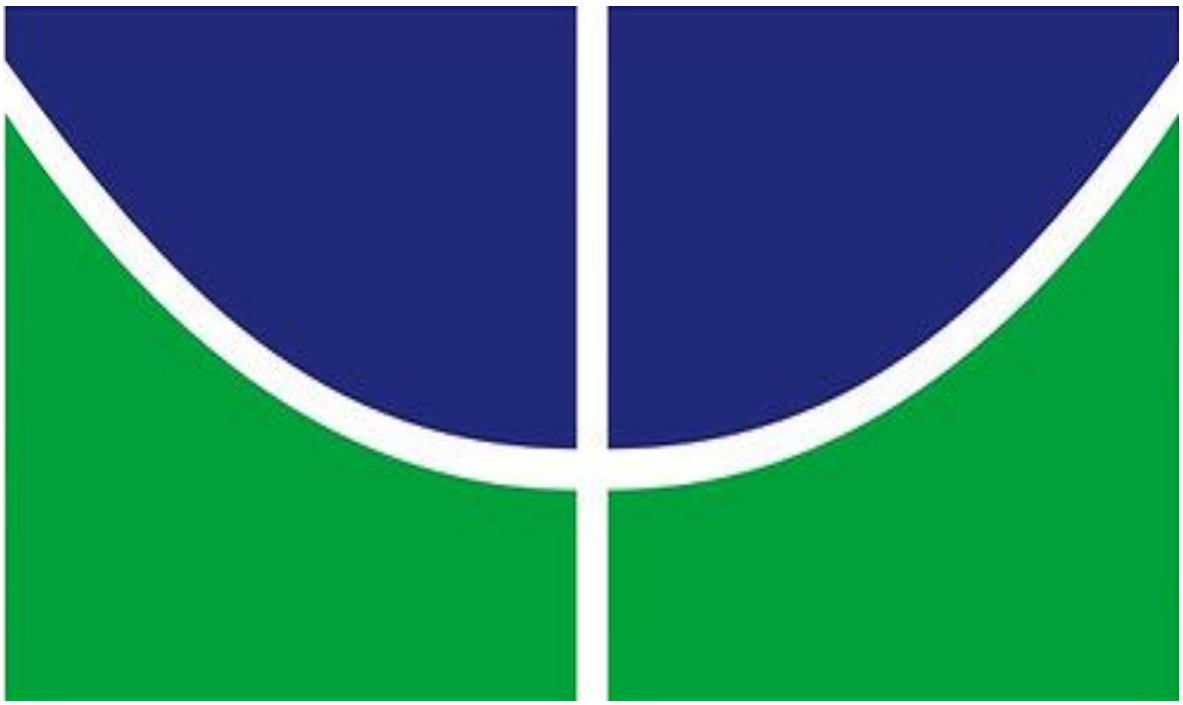

Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P)

VALMIRO JOSE RANGEL GALVIS

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P)

VALMIRO JOSÉ RANGEL GALVIS

ORIENTADOR: PAULO ROBERTO DE LIRA GONDIM

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

PUBLICAÇÃO: PPGENE.DM – 539/2013
BRASÍLIA/DF: MAIO – 2013

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P)

VALMIRO JOSÉ RANGEL GALVIS

DISSERTAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISÍTOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA.

APROVADA POR:

_____

PAULO ROBERTO DE LIRA GONDIM, Dr., ENE/UNB (ORIENTADOR)

_____

SIDNEY CERQUEIRA BISPO DOS SANTOS, Dr. FGL (EXAMINADOR EXTERNO)

_____

FABIO MOREIRA COSTA, Dr., UFG (EXAMINADOR EXTERNO)

BRASILIA/DF, 29 DE MAIO DE 2013.

FICHA CATALOGRÁFICA

GALVIS, VALMIRO J. RANGEL

Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P).

2013. XXII, 179p. 297 mm

(ENE/FT/UnB, Mestre, Engenharia Elétrica, 2013).

Dissertação de Mestrado – Universidade de Brasília.

Faculdade de Tecnologia.

Departamento de Engenharia Elétrica.

1. Vídeo sob Demanda 2. Redes P2P
3. Distribuição de vídeo 4. SVC

REFERÊNCIA BIBLIOGRÁFICA

GALVIS, V. J. R. (2013). Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P). Dissertação de Mestrado em Engenharia Elétrica, Publicação PPGENE.DM – 539/2013, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 179p.

_____

Valmiro José Rangel Galvis

UNIVERSIDADE DE BRASÍLIA

Faculdade de Tecnologia.

Departamento de Engenharia Elétrica.

70910-900 – Brasília – DF – Brasil.

Dedicado aos meus pais Valmiro e Luisa, e às minhas irmãs e os meus irmãos: Taty, Luchy, José e Cesar.

AGRADECIMENTOS

Agradeço a Deus por ter me dado a oportunidade de vir morar no Brasil para cumprir o sonho de ser mestre.

A minha família por ter me apoiado nesse tempo que tenho estado longe deles, mas mesmo estando longe me fizeram sentir que estavam por perto, me cuidando e me acompanhando.

Ao meu orientador, pela sua paciência, pelo apoio, por ter me recebido como seu orientando e ter me guiado no desenvolvimento do meu projeto de mestrado, Prof. Paulo muito obrigado.

Ao pessoal do LabTVDI da UnB, que além de ser meus companheiros de trabalho se tornaram os meus amigos, obrigado Lili, Vanice, Claudio, Vinicius.

Às pessoas que conheci em Brasília e às que conviveram comigo no dia a dia fazendo me sentir como em família com elas. Obrigado Rodrigo, Mariana, Paulo, Bruno, Wladimir, Renata.

RESUMO

Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P)

Nos últimos anos tem havido considerável esforço de pesquisas sobre as redes par-a-par (P2P, do inglês *peer-to-peer*), tanto aquelas voltadas para distribuir vídeo ao vivo (*Live Streaming*) quanto sob demanda (VoD, do inglês *Video-on-Demand*). Apesar do uso mais conhecido das redes P2P envolver sistemas de compartilhamento de arquivos, os sistemas de distribuição de vídeo na Internet também têm visto o seu desenvolvimento potencializado por este tipo de redes, consideradas um paradigma promissor na distribuição de vídeo em larga escala, permitindo solucionar alguns dos principais gargalos em arquiteturas cliente-servidor.

Por outro lado, o padrão de codificação de vídeo escalável H.264/SVC *(Scalable Video Coding)* surgiu como uma opção capaz de permitir que um usuário receba parte do *streaming* de acordo com sua capacidade de processamento, resolução da tela e capacidade da rede local em que se encontra conectado. O vídeo é codificado em camadas, e receber a camada base é suficiente para realizar a apresentação, sendo possível melhorar a qualidade percebida com o emprego de camadas de reforço.

A interação entre as áreas de redes P2P e de vídeo escalável tem sido investigada na literatura, sendo constatado que a codificação de vídeo escalável pode contribuir para a melhora da qualidade percebida pelo usuário neste tipo de redes. Todavia, o mapeamento entre a qualidade de serviço provida pela rede e a qualidade de experiência, ainda que realizado para outros sistemas P2P, ainda não tem sido realizado para vídeo escalável, até onde foi possível observar na literatura disponível.

Existe então uma motivação no desenvolvimento de uma relação quantitativa, capaz de mapear a qualidade de serviço provido pela rede em qualidade subjetiva em sistemas de distribuição de vídeo escalável sob demanda. Em outras palavras, é necessário propor uma relação quantitativa entre a QoS e a QoE, a fim de estimar a provável QoE vivida pelo usuário de acordo com parâmetros de rede e de aplicação.

Assim, após levantamento de possíveis mapeamentos QoS-QoE propostos na literatura, o presente trabalho busca propor e avaliar um preditor de QoE, passível de ser integrado em esquema adaptativo. O preditor é avaliado com base em simulação a eventos discretos em um conjunto de experimentos envolvendo diferentes classes de sequências de vídeo.

ABSTRACT

Quality of Service and of Experience in Scalable Video Distribution over Peer-to-Peer Networks.

In recent years there has been a considerable research effort on video distribution over peer-to-peer networks, considering live streaming and video-on-demand options. Although the most well known use of P2P systems involves file sharing, video distribution systems on the Internet have also seen the development enhanced by this type of networks, seen as a promising paradigm in video distribution on large scale, allowing to solve some of the major issues, as bottlenecks, in Client/Server architectures.

On the other hand, the H.264/SVC (Scalable Video Coding) standard has emerged as an option that would allow users to receive part of the stream according to their processing capacity, screen resolution and capabilities of the local network in which it is connected. The video is encoded in layers, and receiving the base layer is sufficient to carry out the playback, and it is possible to improve the perceived quality with the use of enhancement layers.

The interaction between P2P networks and scalable video has been researched in the literature, leading to the conclusion that the scalable video coding can contribute to the improvement of the quality perceived by the user in such networks. However, the mapping between the quality of service provided by the network and the quality of experience, even if done to other P2P systems, has not been performed for scalable video, as far as we can see in the available literature.

Then, there is a motivation to develop a quantitative relationship, capable of mapping the quality of service provided by the network in subjective quality in scalable video on demand distribution systems. In other words, it is necessary to propose a quantitative relationship between QoS and QoE, in order to estimate the likely QoE experienced by the user according to network and application parameters.

In this sense, after surveying possible mappings QoS-QoE proposed in the literature, this dissertation seeks to propose and evaluate a QoE predictor, which can be integrated into an adaptive scheme. The predictor is evaluated by the use of discrete event simulation and a set of experiments involving different classes of video sequences. The statistical analysis shows an excellent performance of this predictor.

SUMÁRIO

Siglas

3GPP	*Third Generation Partnership Project*
ANOVA	*Analysis of Variance*
AVC	*Advanced Video Coding*
BER	*Bit Error Rate*
BM	*Buffer Map*
CBR	*Constant Bitrate*
CDF	*Cumulative Distribution Function*
CDN	*Content Distribution Network*
CGS	*Coarse-Grain Quality Scalability*
CIF	*Common Intermediate Format*
DOE	*Design of Experiments*
DONet	*Data-Driven Overlay Network*
DHT	*Distributed Hash Table*
EDF	*Excessive Duplicated First*
FIFO	*First In, First Out*
FGS	*Fine-Grain Quality Scalability*
Fps	*Frames Per Second*
FR	*Frame Rate*
GOP	*Group of Pictures*
GW	*Gentle Walk*
IETF	*Internet Engineering Task Force*
IPDV	*Instantaneous Packet Delay Variation*
IQA	*Initial Quality Adaptation*
ISP	*Internet Service Provider*
ITU-T	*International Telecommunication Union*
JVT	*Joint Video Team*
JSVM	*Joint Scalable Video Model*
LRU	*Least Recently Used*
LRS	*Least Reusable First*
LSSF	*Least Server Supplementary First*
LUc	*Latest Useful Chunk*
MCP	*Motion-Compensated Prediction*
MDC	*Multiple Description Coding*
MGS	*Medium-Grain Quality Scalability*
MOS	*Mean Opinion Score*
MPEG	*Moving Picture Experts Group*
MPF	*Motion Prediction Flag*
MSE	*Mean Squared Error*
MTU	*Maximum Transmission Unit*
NAL	*Network Abstraction Layer*
P2P	*Peer to Peer*
PCA	*Principal Component Analysis*
PDV	*Packet Delay Variation*
PER	*Packet Error Rate*
PQA	*Progressive Quality Adaptation*
PSNR	*Peak Signal-to-Noise Ratio*
PSW	*Priority Sliding Window*
QCIF	*Quarter CIF*

QoE	*Quality of Experience*
QoS	*Quality of Service*
QoV	*Quality of Video*
RM	*Rapid Movement*
RMSE	*Root Mean Square Error*
RUp	*Random Useful Peer*
RTP	*Real Time Protocol*
RTT	*Round Trip Times*
SBR	*Sender Bitrate*
SM	*Slight Movement*
SNR	*Signal-to-Noise Ratio*
SSIM	*Structure Similarity Index Metric*
SVC	*Scalable Video Coding*
TCP	*Transport Control Protocol*
UDP	*User Datagram Protocol*
VBR	*Variable Bitrate*
VCEG	*Video Coding Experts Group*
VCR	*Videocassette Recorder*
VoD	*Video on Demand*
VQEG	*Video Quality International Expert Group*
VQM	*Video Quality Metric*

FIGURAS

LISTA DE TABELAS

1 – INTRODUÇÃO

Nos últimos anos, redes com uma estrutura baseada em arquitetura par-a-par (P2P, do inglês *peer-to-peer*) têm evoluído até o ponto de fazerem parte das bases de sistemas amplamente usados. Assim, esse tipo de rede não é mais só de interesse da comunidade dos pesquisadores, e tem sido utilizada para um grande número de aplicações utilizadas no dia a dia. Enquanto mais usuários estejam sendo atraídos pelos aplicativos que usam redes P2P, estas redes se farão cada vez mais presentes no dia-a-dia.

O uso mais conhecido das redes P2P envolve sistemas de compartilhamento de arquivos; adicionalmente, os sistemas de distribuição de vídeo na Internet também têm visto o seu desenvolvimento potencializado por este tipo de rede. Estas redes têm aparecido como um paradigma promissor na distribuição de vídeo em larga escala, por causa da sua abordagem descentralizada, ao contrário da abordagem das arquiteturas tradicionais cliente-servidor. Redes P2P não pressupõem um servidor central que seja responsável pela distribuição e armazenamento do conteúdo. Melhor ainda, todos os usuários interconectados podem contribuir com os seus recursos e ajudar na distribuição do conteúdo, podendo atuar tanto como servidores quanto como clientes.

