previamente preparadas (ZATTOO. . . , 2011). Esse pode ser considerado um sistema puramente *"P2P Live Stream"* por ser usado apenas em transmissões ao vivo. Como foi desenvolvido para audiências específicas, este sistema faz uso de três servidores: servidor de vídeo que converte o sinal de televisão para ser transmitido na Internet, e o servidor de autenticação que fornece *tickets* de tempo limitado ao usuário, o servidor *tracker*, que verifica a validade do *ticket* que o usuário apresenta e retorna a lista de pares que fazem parte da rede sobreposta do canal de vídeo solicitado.

Mesmo que o Zatoo seja um sistema de distribuição P2P em malha, ele constroi uma distribuição *push-based* em cima de um circuito virtual, o que seria algo similar a uma arvore de distribuição dentro da malha. Como este sistema usa codificação de rede, a camada base da codificação é entregue pelos pares na organização em malha (*pull-based*), e as camadas que conferem mais qualidade ao vídeo, são distribuídas por meio de tal circuito virtual. Esse mecanismo tenta beneficiar-se das vantagens de uma arquitetura híbrida de distribuição P2P. Quando pares do circuito virtual mudam de canal ou saem do sistema, o mesmo tem tempo de se re-

cuperar pois os dados afetados conferem apenas qualidade, não potencializando interrupção na reprodução do vídeo. Também é importante ressaltar que este sistema faz uso de TCP ou UDP, dependendo das condições da rede e das restrições nos clientes (CHANG; JAMIN; WANG, 2009).

Os sistemas até agora avaliados não citam em seus mecanismos de tomada de decisão se eles realizam uso da pré-busca ou não. Então, é necessário estudar os sistemas que afirmam usar tal pré-busca explicitamente com objetivo de obter conclusões sobre os potenciais benefícios dessa tecnologia.

3.2 Sistemas de Transmissão de Vídeo P2P que Usam Pré-busca

A pesquisa de (SHARMA; BESTAVROS; MATTA, 2005), foi um dos trabalhos pioneiros ao abordar uma política de distribuição de vídeo com pré-busca, objetivando a redução de carga no servidor. É proposto um protocolo de pré-busca distribuído em que os pares obtém partes do vídeo à frente de seus pontos de reprodução. O intuito dessa solução é de *(i)* poder ajudar outros pares com tal pré-busca ou *(ii)* usá-la caso um par do qual se recebe o vídeo saia do sistema sem aviso prévio. A solução abordada é diferente do mecanismo *cache-and-relay* uma vez que, neste, quando um par do qual se recebe o vídeo sai do sistema sem aviso prévio, o par receptor é obrigado a buscar o fluxo de vídeo no servidor imediatamente, consequentemente, causando descontinuidade na reprodução, aumento na carga do servidor e da rede.

Por meio de análises matemáticas e simulações de eventos discretos, é mostrado que o uso de redes de distribuição *multicast* assíncronas contendo vários pares emissores aliada ao uso da pré-busca reduz bastante a carga de *upload* do servidor e torna-se uma solução escalável. Neste sistema, considera-se que os receptores de vídeo possuem um *buffer* de recepção limitado, portanto, é usada uma atualização de *buffer* circular. No mecanismo de pré-busca adotado, quando um novo par chega ao sistema, este encarrega-se de pedir aos pares anteriores (linearmente em ordem dos mais recentes para os mais antigos) o fluxo extra, caso estes possuam em seus *buffers*. Uma vez que o novo par encontra outro para receber a pré-busca, ele continua essa conexão até acabar de obter o vídeo ou o par emissor sair do sistema. Esse trabalho também preocupa-se em modelar matematicamente o tempo envolvido nessa busca por pares iniciais quando um par acaba de chegar ao sistema.

Em (ZHENG; SHEN; LI, 2005) se investigou uma política de pré-busca com fins de minimizar o tempo de *seek* quando usuários de sistemas de vídeo P2P realizam operações de VCR,

ou seja, quando tentam mudar o pedaço de vídeo corrente que está sendo reproduzido. A validação da proposta se dá por análise de *logs* de uma sessão de vídeo P2P previamente gravada, sendo que não foi feito uso de pré-busca, portanto, o artigo expõe os ganhos que poderiam ser atingidos neste cenário caso o algoritmo de pré-busca proposto fosse usado. São extraídas várias medidas estatísticas relativas a padrões de comportamento quando um usuário assiste aos vídeos. Os padrões obtidos por este trabalho indicam que a modelagem de operações de VCR aleatórias (*random seeks*) conseguem modelar de forma satisfatória o comportamento das operações de VCR dos usuários em sistemas reais.

Os autores argumentam que o problema de decidir quais pedaços de vídeo escolher dentre um grupo bem delimitado assemelha-se ao problema da teoria da quantização escalar (WANG; OSTERMANN; ZHANG, 2002). Por meio de analogias, adapta-se o algoritmo de *Lloyd* (que resolve o problema da quantização escalar) à realidade do problema investigado. A partir daí é proposto um algoritmo de pré-busca hierárquico que procura dividir os pedaços de vídeo para os quais se deseja realizar o *seek* em sub-blocos sempre de 2, dentro de cada sub-bloco se aplica o algoritmo adaptado e *Lloyd* para decidir em quais pedaços deve-se realizar a pré-busca. O algoritmo proposto também implementa uma politica de atualização de *buffer*, supondo um *buffer* limitado. A decisão sobre quanto de *buffer* reservar para a pré-busca é modificada dinamicamente baseando-se na popularidade dos pedaços de vídeo a serem obtidos.

Outros autores alegam que em casos mais realísticos os usuários não realizam *seeks* aleatórios, mas sim baseados em outros fatores. Objetivando levar essa característica em consideração, (HE; GUAN, 2009) realiza uma modelagem matemática para definir formalmente o problema por meio de programação linear. A definição formal do problema leva em conta que os usuários normalmente avançam ou retrocedem conforme uma dada lógica (avanços no final do vídeo são menores, no começo são maiores, no meio tende-se a avançar e retroceder de forma mais intercalada, etc.). Além disso, a modelagem também considera medidas de popularidade dos pedaços de vídeo. Com a solução das equações montadas, obtém-se, em um dado momento, a que taxa cada pedaço de vídeo futuro (que o cliente ainda não precisa no momento) deve ser obtido. Desta forma o algoritmo de pré-busca ordena os pedaços de vídeo com maiores prioridades, definidas pela solução da equação montada, e os obtém um a um.

O algoritmo de pré-busca é sempre retroalimentado pelas medidas de probabilidade de popularidade que vão sendo atualizadas a cada novo *seek* nos vídeos do sistema. Os resultados da proposta são validados por meio de simulação de eventos discretos. Diferentemente de outras pesquisas da área que envolvem simulação, considerou-se que todos os pares envolvidos no sistema possuem capacidade de *upload* de, no mínimo, duas vezes a qualidade do vídeo trans-

mitido. Como resultados das simulações é mostrado que consegue-se obter uma minimização no tempo entre uma operação de *seek* e reinício da reprodução do vídeo.

Outro artigo que explora análise de logs é o (HUANG; LI; ROSS, 2007) em que foram analizados *logs* de distribuição de vídeo do MSN Videos durante um mês. Foram realizadas caracterizações sobre as bandas de *download* dos usuários que puderam ser percebidas pelo servidor, além disso, identificou-se a existência de vídeos que são mais populares para estimar a quantidade de usuários simultâneos. Em cima de tais caracterizações foram realizadas simulações que mostram o quanto o servidor de vídeo poderia obter de economia caso o P2P VoD fosse usado e caso este adicionasse a pré-busca. Os resultados indicam que poderiam ter havido ganhos muito grandes na economia de banda do servidor de vídeo.

Foi proposto um algoritmo de P2P com estrutura predominantemente em árvore, denominado *no-prefetching*, e dois algoritmos de pré-busca, denominados *greedy* e *water-leveling*, que trabalham em cima dessa topologia P2P. Além de simular e economia que poderia ser alcançada do servidor de vídeos do MSN, os autores também simulam a diferença de desempenho entre usar ou não a pré-busca na distribuição de vídeo em um cenário hipotético. Tal simulação é realizada por um simulador de eventos discretos implementado para este propósito. A montagem de tal cenário foi inspirada nos parâmetros coletados das caracterizações de tráfego realizadas previamente. As simulações indicam que as duas políticas de pré-busca obtém um resultado similar. É mostrado que é possível a obtenção de uma economia de mais de 5 vezes na banda de *upload* do servidor quando tais políticas são usadas. Como este trabalho realiza sua avaliação usando um simulador de eventos discretos, não se leva em consideração algumas variáveis inerentes da infraestrutura de rede, além de não analisar o qualidade do conteúdo recebido na perspectiva do cliente.

Tendo em vista ser também fundamental analisar a qualidade de vídeo percebida pelo cliente, a pesquisa de (SHEN et al., 2006) estuda como o uso de políticas de pré-busca podem melhorar a qualidade percebida pelo usuário, fazendo uso de codificação de rede, ou seja, dividindo os vídeos em fluxos independentes que necessitam ser recriados nos pares. Argumenta-se que é interessante dividir o vídeo em vários fluxos diferentes pois caso um dos fluxos não chegue (seja por congestionamento na rede ou saída de um par emissor) o receptor sofre apenas uma pequena perda de qualidade do vídeo, e não interrupção abrupta do mesmo. Em particular na aplicação da pré-busca, tal par receptor pode nem perceber perda de qualidade caso ele já tenha recebido pré-busca e possuam um *buffer* acumulado suficiente. No sistema proposto, quando um novo par chega ao sistema, ele contata o servidor e o servidor indica quais pares enviarão o fluxo os sub-fluxos de vídeo. Quando um dado par começa a enviar o fluxo de vídeo

ao receptor, caso o emissor possua banda de *upload* extra, este envia um fluxo extra que corresponde à pré-busca. O problema investigado pelo artigo é sobre como determinar a qualidade e quantidade dos sub-fluxos de vídeo a serem enviados à um novo par de forma que todos os pares consigam receber uma quantidade razoável de pré-busca.

São propostos dois algoritmos de pré-busca o primeiro é denominado de "algoritmo ótimo *off-line*", neste algoritmo sabe-se à *priori* a capacidade de *download/upload* de todos os pares e os tempos no qual os mesmos chegam ao sistema. Com essa informação é determinado uma alocação ótima dos fluxos para maximizar a qualidade percebida pelos usuários. Tal algoritmo ótimo foi elaborado para delinear o limite máximo de desempenho que o sistema pode obter com vistas a obter uma base de comparação com o outro algoritmo *on-line* proposto. O segundo algoritmo é o denominado *heurístico*, o servidor principal realiza estimativas sobre a capacidade agregada de *upload* do sistema, baseando-se nesta, o mesmo consegue decidir quantos sub-fluxos serão gerados quando um novo par chega ao sistema. Caso se utilize a codificação de rede por múltiplos descritores (MDC), como cada descritor de vídeo possui a mesma importância, os fluxos de pré-busca são alocados com preferência aos pares de menor nível de *buffer*. Caso se use codificação em camadas, é usada uma formula que leva em consideração o grau de importância relativa entre as camadas de vídeo e o nível de *buffer* do receptor em questão. Os resultados indicam que a codificação usando múltiplos descritores obtém uma performance maior (diferença de até 1db na qualidade do vídeo) do que a abordagem usando múltiplas camadas. No caso do MDC, a diferença na qualidade do vídeo entre usar ou não a pré-busca foi de até 2db.

Como foi abordado anteriormente, a política de escalonamento de pedaços em sistemas P2P pode ser classificada em duas principais: *pull-based* e *push-based*. Existe a arquitetura híbrida, denominada, *push-pull* que realiza uma combinação dessas duas abordagens. Esta arquitetura é usada em (GAO et al., 2010), foi proposto um sistema de P2P VoD de topologia híbrida do tipo *push-pull* que faz uso de codificação de rede, suporta operações de VCR e realiza pré-busca. A abordagem *push-pull* utilizada funciona da seguinte forma: o vídeo é divido em sub-fluxos e quando um par solicita um pedaço do vídeo relativo a um fluxo a outro par, o par emissor infere que o receptor estaria interessado no fluxo inteiro, então, a partir daí o par emissor envia ao receptor sem a necessidade de futuras solicitações, ou seja, realizando uma operação de *push*.O receptor necessita avisar explicitamente ao emissor caso não necessite mais do fluxo para que seja cessada a transmissão.

A pré-busca ocorre quando o par receptor solicita o fim do envio de um dado sub-fluxo, nesse momento, o emissor seleciona um grupo de pedaços aleatórios de vídeo que ele acredita

que o receptor não tenha e os envia sem o uso de codificação de rede. É especificado que a decisão de quando parar o envio da pré-busca se baseia em um algoritmo que usa funções ponderadas. Tanto detalhes da política de atualização de *buffer* quanto do algoritmo supracitado, não são fornecidos. O trabalho não implementa e nem simula o que foi proposto, no entanto, ele apresenta uma visão geral de como seria o funcionamento do sistema por meio de diagrama de blocos enfatizando a comunicação entre os diferentes módulos a serem implementados.

A pesquisa de (BIAO; ZHEN; YAOHUA, 2010) propõe uma nova topologia de distribuição P2P VoD baseada em DHTs hierárquicas de dois níveis e que faz uso de pré-busca. As hierarquias possuem o propósito de separar a abrangência geográfica dos pares. Existem as DHTs locais que agrupam pares que estariam próximos geograficamente e uma DHT global que serve como ponte no encaminhamento de informação entre pares de DHTs locais distintas. Esse sistema implementa uma pré-busca em que os pares solicitam o fluxo extra de outros pares até que o tamanho do *buffer* construído chegue a um determinado limite. A validação da proposta se dá por meio de simulação, onde, diferente dos outros trabalhos, considera-se que a capacidade de banda para pré-busca é de, no mínimo, a taxa de codificação do vídeo.

Uma outra abordagem faz uso de pares denominados "ajudantes", estes possuem o objetivo de reduzir a carga de *upload* do servidor, tais ajudantes são pares que participam da rede unicamente para realizar *upload*, não sendo clientes (ZHANG et al., 2009). Esta abordagem se assemelha com as CDNs, no entanto, os pares ajudantes não são propriedade de uma empresa, mas sim usuários que estão dispostos a entrarem na rede P2P unicamente para ajudar. O algoritmo proposto leva em consideração a dinamicidade de tais ajudantes. As previsões realizadas por meio de simulação de eventos discretos mostram que para um vídeo de 512Kbps, são necessários 241 nós ajudantes para manter uma economia considerável no *upload* do servidor de vídeo. O trabalho citado se assemelha com o desta dissertação pois no contexto deste trabalho, os clientes do sistema que possuem banda de *upload* grande também atuam como "ajudantes"por enviar a pré-busca, no entanto, estes também participam da distribuição como clientes.

Como pôde ser visto anteriormente, a tendência atual aponta para a utilização da pré-busca como forma de obter potenciais benefícios para, principalmente, os servidores. Em geral, as pesquisas nesta área envolvem, em sua maioria, simulação por ser uma forma mais fácil, rápida e barata de se investigar os mecanismos de pré-busca. No entanto, as pesquisas citadas relevam a existência da topologia de rede simulada, bem como a possibilidade de ocorrerem perdas além de não investigarem de forma direta o papel da camada de transporte no desempenho global do sistema. As pesquisas que focam na qualidade percebida pelo usuário são as que abordam

codificação de rede, as demais não observam também esta característica adicional. Tendo em vista estes pontos não investigados nas pesquisas apresentadas, este trabalho procura contribuir nestes aspectos, simulando uma topologia de rede com características mais realísticas, com presença de perdas e congestionamentos nos enlaces, considerando diferentes bandas em pares clientes, avaliando diferentes camadas de transporte e analisando a percepção dos clientes em tal difusão.

4 Ambiente de Simulação

Neste capítulo são expostas as decisões tomadas no que concerne o ambiente de simulação usado neste trabalho. É discutido o motivo da escolha do simulador de rede usado, da importância em simular tráfego de fundo e qual ferramenta foi usada para tal geração. São expostos detalhes do *framework* desenvolvido, tais como: arquivos existentes, topologia implementada, parâmetros existentes, tipos de gráficos possíveis, etc. Por fim são apresentados os parâmetros usados na análise dessa dissertação.

4.1 Simulador de Rede

Para realização das simulações deste trabalho foi usado o simulador de rede Network Simulator 2 (NS2) (THE. . . , 2011), em sua versão 2.34. Esse simulador foi inicialmente concebido em 1989 como sendo uma variante do "REAL Network Simulator"da *Cornell University*. No decorrer dos anos ele seu desenvolvimento foi apoiado por várias instituições de notável representação mundial, entre algumas: DARPA (*Defense Advanced Research Projects Agency*), NSF (*National Science Foundation*, EUA), XEROX PARC (*Palo Alto Research Center*) e *Sun Microsystems*. Ao longo de mais de 20 anos de desenvolvimento esta ferramenta adquiriu grande maturidade e atualmente agrega módulos e funcionalidades das mais variadas que também foram implementadas por pesquisadores independentes. Além de tudo isso, o NS2 é uma ferramenta de código aberto, tendo em sua licença permissão para distribuição gratuita, adicionalmente, é uma ferramenta que já foi portada para os mais difundidos sistemas operacionais (Windows, Linux, FreeBSD, SunOS, etc).

O NS2 implementa vários protocolos de transporte, entre eles, dois que foram usados nessa dissertação são: TCP e UDP. Esse simulador foi desenvolvido na linguagem C++, o uso de tal linguagem garante aos módulos implementados, principalmente, eficiência de execução. A interface com o usuário que deseja realizar uma simulação é realizada por meio da linguagem interpretada Otcl/tcl. Esta linguagem permite que vários parâmetros de simulação sejam variados sem que seja necessário modificar o código-fonte do NS2 e recompilá-lo.

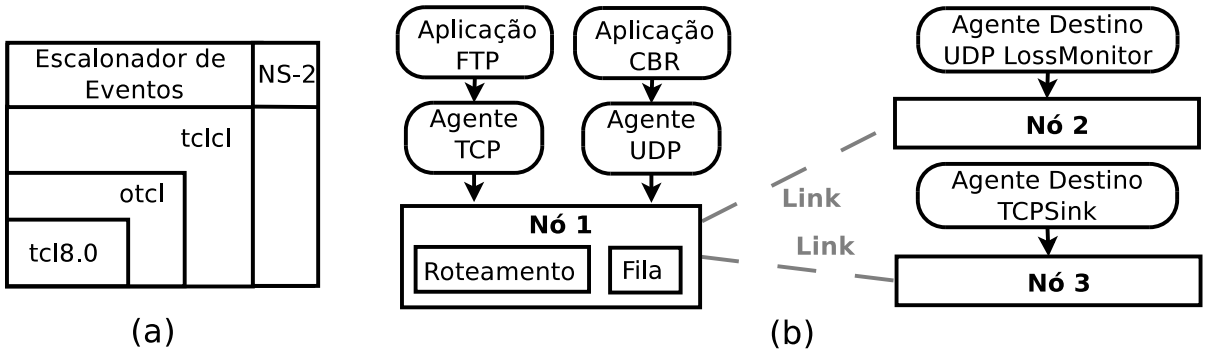

Figura 4.1: Visão geral do simulador NS2 com (a) diagrama de blocos de sua estrutura (NS..., 2001) e (b) como se implementa o uso de diferentes protocolos de transporte/aplicação

A Figura 4.1a mostra um esquemático simplificado da hierarquia entre as linguagens de programação que compõe o NS2, nota-se que a única linguagem com a qual o usuário se interage é o "tcl8.0". A Figura 4.1b exibe como são construídas conexões de dados no NS2. Em cada nó pode-se atribuir pilhas de protocolos, no caso da Figura, atribuiu-se a pilha TCP e UDP. Em cima da pilha TCP foi inserida a aplicação de FTP e no topo da pilha UDP foi inserida a aplicação CBR (*Constant Bitrate*). Mais aplicações podem ser inseridas na mesma pilha de protocolo. Os destino de uma aplicação sobre UDP é implementado pela pilha "*LossMonitor*"e o destino de uma aplicação TCP é implementada pela pilha "*TCPSink*". Em suma, o nó 1 envia fluxos de FTP para o nó 3 e, ao mesmo tempo, envia fluxos de CBR para o nó 2.

Caso esta dissertação se propusesse a implementar uma nova funcionalidade no simulador, seria necessária a programação em C++ e OTcl, no entanto, como apenas fez-se uso de funcionalidades já existentes, a programação se deu apenas em tcl8.0.

Em suma, o NS2 foi usado neste trabalho por ser um dos simuladores de rede mais maduros e amplamente testados na comunidade científica, além de possuir uma extensiva documentação que é facilmente acessível *on-line*.

4.2 Geração de Tráfego Sintético

Com objetivo de simular um ambiente de rede com características mais realísticas, adotou-se o uso de geração de tráfego sintético. Tal tráfego consiste em fluxos de transferência de dados entre nós que são posicionados na simulação unicamente para este propósito. Os termos "tráfego sintético"e "tráfego de fundo"serão usados como sinônimo no decorrer deste trabalho. O tráfego de fundo é essencial quando se deseja simular o desempenho de um algoritmo P2P em um ambiente de Internet real. Em um ambiente deste tipo, vários fatores como: atraso, tempo

de ida e volta (RTT), vazão, perdas nos *links* e pacotes que chegam fora de ordem influenciam no desempenho global de sistemas de vídeo P2P.

Tendo em vista a importância da inserção desta característica na simulação deste trabalho, buscou-se por ferramentas que pudessem gerar o tráfego de fundo necessário. A ferramenta escolhida para este fim é denominada *TCP Evaluation Suite* (TCP..., 2007), trata-se de um conjunto de *scripts* implementados no NS2 que possibilitam a geração de tráfego de rede sintético. O propósito principal desta ferramenta é avaliar o desempenho de diferentes protocolos de transporte em uma simulação de rede realística com a presença de um tráfego de fundo elaborado levando em consideração vários aspectos de tráfegos da própria Internet. Nesta ferramenta, o comportamento de um usuário da Internet é modelado por meio de requisições HTTP e leva em consideração tempos de pausa (*thinking time*) que são comuns de ocorrer na mesma. Por outro lado, esta ferramenta também simula a presença de fluxos mais agressivos, por exemplo, transferência de arquivos grandes como DVDs por meio de FTP.

Na nomenclatura do *TCP Suite*, a transferência de arquivos grandes é denominada: fluxo longo e a transferência de arquivos pequenos é denominado: fluxo curto. O tamanho dos arquivos de ambos fluxos longos e curtos seguem uma distribuição de *pareto* (seguindo estatística da Internet) (CROVELLA; BESTAVROS, 1997) (ARLITT; WILLIAMSON, 1996).

Os tempos de início das conexões seguem uma distribuição exponencial. Esta ferramenta também permite que sejam gerados tráfegos de fundo diferentes para cada simulação, no entanto, ela garante que a média da utilização dos *links* de tais tráfegos seja igual em todas as simulações. Esta característica é vantajosa pois permite que aplicações de vídeo P2P de diferentes configurações sejam simuladas em um ambiente que possui um perfil de tráfego de fundo homogêneo, embora diferente.

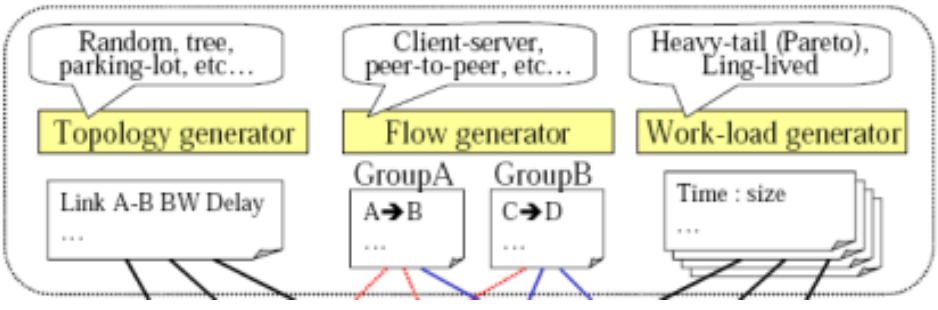

Figura 4.2: Visão geral das funcionalidades do TCP Suite

A Figura 4.2 permite observar uma visão geral da organização do *TCP Suite*, neste trabalho foram usados os módulos "*Topology Gererator*"e "*Work-load generator*". O primeiro se encar-

rega de estabelecer as conexões dos *links* de *backbone*, suportando duas topologias: *dumb-bell* e *parking-lot*. O segundo é responsável por gerar o tráfego de fundo que caracteriza o comportamento da Internet, provendo conexões longas e curtas. O outro módulo exibido na figura (*Flow generator*) se encarrega de gerar fluxos que trafegarão no meio do ambiente congestionado e estarão sujeitos à avaliação. Existe outro módulo que não foi mostrado na figura, o "*analysis*"que é responsável por gerar gráficos de desempenho a partir das simulações realizadas.

4.3 Arquitetura do Framework

O *framework* desenvolvido foi implementado tomando como base a organização de pastas e arquivos do *TCP Suite*. Existem três módulos principais, o de topologia (*topology*) permite criar um arquivo-texto em formato bem definido para especificar as conexões e atrasos dos *links* de *backbone*, pares do sistema P2P e geradores de tráfego de fundo. O módulo *executor* implementa os *scripts* para submeter a simulação para execução, por fim, o módulo *analysis* permite a geração automática de vários gráficos padronizados apartir dos resultados da simulação.

4.3.1 Organização dos Arquivos

Os três módulos principais correspondem, cada um, a uma respectiva pasta.

Topology

A escolha da topologia de rede é de fundamental importância em simulação. Como foi mencionado anteriormente, o *TCP Suite* fornece dois tipos de topologia nativo para o *backbone* da rede: *Dumbbell* e *Parking-lot*.

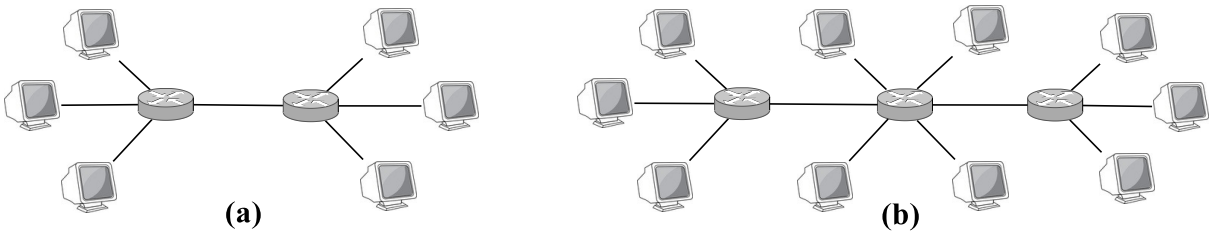

Figura 4.3: Topologias comuns de rede: (a) *Dumbbell* e (b) *Parking-lott*.

A característica principal da topologia *Dumbbell* é permitir que vários usuários concorram por um enlace compartilhado. É possível simular vários cenários simplesmente mudando a capacidade de banda e atraso do *link* central. Já a topologia *Parking-lot* é uma generalização da

anterior, que permite experimentar o fenômeno de múltiplos gargalos e inter-relacionamento de fluxos em ambas direções. Uma limitação que pode ser notada na topologia *Parking-lot* é a sua estrutura rígida de encaminhamento de pacotes, nesta arquitetura os pacotes seguem sempre um mesmo caminho entre uma origem e um destino. Tal deficiência nos motivou a implementar uma nova topologia para simulação.

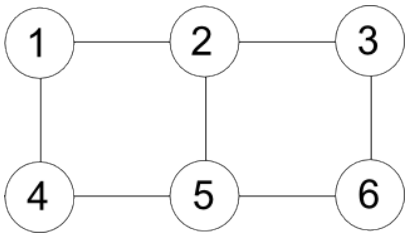

Figura 4.4: Topologia implantada na simulação: *Double Parking-lot*.