Apesar das redes P2P terem sido popularizadas para a distribuição de vídeo na Internet, ainda há problemas a serem adequadamente resolvidos. Um dos principais problemas na adaptação de uma rede P2P para o *streaming* de mídias envolve as restrições de tempo, inerentes ao conteúdo de vídeo, e a natureza heterogênea dos dispositivos dos usuários. No caso do compartilhamento de um arquivo, flutuações da largura de banda poderão alterar o instante de tempo em que o arquivo ficará totalmente disponível, o que, em boa parte dos casos, pode não trazer consequências funestas, a não ser algum atraso que pode ser aceitável; no caso de um usuário que assiste a um *streaming* de vídeo que se supõe que seja contínuo, essas mesmas flutuações podem alterar drasticamente a qualidade percebida pelo usuário. Assim, é preciso que um *streaming* de vídeo ocorra com base em um *bitrate* adequado às restrições de tempo necessárias para uma qualidade, no mínimo, aceitável para o usuário final.

Ao lado das dificuldades ligadas às restrições de tempo, é necessário garantir uma qualidade do vídeo pelo menos aceitável para os nós da rede. Para isso, o padrão de codificação de vídeo escalável H.264/SVC *(Scalable Video Coding)* surgiu como uma opção a ser avaliada e pelo menos aparentemente promissora, pelo fato de que esta codificação permite que um usuário receba parte do *streaming* de acordo com sua

capacidade de processamento, resolução da tela, e capacidade da rede local em que se encontra conectado. Por isto, o fluxo de vídeo ao vivo (*Live Streaming*) está fora do escopo desta dissertação, pois o codificador demora muito tempo no processo de codificação e faz com que a codificação em tempo real não seja possível ainda.

Destaque-se aqui que a codificação H.264/SVC não impõe que a totalidade do vídeo esteja disponível para a sua reprodução, pois o vídeo é codificado em camadas, e receber a camada base é suficiente para tornar possível a apresentação; se a rede oferece recursos suficientes e o usuário tem os recursos necessários para receber e decodificar outras camadas (camadas de reforço), é possível melhorar a qualidade percebida. Uma questão é saber quantas e quais camadas um terminal, com recursos limitados e um nível de qualidade esperada, pode receber.

1.1 - MOTIVAÇÃO

Atualmente, as redes P2P estão sendo usadas para o *streaming* de vídeo, e combinar estas com a codificação de vídeo escalável pode contribuir para a melhora da qualidade percebida pelo usuário neste tipo de redes. A interação entre as redes par-a-par e o padrão H.264/SVC tem sido estudada em referências tais como [64-67, 69, 70, 72, 73].
Por outro lado, alguns trabalhos recentes têm sido feitos buscando um mapeamento de qualidade de serviço em qualidade subjetiva [51, 53-57, 60, 63]. Por isto, existe uma motivação no desenvolvimento de uma relação para poder mapear a qualidade de serviço provido pela rede em qualidade subjetiva em sistemas de distribuição de vídeo sob demanda. Além desse mapeamento de qualidade de serviço (QoS, do inglês *Quality of Service*) em qualidade de experiência (QoE, do inglês *Quality of Experience*), essa relação pode funcionar no outro sentido (partir de um valor de QoE e chegar a valores de parâmetros de QoS), assim valores de parâmetros de rede e de aplicação podem ser prognosticados a partir de um valor de QoE estabelecido.

Ao lado da necessidade de fazer a adaptação do *streaming*, para garantir ao usuário final os requisitos mínimos de qualidade e se ajustar às variações pelas quais passa a rede (por exemplo, de banda), é também preciso medir, dentro dos sistemas de distribuição de vídeo baseados em redes P2P, as mudanças na qualidade percebida em relação às mudanças em parâmetros de rede e de aplicação. Assim, é de grande valia encontrar uma relação, entre os parâmetros de rede e de aplicação e a qualidade entregue ao usuário final, que permita predizer e quantificar o valor da qualidade percebida, evitando assim depender de avaliações subjetivas (de execução cara em

termos de tempo e de recursos, principalmente humanos). A partir dessa relação, considera-se viável propor uma técnica de *streaming* adaptativo, envolvendo o conteúdo a ser transmitido e as taxas de transmissão.

1.2 - OBJETIVOS

1.2.1 - Objetivo Geral

Propor e avaliar uma relação quantitativa entre a QoS e a QoE relativas à transmisão e recepção de vídeo escalável em redes *peer-to-peer*, a fim de estimar a provável QoE vivida pelo usuário de acordo com parâmetros de rede e de aplicação.

1.2.2 - Objetivos Específicos

- Conhecer conceitos e características de codificação de vídeo escalável e de redes P2P;
- Avaliar a qualidade de serviço provido por uma rede P2P para distribuição de vídeo, usando diferentes métricas de QoS;
- Conhecer e avaliar métodos de mapeamento de QoS em QoE, e selecionar o(s) mais adequado(s) para os cenários considerados;
- Propor um mapeamento de QoS em QoE, aplicável ao sistema considerado.
- Avaliar o desempenho da técnica de mapeamento de QoS em QoE proposta, permitindo concluir se, de fato, serve como um bom indicador da QoE percebida pelo usuário.

1.3 - CONTRIBUIÇÕES

As seguintes contribuições podem ser destacadas:

- Levantamento bibliográfico e discussão sobre os sistemas P2PVoD, bem como uma taxonomia relativa a tais sistemas;
- Levantamento bibliográfico e discussão sobre propostas de mapeamento de QoS em QoE relativa a vídeo *streaming* e transmissão multimídia;
- Levantamento sobre simuladores de redes P2P;
- Proposta de preditor de QoE baseado em métricas de QoS, para emprego em um sistema de distribuição de vídeo sob demanda sobre redes P2P.
- Avaliação do preditor de QoS, com base em simulação e análise estatística dos resultados.

1.4 - ORGANIZAÇÃO

O restante desta dissertação está organizado da seguinte forma: o Capítulo 2 apresenta conceitos importantes em redes de distribuição de vídeo sob demanda em redes *peer-to-peer*, além de apresentar uma taxonomia, bem como considerações importantes na hora de projetar e implementar um sistema deste tipo.

No Capítulo 3 são apresentados conceitos sobre vídeo escalável, com foco no padrão H.264/SVC, e com detalhamento dos três tipos de escalabilidade: espacial, temporal e de qualidade, além de estudar a interação entre redes P2P e vídeo escalável.

No Capítulo 4 são apresentadas algumas definições e alguns aspectos relacionados às métricas de qualidade de vídeo (QoV), de qualidade de serviço em redes (QoS), e de qualidade de experiência (QoE). Também são apresentadas ferramentas e métodos usados na literatura para a aferição da qualidade objetiva e da qualidade subjetiva, bem como alguns métodos de mapeamento de qualidade objetiva em qualidade subjetiva (QoS em QoE) propostos por outros autores.

O Capítulo 5 apresenta inicialmente uma descrição detalhada do *testbed* usado nesta dissertação (incluindo o ambiente computacional, os cenários, as técnicas e tecnologias utilizadas) e uma descrição da arquitetura do sistema sendo avaliado para, em seguida, apresentar e analisar os resultados do trabalho.

Finalmente, no Capítulo 6 são apresentadas as conclusões do trabalho desenvolvido e as propostas de trabalhos futuros para a continuidade desta dissertação.

2 – DISTRIBUIÇÃO DE VoD SOBRE REDES P2P

Uma rápida pesquisa na literatura sobre redes P2P mostra várias definições de *peer-to-peer*, principalmente diferenciadas em decorrência da abrangência dada ao termo. Uma definição rigorosa de *peer-to-peer* "puro" fala de sistemas totalmente distribuídos, nos quais todos os nós da rede são equivalentes em termos de funcionalidades e das tarefas que desempenham. Essa definição rigorosa falha, por exemplo, na hora de abranger sistemas que usam o conceito de *tracker* ou de super nós [19].

As arquiteturas de sistemas distribuídos chamadas de *Peer-to-Peer* (P2P) são projetadas para o compartilhamento de recursos por meio de uma troca direta entre os usuários da rede, ao invés de precisar da intermediação ou do suporte de um servidor ou de uma entidade centralizada [19].

Uma outra definição, menos rigorosa e focando aplicações, é apresentada em [20], onde *peer-to-peer* é uma classe de aplicações que tira vantagem dos recursos (armazenamento, processamento, ciclos de CPU, presença humana) disponíveis nas bordas da Internet.

Dentre as citadas aplicações, a distribuição de conteúdo é uma aplicação importante das redes P2P na Internet, que tem recebido uma atenção considerável dos pesquisadores da área. As aplicações de distribuição de conteúdo tipicamente permitem aos computadores pessoais funcionarem, de uma forma coordenada, como um meio de armazenamento distribuído, contribuindo, buscando e obtendo conteúdo digital [19].

Observa-se que as redes P2P têm evoluído como um paradigma promissor na distribuição de vídeo em larga escala, de uma forma mais eficiente do que as arquiteturas tradicionais [1]. Devido à sobrecarga do lado do servidor em sistemas baseados na arquitetura Cliente/Servidor, onde se experimentam gargalos assim que o número de clientes aumenta na rede, tendo perdas na qualidade de serviço (QoS, do inglês, *Quality of Service*) prestado, as redes P2P surgem como uma alternativa para aliviar a carga no servidor nos sistemas de distribuição de conteúdo.

A seguir serão apresentadas classificações e as principais características dos sistemas de distribuição de VoD sobre redes P2P.

2.1 - CLASSIFICAÇÃO DOS SISTEMAS DE DISTRIBUIÇÃO DE VOD SOBRE REDES P2P

Existem duas grandes categorias de sistemas de distribuição de vídeo sobre redes P2P: baseados em *Live Streaming* (vídeo ao vivo) e baseados em *Video-on-Demand* (vídeo

sob demanda). Esses sistemas são chamados de P2PLive e P2PVoD [2], respectivamente.

Este trabalho foca em sistemas P2PVoD considerando VoD como um dos serviços mais importantes entre as aplicações multimídia [3] e um fator fundamental para atrair os consumidores para os serviços de IPTV [2].

Outra das questões a serem consideradas na eleição de VoD é o fato do aumento da complexidade associada ao *live streaming*, especialmente em termos de vídeo escalável (incluindo a escassez de codificadores), porque esta codificação não pode ser feita em tempo real.

Os sistemas P2PVoD podem ser classificadas de acordo com:

- O grau de descentralização da rede P2P.
- A estrutura da rede P2P.
- O mecanismo de descoberta de conteúdo.

2.1.1 - Classificação pelo grau de descentralização da rede

Uma primeira forma de classificar os sistemas de distribuição de P2PVoD é pelo grau de descentralização da arquitetura da rede. Essas arquiteturas podem ser: totalmente descentralizadas, parcialmente centralizadas e híbridas descentralizadas [19].

Nas arquiteturas totalmente descentralizadas todos os nós da rede têm as mesmas funções, agindo como clientes e servidores ao mesmo tempo sem se ter um controle centralizado das suas atividades, como mostrado na figura 2.1. Não existe uma hierarquia na rede [15]. Um exemplo claro desta arquitetura é a famosa rede de compartilhamento de arquivos chamada *Gnutella* [39].

Figura 2.1: Rede descentralizada (Adaptada de [13])

Nas arquiteturas parcialmente centralizadas o fundamento é o mesmo das totalmente descentralizadas. Mas, alguns dos nós assumem funções mais importantes dentro da rede atuando como índices centrais para compartilhamento de arquivos dos pares locais. Esses nós são chamados de Super Nós ou Super Pares [5, 19, 20] e estes nós não constituem pontos localizados de falha na rede P2P porque eles são escolhidos dinamicamente e, no caso de falha, a rede automaticamente os substituirá por outros nós. Um exemplo deste tipo de estrutura é mostrado na figura 2.2.

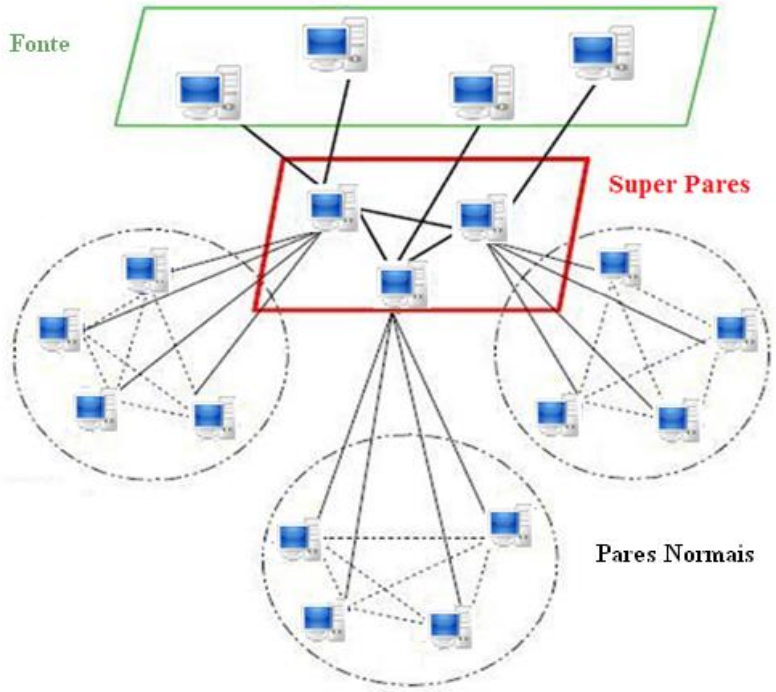

Figura 2.2: Rede parcialmente centralizada (Adaptada de [70]).

Nas arquiteturas híbridas descentralizadas há um servidor central (*tracker*) que facilita a interação entre os pares mantendo diretórios de metadados, que descrevem o conteúdo armazenado em cada par [3, 5, 21]. E as interações para a obtenção dos conteúdos podem ser entre pares somente ou com o servidor, mas o recomendado é que um par só procure conteúdo no servidor como último recurso. Uma desvantagem desta arquitetura é que tem um ponto único de falha localizado (o servidor central) [8, 2] que tipicamente faz com que o sistema seja inerentemente não escalável e vulnerável à censura, a falhas técnicas e a ataques maliciosos [19]. A figura 2.3 mostra a estrutura deste tipo de arquitetura.