No *framework* desenvolvido foi implementada a topologia que denominamos "*Double Parking-lot*", esta permite expressar um ambiente de núcleo de rede mais elaborado. Nesta topologia pacotes podem ter caminhos diferentes entre uma origem e um destino apresentando a mesma distância em saltos. A numeração exibida na Figura 4.4 é refente à identificação dos nós de *backbone* usados na simulação.

Dentro deste módulo, constam os arquivos:,

- **topogen.tcl**: Gera dois arquivos de saída que definem a topologia do *Backbone* (*model-topology*) e nós que geram o tráfego de fundo (*model-flow*);
- **cbrgen.tcl**: Gera um arquivo que define os pares do sistema de distribuição P2P. Esse script implementa as diferentes ordens de chegada possíveis que são abordadas mais adiante neste capítulo. A saída é o arquivo *model-cbrs*;
- **execute.sh**: Permite especificar todos os parâmetros a serem usados na criação da topologia, chamando a execução dos dois *scripts* abordados anteriormente;
- **create_topologyies.sh**: Chama a execução do *script* anterior várias vezes para criar modelos de simulação que variam a quantidade de pares com capacidades diferentes de *upload*/*download*;

Executor

Os arquivos referentes à execução da simulação são:

- **script-udp.tcl**: Responsável por implementar a simulação usando protocolo UDP;
- **script-tcp.tcl**: Responsável por implementar a simulação usando protocolo TCP;
- **roda_experimento.sh**: Documenta os parâmetros possíveis e permite chamar o script de execução adequado, podendo usar *cluster* ou não;
- **submit_jobs.sh**: Permite submeter várias execuções do *script* anterior para rodar em um sistema em *cluster*;

Analysis

Contém os *scripts* que geram gráficos para cada simulação. Os gráficos são gerados fazendo-se uso do programa *gnuplot*.

- **compare_servers.sh**: Gera gráficos comparativos entre as taxas de upload e número de clientes simultâneos quando se usa ou não a pré-busca;
- **compare_links.sh**: Gera gráficos das taxas de *upload* e *download* encaminhadas em nós de *Backbone*, além de gerar gráficos de barras exibindo as perdas;
- **gera_discont_simples.sh**: Permite gerar dois tipos de gráficos que avaliam a qualidade de recepção de vídeo do usuário: um indicando a quantidade de interrupções que ocorreram na reprodução do vídeo e outro indicando a duração das mesmas;
- **gera_graficos.sh**: Possibilita gerar gráficos comparativos entre taxas de *upload* e *download* de cada par do sistema P2P e gráficos que agrupam os mesmos de 3 em 3 para exibir a evolução de seus respectivos *buffer-levels* e *playback-pointers*.
- **generate_statistics.sh**: Este *script* é o executável principal deste módulo pois agrupa a chamada de todas as funcionalidades em um único lugar, documentando os tipos de gráficos possíveis e permitindo a geração dos mesmos de forma seletiva;

4.3.2 Parâmetros Existentes

Como o *framework* foi concebido de forma configurável, é possível alterar seu comportamento por meio da escolha de diversos parâmetros existentes. É permitido variar de forma fácil 14 parâmetros, dentre eles: (i) número de nós do *Backbone*, (ii) número de conexões longas, (iii) número de conexões curtas, (iv) tempo de simulação, (v) qualidade do vídeo, capacidades de *dowloand/upload* dos *links* de (vi) *Backbone* e (vii) pares, (viii) quantidade de pares do

sistema P2P, (ix) quantidade mínima de vídeo antes de iniciar a reprodução do vídeo, (x) quantidade mínima de vídeo antes de parar de receber fluxos do servidor, (xi) taxa e (xii) ordem de chegada dos pares ao sistema, (xiii) duração do intervalo para executar a tomada de decisão da política de pré-busca dos pares e (xiv) camada de transporte usada nos fluxos da aplicação. De tais parâmetros, é pertinente destacar um deles, a ordem de chegada dos pares ao sistema.

O *framework* desenvolvido considera a existência de dois tipos de pares com características de *upload*/*download* diferentes, o que é necessário para que se consiga simular a política de pré-busca *water-leveling* explicada no capítulo anterior. Com a existência de tais capacidades diferentes nos pares, é necessário considerar a ordem em que os mesmos chegam ao sistema. Em geral, pesquisas que realizam simulação de sistemas P2P tomam uma das duas atitudes listadas a seguir: (i) não consideram a existência de pares com capacidades de *upload*/*download* diferentes, ou (ii) consideram a ordem de chegada de forma intercalada. Esta ordem de chegada indica que os pares com diferentes bandas de *upload* chegam de forma intercalada ao sistema.

A ordem de chegada é considerada no *framework* e nas avaliações desta dissertação pois o algoritmo P2P usado para formar a rede sobreposta de distribuição se baseia em uma arquitetura em árvore. Como é bem conhecido, a estrutura de distribuição em árvore possui uma topologia de encaminhamento estática, portanto, a ordem de chegada dos pares pode impactar bastante no desempenho global. No *framework*, foram implementados 4 casos de ordem de chegada: a melhor ordem, a ordem intercalada, a pior ordem e a ordem aleatória.

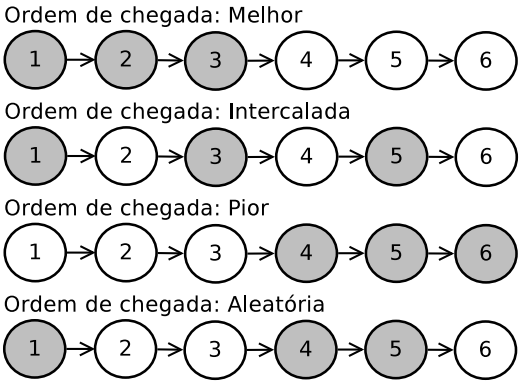

Figura 4.5: Ilustração das ordens de chegada consideradas na simulação.

Na Figura 4.5 os círculos escurecidos correspondem a pares do tipo 1 (*upload* maior) e os círculos de fundo branco correspondem a pares do tipo 2 (*upload* menor). Os números dentro dos círculos correspondem à ordem em que os mesmos chegam ao sistema. Na ordem de chegada aleatória, a posição dos nós tipo 1 e 2 são sorteadas, sendo que ambos os tipos possuem a mesma chance de serem alocados em qualquer posição de chegada.

4.3.3 Diagrama de Blocos

Esta subseção tem por objetivo explicar de forma resumida o funcionamento do *framework* desenvolvido. A Figura 4.6 exibe um diagrama que separa as funcionalidades em 5 blocos distintos mas interdependentes: "Inicia *Backbone*", "Inicia *No-Prefetching*", "Tráfego de Fundo", "Gerencia *Water-leveling*"e, por fim, "*Logs* e Controles".

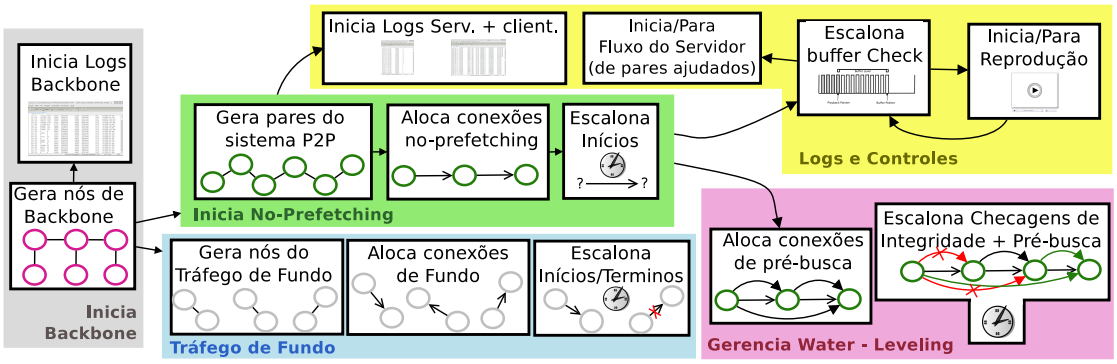

Figura 4.6: Diagrama funcional das funções principais na implementação da simulação.

A dependência entre os blocos é representada pelas setas presentes. O bloco "Inicia *Backbone*"(em cinza) organiza as funcionalidades responsáveis por criar a base da simulação, inicializando os nós de *backbone* e também associando a tais nós as verificações para aferimento de tráfego passantes, perdas e tamanho da fila de roteamento para fins de *log*. Posteriormente, as inicializações pertinentes ao tráfego *no-prefetching* do bloco "Inicia *No-Prefetching*"(em verde) e do tráfego de fundo do bloco "Tráfego de Fundo"(em azul) são realizadas simultaneamente.

Os pares participantes do tráfego de fundo são completamente independentes do resto da simulação, em tal módulo (Tráfego de Fundo) são criadas as conexões, os tempos de início e as pausas de *thinking time* são escalonadas no decorrer da execução da simulação. O bloco "Inicia *No-Prefetching*"(em verde) encarrega-se de gerar a quantidade de pares necessários, alocando as conexões segundo o algoritmo P2P baseado em árvore e escalonando o início das transmissões.

Após inicializar os pares da difusão *no-prefetching*, é executada uma função do módulo "*Logs* e Controles"(em amarelo) que anexa a cada par integrante do sistema P2P um arquivo de *log*, incluindo o servidor. Neste *log* são guardadas informações sobre: quantidade de vídeo recebido e enviado, taxas de recepção e envio, quantidade de perdas de pacotes e o ponto de reprodução do vídeo (a reprodução do vídeo também é simulada pelo sistema). No *log* do servidor, as informações armazenadas são: quantidade de *bytes* enviados, número de clientes simultâneos, taxa de envio e quantidade de *bytes* perdidos.

Caso o *water-leveling* seja simulado, as ações necessárias são tomadas, necessariamente,

após a completa execução dos módulos: "Inicia *Backbone*"e "Inicia *No-Prefetching*". Os passos necessários para simulação do *water-leveling* são exibidos no bloco "Gerencia *Water-Leveling*"(em lilaz). Como é sabido que um dado par pode enviar pré-busca a todos os que chegaram depois dele, este módulo já pré-aloca todas as conexões de pré-busca que são possíveis de serem usadas quando necessário. Em seguida, são escalonadas as checagens para envio de pré-busca a novos pares e finalização de tais fluxos quando necessário. Relembrando, a decisão sobre para qual par deve-se enviar a pré-busca é baseada na comparação entre o *buffer level* de dois pares que chegaram depois do par emissor. Tal verificação deve ser realizada periodicamente para assegurar que o algoritmo de pré-busca comporte-se da forma correta. Em relação às checagens de integridade, as mesmas são escalonadas de forma a impedir situações não condizentes com a realidade. Por exemplo, um dado par N não pode possuir em seu *buffer-pointer* um pedaço de vídeo que o par N-1 ainda não tenha, uma vez que é necessário que os pares mais antigos sempre possuam pedaços de vídeo mais avançados do que os pares mais recentes. Quando os pedaços de vídeo entre dados pares N e N+1 estão praticamente sincronizados, a checagem em questão encarrega-se de desativar a pré-busca recebida por N. Nesse caso específico, o emissor da pré-busca encerrada envia a mesma para outro par mais recente.

Em paralelo à execução das funções do bloco "Gerencia *Water-Leveling*", também são executadas outras funções do bloco "*Logs* e Controles". Neste são escalonadas as verificações de *buffer* e executadas as decisões de iniciar ou parar a reprodução do vídeo. Como cada par possui em seu *buffer* o vídeo que recebeu, é feita uma simulação do consumo de tal vídeo, a taxa de consumo é a mesma taxa de sua codificação (512Kbps). O módulo "Inicia/Para fluxo do servidor"permite fazer com que um par deixe de ser assistido pelo servidor (caso este seja) e passe a usar o seu *buffer*. Tal tomada de decisão faz uso da verificação de *buffer* escalonada pela função "Escalona *Buffer Check*", desta forma, a função "Inicia/Para fluxo do Servidor"sabe quando um dado cliente que é assistido pelo servidor possui um *buffer* razoável e pode deixar de receber tal fluxo. Esta função é importante pois possibilita a economia de banda no servidor.

4.4 Parâmetros de Simulação

De posse do *framework* que foi concebido, é necessário escolher os parâmetros para execução da simulação. A correta escolha dos parâmetros é fundamental para a execução bem-sucedida de uma simulação. Deve-se usar parâmetros que representem ao máximo a heterogeneidade de um ambiente realista e, ao mesmo tempo, preocupar-se para não adicionar muitas variáveis para se ter melhor controle dos impactos causados por cada parâmero específico. A Tabela 4.1 exibe os principais parâmetros de configuração usados na simulação deste trabalho.

Tipos de Pares	2
Características do Par Tipo 1	*Upload*/*Download* 765*Kbps*/2.268*Mbps*
Características do Par Tipo 2	*Upload*/*Download* 256*Kbps*/768*Kbps*
Qualidade do Vídeo	512*Kbps*
Quantidade de Nós	60
Tempo de Simulação	1200 segundos
Tempo de Vídeo	900 segundos
Tráfego de Fundo (Conexões Longas)	120
Tráfego de Fundo (Conexões Curtas)	200
Enlaces de *Backbone*	100*Mbps*
Número de nós do *Backbone*	6

Tabela 4.1: Descrição dos parâmetros usados na simulação.

Os parâmetros para os atrasos do *backbone* foram configurados segundo o padrão do *TCP Suite*, que é um valor extraído de uma distribuição exponencial com média de 15 *msecs*. Ou seja, para passar pelo *backbone*, o atraso seria em média de 30 a 45 *msecs* (2 ou 3 saltos). A simulação foi realizada com 60 pares clientes o que corresponde a um número de clientes compatível (proporcionalmente) com o número de fluxos do tráfego de fundo. Para aumentar o número de clientes do sistema P2P, seria necessário aumentar a capacidade dos enlaces de *Backbone* e a quantidade de fluxos do tráfego de fundo. As velocidades dos enlaces do *backbone* estão com 100 Mbps, o que é um valor compatível com a quantidade de pares que são inseridos na simulação (700 pares). Além disso, são usados dois tipos de tráfego de fundo segundo o padrão do *TCP Suite* para gerar um congestionamento moderado no *backbone*, sendo 120 conexões longas caracterizadas por transferência de arquivos grandes pela rede (como DVDs), e 200 conexões curtas caracterizadas por transferência de páginas web. O tamanho dos arquivos segue uma distribuição de Pareto (seguindo a estatística da Internet) e os tempos entre chegadas de novas conexões curtas é regulado por uma distribuição exponencial representando períodos de *thinking time*, quando os usuários não estão requisitando novas páginas web.

A Tabela 4.4 exibe perfis de pares P2P, o de tipo 1, que possui um *upload* maior e o de tipo 2 que possui um *upload* menor. Com relação à infraestrutura P2P colocada sobre essa topologia e o tráfego descrito anteriormente, temos a seguinte parametrização: (i) o intervalo entre o tempo de chegada dos pares P2P no sistema segue uma distribuição exponencial com $\lambda = 0,15$ pois esta taxa permite que todos os pares cheguem ao sistema até, no máximo, a metade da simulação; (ii) os pares foram posicionados no *backbone* de forma aleatória, ou seja, cada par possui chance igual de ser alocado em qualquer nó de *backbone*, desde que seja diferente do nó

alocado para o servidor de conteúdo.

As capacidades de recebimento e envio dos pares é baseada na em realidades de usuários de Internet a cabo e ADSL do Brasil. Como, em geral, as referidas bandas são assimétricas nos usuários domésticos, também resolvemos adicionar essa característica na simulação. Repare que a taxa de *download* é sempre 3 vezes maior que a taxa de *upload*. As taxas de *upload* também foram escolhidas de tal forma que os pares de tipo 1 não tenham capacidade de enviar um vídeo de 512*Kbps* por si só e que os pares de tipo 2 consigam e sobre ainda um pouco de banda. Essa decisão demonstra o compromisso em simular a existência de ambos pares que possuem boa capacidade de *upload* dos que possuem capacidades mais restritas.

4.4.1 Foto da Simulação

O NS2 possui uma ferramenta, denominada NAM (*The Network Animator*), que permite visualizar graficamente a organização física dos nós em uma simulação. A Figura 4.7 exibe a organização de uma das simulações deste trabalho.

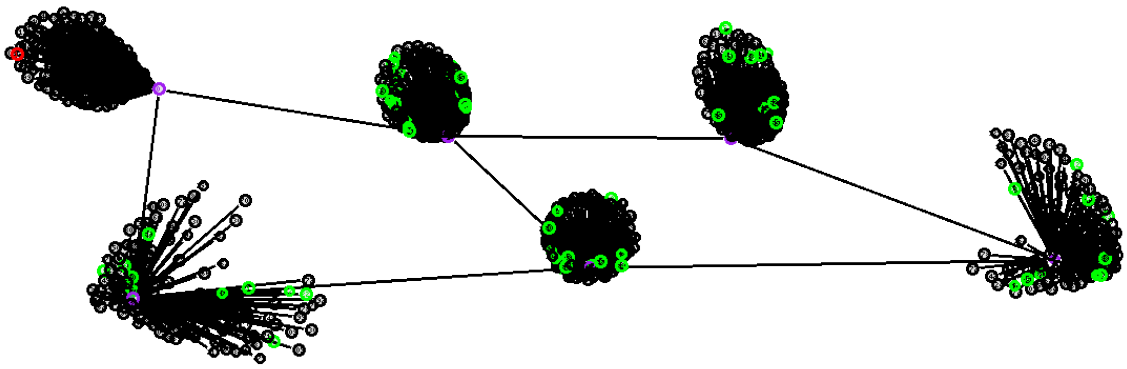

Figura 4.7: Ilustração da organização dos pares em uma simulação típica.

O servidor de vídeo é representado pelo nó em vermelho, os nós geradores de tráfego de fundo estão representados na cor preta, os nós de *backbone* estão em roxo e, por fim, os pares que participam na distribuição de vídeo P2P estão na cor verde. Visivelmente pode-se perceber como a quantidade de pares que geram tráfego de fundo é bem maior que a quantidade de pares que participam da distribuição de vídeo P2P. Para cada fluxo longo e curto, são inseridos dois pares, um correspondendo ao emissor e outro correspondendo ao receptor, como as conexões longas e curtas totalizam 320 (correspondendo a 120 conexões longas e 200 conexões curtas), são inseridos 640 pares no sistema contra os 60 pares do vídeo P2P, totalizando 700 pares.

5 *Análise dos Resultados*

Este capítulo visa aplicar a metodologia usada no presente trabalho, explicitando as análises sobre os resultados obtidos. Inicialmente é exibida a metodologia que visa apresentar a abordagem utilizada nas simulações e os respectivos resultados. Após isso, este capítulo termina com as considerações finais, apresentando as explicações sobre alguns aspectos não considerados.

5.1 Metodologia

Com objetivo de simular os diferentes estados que descrevem a capacidade de *upload* agregada do sistema, variou-se os tipos dos 60 pares participantes na transmissão de vídeo. A variação ocorreu de forma que se pudesse caracterizar os estados: *deficit*, *balanced* e *surplus*.

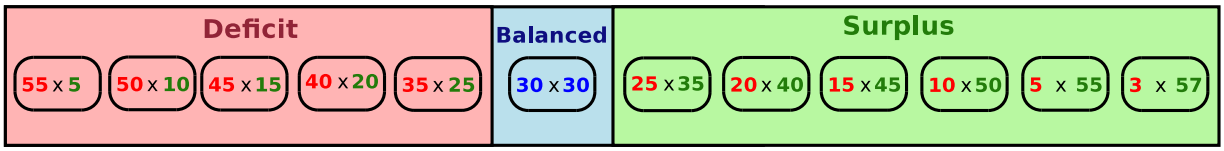

Figura 5.1: Representação das variações nos tipos dos pares.

A Figura 5.1 exibe todas as variações consideradas. Os retângulos de cantos arredondados denotam a execução de uma simulação, os números em verde indicam os pares de tipo 1 e os números em vermelho os de tipo 2. Para a análise de um cenário qualquer (seja variando ordem de chegada, camada de transporte, tráfego de fundo, etc.), é necessário que seja simulado todas as variações descritas na Figura 5.1. Consequentemente, para cada cenário diferente são necessárias 12 simulações.

Em todos os cenários simulados, considera-se como padrão o (i) uso do protocolo de transporte UDP, a (ii) ordem de chegada aleatória e a (iii) presença do tráfego de fundo. Essas características só mudam quando sua mudança faz parte dos objetivos da análise em questão, sendo devidamente explicitada. Em todos os outros casos, pode-se considerar que essas características foram usadas.

Abaixo segue a seguência de aspectos investigados:

- O primeiro cenário a ser simulado consiste na variação da ordem de chegada dos pares, a importância na variação desta característica já foi explicitada na Seção 2.3.2. Levou-se em consideração 4 ordens de chegada: pior ordem, ordem intercalada, melhor ordem e ordem aleatória;
- Posteriormente, a simulação abordou a presença ou não de tráfego de fundo;
- Em seguida, estudou-se o impacto na escolha da camada de transporte considerando os protocolos TCP e UDP. Vale ressaltar que devido a limitações encontradas no simulador, não foi possível simular de forma satisfatória uma terceira camada de transporte que também seria considerada, o DCCP. A variação na camada de transporte foi analisada tanto na perspectiva do servidor (economia de *upload*) quanto na dos enlaces de dados (perdas).
- Por fim, a ultima análise aborda a qualidade de vídeo percebida pelos usuários do sistema, principalmente quando se usa a pré-busca e se varia a camada de transporte utilizada.

5.2 Variação na Ordem de Chegada

Conforme descrito previamente, os pares possuem capacidades de *upload* e *download* diferentes, caracterizando uma banda assimétrica. Desta forma, mesmo quando o sistema encontra-se em um modo *surplus*, a ordem de chegada pode impactar tanto positiva quanto negativamente. É importante ressaltar que nestes experimentos houve inserção de tráfego de fundo, afim de torná-los mais realísticos.

Nas Figuras 5.2 a 5.5, o eixo Y representa a razão entre a taxa média de *upload* do servidor durante a simulação e a taxa de codificação do vídeo. O eixo X representa a razão entre as demandas e ofertas. No caso das Figuras do tipo (a), o eixo X representa a razão oferta sobre demanda, indicando que quando tal razão é 1 o sistema está no modo balanceado, que é caracterizado pela existência de um mesmo número de pares do tipo 1 e 2. À medida que o eixo X aumenta de valor, representa-se casos nos quais os pares do tipo 1 concentram-se em maior proporção do que os de tipo de 2, o que caracteriza o cenário *surplus*. Já nos gráficos do tipo (b), o eixo X representa a razão: demanda sobre oferta, indicando que à medida que tal razão aumenta, existem mais pares do tipo 2 do que 1, gradativamente, caracterizando o cenário deficit.

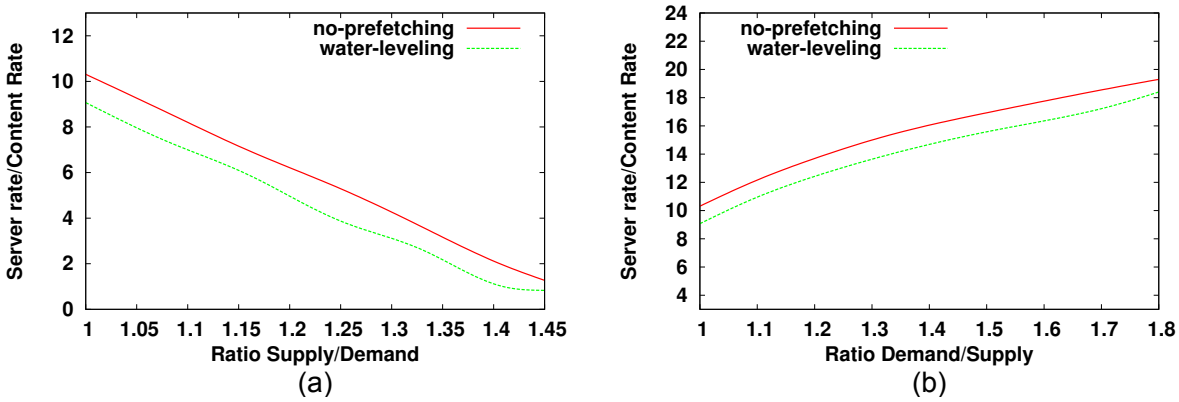

Figura 5.2: Desempenhos na melhor ordem de chegada (a) modo *surplus*; (b) modo *deficit*.

Embora pareça contra intuitivo, na comparação entre as Figuras 5.2 e 5.3, percebe-se que é melhor para o servidor quando os pares chegam ao sistema na ordem intercalada, do que quando chegam na melhor ordem. Isto ocorre pois pares do tipo 1 começam mandando pré-busca aos imediatamente posteriores na ordem de chegada, os fluxos suplementares só são transmitidos para os pares mais recentes em duas hipóteses: (1) quando ele termina de transferir o vídeo inteiro, (2) quando o seu ponto de tocar o vídeo (*playback point*) iguala-se ao do par imediatamente anterior.

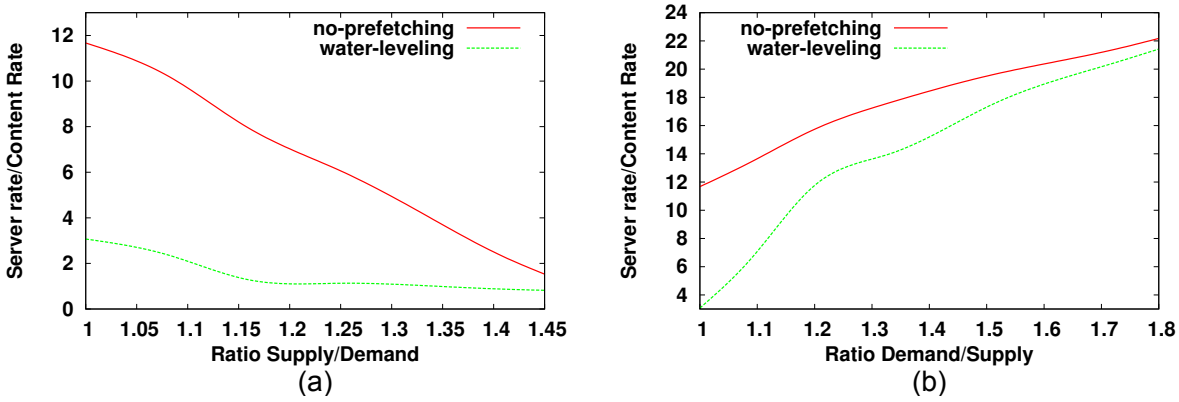

Figura 5.3: Desempenhos na ordem média de chegada (a) modo *surplus*; (b) modo *deficit*.

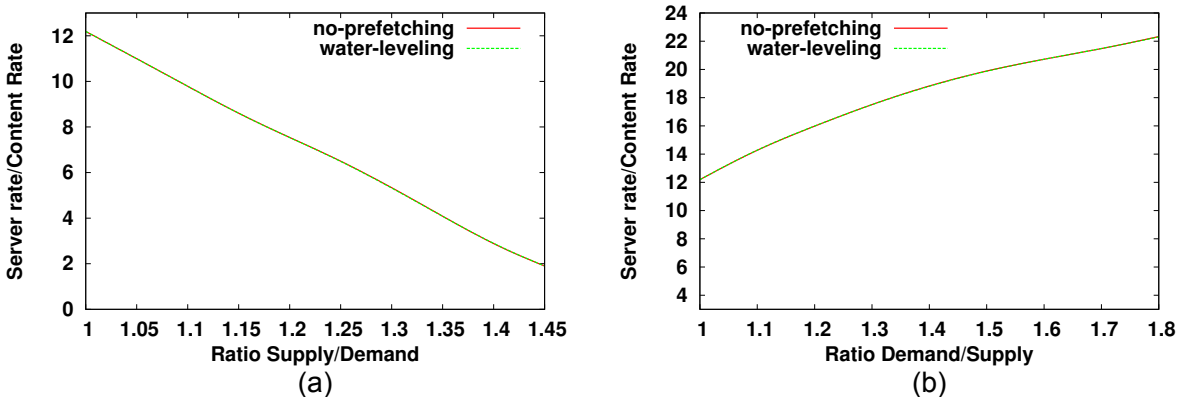

Figura 5.4: Desempenhos na pior ordem de chegada (a) modo *surplus*; (b) modo *deficit*.