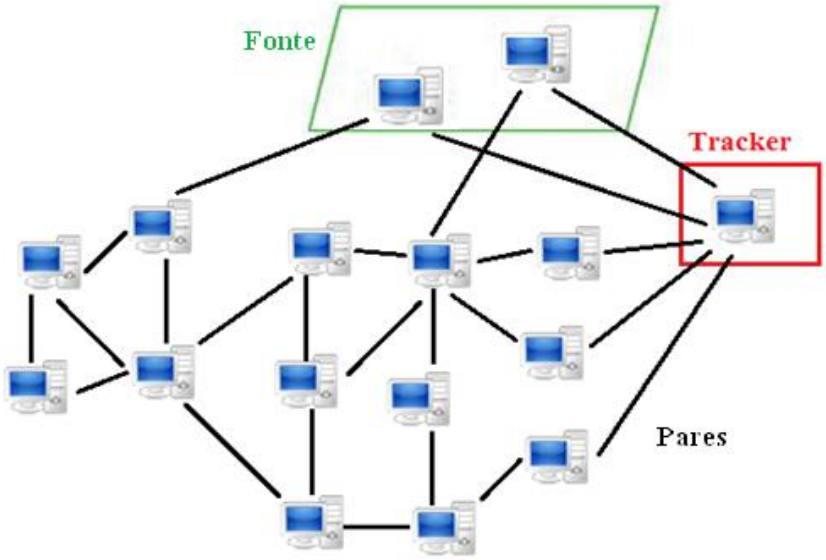

Figura 2.3: Rede híbrida descentralizada (Adaptada de [70]).

2.1.2 - Classificação de acordo com a estrutura da rede

Um sistema P2PVoD pode ter uma estrutura baseada em árvore de *Multicast*, baseada em malha ou uma estrutura híbrida (uma combinação entre árvore e malha) [2, 3, 19, 89].

2.1.2.1 - Estrutura baseada em árvore de *multicast*

Os sistemas P2P baseados na estrutura de árvore (onde a fonte do vídeo é a raiz da árvore) foram desenhados originalmente para funcionar como o IP *Multicast* na camada de aplicação, independentemente do suporte da camada de rede subjacente. Os usuários no *overlay* estão sincronizados e recebem o conteúdo na ordem em que a fonte o envia, e o desafio nesse tipo de estrutura é como arranjar os usuários assíncronos dos sistemas P2PVoD eficientemente e oferecer suporte a operações de VCR (avançar ou atrasar o vídeo, pular a outra parte do vídeo, etc.) [2].

A estrutura de árvore (*tree*) pode ser de árvore simples ou de múltiplas árvores. A estrutura de árvore simples não é muito utilizada porque os nós folha (nós finais da árvore) não contribuem com seus recursos à rede, e de que na hora que um nó falha todos os seus nós filhos ficam sem o *streaming* [4]. Além disso, o processo de recuperação de uma falha de um nó não é suficientemente rápido para lidar com a volatilidade dos pares, tornando o sistema ineficiente [2].

Para lidar com os diferentes problemas da estrutura de árvore simples foram estudadas aplicações de árvore de *multicast* [2, 8, 4, 11, 13]. Onde o servidor é a raiz, mas os pares são arranjados em subárvores no sistema de acordo com os seus tempos de chegada em sessões ou gerações [2]. A diferença básica entre as estruturas de árvore é a

forma de como são formadas e como são agrupados os nós dentro da rede.

Uma primeira abordagem consiste em criar sessões de acordo com um tempo limiar T. Usuários que chegarem ao sistema dentro desse tempo limiar constituem uma sessão. Junto com o servidor, clientes que pertencem à mesma sessão formam uma árvore de *multicast* na camada de aplicação, que é chamada de árvore base. O servidor transmite o vídeo normalmente e, quando um novo nó chega ao sistema, ele pega o streaming de um dos nós que já faz parte da árvore base, enquanto a parte que perdeu do vídeo ele obtém de um dos nós da árvore que já o tem armazenado no cache.

Outro tipo de estrutura de árvore é o *Cache-and-Relay* [2] onde, além do tempo de chegada, é utilizado o tamanho da janela deslizante do cache dos participantes para agrupá-los em *clusters*. O usuário "A" que chega primeiro ao sistema obtém o vídeo diretamente do servidor e armazena o conteúdo no cache usando uma janela deslizante, centrada no seu ponto de reprodução, e encaminha o vídeo para os pares que cheguem depois ao sistema, cuja parte de interesse do vídeo esteja ainda dentro da janela deslizante de A. Esses usuários de forma similar a "A" encaminham o vídeo para outros usuários, formando, assim, um cluster. Um usuário "B" que chegar fora dessa janela é forçado a obter o vídeo diretamente do servidor e a formar outro cluster de forma similar a "A", junto com os usuários que chegarem depois dele.

Outra forma de agrupar os pares pode ser em gerações e em sessões. O vídeo é dividido em blocos de recuperação (*Retrieval Block*), onde cada bloco é equivalente a uma unidade de tempo de reprodução. Os Blocos-R são numerados de acordo com sua posição com respeito ao tempo no vídeo, de 1 até o cumprimento total do vídeo. Os nós participantes contribuem com o sistema com um espaço na memória (*buffer*) equivalente a *n* unidades de tempo de reprodução (n Blocos-R) para armazenar em cache o conteúdo mais recente do *streaming* que estão recebendo. Cada cliente pode servir outros clientes que chegarem depois dele, entre o seu tempo de chegada (tchegada) e o tempo de chegada mais o tamanho do buffer (TBuffer), ou seja no intervalo [tchegada, tchegada+TBuffer]. Os clientes que tiverem o mesmo grupo de Blocos-R armazenados em cache são agrupados em gerações. Os clientes dessas gerações, excluindo o servidor, formam uma sessão de vídeo [8].

Ao invés de as sessões serem organizadas pelo tempo de chegada e o tamanho do buffer, outra forma pode ser que o servidor comece uma sessão de transmissão de *broadcast* do vídeo a cada *m* minutos. Assim, os pares que entrarem no sistema desde o início da transmissão até o minuto *m* formam a sessão 1. Do mesmo jeito, os pares que

chegarem depois ao sistema formam as sessões 2, 3, etc. Em cada sessão, os pares que chegarem primeiro ao sistema são os filhos diretos do servidor e os pares que chegarem depois que a largura de banda do servidor esteja totalmente ocupada, são redirecionados para se converterem em descendentes dos que chegaram primeiro. Para reforçar o esquema de transmissão do vídeo, o servidor envia uma lista aleatória de alguns dos participantes a cada usuário que está entrando ao sistema. Quando um usuário entra tarde em uma sessão e perde a parte inicial do vídeo, ele seleciona alguns dos nós da lista aleatória, que o servidor lhe enviou, para baixar a parte inicial que ainda não assistiu [4].

Outra abordagem consiste em dividir a árvore em camadas. Cada nível da árvore é uma camada e, além do mais, é introduzido o conceito de múltiplos pais para dotar o sistema de maior robustez contra a falha de um nó do que os sistemas com árvores de um pai só. As camadas são divididas com respeito ao segmento de vídeo que os pares de cada camada estejam assistindo. Portanto, a camada *i* é a camada dos pares que estão assistindo o segmento *i* do vídeo. Cada par que quer entrar no sistema contata o servidor central da rede para que este lhe indique os nós candidatos a pai dele na camada imediatamente acima da camada na qual ele quer entrar [11]. A figura 2.4 mostra uma estrutura baseada em árvore.

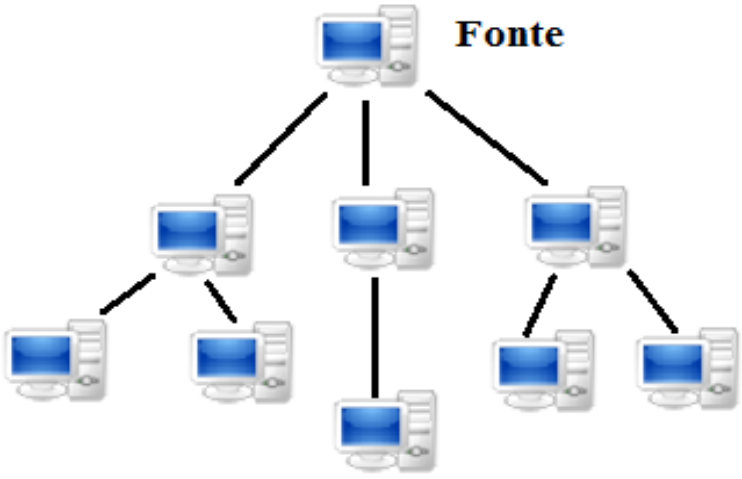

Figura 2.4: Estrutura baseada em árvore (Adaptada de [2]).

2.1.2.2 - Estrutura baseada em malha

Os sistemas baseados na estrutura de malha têm mostrado um melhor desempenho do que os sistemas baseados na estrutura de árvore, e a maioria das pesquisas atuais sobre VoD são baseadas nesta estrutura [12]. Eles têm uma robustez intrínseca, uma boa resposta à volatilidade dos pares, adaptabilidade aos ambientes dinâmicos e um uso

eficiente da largura de banda disponível nos pares. Mas, como esses sistemas não têm uma estrutura bem definida, a descoberta do conteúdo e o roteamento das mensagens são mais complicados [23].

Uma malha pode ser totalmente conectada (*fully connected*), onde todos os pares têm conexões com todos os outros pares da rede, ou parcialmente conectada (*partially connected*), onde alguns dos pares estão conectados a mais de um par na rede. Uma das questões que varia na malha parcialmente conectada é a forma como são selecionados os vizinhos, por exemplo se é um grafo n-randomico, os pares têm "n" vizinhos que são selecionados aleatoriamente (GNR, do inglês *Random N-Regular Graph*), ou se é um grafo onde o número de vizinhos é selecionado de acordo com uma distribuição de Poisson (GNP, do inglês *Poisson N-Regular Graph*) [92].

Atualmente existem várias propostas sobre sistemas de P2PVoD que usam a estrutura de malha para a construção do *overlay* na camada de aplicação [1, 5, 6, 7]. Onde não existe a relação pai/filho como na estrutura de árvore, mas sim, de vizinhos ou relações de vizinhança entre os pares. A diferença básica entre os sistemas que usam este tipo de estrutura é no modo como é formada a malha e como um par seleciona os seus vizinhos e como ele distribui o conteúdo aos seus vizinhos.

Uma abordagem muito usada é a de que cada par que quer se unir à rede tem que contatar um *tracker* que lhe envia uma lista dos pares interessados no mesmo vídeo. O novo par seleciona aleatoriamente entre 8 e 10 pares dessa lista para estabelecer parcerias. Com as parcerias estabelecidas o par fica pronto para compartilhar largura de banda e trocar informações sobre a disponibilidade dos dados, e através de um escalonador procurar os segmentos de vídeo que precisa [1]. Esse esquema de construção da rede é chamado de *Data-Driven Overlay Network (DONet)* [24] ou *Swarming* [22, 27, 70] onde os pares são agrupados em enxames.

Um método parecido com o anterior pode ser o de formar a rede do mesmo jeito, mas com a diferença de que quando um par começa a assistir a um vídeo, ele informa ao *tracker* que está retransmitindo esse vídeo e também quando não tem mais o vídeo armazenado no cache e vai parar de retransmitir. Quando um par quer começar a assistir um vídeo, ele contata o *tracker* para encontrar pares que tenham esse vídeo. E é assim que se faz a seleção dos vizinhos [5].

Outra técnica é criar os enxames de acordo com o ponto de reprodução de cada par. Agrupar os pares de acordo com o ponto de reprodução garante uma troca de segmentos de interesse entre os pares do mesmo enxame. Mas, para incentivar a troca

de segmentos com pares dos enxames que estejam mais na frente, do ponto de vista do ponto de reprodução do vídeo, o servidor força o envio dos segmentos, que os pares que estejam mais na frente precisem, para os pares mais atrasados. Isto é feito, com a finalidade de dotar a esses pares com segmentos de interesse para trocar com os usuários mais avançados do sistema, que já têm a primeira parte do vídeo assistida armazenada no seu cache [6].

Uma abordagem interessante é a do InstantLeap (pulo instantâneo). Neste sistema, o vídeo é dividido em *m* segmentos. Um usuário faz parte do grupo *i* se o seu ponto atual de reprodução cai dentro do segmento *i* do vídeo. Os pares dos grupos *i*-1, *i* e *i*+1 provavelmente têm conteúdo de buffer que se sobreponha, sendo, assim, possíveis pares provedores entre si. Um par do grupo *i* tem duas listas de vizinhos: uma lista que é um subconjunto do seu grupo *i* e dos grupos adjacentes *i*-1 e *i*+1 para descarga e troca de blocos do vídeo; a segunda lista inclui pares que não estejam nos grupo *i*-1, *i* e *i*+1, para que, no caso do par precisar pular do seu ponto de reprodução para outro, ele possa usar um nó dessa lista como um atalho para alcançar essa outra parte do vídeo.

Um exemplo de uma estrutura baseada em malha pode ser visto na figura 2.5.