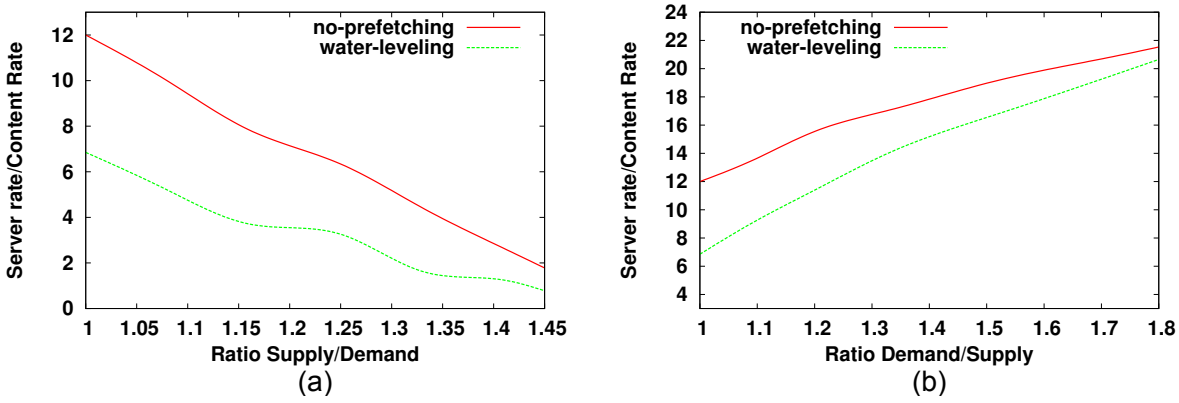

Figura 5.5: Desempenhos na ordem de chegada aleatória (a) modo *surplus*; (b) modo *deficit*.

Na melhor ordem, os pares do tipo 1 ficam um tempo inicial considerável servindo a pré-busca aos imediatamente posteriores a eles, neste caso, a outros pares tipo 1 do sistema. Com isso, sobra pouco tempo (na simulação simulação) para os pares tipo 1 servirem a taxa extra aos que são de tipo 2. Do ponto de vista do servidor, somente os pares imediatamente posteriores aos de tipo 2 são importantes para receberem a pré-busca, pois quando os mesmos possuem um reservatório de vídeo (*buffer level*) razoável, tal reservatário é usado para economizar banda do servidor.

Já na ordem de chegada intercalada, como os pares são intercalados, normalmente os de tipo 2 são melhor servidos pela pré-busca. A Figura 5.4 evidencia que a ordem de chegada do pior caso faz com que o desempenho do *water-leveling* seja igual ao *no-prefetching* uma vez que os pares iniciais são sempre de tipo 2, quando só pares de tipo 1 começam a chegar, eles apenas podem ajudar os posteriores a eles, que também serão de tipo 1, dessa forma, o ganho desses pares não potencializa economia de banda no servidor, por serem sempre imediatamente

posteriores a pares de tipo 1 e não assistidos pelo servidor.

A Figura 5.2 evidencia que o desempenho do *water-leveling* não foi muito melhor do que o *no-prefetching*, (economia média de 17%) em ambos casos *surplus* (a) e deficit (b) pelos motivos explicados anteriormente. Na Figura 5.3, percebe-se uma grande economia pelo uso da pré-busca, os maiores ganhos são no modo balanceado e surplus ($x >= 1$ na Figura 5.3a), o *upload* do servidor chega a ser até 3,8 vezes menor no caso balanceado (economia de 73%) e checou a 6,19 vezes no caso *surplus* (economia de 83%). A média da economia nas simulações deste caso foi de 47%.

A economia do servidor na ordem de chegada aleatória pode ser vista na Figura 5.5. Como nos casos anteriores, os maiores ganhos encontram-se no modo *surplus*, neste modo, a economia chegou a ser de 63%. No modo balanceado a banda economizada foi de, praticamente, 43%. A Figura 5.6 exibe um gráfico do tipo *box ploting* um comparativo entre as três ordens de chegada analisando várias métricas.

O gráfico do tipo *box ploting* permite verificar em uma só representação 5 medidas principais: maior número, menor número, primeiro quartil, mediana e terceiro quartil. A barra azul mais fina representa a diferença entre a maior e menor economias obtidas. A barra mais grossa dentro de cada intervalo representa o intervalo entre o primeiro e terceiro quartil, por fim, a marcação escurecida dentro do intervalo descreve a mediana.

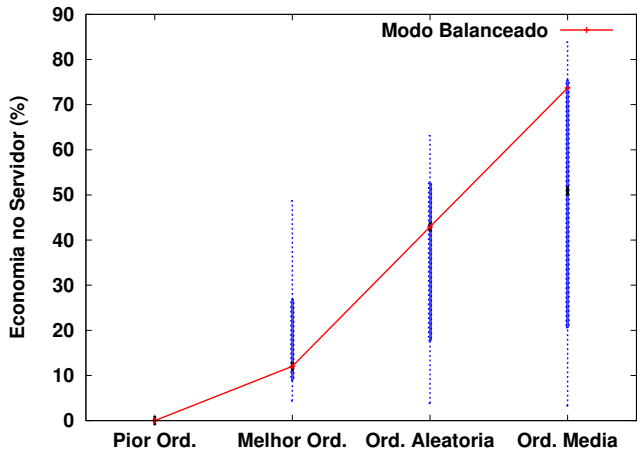

Figura 5.6: Comparativo entre as três ordens de chegada.

Adicionalmente, a Figura 5.6 também representa uma curva que descreve a variação da economia obtida pelo servidor no modo balanceado ($x = 1$ nas Figuras do tipo (a)) nas 4 ordens de chegada analisadas. O caso balanceado é o mais importante de se destacar pois representa um cenário durante o qual sistemas de P2P reais tentem a trabalhar por mais tempo. Com

exceção da ordem de chegada média, e economia do servidor no modo balanceado coincidiu com a mediana obtida nas simulações das ordens de chegada: melhor e aleatória. Esse resultado indica que o modo balanceado da ordem de chegada média não representa um caso típico neste cenário.

5.3 Impacto do Tráfego de Fundo

O tráfego de fundo adotado nas simulações gerou uma utilização muito grande nos *links* de *backbone*. Esta alta utilização é decorrente do elevado número de conexões que foram configuradas para serem estabelecidas (320 conexões) e devido às mesmas serem por meio do protocolo de transporte TCP. O protocolo TCP tem a característica principal de regular a sua janela de recepção/envio de acordo com a velocidade que a rede consegue entregar, dessa forma, as conexões do tráfego de fundo ocuparam toda banda disponível dos *links* de *backbone*.

A Figura 5.7 exibe a utilização de um *link* de *backbone* na simulação da ordem de chegada aleatória, usando UDP, no modo balanceado com uso de pré-busca. A figura compara a utilização nos *links* quando não há presença de tráfego de fundo (a) e quando a mesma existe (b). Sem a presença do tráfego de fundo a utilização dos *links* refere-se apenas à transmissão do vídeo no sistema P2P com as respectivas pré-buscas. Foi constatado que com a inserção do tráfego de fundo, a utilização dos *links* de *backbone* foi superior a 90% em grande parte do tempo de simulação. Pela Figura 5.7 percebe-se que a utilização *link* de *backbone* 4->5 foi bem alta em ambos sentidos (*upload* e *download*).

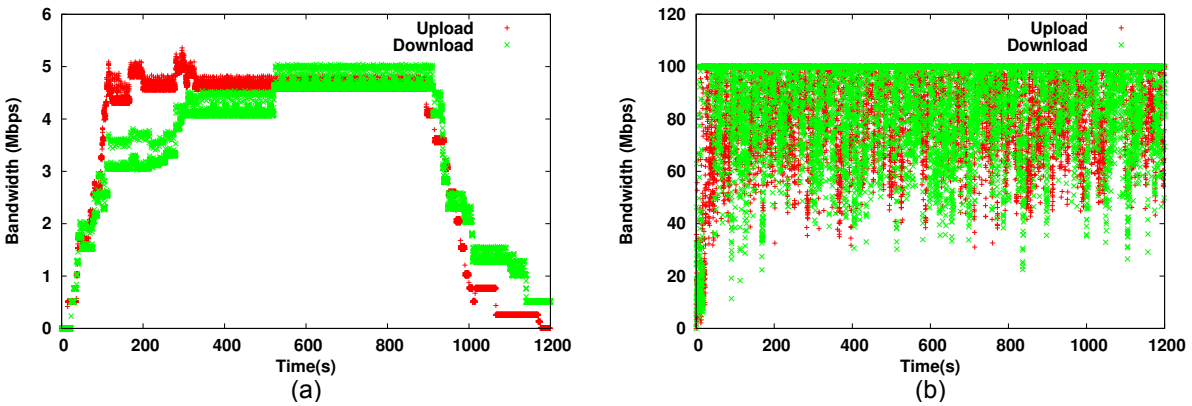

Figura 5.7: Utilização do *link* de *backone* 4->5 na simulação do *water-leveling* (a) sem tráfego de fundo e (b) com tráfego de fundo.

A inserção do tráfego de fundo pode impactar negativamente os clientes do sistema P2P

uma vez que o alto congestionamento pode gerar perdas nos *links*, tais perdas podem ser tanto referentes a dados inerentes ao tráfego de fundo quanto dados do sistema de distribuição de vídeo. Com objetivo de analisar o impacto do tráfego de fundo nos clientes, verificou-se como o *buffer level* de três clientes consecutivos comportou-se na presença e ausência de tal tráfego.

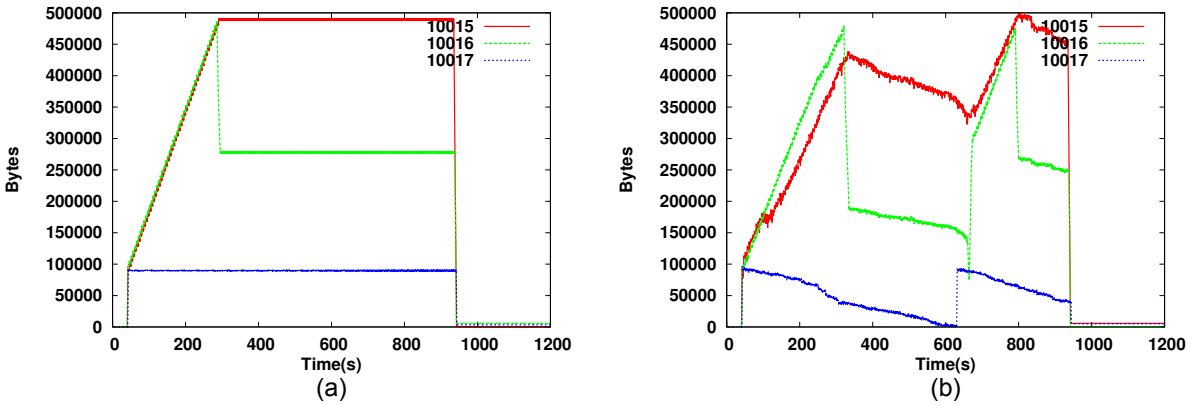

Figura 5.8: Impacto do tráfego de fundo no *buffer level* de 3 pares-cliente no modo balanceado (a) sem tráfego de fundo e (b) com tráfego de fundo.

A Figura 5.8 exibe o *buffer level* os pares 10015, 10016 e 10017. Quando não há presença do tráfego de fundo os pares podem receber o vídeo e efetuar a pré-busca sem que hajam perdas, desta forma, o *buffer level* sempre aumenta ou, no pior caso, mantém-se o mesmo. No caso da figura 5.8a, os pares 10015 e 10016 conseguiram construir *buffer levels* altos pois receberam pré-busca. No caso específico do par 10016, verificou-se que ele decidiu fazer uso do seu b*uffer level* para economizar banda do servidor, devido a isso, visualiza-se uma queda brusca de tal *buffer* por volta dos 300 segundos de simulação. Já o par 10017 não recebeu pré-busca durante a simulação inteira, devido a isso, seu *buffer level* sempre manteve-se baixo. Analisando a Figura 5.8, verifica-se que o tráfego de fundo potencializou perdas, as mesmas deterioraram aos poucos o *buffer level* de todos os 3 pares analisados. Em especial o par 10017, por não ter recebido pré-busca, esgotou seu *buffer level* por volta dos 600 segundos, o que gerou interrupção na visualização do vídeo. Mesmo com as perdas., os outros pares (10015 e 10016) conseguiram recuperar seus *buffer levels* no decorrer da simulação não havendo esgotamento do *buffer*.

Do ponto de vista do servidor, é necessário analisar como a economia em sua banda foi afetada pelas as perdas geradas decorrente da inserção do tráfego de fundo. Em ambos os casos, usando o *no-prefetching* ou *water-leveling*, deve existir o impacto na economia do servidor pois com a ocorrência das eventuais perdas, os pares e o servidor devem enviar conteúdo a mais para compensar as mesmas.

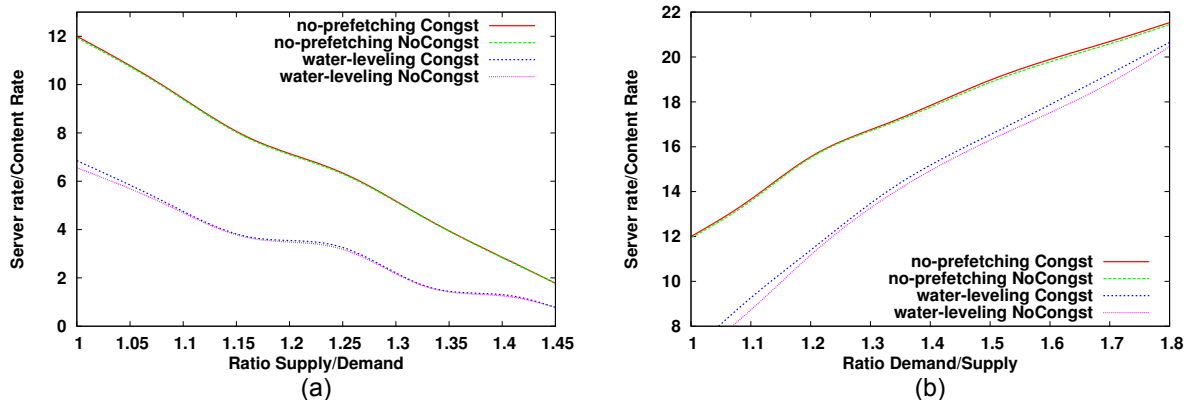

Figura 5.9: Economia na banda de *upload* do servidor com e sem tráfego de fundo (a) no modo *surplus* e (b) no modo *deficit*.

Pelos resultados exibidos na Figura 5.9, a diferença entre as economias no servidor com e sem a presença do tráfego de fundo foi bem pequena. Em termos numéricos, a média da economia no servidor quando existe tráfego de fundo é de 35,71%. A economia aumenta para 36,82% quando tal tráfego está ausente. Em suma, a diferença na economia do servidor foi de 1,11%, demonstrando que, embora a inserção do tráfego de fundo possa impactar os clientes comprometendo seus *buffer levels*, do ponto de vista do servidor, tal inserção não gera impactos relevantes. Acredita-se que o principal motivo deste resultado é decorrente da característica dos fluxos UDP não possuírem mecanismos de controle de congestionamento. A ausência de tais mecanismos torna estes fluxos indiferentes às perdas e congestionamentos existentes na rede.

Note que caso esta análise se tratasse de uma distribuição de vídeo P2P ao vivo (em vez de VoD), só seria aplicável comprar o desempenho do *no-prefetching* pois o *water-leveling* supõe que os pares possuem pontos de reprodução de vídeo diferentes (o que deve-se minimizar ao máximo em transmissões ao vivo). Além disso, o tráfego de fundo não geraria impacto algum na economia do servidor uma vez que não haveriam retransmissões.

5.4 Impacto da Camada de Transporte

A escolha da camada de transporte pode ser decisiva na construção de um sistema de distribuição de vídeo P2P. Existem duas principais camadas de transportes que são amplamente implantadas na Internet, o TCP e o UDP. Quando a aplicação opta por usar o UDP, a mesma deve encarregar-se de lidar com variações inesperadas na taxa de transmissão, perdas de pacotes, atrasos, *jitter* e etc. Por outro lado, quando se usa o TCP, todas estas e outras características são melhor tratadas, poupando a aplicação destas decisões. Ambos protocolos possuem seus

pontos positivos e negativos, a análise desta seção visa exibir os compromissos decorrentes da escolha de qual protocolo usar, nas perspectiva do servidor e nos enlaces de *backbone*.

5.4.1 Perspectiva do Servidor

Ao passo em que o protocolo UDP consegue entregar o vídeo em taxas de transmissão constantes e é indiferente à situação de congestionamento na rede, o TCP possui taxas de transmissão variáveis e responde a tais congestionamentos. Na simulação desta seção, usou-se a ordem de chegada aleatória e analisa-se o modo balanceado, ou seja, quando existem 30 pares do tipo 1 e 30 pares do tipo 2. A Figura 5.10 exibe o resultado da economia no servidor de vídeo no modo *surplus* com ambos protocolos de transportes abordados anteriormente. Pode-se perceber que quando houve uso do TCP, a economia apresentou-se moderadamente variável. Tal variação é devida à capacidade dos fluxos deste protocolo se ajustar a diferentes situações de congestionamento da rede. Portanto, o tráfego de fundo causa razoável impacto no desempenho da transmissão de vídeo por meio de TCP. Além da variação,a economia alcançada pelo servidor usando o TCP obteve uma média de, praticamente, 20%. Já com o uso do UDP, a economia média foi de 35%.

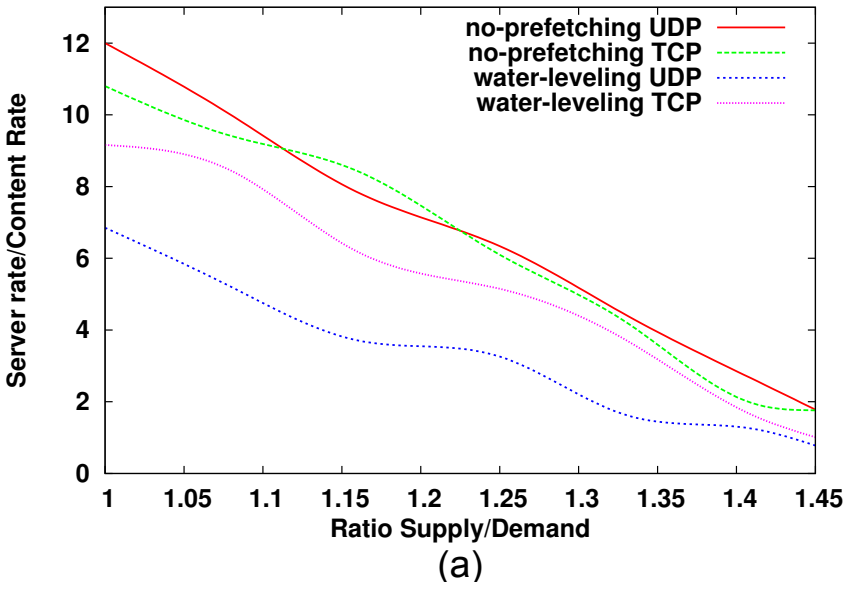

Figura 5.10: Desempenhos usando a ordem de chegada aleatória, no modo surplus.

Comparando a diferença de economia no modo balanceado ($x = 1$), a diferença é ainda maior, sendo a do UDP de, praticamente, 43% e a do TCP de 15%, ou seja, usando o UDP no modo balanceado, este protocolo de transporte foi 28% mais econômico. É interessante notar que quanto mais clientes o servidor de vídeo possui, maior será sua banda de *upload*, portanto,

a pré-busca deve criar o *buffer* nos demais pares para que um número mínimo necessite ser assistido pelo servidor. Com intuito de analisar a dinâmica envolvida em tais pares assistidos, foi gravado a quantidade de clientes simultâneos que o servidor possuiu ao longo da simulação.

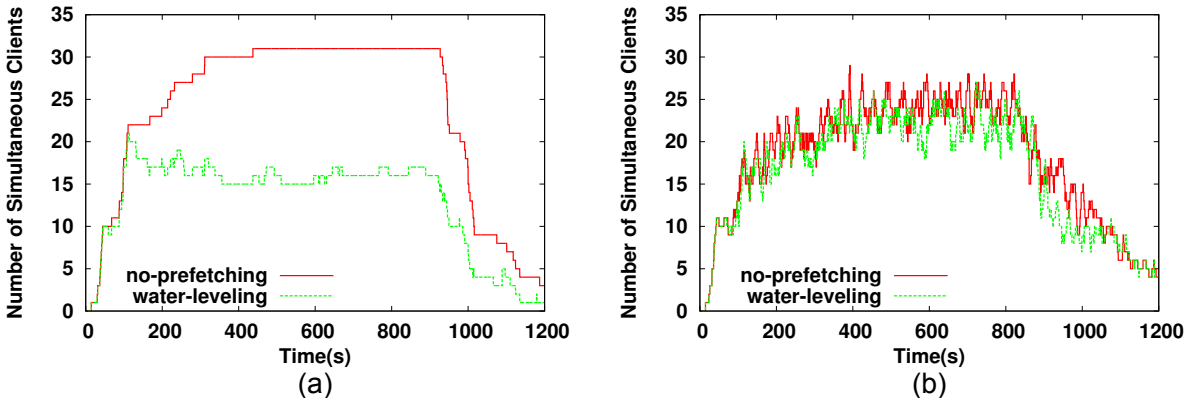

Figura 5.11: Variação da quantidade de clientes simultâneos **no servidor** de vídeo quando usou-se o (a) UDP e o (b) TCP.

A Figura 5.11 indica que quando usou-se o UDP, o servidor de vídeo possuiu uma certa estabilidade na quantidade de seus clientes, já usando o TCP, verifica-se uma moderada variação. Acredita-se que o principal motivo deste comportamento decorre de sua característica de possuir mecanismos de controle de congestionamento, como o *slow start*. O TCP também implementa a retransmissão de forma que o tamanho de sua janela de envio diminui bastante quando a mesma acontece. Como ocorrem perdas eventuais devido ao tráfego de fundo, é provável que as retransmissões aliadas com o *slow start* tenham sido decisivas pelo seu desempenho menor, como mostrado pela grande variação dos clientes simultâneos do servidor (Figura 5.11).

5.4.2 Perspectiva dos Enlaces

Como o UDP não é amigável com os fluxos TCP, é intuitivo pensar que ele possa ser um causador de maiores perdas.Com intuito de avaliar o impacto destas perdas nos enlaces, em todas as simulações registrou-se as perdas em ambas direções em todos os *links* do *backbone*.

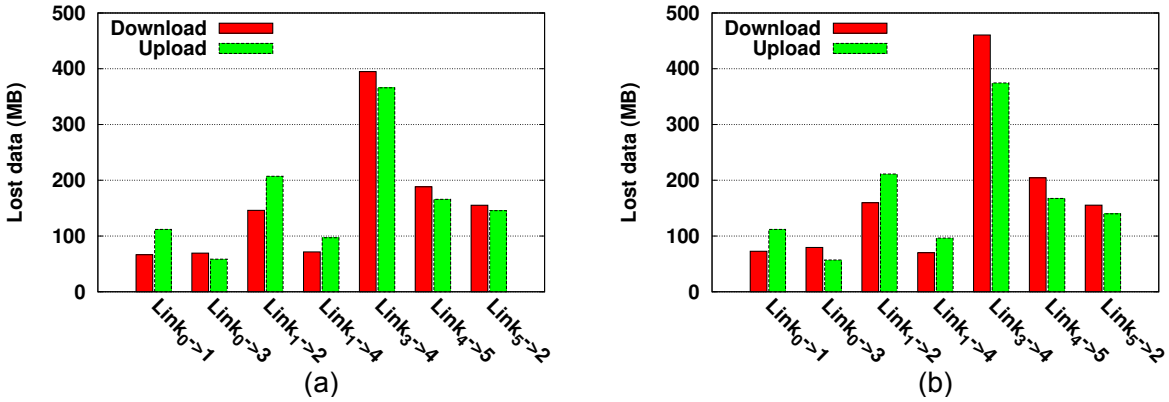

Figura 5.12: Perdas nos enlaces de backbone usando o (a) TCP e o (b) UDP.

As perdas referem-se aos enlaces de *backbone* por onde trafegam tanto os vídeos do sistema quanto o tráfego de fundo. Para tal análise, fez-se uso do cenário em que espera-se haver maior tráfego de dados, ou seja, usando-se a pré-busca com 55 pares de tipo 1 e 5 pares de tipo 2. Espera-se que haja maior tráfego de dados pois neste cenário existe o maior número de pares do tipo 1 (com *upload* maior), portanto, nesta situação será gerada uma quantidade maior de pré-busca em relação aos outros cenários. Contrariamente ao esperado, os resultados demonstram que as diferenças nas perdas foram na mesma ordem de grandeza. Neste caso, o esperado seria uma perda de proporções maiores nos *links* de *backbone* quando se usa o UDP em relação ao TCP, uma vez que o UDP não possui controle de fluxo e nem de congestionamento. Apenas no link_3->4 (que na topologia *parking lot* é a conexão entre o nó 3 e 4 da Figura 2), que a diferença foi 17% (400 e 450Mb) quando o UDP foi usado na camada de transporte. Muito provavelmente, neste *link* em que ocorreram mais perdas deve ser um dos dois *links* adjacentes ao nó onde está ligado o servidor de vídeo.

5.5 Percepção de Qualidade dos Clientes

Esta seção visa exibir características que podem ser observadas pelos clientes do sistema de distribuição de vídeo P2P, tais características podem ser decisivas para satisfação ou não dos clientes. Neste caso, para cada par é registrada a quantidade de vezes que o vídeo parou de tocar, representado por um diagrama de frequências. Adicionalmente, é armazenado o tempo que o vídeo permaneceu parado, representado pelo diagrama *boxplot* - sendo a mediana o centro do box, os quartis representam os contornos e os *outliers* ficam do lado de fora. Como foi feito na seção anterior, n análise desta seção foram usados os resultados de simulações no modo balanceado, ou seja, quando existem 30 pares com banda de *upload* alta e 30 pares com banda

de *upload* baixa.

5.5.1 Camada de Transporte UDP

O UDP foi o primeiro protocolo de transporte a ser analisado neste contexto. É bem sabido que do ponto de vista do cliente, é indesejável que o *playback* do vídeo tenha muitas interrupções, no entanto, quando as mesmas ocorrem, a ideia é que o mesmo retorne o mais rápido possível (Figuras 5.13 à 5.16). Na Figura 5.13, avaliamos como a política de pré-busca *water-leveling* pode melhorar de forma considerável a quantidade de interrupções nos pares quando se usa o UDP. É possível afirmar que a política de pré-busca melhorou bastante o número de interrupções no vídeo em comparação ao P2P *no-prefetching* (quando não há pré-busca).