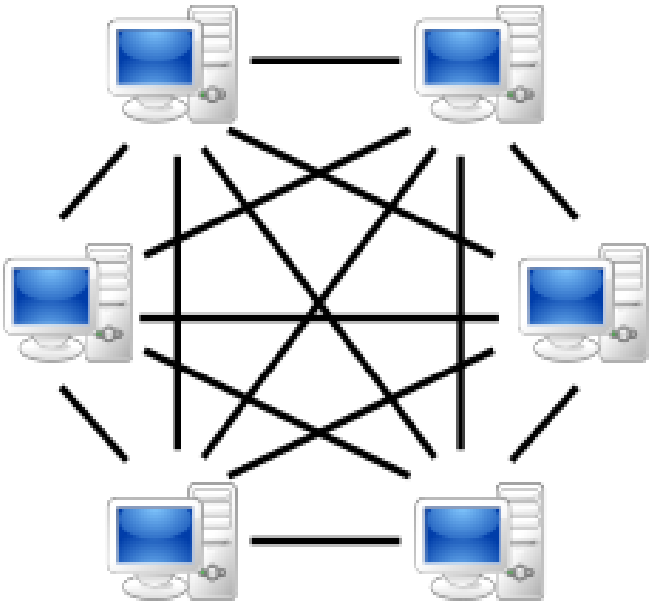

Figura 2.5: Estrutura baseada em Malha (Adaptada de [13])

2.1.2.3 - Estrutura híbrida (Árvore - Malha)

São poucos os sistemas que usam essa estrutura por causa da complexidade. Um exemplo desses sistemas é o TAG [3]. O TAG combina as estruturas de árvore e malha para o escalonamento dos dados de mídia em ambientes de VoD. O TAG usa principalmente a estrutura de malha para entregar os dados, mas a seqüência é

determinada pela estrutura de árvore. Este tipo de arquitetura aumenta a dificuldade da implementação do sistema, e ainda sofre com a descontinuidade na reprodução do vídeo. Um exemplo de estrutura híbrida pode ser visto na figura 2.6.

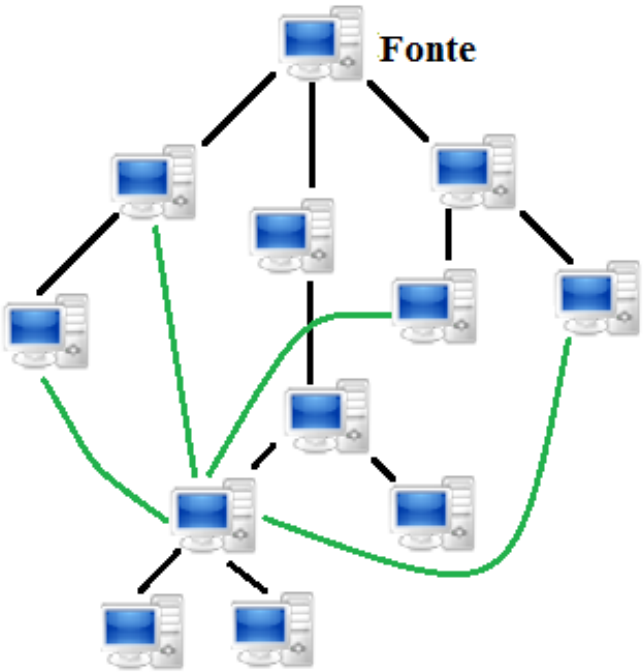

Figura 2.6: Estrutura híbrida (Adaptada de [3]).

2.1.3 - Classificação de acordo com o mecanismo de descoberta do conteúdo

Sem exceção, todos os sistemas P2P são baseados em um dos seguintes métodos para a disseminação e a busca de conteúdo entre os pares da rede [5, 88]:

- Tabelas Hash Distribuídas (DHT, do inglês *Distributed Hash Table*)
- Índices centralizados (*Trackers*)
- *Gossip*

2.1.3.1 - Mecanismos baseados em DHT

Os sistemas que usam o mecanismo do DHT [10-12] utilizam uma função *hash* para atribuir uma chave a um arquivo e aos nós do sistema de acordo com um parâmetro único, em uma operação que é chamada de *hashing*. No caso dos arquivos, pode ser pelo nome e no caso dos nós pode ser pelo endereço IP. Depois de realizar o processo de *hashing*, cada nó do sistema tem que cuidar de um determinado número de chaves. Uma forma de distribuir as chaves é dividindo o número de chaves (k) pelo número de nós (n). Assim, cada nó tem que cuidar de k/n chaves.

Com base no que foi exposto acima, quando um nó for procurar por um vídeo em particular no sistema, ele procura pelo nó que têm que cuidar da chave associada a esse vídeo. Quando um nó abandona a rede, as chaves têm que ser redistribuídas entre

os nós. Portanto, este método de DHT não é recomendado para sistemas onde os pares apresentem uma volatilidade alta, entrando e saindo do sistema frequentemente, porque precisariam de muitas atualizações, o que poderá trazer sobrecarga de mensagens de controle e de atualização. Uma proposta para lidar com essa dinamicidade do sistema é um método de atualizações "preguiçosas" chamado de t-DHT (*Temporal DHT*), descrito em [9].

Em [10], D. Wang e C.K. Yeo usam um roteamento DHT para criar atalhos entre os pares na rede para diminuir o tempo de busca de um segmento de vídeo.

Em [11], os autores usam uma estrutura baseada em DHT, chamada anel *CHORD* [14], e propõem um sistema de múltiplas camadas, onde cada camada é um anel lógico *CHORD*. A união das diferentes camadas forma uma estrutura cilíndrica. As camadas são definidas de acordo com os segmentos de interesse dos pares do sistema. Pares interessados no mesmo conteúdo são agrupados em uma mesma camada. O vídeo é transmitido de camada em camada, a partir da camada mais próxima ao servidor (primeiros segmentos do vídeo), embora um nó filho de uma camada inferior possa pegar os segmentos de vídeo de diferentes pais na camada superior (múltiplos pais).

As estruturas baseadas em anéis *CHORD* são as mais difundidas entre os sistemas que usam o DHT. As variações entre os sistemas dependem do número de anéis e de como é mantida e gerenciada a estrutura e de como é feita a busca do conteúdo entre os participantes.

2.1.3.2 - Mecanismos baseados em índices centralizados

Os mecanismos baseados em índices centralizados têm um *tracker*, que é um servidor central onde estão indexados os pares que têm o conteúdo de interesse que se está procurando, e também pode armazenar informação importante, como a parte do conteúdo que têm capacidade de enviar, o tempo que têm desde sua entrada na rede, sua atividade, etc. Os pares do sistema periodicamente informam ao *tracker* quais partes do vídeo eles possuem enviando o mapa do seu buffer (*Buffer Map*) e mensagens indicando que estão ativos no sistema. Assim, quando um usuário precisar de um conteúdo, ele envia um pedido para o *tracker* que, baseado na informação do conteúdo disponível nos pares, redireciona o usuário para o par ou os pares que possuem o conteúdo de interesse para ele. Depois de receber estes candidatos a vizinhos fornecidos pelo *tracker*, de acordo ao critério de seleção de vizinhos, o par selecionar os pares que possam fornecer de uma forma melhor o vídeo.

Exemplos de sistemas que usam esse mecanismo foram descritos na seção 2.1.2.2.

2.1.3.3 - Mecanismos baseados em *Gossip*

Esse tipo de mecanismo é adequado para estruturas que não têm uma topologia definida (malha), apresentando maior descentralização porque não precisa de um servidor central para buscar conteúdo na rede; o uso de um *tracker* pode ser opcional para dotar o sistema de maior robustez para lidar com falhas e com a volatilidade dos nós. Um par em busca de um conteúdo começa a trocar o mapa do seu *buffer* com os seus vizinhos para saber quais partes do vídeo os seus vizinhos possuem e para informar quais partes ele tem, esse mecanismo é conhecido como método baseado na fofoca (*gossip*) [90]. Quando um par está interessado em um determinado *chunk* que os seus vizinhos não possuem, uma alternativa é espalhar (difundir) mensagens na rede, enviando uma mensagem de que está procurando por esse *chunk* para os seus vizinhos, que encaminham essa mensagem para os seus vizinhos e assim por diante. Esse método é conhecido como *flooding* [19, 91].

Um exemplo claro do funcionamento do mecanismo de *gossip/flooding* pode ser encontrado em [88].

Os mecanismos expostos para a descoberta de conteúdo na rede também são os mesmos usados para o gerenciamento e a manutenção do *overlay*.

2.1.4 - Outras considerações importantes dentro das redes P2P

Existem outros aspectos a serem levados em conta dentro das redes par-a-par. Tais aspectos pela sua natureza não se enquadram dentro das classificações feitas anteriormente, mas precisam ser esclarecidos para um melhor entendimento do funcionamento deste tipo de rede.

2.1.4.1 - Intercâmbio de informação entre os pares

Para a transmissão de um *streaming* a informação pode ser dividida em partes independentes e autônomas chamadas *chunks* ou pode ser organizada em um ou mais fluxos de pacotes individuais. Nesta dissertação vamos focar na transmissão baseada em *chunks* para sistemas de *streaming* de vídeo. Essa transmissão pode ser feita usando quatro abordagens: *push*, *pull* e *push-pull* [70, 82, 83, 86].

Na abordagem *push* os *chunks* são enviados de um par (pai) para o outro (filho) sem perguntar qual *chunk* tem que enviar. Isto pode ser uma causa de ineficiência

porque se o par tiver vários pais, e estiver em uma rede sem uma estrutura definida, pode receber o mesmo *chunk* várias vezes ou deixar de receber alguns outros. Geralmente é associada a redes com estruturas baseadas em árvore [82].

Pull é o contrário a *push*, porque um par pede para um vizinho os pacotes que precisa sem saber se o par tem ou não tem o *chunk* que ele precisa. A duplicação de *chunks* não é um problema, mas um par pode ficar sem *streaming* se não encontrar um vizinho que possua o *chunk* que está procurando. Este tipo de abordagem geralmente é associado a redes sem uma estrutura definida como as baseadas em malha, onde um par tem vários vizinhos para garantir uma diversidade dos pacotes e reduzir a probabilidade de ficar sem *streaming*.

Um caso especial da abordagem anteriormente mencionada é o caso de intercâmbio de tabelas de estado, onde os pares periodicamente trocam com os seus vizinhos um *bitmap* do seu buffer (1 quer dizer que tem o *chunk* no buffer, 0 não o tem), fazendo com que um par saiba quais *chunks* possui cada um dos seus vizinhos e possa pedir os que precisar para o vizinho adequado. Uma desvantagem diz respeito à escalabilidade da rede, pois se a rede for muito grande podem ser gerados congestionamentos ou atrasos dentro da rede por causa da quantidade de mensagens a mais que tem que ser geradas para fazer a troca de informações entre os pares. Cabe resaltar que, para a distribuição de *chunks*, há pouca diferença entre *push* e *pull* em um ambiente onde cada par tem (potencialmente) conhecimento completo dos estados dos outros pares [84].

Uma alternativa, um método híbrido entre *push* e *pull*, surgiu para equilibrar as desvantagens das duas abordagens. O mecanismo de *pull* pode lidar bem com a volatilidade dos pares e diversas condições de rede, mas tem problemas com aplicações que são sensíveis ao atraso [70, 85]. No início da transmissão os pares estão no mecanismo de *pull*, mas assim que o par receber um pacote o pacote é encaminhado para os seus vizinhos sem precisar de um pedido explícito. O contrário também pode ser feito, o par inicialmente está em modo *push* (assim que o par entra na rede um par emissor envia os pacotes sem um pedido explícito) e depois muda para o modo *pull* para obter os pacotes do *streaming* que faltam e que não recebeu quando estava no modo *push* para reconstruir a sequência de vídeo que está assistindo [86].

2.1.4.2 - *Scheduling* (Escalonador)

Depois de definidas as formas de intercâmbio de informação entre os pares, é preciso falar sobre a forma como um par decide qual *chunk* ele vai enviar e qual par vai ser o par de destino desse *chunk* selecionado. Quem se encarrega desta função é o escalonador dos pacotes.

Uma estratégia conhecida para selecionar os *chunks* é a de selecionar o último *chunk* útil que o par tem na sua janela (LUc, do inglês *Latest Useful Chunk*). O par de destino pode ser selecionado, por exemplo, aleatoriamente (RUp, do inglês *Random Useful Peer*). Se o par não tem muito conhecimento dos vizinhos, podem ser usadas duas estratégias, uma para seleção do *chunk* baseada no *deadline* de reprodução, onde o *chunk* com o menor *deadline* é selecionado e outra para selecionar o par, onde um par é selecionado como alvo se precisa um *chunk i* e possui o último *chunk k* com o menor tempo de geração, isto é o tempo de envio do *chunk k* [84].

2.2 - CONSIDERAÇÕES NO DESENHO DE SISTEMAS DE P2PVoD

No desenho de um sistema P2PVoD têm que ser considerados vários fatores. Em [3] os autores propõem considerar o tamanho dos segmentos (o tamanho de cada *chunk* no qual o vídeo é dividido para a sua transmissão) e a estratégia usada para replicar os segmentos na rede, com a finalidade de garantir a maior disponibilidade dos segmentos entre os pares, para satisfazer as demandas de visualização dos usuários sem incorrer em uma sobrecarga excessiva no sistema. A seleção de *chunks* também é um fator importante, assim como a forma como são escalonados para serem recebidos e para serem descartados do buffer, com técnicas já conhecidas na literatura como FIFO (*First In, First Out*), LRU (*Least Recently Used*), LRS (*Least Reusable First*), EDF (*Excessive Duplicated First*) e LSSF (*Least Server Supplementary First*) [26].

Além das questões anteriormente mencionadas, em [8] são considerados fatores essenciais, tais como a entrada de um nó na rede. A entrada do nó tem que ser o mais rápida possível. A localização e a recuperação de falhas também têm que ser rápidas e imperceptíveis para os usuários. O sistema também tem que ter um manejo adequado das petições assíncronas dos clientes e um overhead de controle baixo.