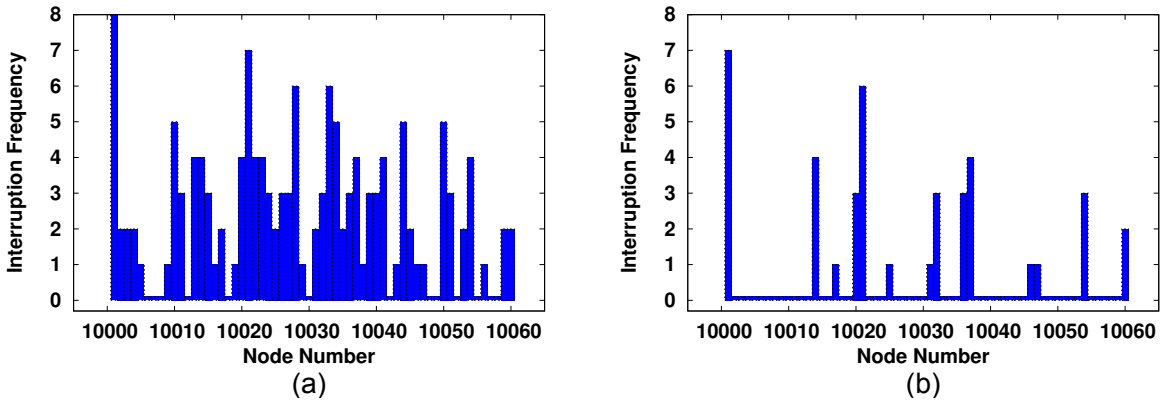

Figura 5.13: Frequência das paradas no *playback* do vídeo na ordem de chegada aleatória usando UDP no modo balanceado (a) *no-prefetching* (b) *water-leveling*.

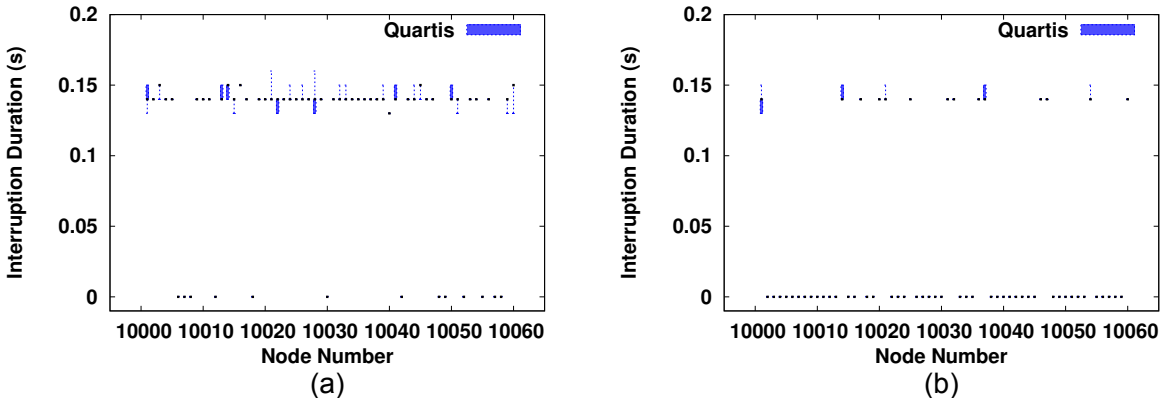

Figura 5.14: Continuidade no *playback* do vídeo na ordem de chegada aleatória usando UDP no modo balanceado (a) *no-prefetching* (b) *water-leveling*.

Se considerarmos que 2 interrupções no vídeo com sendo uma quantidade máxima aceitável durante a simulação inteira, podemos dizer que houve uma redução de 75% na existência das mesmas quando se fez uso da pré-busca. Na Figura 5.14 é possível ver que o tempo decorrido dentre o vídeo parado até voltar a tocar, foi em geral, muito baixo (menor que 0,2 s). Esses resultadores indicam que quando ocorreram interrupções, as mesmas duraram muito pouco em ambos casos (*no-prefetching* e *water-leveling*). Além disso, se comprova que sistemas de vídeo os quais fazem uso do UDP tendem a possuir *startup-delays* mais baixos.

5.5.2 Camada de Transporte TCP

Também optou-se por analisar esta camada de transporte devido a sua grande utilização nos sistemas de distribuição de vídeo na Internet. A Figura 5.15 mostra como o *water-leveling* reduziu bastante o a quantidade de interrupções no *playback* do vídeo quando a camada de transporte é o TCP, no entanto, vários pares ainda ficaram com interrupções na ordem de grandeza em 100, o que não é considerado aceitável. Caso tomemos como medida o mesmo parâmetro da subseção anterior, verificamos que no *no-prefetching* existiram 42 pares com mais de 2 interrupções, quando se usou a pré-busca, este número caiu para 34, ou seja, houve uma redução de 19% na quantidade de pares com mais de 2 interrupções.

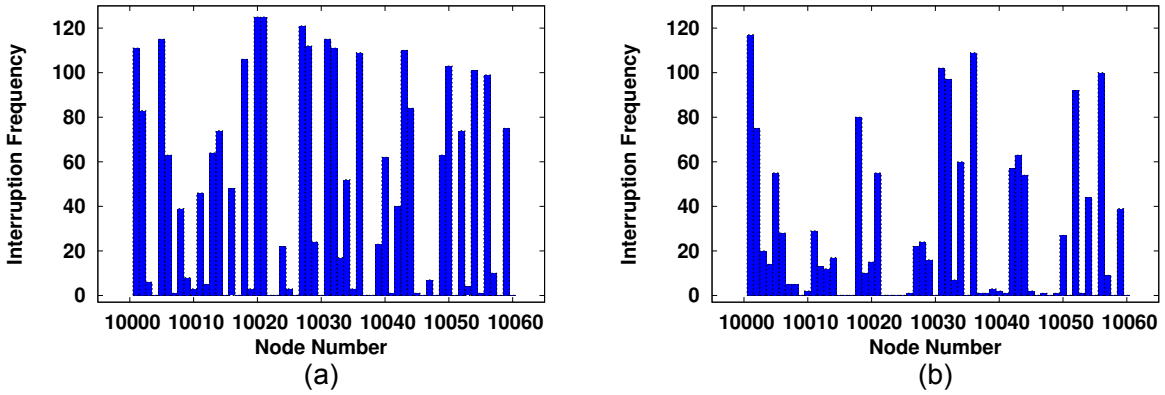

Figura 5.15: Frequência das paradas no *playback* do vídeo na ordem de chegada aleatória usando TCP no modo balanceado (a) *no-prefetching* (b) *water-leveling*.

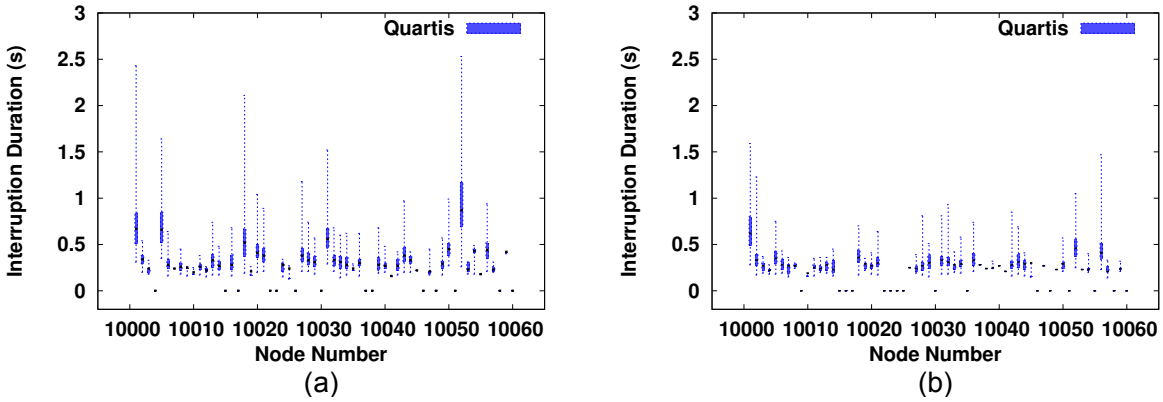

Figura 5.16: Continuidade no *playback* do vídeo na ordem de chegada aleatória usando TCP no modo balanceado (a) *no-prefetching* (b) *water-leveling*.

Na Figura 5.16 é exibido o tempo decorrido entre uma parada e a retomada da reprodução do vídeo usando o TCP. Também percebe-se que houve redução neste tempo com a adoção do *water-leveling*, no entanto, tal redução não foi muito acentuada. As reduções mais significativas na continuidade ocorreram em apenas 5 pares, o que representa menos de 9% de todos os pares do sistema. Em suma, estes resultados mostram que o TCP não potencializou ganhos muito relevantes para a percepção de qualidade dos clientes.

5.6 Avaliação dos Resultados

Pôde-se concluir que a ordem de chegada intercalada é a que confere maior economia para o servidor quando se usa a pré-busca, no entanto, esta ordem corresponde a um cenário ótimo. A ordem aleatória, que trata-se da ordem mais realista, conferiu uma economia de, praticamente, 43% no modo balanceado enquanto que no cenário ótimo (ordem intercalada) a economia chegou a 73%. Embora não tenha sido dito explicitamente na proposta de referência (HUANG; LI; ROSS, 2007), pode-se inferir os resultados obtidos basearam-se no cenário que consideramos ser ótimo por serem próximos dos nossos neste caso.

Na avaliação do impacto do tráfego de fundo pode-se concluir que o tráfego de fundo não possui impacto relevante na economia de banda percebida pelo servidor quando se usa o protocolo de transporte UDP, por outro lado, a Figura 5.8 mostrou que a presença de tal tráfego pode impactar de forma significativa os clientes.

Na análise do impacto na escolha da camada de transporte no servidor, concluiu-se que o uso do UDP potencializou a obtenção de uma economia maior no servidor se comparado ao uso

do TCP. A diferença de economia chegou a 28% no modo balanceado. Percebeu-se que com o uso do TCP a quantidade de clientes simultâneos no servidor variou bastante. Acredita-se que tamanha variação seja decorrente da característica de transmissão em rajadas do TCP. No que tange as perdas de enlace comparando o uso do TCP com UDP, o resultado contrariou o esperado indicando que quando se usou o UDP ocorreram 17% de perda a mais em relação ao uso do TCP (apenas no enlace mais sobrecarregado).

Tratando-se das percepções do cliente, o gráfico de frequência possibilitou observar que o uso da pré-busca na difusão em UDP possibilitou uma redução considerável na quantidade de interrupções percebidas pelos usuários. No que tange o tempo entre a parada e retomada da reprodução do vídeo, percebeu-se que independentemente do uso da pré-busca, tais intervalos foram pequenos (em torno de 0,2s). Na análise usando TCP, percebeu-se que a pré-busca também reduziu de forma considerável a quantidade de pares com frequências altas de interrupção. No entanto, não houve grande redução no tempo de retomada na reprodução do vídeo.

Se for considerar uma análise de compromissos para o sistema entre a camada de transporte e o uso da pré-busca, ficou claro que a camada de transporte tem maior impacto na percepção da qualidade de vídeo para os usuários. É importante notar que o uso da pré-busca foi mais efetivo quando se utilizou a camada de transporte UDP do que TCP. Isso pode ser explicado pelo UDP não tratar as perdas ocorridas por meio de retransmissão, deixando esta tarefa a cargo da aplicação. Embora esta característica do UDP possa gerar uma certa injustiça na utilização dos enlaces, a ordem de grandeza nas perdas mostrou-se ser equivalente ao longo da maioria dos enlaces (apenas um enlace possui uma diferença de 17%).

6 *Conclusões e Trabalhos Futuros*

O presente trabalho investigou compromissos de projeto para um sistema P2P de distribuição de vídeo, considerando as perspectivas do servidor, da rede e particularmente a continuidade de reprodução de vídeo percebido pelos usuários. O trabalho parte das políticas de pré-busca propostas em (HUANG; LI; ROSS, 2007), distinguindo-se ao avaliar o desempenho da pré-busca em um simulador de redes sob condições menos idealistas, incluindo atrasos, perdas nos enlaces, tráfego de fundo e distintos protocolos da camada de transporte.

Na fundamentação teórica foram apresentados várias técnicas desenvolvidas ao longo do tempo que objetivam alcançar melhor desempenho na transmissão de vídeos e arquivos na Internet. A análise focou-se na difusão de vídeo por ser relacionado diretamente com este trabalho. Após a análise realizada, percebeu-se que nem as diferentes técnicas na camada de transporte, *multicast* e redes de distribuição de conteúdo se mostraram amplamente satisfatórias para atender aos requisitos para aplicação em larga escala e com custo reduzido. Conclui-se que o P2P figura como o mecanismo com maior potencial de resolver estes problemas devido à sua elevada flexibilidade em usar recursos de forma distribuída. Ainda neste capítulo concluiu-se que o algoritmo de pré-busca *water-leveling* apresentou-se como mais interessante de ser estudado em profundidade neste trabalho em decorrência de aplicar uma maior justiça coletiva.

Nos trabalhos relacionados foram explicitados alguns sistemas de transmissão de vídeo que não afirmam se fazem ou não o uso da pré-busca e outros que afirmam usá-la de forma mais explícita. Os resultados das pesquisas em sistemas que usam a pré-busca mostraram-se promissores e serviram como apoio para esta dissertação no sentido de exibir como as diferentes soluções são propostas e podem ser utilizadas para beneficio real dos usuários. Foi possível observar lacunas em alguns trabalhos citados, tais lacunas motivaram a investigação de alguns aspectos deste trabalho. Por exemplo, a ideia de avaliar a qualidade percebida tanto pelo servidor quanto pelos clientes.

No que tange o ambiente de simulação, o simulador de redes usado foi uma boa escolha por já possuir uma grande estabilidade em sua implementação e comprovada fidelidade na im-

plementação da pilha de protocolos do TCP/IP. O uso do TCP Suite possibilitou a criação de um ambiente simulado menos idealizado por gerar tráfego de fundo coerente com estatísticas da Internet. A topologia de rede usada, juntamente com as duas ferramentas citadas anteriormente, constituem a base do ambiente de simulação montado. Com a existência dos pares que geram tráfego de fundo foi possível constatar visualmente (pela foto tirada da simulação) o quão carregado estava o cenário de simulação.