Outra forma de melhorar o desempenho dos sistemas de P2PVoD, do ponto de vista das operações de VCR, é o uso de uma técnica chamada *prefetching*, que consiste em antecipar os segmentos a serem reproduzidos no tocador de vídeo. Já foi se mostrado que os sistemas que implementam o *prefetching* podem obter um desempenho

muito melhor, com respeito a aliviar a carga do servidor, do que os sistemas que não empregam essa técnica [12]. O *prefetching* pode ser feito de uma forma sequencial, aleatória, procurando os *chunks* mais raros primeiro, estabelecendo relações entre os segmentos de um vídeo (intra-vídeo) [4] ou estabelecendo relações entre vídeos (inter-vídeo) [10].

Uma questão importante no projeto de sistemas P2PVoD é a possibilidade de comportamento egoísta por parte de alguns pares. Uma proposta interessante para lidar com este problema é a de [6], que usa o mecanismo de incentivo ao compartilhamento dos recursos do BitTorrent [22, 25] (um dos sistemas mais populares do mundo para o compartilhamento e distribuição de arquivos), conhecido como t*it-for-tat*. Este mecanismo incentiva os clientes para a troca de arquivos dotando os pares mais atrasados (do ponto de vista do tempo de reprodução do vídeo) com segmentos de interesse para os pares mais adiantados, pois os pares mais avançados têm armazenados em seu buffer segmentos que já assistiram e que são de interesse para os pares mais atrasados, mas não estão interessados nos segmentos que os pares mais atrasados têm porque já assistiram essa parte do vídeo.

Cabe mencionar ainda aspectos de confiança computacional, fundamentais para o adequado funcionamento de qualquer sistema P2P. Tais aspectos estão fora do escopo desta dissertação, sendo indicada ao leitor a consulta a [93] e [94].

2.3 - EXEMPLOS DE SISTEMAS P2P DE DISTRIBUIÇÃO DE VÍDEO

Atualmente, existem muitos clientes P2P que oferecem *streaming* de vídeo de graça, mas sem a funcionalidade do vídeo escalável. Entre os mais proeminentes temos:

- Goalbit [28]
- Joost [29]
- Gridmedia [30]
- CoolStreaming [31]
- PPLive [32]
- Tribler [33]

A maioria dos clientes P2P tem o mesmo principio de funcionamento: oferecem shows profissionais de TV junto com conteúdo particular, e faturam dinheiro fazendo publicidade. A vantagem para as estações de TV é que, além de fazer com que os seus programas fiquem disponíveis ao redor do mundo, transferem uma parte dos custos do

broadcasting para os usuários individuais. Dos clientes anteriormente mencionados, apenas o Goalbit é de código aberto.

2.4 - CONCLUSÃO

Neste capítulo, foram apresentados conceitos básicos sobre redes P2P, junto com uma classificação de tais redes, de acordo com o grau de descentralização, os mecanismos de descoberta de conteúdo e o escalonador utilizado pelos pares para enviar/obter *chunks*. Além dos conceitos e da classificação feita, que serão úteis para a elaboração da proposta, foram discutidas algumas considerações no desenho de sistemas de P2PVoD e listados alguns dos clientes P2P mais usados na distribuição de vídeo.

3 – CODIFICAÇÃO DE VÍDEO ESCALÁVEL

3.1 - CARACTERÍSTICAS E BENEFÍCIOS DA CODIFICAÇÃO DE VÍDEO ESCALÁVEL

O termo ”escalável”, no contexto de vídeo, se refere à codificação de uma seqüência de imagens de uma forma progressiva (i.e. escalável), fazendo com que a estrutura interna do vídeo codificado permita um ajuste adequado entre *bitrate* e a qualidade subjetiva. A flexibilidade adicional é fornecida se parte do *bitstream* do vídeo puder ser descartada, e ainda assim, o resultado for uma sequência válida de vídeo. Para fazer isso é necessária uma estrutura de camadas, dentro do vídeo codificado, onde possa se diferenciar a informação básica e as partes que representam apenas detalhes. Assim, um vídeo pode ser ajustado de uma forma rápida e fácil para se adaptar às mudanças das condições da rede ou às capacidades dos usuários finais. Levando isso em conta, foi criado o padrão H.264/SVC (*Scalable Video Coding*).

Algumas vantagens que oferece esta codificação, se comparada com a codificação de vídeo não escalável, são a seguir listadas:

- No caso de *simulcast*, várias versões do mesmo vídeo deverão estar disponíveis para serem entregues a diferentes usuários com diferentes capacidades de processamento, largura de banda, e armazenamento. Obviamente, essas diferentes versões têm um maior grau de redundância e, mesmo que tenham diferentes *bitrate*, todos os *streams* representam o mesmo conteúdo. A codificação de vídeo escalável se esforça para reduzir essa redundância e pode, portanto, produzir um fluxo de vídeo que requer um espaço de armazenamento significativamente menor do que a soma total de todas as versões de um fluxo de vídeo de simulcast [43].
- *Streams* de vídeo escalável podem ser codificados com diferentes *bitrates*. Assim, pode-se ter uma graduação fina, aumentando as opções de escolha de *bitrate* para um conjunto amplo de valores possíveis.
- O gerenciamento de diferentes versões de *bitrate* do mesmo vídeo é evitado. Com o vídeo escalável um só *bitstream* pode servir para vários usuários com diferentes necessidades. Como conseqüência o ajuste do *bitrate* é simplicado. Para ajustar não precisa trocar entre dois *bitstreams* separados, porque essa operação pode ser feita usando o mesmo *stream* de vídeo. Esta conveniência

melhora a flexibilidade do *stream* de vídeo e aumenta a robustez do sistema contra falhas do *link* de transmissão.

- Codificando o vídeo com uma estrutura de camadas, pode se associar diferentes graus de importância a cada uma das camadas. A camada base de um *stream* de vídeo escalável contém informação que é fundamental para a reprodução (*playback*) do vídeo. Assim, a camada base representa a parte mais importante do *stream* de vídeo. Assim que a ordem das camadas de reforço aumenta, a sua importância diminui.

  A vantagem desta estrutura em camadas é que partes específicas do vídeo podem ser priorizadas. Em uma rede com uma largura de banda limitada essa priorização permite preferir pacotes de dados que são essenciais para a reprodução do vídeo. Portanto, nos casos onde não existe suficiente largura de banda para receber o vídeo completo, é possível oferecer pelo menos uma versão de baixa qualidade do vídeo.
- Especialmente para redes P2P a codificação de vídeo escalável oferece outra vantagem que está relacionada às vantagens anteriormente mencionadas. Geralmente, todos os pares de uma rede não formam um grupo homogêneo, diferenciando pela largura debanda disponível ou pelas capacidades de computação. Assim, no caso do *simulcast*, demandariam *bitrates* diferentes do vídeo e, consequentemente, também precisariam diferentes *streams*. Isto levaria ao problema de ter que quebrar o *overlay* em diferentes subgrupos. Cada subgrupo compartilha só uma versão do *stream* de vídeo [38, 40].

Na figura 3.1, pode-se observar a diferença entre o uso do *simulcast* e o uso do vídeo escalável em uma rede P2P. No primeiro caso a fonte envia 3 *bitrates* diferentes para o mesmo vídeo. Os pares são agrupados de acordo com o *bitrate*. No caso do *simulcast* os *bitrates* são representados por 3 *streams* diferentes. Mas, pares com *bitrates* diferentes não podem trocar pacotes, e os pares com *bitrates* iguais têm que formar subgrupos separados. Essa separação faz com que se tenham problemas na rede. Se a largura de banda muda, os pares poderão solicitar um *bitrate* diferente do vídeo que estão assistindo. A única forma para mudar de *bitrate*, no *simulcast*, é solicitar um *stream* diferente. Como os *streamings* são independentes, a mudança de um para outro precisa que o par abandone um grupo e entre em outro. O que levaria tempo adicional, além do problema de ter que fazer de novo o processo de criação da vizinhança [43].

O SVC supera o problema da mudança oferecendo um *stream* que pode fornecer vários *bitrates*. Assim, todos os pares podem fazer parte do mesmo grupo, independente do *bitrate*. Isto simplifica o processo de mudança de *stream* e melhora a robustez do sistema. Além disso, a rede P2P se vê beneficiada também pelo fato de ter um grupo maior de pares no mesmo grupo [38]. O desempenho de uma rede P2P depende do número de pares presentes nela, quanto maior o número de pares, maior a capacidade da rede. E quanto maior o número de pares, mais rápida a resposta da rede para se recuperar da falha de um dos pares.

Entretanto, é relevante falar que a flexibilidade do SVC produz certa quantidade de sobrecarga de dados que reduzem a eficiência da codificação do *stream* de vídeo. Além disso, a complexidade dos codificadores e decodificadores de vídeo escalável é consideravelmente maior, se comparada com os codecs utilizados para vídeo não escalável [40].

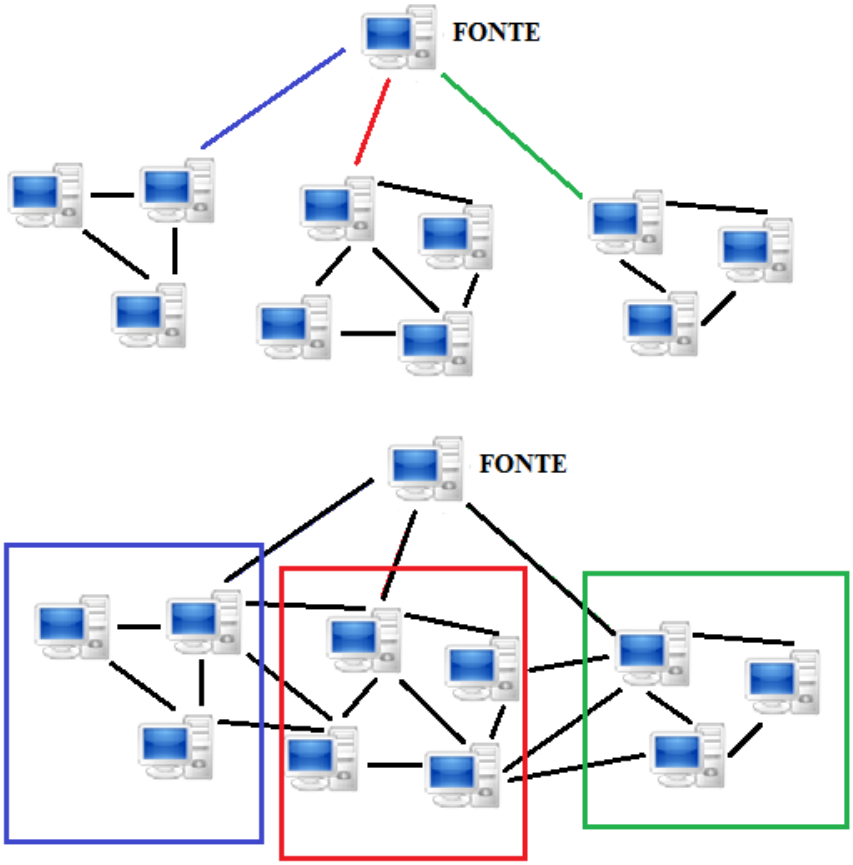

Figura 3.1: Simulcast e vídeo escalável (Adaptada de [43]).

O H.264/AVC abrange uma VCL (*Video Coding Layer*), que foi projetada para representar eficientemente o conteúdo do vídeo e uma NAL, que formata a

representação VCL do vídeo e fornece informações do cabeçalho de forma adequada para transporte por uma variedade de camadas de transporte ou de mídias de armazenamento.

3.2 - PADRÃO H.264/SVC

Nos últimos anos, o vídeo escalável tem sido uma área de alto interesse para os pesquisadores da comunidade de codificação de vídeo, sendo o padrão de Codificação de Vídeo Escalável (SVC) uma extensão que faz parte do padrão H.264/AVC [41] desenvolvida pelo *Joint Video Team* (JVT) do Grupo de Codificação de Vídeo da ITU-T (VCEG – *Video Coding Experts Group*) e pelo Grupo de Especialistas em Imagens em Movimento da ISO/IEC (MPEG – *Moving Pictures Expert Group*).

Como é uma extensão, o SVC depende das principais decisões de desenho do padrão de codificação de vídeo H.264/AVC. Portanto, usa conceitos como codificação hibrida de vídeo (predição inter-*frame* combinada com predição intra-*frame*), codificação da transformada discreta do cosseno (DCT, do inglês *Discrete Cosine Transform*), transformada discreta do cosseno inversa (IDCT, do inglês *Inverse Discrete Cosine Transform*), transformada de Hadamard e transformada de Hadamard inversa, bem como a estrutura de macroblocos subjacente. Uma descrição detalhada desses conceitos pode ser encontrada em [38]. A principal característica do SVC com respeito aos padrões que deram origem a ele, é a estrutura em camadas do vídeo codificado, permitindo que partes específicas do *bitstream* sejam agrupadas de acordo com sua importância na qualidade final do vídeo.

Dentro do padrão SVC três diferentes possibilidades são usadas para dividir o vídeo em várias camadas. Essas três possibilidades são: escalabilidade espacial, escalabilidade temporal e escalabilidade de qualidade ou SNR (*Signal-to-Noise Ratio*) [54].

3.2.1 - Escalabilidade Temporal

A escalabilidade temporal basicamente significa alterar a taxa de quadros (fps – *frames* por segundo) do vídeo. Simplesmente descartar quadros aleatórios da sequência de vídeo não é factível, porque outros quadros podem depender deles para a compensação de movimento. Assim, para fornecer várias taxas de quadros, os quadros devem ser codificados de uma certa forma, chamada de estrutura de predição hierarquica [37].

A figura 3.2 apresenta um exemplo de escalabilidade temporal.

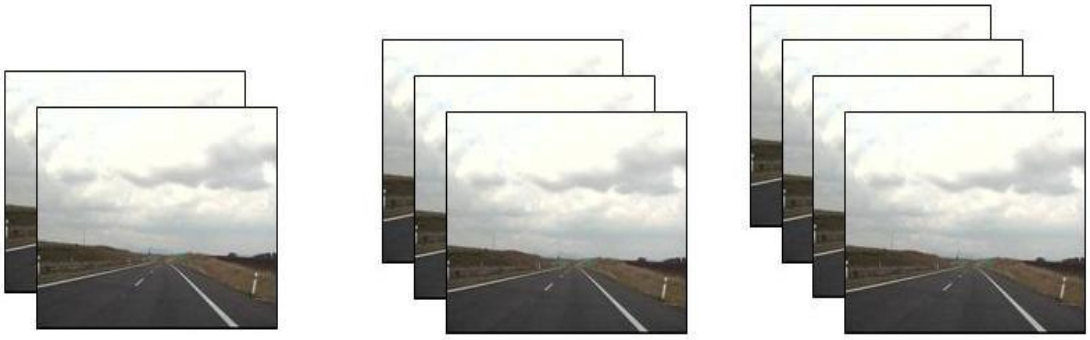

Figura 3.2: Escalabilidade temporal (Adaptada de [95]).

Este tipo de escalabilidade pode ser encontrado na figura 3.3. Os números embaixo dos quadros indicam a ordem dentro do *bitstream* codificado. A figura 3.3(a) apresenta quadros consecutivos e as suas dependências da predição de movimento. Os quadros de referência não são escolhidos arbitrariamente, mas sim com uma hierarquia. Na figura esses níveis de hierarquia são diferenciados pela cor e são denotados por T_k, com *k* sendo o identificador de uma camada temporal. Quadros entre dois quadros consecutivos da taxa de quadros mais baixa (maior hierarquia T_0) junto com o quadro T_0 subsequente são chamados de Grupo de imagens (GOP – *Group Of Pictures*). A questão mais importante desta predição é que quadros de uma camada temporal *k* somente dependem de quadros da mesma camada ou de camadas mais baixas, e camadas com identificadores maiores do que *k* podem ser descartadas do *bitstream*.

As figuras 3.3 (b) e (c) apresentam mais exemplos de predição hierárquica. Na figura 3.3 (b) os quadros estão arranjados de uma forma não diádica, onde a taxa de quadros não se incrementa por um fator de 2 de uma camada para a seguinte. A figura 3.3 (c) mostra a possível solução, onde a predição de movimento ainda é feita de uma forma hierárquica, enquanto se evita um atraso estrutural no processo de decodificação da sequência. Pode-se ver a falta de predição para trás, sem imagens de referência do futuro, e a ordem no incremento dos números dos quadros no streaming codificado [37].

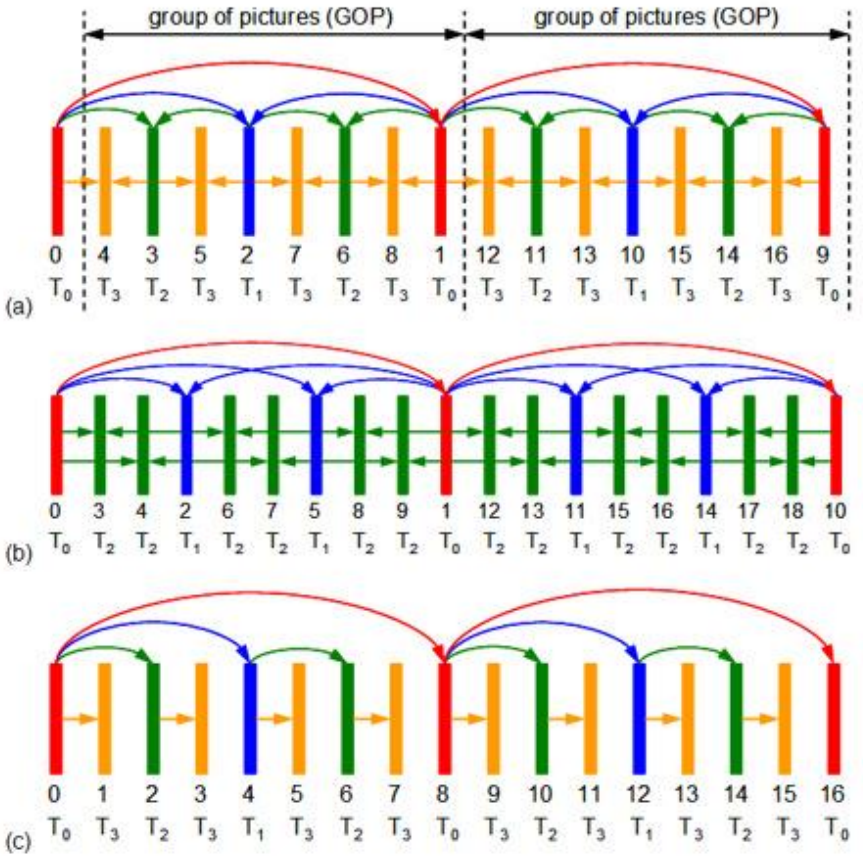

Figura 3.3: Estrutura de predição hierarquica[1].

Basicamente, o conceito de resoluções temporais diferentes pode ser feito usando apenas o padrão H.264/SVC. A flexibilidade avançada do H.264/AVC para escolher e controlar quadros de referência permite o uso da predição hierárquica de movimento [42]. O SVC adiciona métodos para sinalizar e marcar as camadas temporais para subsequentemente extraí-las sem dificuldade.

3.2.2 - Escalabilidade Espacial

A escalabilidade espacial corresponde à escalabilidade usada para prover diferentes resoluções das imagens. Cada nova camada de um *bitstream* como escalabilidade espacial melhora a resolução final da imagem. A figura 3.4 apresenta um exemplo de escalabilidade espacial.

[1] http://ip.hhi.de/imagecom_G1/savce/images/fig_5.jpg

Figura 3.4: Escalabilidade Espacial (Adaptada de [95]).

A figura 3.5 mostra a ideia da escalabilidade espacial com uma camada base S_0 e duas camadas de reforço (S_1 e S_2). Neste caso, não só o tamanho mas também o número de quadros por camada é diferente, o que leva a uma combinação de escalabilidade espacial (S_k) e temporal (T_k).

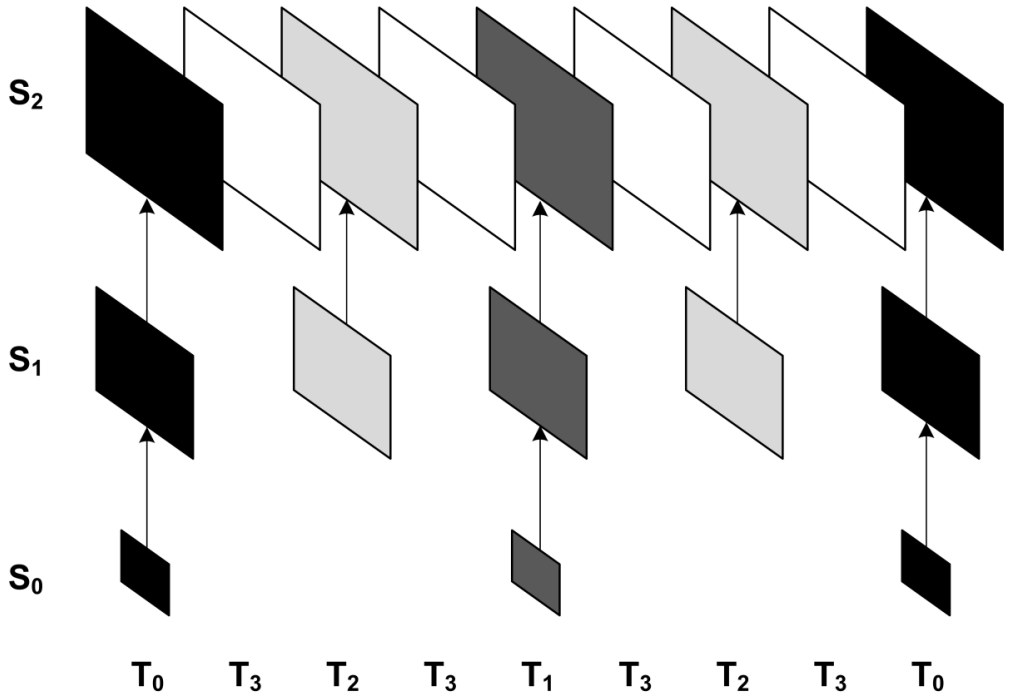

Figura 3.5: Escalabilidades espacial e temporal (Fonte: [95]).

As setas verticais na figura anterior descreven o conceito da predição inter-camadas. Este método de predição se esforça para reutilizar a maior quantidade informação possível de uma camada para a seguinte. Isto evita a redundância entre camadas e subsequentemente melhora a eficiência na codificação. Como é feito na

predição de movimento dentro de uma camada, no caso da predição inter-camadas primeiro é predita a imagem final a partir da imagem correspondente na camada de referência, e finalmente apenas as diferenças com relação à imagem atual (também chamadas de residuais) são codificadas [38].

A forma mais eficiente para fazer a predição inter-camadas seria depender da imagem completamente reconstruída ou decodificada da camada subjacente. Este método direto, no entanto, aumentaria significativamente a complexidade do decodificador, pelo fato de ter que decodificar totalmente todas as camadas subjacentes. Para manter o processo de decodificação simples e evitar múltiplos laços de decodificação, o padrão SVC fornece métodos de predição inter-camadas que permitem a chamada compensação de movimento de laço simples [40]. Com a compensação de laço simples, a estimação do movimento no codificador é realizada em todas as camadas, enquanto o processo dispendioso de compensação de movimento no decodificador somente é requerido nas camadas alvo. Isto é garantido se a informação das camadas inferiores pode ser usada na camada alvo sem decodificação.

Mesmo que a compensação de movimento de laço simples ligeiramente diminua a eficiência da codificação, ela simplifica significativamente a estrutura do decodificador [38, 40]. Assim, as técnicas de predição inter-camadas que vão ser descritas a seguir reciclam informação de camadas de níveis inferiores sem decodificá-los completamente.

3.2.2.1 - Predição de movimento inter-camadas

A predição de movimento convencional do H.264/SVC e dos padrões similares tem dois componentes principais que são codificados para cada macrobloco B ou P: o vetor de movimento e a informação residual correspondente. Segundo isto, os primeiros dois métodos de predição inter-camadas se concentram nessas duas partes. Uma vez que é provável que os vetores de movimento não mudem muito de uma camada para a seguinte (excetuando o fator de escalamento), eles oferecem uma chance muito boa para reduzir a redundância. A predição de movimento inter-camadas faz uso deste fato copiando os vetores de movimento e informação adicional (partição de macroblocos e os índices das imagens de referência) em camadas que estão acima na hierarquia.

3.2.2.2 - Predição residual inter-camadas

Além dos vetores de movimento, o SVC também fornece os recursos para reutilizar a informação residual. Portanto, somente o sinal residual da diferença entre a camada "*upsampled*" e a sua correspondente camada inferior precisa ser codificada. Apesar das

predições de movimento e residual inter-camadas explorarem juntas a maioria da informação apresentada na camada inferior, ainda não precisam decodificar a camada de referência. Assim, o conceito de compensação de movimento de laço simples ainda é mantido.

3.2.2.3 - Predição intra inter-camadas

Esta técnica de predição vai um passo além. A predição intra inter-camadas usa uma versão reconstruída da imagem correspondente da camada de referência como predição. Para isto precisa da decodificação da camada inferior e, portanto, apenas é permitida para macroblocos intra-preditos. Estes tipos de macroblocos são preditos sem referência de movimento dos outros quadros (não usam nenhuma forma de compensação de movimento), o que garante que possam ser decodificados sem ter que rodar um laço separado de compensação de movimento.

3.2.3 - Escalabilidade de Qualidade (SNR – *Signal-to-Noise Ratio*)

Durante o processo de codificação a informação da textura de cada macrobloco é transformada para o domínio da frequência. Por razões de eficiência os coeficientes da transformada são quantizados antes de serem codificados [41]. Desta forma, os coeficientes são mapeados sobre um número limitado de níveis de quantização. O que a escalabilidade de qualidade (também chamada SNR ou de fidelidade) faz, basicamente, é afetar o número de níveis de quantização. Desse modo, a escalabilidade de qualidade consegue esse refinamento na qualidade da imagem diminuindo gradualmente o tamanho de passo da quantização. Quanto menor o tamanho de passo da quantização, maior o número de níveis de quantização, e consequentemente, tem-se uma graduação mais fina da transformada dos coeficientes. Finalmente, isto leva a uma qualidade melhorada da imagem com um alto grau de detalhes [42].

A figura 3.6 apresenta um exemplo de escalabilidade SNR.

Figura 3.6: Escalabilidade SNR (Adaptada de [95]).

Olhando mais de perto, este método pode ser considerado um caso especial da escalabilidade espacial. Se a resolução da imagem não muda de uma camada espacial para a seguinte, o refinamento também é realizado somente utilizando um menor tamanho de passo de quantização [40]. Esta ideia de dividir o vídeo em várias camadas de qualidade é chamada de escalabilidade de qualidade de granularidade grossa (CGS – *Coarse Grain Quality Scalability*). Por esse motivo as camadas CGS podem ser consideradas casos especiais das camadas espaciais. Uma desvantagem da CGS é inerente às camadas com escalabilidade espacial. A predição de movimento é feita separadamente em cada camada espacial, como pode ser visto na figura 3.7 (a), o que faz com que a mudança entre duas camadas espaciais só seja possível em pontos bem definidos dentro do *stream* do vídeo (só nos quadros I) [42].