Em relação à análise dos resultados, o maior ganho na taxa de *upload* do servidor proporcionado pela pré-busca foi uma economia de 3,8 vezes (economia de 73%), no modo balanceado. No entanto, esta economia é condicionada ao uso do UDP como protocolo de transporte e quando os pares chegam de forma intercalada ao sistema. No caso da ordem de chegada aleatória, usando-se também UDP, a economia de banda obtida no servidor foi de praticamente duas 2 vezes (42,9%), que corresponde a um ganho menor do que apresentado em (HUANG; LI; ROSS, 2007). Entretanto, acreditamos que tal ganho tenha sido menor pois podemos inferir que o referido trabalho usou a ordem de chegada intercalada, embora não tenha deixado isso explícito, tal crença é em decorrência dos resultados desta ordem serem similares ao nosso.

Ressalta-se ainda que a ordem de chegada aleatória tende a ser um cenário mais próximo ao que acontece em redes P2P na Internet. Pelo uso do TCP como protocolo de transporte percebeu-se que a economia propiciada ao servidor (mesmo usando pré-busca) foi consideravelmente inferior ao uso do UDP (com 28% de diferença). No ponto de vista da rede, as perdas ocasionadas pelo uso do UDP não foram muito distantes das perdas ocorridas usando o TCP (diferença de 17% em apenas um dos links), portanto, conclui-se que é viável o uso do UDP na distribuição de vídeo sob este ponto de vista. A análise que aborda o impacto do tráfego de fundo (quando se usa UDP) também aponta para esta mesma conclusão, ou seja, o UDP é mais recomendado neste aspecto por ter não ter ocasionado perda de desempenho considerável na economia do servidor quando se comparou a presença ou ausência de tráfego de fundo. A camada de transporte DCCP também foi investigada neste trabalho, no entanto, não possível realizar a simulação utilizando a mesma devido a restrições encontradas no simulador.

Na análise de continuidade do vídeo percebido pelos clientes, os resultados mostram que a escolha da camada de transporte teve mais impacto do que o uso da pré-busca. No entanto, por mais que as interrupções no *playback* do vídeo ocorridas usando o UDP tenham durado muito pouco, as mesmas são suficientes para gerar insatisfação do usuário. Neste contexto, pode-se concluir que o uso da pré-busca beneficia bastante ambos o provedor de conteúdo e os clientes.

Por ter-se feito uso da arquitetura de P2P proposta em (HUANG; LI; ROSS, 2007) no desenvolvimento da simulação deste trabalho, foram adotadas premissas equivalentes. Nas

simulações não foi considerado que usuários possam realizar operações de VCR, ou seja, que consigam avançar e retroceder na reprodução do vídeo recebido. Caso esta característica fosse considerada, seria necessário efetuar uma revisão completa no algoritmo de construção da rede P2P, uma vez que neste novo cenário os pares poderão, em dado momento, mudar a alocação de sua banda destinada a necessidades de tempo real para outros pares. Lembrando que na arquitetura P2P usada, as bandas referentes às necessidades de tempo real não mudam, são estáticas.

Neste trabalho considerou-se que não existe *peer-churn*, ou seja, a possibilidade da existência dos denominados pares dinâmicos. Esta característica retrata a possibilidade dos pares entrarem e saírem do sistema de distribuição P2P de forma arbitrária. A inserção do *peer-churn* implicaria em ter que desenvolver algoritmos de realocação de fluxos de vídeo, além disso, seria adicionada mais uma variável a ser analisada nas simulações.

Por fim, neste trabalho se abstraiu aspectos relativos às decisões tomadas pela camada de aplicação em relação a eventuais perdas. Dessa forma, deixa-se como trabalho futuro a simulação de um protocolo na camada de aplicação que possui mecanismos mais reativos a tais perdas, como o RTP (*Real Time Protocol*), por exemplo.

Também como trabalhos futuros, permanecem duas primeiras características citadas anteriormente que não foram consideradas/implementadas: (i) o *peer-churn*, que permitirá representar modelos de entrada/saída de pares e (ii) a possibilidade dos usuários realizarem operações de VCR.

As simulações realizadas consideram que as capacidades de *upload*/*download* de um dado par ficam o tempo todo disponíveis para a aplicação, seria importante simular a característica da existência do tráfego de fundo saindo dos pares que participam da rede P2P pois na prática as aplicações de um dado computador competem entre si pelo uso da rede disponível.

Outra linha de trabalho futuro pode investigar o uso da extensão "*TCP Cradle*"(JANSEN; MCGREGOR, 2007) do NS-2 que permite usar a pilha TCP/IP real da máquina dentro do ambiente de simulação. Embora esta abordagem torne a simulação mais lenta, é possível desenvolver aplicações na máquina real que podem funcionar dentro do ambiente de simulação.

Em relação ao tráfego de fundo, existe um programa denominado TMIX (WEIGLE et al., 2006) que permite converter os traces de transmissões de rede reais para serem inseridos dentro de simulações do NS2. De forma a adicionar mais realismo ao tráfego de fundo também deixamos como trabalho futuro incorporar esta ferramenta ao framework desenvolvido.

Referências Bibliográficas

1080P HD Is Coming to YouTube. Novembro 2009. Disponível em: <http://youtube-global.blogspot.com/2009/11/1080p-hd-comes-to-youtube.html>.

51VOD.CN. Junho 2006. Disponível em: <http://web.archive.org/web/20060628205635/http://www.51vod.cn/news/news20.asp>.

ALBUQUERQUE, C.; PROENçA, T.; OLIVEIRA, E. Tvoip: Tv sobre ip arquiteturas para transmissão em larga escala. *Simpósio Brasileiro de Computadores (SBRC), Em Minicursos do*, Maio 2006.

ANDERSON, D. P. et al. Seti@home: an experiment in public-resource computing. *Commun. ACM*, ACM, New York, NY, USA, v. 45, n. 11, p. 56–61, nov. 2002. ISSN 0001-0782.

ARLITT, M. F.; WILLIAMSON, C. L. Web server workload characterization: the search for invariants. *SIGMETRICS Perform. Eval. Rev.*, ACM, New York, NY, USA, v. 24, p. 126–137, Maio 1996.

BANCO Mundial: Indicadores de Desenvolvimento Mundial. Agosto 2011. Disponível em: <http://www.google.com.br/publicdata-/explore?ds=d5bncppjof8f9_&ctype=l&strail=false&nselm=h&met_y=it_net_user_p2&scale_y=lin&ind_y=fa
294958800000&tend=1282878000000&hl=pt-BR&dl=pt-BR&icfg&iconSize=0-.5&uniSize=0.035>.

BIAO, C.; ZHEN, L.; YAOHUA, L. A special topology for files pre-fetch in large scale video on demand. In: *Biomedical Engineering and Computer Science (ICBECS), 2010 International Conference on*. [S.l.: s.n.], 2010. p. 1 –4.

CHANG, H.; JAMIN, S.; WANG, W. Live streaming performance of the zattoo network. In: *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. New York, NY, USA: ACM, 2009. (IMC '09), p. 417–429. ISBN 978-1-60558-771-4.

CIGNO, R. L.; RUSSO, A.; CARRA, D. On some fundamental properties of p2p push/pull protocols. In: *Communications and Electronics, 2008. ICCE 2008. Second International Conference on*. [S.l.: s.n.], 2008. p. 67–73.

CLARKE, I. et al. Freenet: A distributed anonymous information storage and retrieval system. In: *INTERNATIONAL WORKSHOP ON DESIGNING PRIVACY ENHANCING TECHNOLOGIES: DESIGN ISSUES IN ANONYMITY AND UNOBSERVABILITY*. [S.l.]: Springer-Verlag New York, Inc., 2001. p. 46–66.

CROVELLA, M. E.; BESTAVROS, A. Self-similarity in world wide web traffic: evidence and possible causes. *IEEE/ACM Trans. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 5, p. 835–846, December 1997.

DATAGRAM Congestion Control Protocol (DCCP). Outubro 2004. Disponível em: <http://tools.ietf.org/html/draft-ietf-dccp-spec-08>.

EL-SAYED, A.; ROCA, V.; MATHY, L. A survey of proposals for an alternative group communication service. *Network, IEEE*, v. 17, n. 1, p. 46–51, Jan 2003. ISSN 0890-8044.

GAO, L. et al. The design of a p2p video-on-demand prototype system. In: *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*. [S.l.: s.n.], 2010. v. 7, p. 473–477.

HE, Y.; GUAN, L. Prefetching optimization in p2p vod applications. In: *Advances in Multimedia, 2009. MMEDIA '09. First International Conference on*. [S.l.: s.n.], 2009. p. 110–115.

HOLBROOK, H. W.; CHERITON, D. R. Ip multicast channels: Express support for large-scale single-source applications. In: *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*. New York, NY, USA: ACM, 1999. (SIGCOMM '99), p. 65–78. ISBN 1-58113-135-6.

HUANG, C.; LI, J.; ROSS, K. W. Can internet video-on-demand be profitable? In: *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2007. (SIGCOMM '07), p. 133–144. ISBN 978-1-59593-713-1.

ICQ.COM community, people search and messaging service. Junho 2011. Disponível em: <http://www.icq.com>.

JANSEN, S.; MCGREGOR, A. Validation of simulated real world tcp stacks. In: *Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come*. Piscataway, NJ, USA: IEEE Press, 2007. (WSC '07), p. 2177–2186. ISBN 1-4244-1306-0.

LARZON, L. et al. *UDP Lite for Real Time Multimedia Applications*. [S.l.], 1999.

LI, C.; CHEN, C. Measurement based ppstream client behavior analysis. In: *Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on*. [S.l.: s.n.], 2009. v. 1, p. 341–345.

LIANG, W. et al. On characterizing ppstream: Measurement and analysis of p2p iptv under large-scale broadcasting. In: *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. [S.l.: s.n.], 2009. p. 1–6. ISSN 1930-529X.

LIU, Y.; GUO, Y.; LIANG, C. A survey on peer-to-peer video streaming systems. *PeertoPeer Networking and Applications*, IEEE: Proceedings of the First Workshop on Secure Network Protocols, v. 1, n. 1, p. 18–28, 2008.

LIU, Z. et al. Uusee: Large-scale operational on-demand streaming with random network coding. In: *INFOCOM, 2010 Proceedings IEEE*. [S.l.: s.n.], 2010. p. 1–9. ISSN 0743-166X.

LU, Y. et al. Assessing the quality of experience of sopcast. *Int. J. Internet Protoc. Technol.*, Inderscience Publishers, Inderscience Publishers, Geneva, SWITZERLAND, v. 4, p. 11–23, March 2009. ISSN 1743-8209.

MENDES, J.; SALVADOR, P.; NOGUEIRA, A. P2p-tv service and user characterization. In: *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*. [S.l.: s.n.], 2010. p. 2612–2620.

NATARAJAN, P. et al. Sctp: an innovative transport layer protocol for the web. In: *Proceedings of the 15th international conference on World Wide Web*. New York, NY, USA: ACM, 2006. (WWW '06), p. 615–624. ISBN 1-59593-323-9.

NS by Example. Junho 2001. Disponível em: <http://nile.wpi.edu/NS>.

RELIABLE UDP Protocol. Fevereiro 1999. Disponível em: <http://www.ietf.org/proceedings-/44/I-D/draft-ietf-sigtran-reliable-udp-00.txt>.

RFC1112 - Host Extensions for IP Multicasting. Agosto 1989. Disponível em: <http://www-.ietf.org/rfc/rfc1112.txt>.

RFC1633 - Integrated Services in the Internet Architecture: an Overview. Junho 1994. Disponível em: <http://www.ietf.org/rfc/rfc1633.txt>.

RFC2475 - An Architecture for Differentiated Services. Dezembro 1998. Disponível em: <http://www.ietf.org/rfc/rfc2475.txt>.

SENTINELLI, A. et al. Will iptv ride the peer-to-peer stream? [peer-to-peer multimedia streaming]. In: . [S.l.: s.n.], 2007. v. 45, n. 6, p. 86–92. ISSN 0163-6804.

SHARMA, A.; BESTAVROS, A.; MATTA, I. dpam: a distributed prefetching protocol for scalable asynchronous multicast in p2p systems. In: *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*. [S.l.: s.n.], 2005. v. 2, p. 1139–1150. ISSN 0743-166X.

SHEN, Y. et al. On the design of prefetching strategies in a peer-driven video on-demand system. In: *Multimedia and Expo, 2006 IEEE International Conference on*. [S.l.: s.n.], 2006. p. 817–820.

TCP Evaluation Suite. Setembro 2007. Disponível em: <http://netlab.cs.ucla.edu/tcpsuite/>.

THE Network Simulator - NS-2. Junho 2011. Disponível em: <http://www.isi.edu/nsnam/ns-/>.

WANG, Y.; OSTERMANN, J.; ZHANG, Y. *Video Processing and Communications*. [S.l.]: Prentice Hall Publishing Company, 2002.

WEIGLE, M. C. et al. Tmix: a tool for generating realistic tcp application workloads in ns-2. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 36, p. 65–76, Julho 2006. ISSN 0146-4833.

YIN, H. et al. Design and deployment of a hybrid cdn-p2p system for live video streaming: experiences with livesky. In: *ACM Multimedia'09*. [S.l.: s.n.], 2009. p. 25–34.

ZATTOO - Wikipedia, the free encyclopedia. Junho 2011. Disponível em: <http://en-.wikipedia.org/wiki/Zattoo>.

ZHANG, H. et al. Scaling peer-to-peer video-on-demand systems using helpers. In: *Image Processing (ICIP), 2009 16th IEEE International Conference on*. [S.l.: s.n.], 2009. p. 3053–3056. ISSN 1522-4880.

ZHANG, L. et al. Rsvp: a new resource reservation protocol. *Network, IEEE*, v. 7, n. 5, p. 8–18, Setembro 1993.

ZHENG, C.; SHEN, G.; LI, S. Distributed prefetching scheme for random seek support in peer-to-peer streaming applications. In: *Proceedings of the ACM workshop on Advances in peer-to-peer multimedia streaming*. New York, NY, USA: ACM, 2005. (P2PMMS'05), p. 29–38. ISBN 1-59593-248-8.