Assim, o SVC também considera outros recursos para a escalabilidade de qualidade além da CGS. Todas essas alternativas ocorrem dentro da mesma camada espacial e, portanto, todas as camadas de qualidade contam com o mesmo laço de compensação de movimento. Nas figuras, 3.7 (b), (c) e (d), este fato é apontado pela falta da brecha entre as camadas de qualidade, porque todas as camadas de qualidade dentro da mesma camada espacial estão baseadas no mesmo laço de predição de movimento. E a questão mais importante agora é resolver o problema de qual nível de qualidade da imagem de referência vai ser usado para a compensação de movimento.

Uma possível solução, figura 3.7. (b), é a Granularidade de Escalabilidade Fina (FGS - *Fine-Grain Quality Scalability*). A compensação de movimento é feita usando o nível mais baixo de qualidade da imagem de referência. A principal vantagem da FGS é que a estimação e a predição do movimento não sempre usam o mesmo nível de qualidade da imagem de referência. Isto é garantido porque ao menos a camada base sempre está disponível no receptor. Se a predição de movimento é somente feita com a camada de granularidade fina mais baixa, uma perda de pacotes de refinamento não influencia o *loop* de compensação de movimento. Este fato permite escalar o *bitrate* de um *stream* em um nível baseado em pacotes, simplesmente descartando pacotes específicos fora da camada de reforço. Por isso, a FGS garante que o codificador e o decodificador estejam sincronizados em todo momento. Porém, esta solução reduz a eficiência da codificação, porque a informação da camada de reforço não pode ser explorada para predição.

A maneira obvia de reutilizar a maior quantidade de informação da imagem de referência que for possível seria a de fazer a predição de movimento nas camadas de

qualidade mais alta. Esta técnica pode ser vista na figura 3.7 (c). Enquanto a imagem de referência é usada de uma forma muito eficiente, o *stream* se torna susceptível a um fenômeno chamado *drift*. O *drift* descreve uma situação na qual os *loops* de predição de movimento do codificador e do decodificador não estão trabalhando mais com a mesma imagem de referência. Se alguns pacotes da camada de reforço foram descartados durante a transmissão, o decodificador não pode detectar esta situação, e usa uma imagem de referência de qualidade inferior para a compensação de movimento como o codificador fez. Na figura 3.7 (c) a estimação do movimento no codificador é feita na camada de reforço, enquanto que para o terceiro *frame* apenas foi recebida a camada base no decodificador.

Portanto, SVC introduz uma troca justa entre CGS e FGS, para refinar a qualidade do vídeo, enquanto mantém o *drift* em um nível aceitável. A idéia da escalabilidade de qualidade de granularidade média (MGS – *Medium-Grain Quality Scalability*) é explicada na figura 3.7. (d). A melhora com respeito à já mencionada FGS repousa na flexibilidade da escolha da camada de qualidade que será utilizada para a predição de movimento. Desta forma, a predição de movimento pode ser realizada usando uma camada de reforço, mas com atualizações periódicas na camada base. Essas atualizações periódicas acontecem nas chamadas imagens chaves, e sincronizam o *loop* de compensação de movimento do decodificador com o *loop* do codificador. Desta forma, a MGS garante que os efeitos do *drift* sejam reduzidos [41].

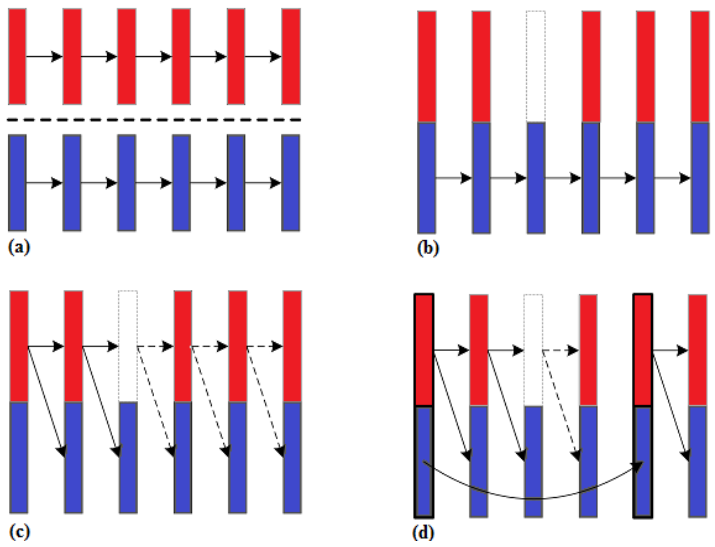

Figura 3.7: Métodos de escalabilidade de qualidade[2].

[2] http://ip.hhi.de/imagecom_G1/savce/images/fig_8.jpg

Com o conceito de MGS, qualquer unidade NAL de uma camada de reforço pode ser descartada de um *bitstream* com escalabilidade de qualidade, e assim a codificação de escalabilidade de qualidade baseada em pacotes é fornecida. Além disso, SVC oferece dois recursos para a codificação de escalabilidade de qualidade:

- Particionamento dos coeficientes transformados: O SVC oferece a possibilidade de distribuir os coeficientes da transformada da camada de reforço entre várias fatias. O primeiro e o último índice dos coeficientes escaneados da transformada são sinalizadas nos cabeçalhos das fatias, e a fatia de dados inclui apenas os níveis dos coeficientes da transformada para escanear os índices dentro do intervalo sinalizado. Assim, a informação para uma imagem de refinamento de qualidade pode ser distribuída ao longo de várias unidades NAL.
- Reescritura SVC-a-AVC: SVC também suporta a criação de bitstreams que podem ser convertidos em um dos perfis não escaláveis do H.264/SVC, usando um processo de reescritura que é menos complexo do que a transcodificação do bitstream SVC.

3.2.4 - Escalabilidade Combinada

Todos os três tipos de escalabilidade anteriormente mencionados podem ser combinados entre eles.

A seguir, na figura 3.8 é mostrada a estrutura esquemática do *bitstream*. A classificação mais importante se refere às camadas espaciais do *stream* do vídeo SVC. Dentro de uma camada espacial podem se encontrar uma ou mais camadas temporais ou de qualidade.

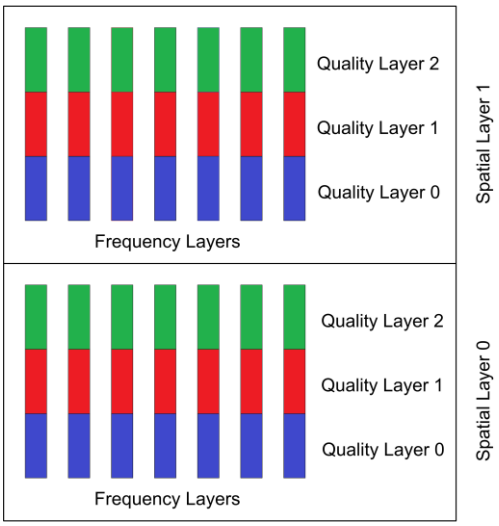

Figura 3.8: Estrutura esquemática do *Bitstream* (Adaptada de [38]).

A estrutura do SVC é organizada em camadas de dependência. Essa dependência normalmente representa uma camada com escalabilidade espacial específica. Em alguns casos, pode haver dependência de duas camadas com resolução espacial. As dependências das camadas são apresentadas pelo número identificador.

A resolução espacial não diminui em relação à primeira camada e à camada seguinte. Quando uma camada de referência é uma camada de dependência com resolução espacial e esta possui representações de qualidade diferentes, ela precisa ser sinalizada para saber qual será utilizada na predição inter-camadas.

Para permitir uma adaptação simples do *bitstream*, o SVC fornece alguns meios pelos quais os *substreams* que estão contidos em um *bitstream* escalável completo possam ser identificados facilmente.

O codificador de vídeo usado foi o JSVM, codificador que será explicado no Apêndice II.

3.3 - CODIFICAÇÃO DE VÍDEO ESCALÁVEL E AS REDES P2P

Embora ambas as áreas (vídeo escalável e redes *peer-to-peer*) tenham chamado a atenção de muitos pesquisadores e instituições de pesquisa nos últimos anos [64] [65] [66], a quantidade de *papers* científicos falando da sua interação é pequena. Embora às vezes haja vários *papers*, boa parte foi publicada pelos mesmos autores e versam sobre os mesmos sistemas P2P. O que mais tem chamado a atenção para os sistemas P2PVoD usando codificação de vídeo escalável (SVC) é o fato de proverem uma adaptação eficiente a recursos heterogêneos e às dinâmicas da rede, além das vantagens do uso de redes P2P na distribuição do vídeo, como a redução da carga no servidor pela participação dos pares na distribuição do conteúdo em larga escala [1].

A seguir descrevemos alguns sistemas que se baseiam na interação entre P2P e SVC.

Em [64] os autores exploram as vantagens das redes P2P e do SVC para propor um mecanismo de *streaming* eficiente e adaptável. O SVC é usado para garantir a entrega "suavizada" do conteúdo de vídeo entre os pares da rede (*Smooth Video Delivery*) escolhendo os pares que contribuem, com diferentes partes do vídeo, de acordo com a QoS oferecida. O vídeo é codificado para oferecer unicamente escalabilidade temporal (7.5, 15 e 30 fps), apresenta uma resolução QCIF (176x144) e só uma opção de escalabilidade SNR. A camada mais importante do vídeo (a camada base) é obtida do par com a melhor QoS oferecida. A QoS oferecida é calculada usando

a largura de banda que o par está disposto a contribuir durante a transmissão do vídeo. A largura de banda disponível, entre o transmissor e o receptor, é estimada por meio do RTT (*Round Trip Time*).

A escolha do RTT é justificada pelo fato de que ele pode ser facilmente medido e por ser o parâmetro mais representativo da qualidade do *link*, segundo os autores [64], porque o congestionamento em determinado *link* pode ser detectado se o valor do RTT aumenta muito. Além disso, de acordo com o seu RTT, um par pode ser marcado como não apto para a sessão de *streaming*, o que causa a procura por outros pares disponíveis e com um RTT menor na rede P2P, ou, se não tiver nenhum par disponível com um melhor RTT, é feita uma adaptação do *streaming*, onde o par reajusta as camadas de vídeo recebidas dinamicamente, fazendo com que a qualidade do vídeo seja adaptada de acordo com as condições da rede.

Cabe resaltar: que a abordagem de [64] é centrada no receptor, que é quem decide quais pares serão os seus vizinhos, avalia a qualidade deles e decide se troca os vizinhos ou solicita uma adaptação do *streaming*; que a aferição da QoS é feita de uma forma dinâmica para contrarrestar os fatos de que os pares podem abandonar a rede ou deixar de contribuir com conteúdo; que a largura de banda pode variar a cada momento de acordo com as condições da rede; que pares novos podem entrar na rede e oferecer melhor largura de banda; e que muito tráfego pode gerar mais perdas de pacotes. Assim, a avaliação da QoS depende altamente de como são selecionados os pares e de quão rápido são detectadas as falhas de pares dentro da rede. Essas falhas são detectadas de duas formas: se o par não recebe nenhuma resposta do par que está enviando o vídeo, ou se o número de pacotes recebidos armazenados no *buffer* é menor do que um valor limiar específico.

As simulações para os testes de desempenho em [64] foram feitas com a qualidade do *streaming* de um vídeo escalável enviado sobre uma rede *peer-to-peer* com uma topologia similar à de uma rede *Gnutella* [39] usando o *software*, para simulação de redes p2p de propósito geral, *Network Simulator* (NS-2). A avaliação, de fato, mostra uma melhora na taxa de vazão do vídeo nos pares e uma melhor qualidade do vídeo recebido com relação a outras abordagens de *streaming* convencionais. A qualidade do vídeo é avaliada usando o PSNR (item 4.1.1 desta dissertação). Uma questão a ser levada em conta é que o uso de uma rede similar à do *Gnutella* implica que a topologia entre os pares é construída de uma forma arbitrária, sem uma estrutura definida do *overlay*.

No artigo de Zhang e Yuan [65], os autores exaltam as características da codificação de vídeo escalável, e mostram as vantagens do uso deste tipo de codificação se usado junto a uma rede P2P para a distribuição de conteúdo de vídeo. Propõem um *streaming* adaptativo sobre uma rede P2P baseada na topologia do *BitTorrent* [25], para otimizar a latência dos *links* e para aproveitar a largura de banda de *upload*, para redes heterogêneas e flutuações de largura de banda. O vídeo é codificado para oferecer cinco camadas de escalabilidade temporal (1.875, 3.75, 7.5, 15 e 30 fps), apresenta duas opções de escalabilidade temporal (resoluções CIF e QCIF) e duas opções de escalabilidade SNR (Q0 e Q1).

No sistema proposto, os pares são organizados de tal forma que cada um possa receber uma camada específica, que pode se auto-adaptar de acordo com as capacidades de largura de banda disponíveis, e os pares trocam pacotes de dados com vizinhos da mesma camada ou recebem pacotes dos pares das camadas superiores.

Como a rede é baseada na topologia usada no *BitTorrent* (rede híbrida descentralizada), apresenta um servidor de vídeo, um *tracker* e os pares. O servidor está dividido em dois servidores: de *upload* e de cache, sendo que o de *upload* lê e empacota o arquivo de vídeo e o envia para o servidor de cache que armazena o vídeo e funciona como uma fonte de fornecimento do *streaming* de vídeo. O *tracker* facilita a interação entre os pares e armazena informação importante sobre o *streaming* de vídeo escalável. Por último, os pares que formam a rede, os quais selecionam os vizinhos baseados no RTT e dando prioridade aos pares de camadas superiores.

Depois de formada a rede é que se aplica o esquema adaptativo proposto em [65], o qual tem dois módulos que fornecem a adaptação a redes heterogêneas e a auto-adaptação às flutuações da largura de banda:

O primeiro módulo, que se encarrega da locação inicial dos pares, é chamado sempre que um par novo entra na rede e começa a solicitar um vídeo. O processo de classificar em qual camada do vídeo se enquadra o par é feito medindo inicialmente a largura de banda do par e selecionando uma camada do vídeo com as correspondentes características de escalabilidade espacial, temporal e de qualidade de acordo com a informação contida no arquivo de *bitstream* do vídeo.

O segundo módulo, que se encarrega da locação progressiva conforme o vídeo está sendo reproduzido, é chamado periodicamente para "otimizar" os vizinhos dos quais o par vai receber partes do conteúdo de vídeo, usando a taxa de dados média do e decidindo sobre este valor se o par pode ser relocado em outra camada (superior ou

inferior segundo o caso) se existem mudanças notáveis nas condições da rede ou na largura de banda disponível. Se alguma relocação for feita, o par tem que transmitir uma mensagem aos vizinhos que dependem dele como uma fonte, para eles atualizarem as suas listas.

Além dessa atualização por realocação de um par, uma atualização periódica é feita também para encontrar vizinhos inativos usando mensagens do tipo *handshake* e *handshakeback* para garantir a interatividade entre os pares e manter uma lista eficiente dos vizinhos, caso um par precise selecionar outro par para substituir um dos seus vizinhos o valor do RTT é usado. Como no caso de [66], a qualidade é medida usando o PSNR, e mostrando, por meio de simulações, como muda a adaptação no *streaming* com respeito a redes de diferentes larguras de banda e a flutuações da largura de banda disponível.

Em [67], usando uma combinação da codificação de vídeo escalável (SVC) com a codificação de vídeo por múltiplos descritores (MDC) na transmissão do vídeo os autores fazem com que pares com capacidades heterogêneas recebam a qualidade de vídeo adequada com a sua capacidade. MDC é usado para aliviar a dependência entre camadas do SVC porque, em comparação com o SVC, cada descrição do MDC pode ser decodificada independentemente.

A combinação consiste em codificar várias camadas SVC em descrições MDC e cada par recebe um número de descritores determinado pela sua capacidade de largura de banda de *upload*. A fonte divide o vídeo em pequenos blocos e cada bloco é associado a um par para sua redistribuição usando UDP. TCP é usado para que a fonte receba *feedback* dos pares sobre a taxa recebida. UDP é usado na distribuição do vídeo porque gera um atraso menor do que o gerado pelo TCP.

Com um mecanismo de controle de fluxo centrado na fonte, sobre uma rede P2P com uma estrutura baseada em malha, dinamicamente otimizam a vazão total da rede e ajustam a qualidade do vídeo para cada par, para se adaptar às condições temporais de capacidade da rede e para reduzir a taxa de perda de pacotes. Usando a plataforma de testes de redes P2P *PlanetLab* [68], a qualidade do sistema é avaliada em função da vazão e da perda dos pacotes, embora para redes em malha com poucos pares, pois os testes são feitos com uma fonte e quatro nós apenas.

Nunes et al. [69], baseados na topologia do *BitTorrent*, como em [64], implementam uma arquitetura para a distribuição de vídeo escalável. O protocolo do *BitTorrent* é modificado para atingir as condições necessárias no *streaming* distribuído

de vídeo escalável: a primeira é que a perda de pacotes de camadas de refinamento não afeta a reconstrução nem a decodificação de camadas inferiores; a segunda é maximizar a qualidade percebida pelo usuário final, adaptando em tempo real o *streaming* de vídeo às capacidades do dispositivo do usuário e à largura de banda disponível; a terceira é implementar mecanismos de incentivo para os pares que mais contribuem, sendo que quanto mais contribuírem, vão receber mais vídeo e com melhor qualidade; e a quarta é o fato dos pacotes serem de tamanho variável e não serem extraídos dos pares da rede de uma fora aleatória ou usando o algoritmo do mais raro primeiro (*rarest first*).

O protótipo usa um algoritmo de janela deslizante priorizada (PSW, do inglês *Priority Sliding Window*), que escolhe os *chunks* adequados que já estão armazenadas em cada par e reserva um tamanho de janela de *download* de *chunk* que prioriza a camada base do vídeo escalável. O tamanho de janela de *download* de *chunk* pode aumentar ou diminuir dependendo de várias medições feitas durante a reprodução do vídeo, por exemplo, largura de banda disponível, latência, carga da CPU, etc.

O sistema proposto em [69] usa um ou mais *trackers* que, como em todos os sistemas baseados no *BitTorrent*, armazenam informação sobre a rede e sobre o conteúdo além do conteúdo em si. Isto é, os *trackers* não enviam conteúdo para os usuários da rede. Quando um par entra na rede, ele tem que contatar um destes *trackers* para ser anunciado dentro da rede e para pedir informação sobre outros pares que estejam compartilhando o conteúdo desejado.

A estratégia de seleção dos pares é muito parecida à do *BitTorrent*, mas dá maior prioridade de seleção aos pares que estejam mais perto, do ponto de vista da localização geográfica, para minimizar custos de tráfego dos provedores de serviços de Internet (ISP), minimizar a latência, que é minimizada se os pedidos por pacotes são feitos a pares locais ou pares que estejam dentro da mesma rede do ISP, e para evitar a regulação dos ISPs que limitam a largura de banda para conexões P2P com pares externos. A efetividade do algoritmo é avaliada fazendo a comparação com o algoritmo de seleção de *chunks* padrão do *BitTorrent* do ponto de vista de tempos de *download* e *upload* e do atraso na distribuição de vídeo ao vivo.

A abordagem dos autores em [70] é um pouco diferente daquelas dos autores anteriores, pois, propõem organizar os pares em *clusters*, ou em “pequenos mundos”, como os autores chamam essa abordagem, onde todos os pares de um mesmo *cluster* estão conectados entre si formando uma rede tipo malha dentro do *cluster* e conectados a um superpar (um par com maior capacidade de largura de banda de *upload*) fora do

cluster.

A vantagem de organizar os pares em *clusters* é que cada par pode ser acessível desde qualquer outro par em um número pequeno de saltos, diminuindo o atraso fim-a-fim.

O objetivo é lidar com quatro aspectos importantes: minimizar o atraso entre a fonte e o par receptor para cumprir as restrições de tempo de aplicações em tempo real, ser capazes de lidar efetivamente com a dinamicidade dos pares que entram e saem da rede em qualquer momento, escalabilidade e adaptabilidade no sentido de poder receber um número grande de usuários e poder adaptar e melhorar o *overlay* constantemente baseado nas características dos usuários, e maximizar o uso da largura de banda para que cada par receba a taxa de *streaming* adequada com as suas capacidades independentemente da heterogeneidade da largura de banda.

Dentro da rede podem ser identificadas quatro entidades básicas: o servidor, o *tracker*, os superpares e os pares ordinários. Os superpares, inicialmente, são selecionados entre os pares ordinários pela sua capacidade de *upload*, sendo que são selecionados vários superpares para tratar com a questão de ter um ponto único de falha dentro da rede. Além disso, quando a reprodução do vídeo começa, a lista de superpares da rede é atualizada periodicamente, de acordo com as contribuições de cada par na entrega efetiva do conteúdo.

Os autores lidam com o problema das perdas e melhoram a QoS combinando os mecanismos de *pull* e *push* para seleção de *chunks* (*push-pull*) [71]. Normalmente, os pacotes são empurrados (no modo *push* do mecanismo) da fonte até o par receptor e quando o receptor percebe que estão faltando algumas partes do vídeo, ele entra no modo puxar (*pull*) do mecanismo para puxar da fonte os pacotes que faltam das camadas do vídeo para reconstruir a sequência original.

Os pacotes perdidos são facilmente reconhecidos usando o número de sequência dos pacotes. A heterogeneidade dos pares é atacada usando a codificação de vídeo escalável, e é o usuário quem seleciona a qualidade de vídeo que quer receber, se com alta escalabilidade SNR, com que *frame rate* ou com escalabilidade espacial. Quer dizer, os autores de [72] não implementaram um esquema adaptativo como tal, mas usam a estrutura da codificação de vídeo escalável para melhorar a qualidade percebida pelo usuário.

A efetividade do sistema é medida em termos da razão de perdas de pacotes, do *jitter* e do número de saltos que um pacote faz, no caminho desde a origem até o destino

e comparando os resultados com os obtidos usando *CoolStreaming* [24, 31].

Em [72], os autores exploram o impacto e os *trade-offs* oferecidos pela adaptação de qualidade baseada no SVC com foco nos algoritmos de seleção de camadas SVC durante a transmissão do vídeo. O esquema adaptativo proposto contém dois blocos principais que decidem qual camada corresponde aos recursos disponíveis em cada par: o primeiro bloco é o da adaptação inicial da qualidade (IQA, do inglês *Initial Quality Adaptation*), que determina a maior camada possível que um par pode obter e reproduzir, e o segundo bloco é o da adaptação progressiva da qualidade (PQA, do inglês *Progressive Quality Adaptation*), que periodicamente ajusta a camada em função das mudanças no ambiente da rede, representando a parte dinâmica do esquema. Quando a reprodução do vídeo é iniciada, o bloco IQA é chamado para decidir qual é o nível de qualidade factível baseado nos recursos locais, isto é IQA compara os requisitos de cada camada do *streaming* de vídeo com os recursos locais de um par para descartar as camadas que não são suportadas no processo de decisão. Com base nesta decisão, é feita a seleção dos vizinhos e dos pacotes.

Os vizinhos têm que ser capazes de prover a camada de vídeo selecionada e, depois de contatar os vizinhos, são reservados os espaços para fazer o *upload* e é feita a seleção dos pacotes. O bloco IQA leva em conta três aspectos importantes, derivados da estrutura da codificação, que definem os recursos locais que são relevantes na decisão da qualidade inicial.

O primeiro é o tamanho da tela do dispositivo do usuário, que determina a resolução do vídeo a ser recebido. O segundo é a largura de banda de *download*, que determina o valor máximo do *bitrate* do *streaming* de vídeo que o par pode receber. E o terceiro recurso é a potência de processamento, pois para decodificar um *streaming* de vídeo SVC é necessária mais potência de processamento que para decodificar um *streaming* de vídeo não escalável, ou seja, quanto maior for a potência de processamento, maior o número de camadas que podem ser decodificadas.

Para garantir uma reprodução contínua do vídeo, o bloco PQA é executado periodicamente, para aumentar ou diminuir o número de camadas, dependendo das condições da rede. Como a resolução da tela do usuário não muda durante a reprodução do vídeo, o PQA só adapta a escalabilidade temporal e a escalabilidade SNR. As informações da rede são obtidas em tempo real medindo a disponibilidade dos pacotes na vizinhança e a vazão ativa de *donwload*.

A disponibilidade dos pacotes é medida para saber se os vizinhos têm

capacidade de fornecer a camada atual com os pacotes que têm armazenados nos seus *buffers* e todos os blocos de alta prioridade necessários podem ser obtidos dos vizinhos atuais sem ter que estabelecer novas conexões. Se isto não puder ser garantido, a camada SVC é diminuída para evitar degradação no desempenho até que novos pares tenham sido contatados. A adaptação do *bitrate* é feita usando o estado do *buffer*, um *buffer* cheio mais do que 80%, indica uma alta vazão e se a camada atual é menor do que a determinada pelo IQA o par é alocado em uma camada superior, temporal ou de SNR, e um *buffer* vazio, menos do que 10%, indica baixa vazão e o par é alocado em uma camada inferior, temporal ou espacial.

A adaptação da complexidade depende da carga do processador do dispositivo e é aumentada ou diminuída segundo o caso. As métricas usadas para a avaliação do desempenho do sistema são divididas em dois tipos: métricas de qualidade de sessão e métricas de qualidade SVC. As métricas de qualidade de sessão consideram o atraso para iniciar a reprodução do vídeo, eventos de congelamento da tela, duração desses eventos de congelamento da tela e o atraso relativo de reprodução por par. As métricas de qualidade SVC incluem o número de mudanças de camada durante a reprodução do vídeo e as camadas recebidas por cada par comparadas com a camada inicialmente calculada pelo módulo IQA.

Em [73], os autores modificam o protocolo do *BitTorrent* para habilitar a opção de *streaming* de vídeo escalável. Como nos casos anteriormente apresentados, que trabalham com o *BitTorrent*, existem o servidor, o *tracker* e os pares. O processo de entrada de um par na rede é o mesmo e a formação do *overlay* também.

Por restrições de tempo real os autores usam uma janela deslizante para representar o próximo conteúdo a ser reproduzido, isto faz com que somente partes que pertencem à janela deslizante sejam solicitadas aos vizinhos. Não implementam um esquema adaptativo, mas usando a estrutura de codificação modificam o protocolo para melhorar a qualidade percebida pelo usuário. O vídeo usado é codificado para ter duas resoluções espaciais, CIF e QCIF, e dois níveis de escalabilidade de qualidade. As simulações são feitas usando o NS-2, comparando o vídeo SVC com o mesmo vídeo codificado com o padrão H.264/AVC (não escalável) e os resultados são apresentados em função das unidades NAL não recebidas.

Por fim, cabe observar alguns aspectos ligados a implementações de sistemas reais, onde se verifica a ausência de codificadores SVC. Os parâmetros do codificador (por exemplo, o padrão de codificação, o *bitrate*, a estrutura do GOP, CBR/VBR, etc.)

são definidos levando em conta o método de transmissão e a largura de banda requerida, havendo uma grande liberdade para os desenhos específicos no mundo da Internet. O caso padrão para as aplicações é o uso de *codecs* proprietários (às vezes MPEG-4 – XviD, H.264, algumas vezes SMPTE VC-1 – Windows Media).

3.4 - CONCLUSÃO

Neste capítulo foram apresentados conceitos básicos sobre a codificação de vídeo escalável, bem como o padrão H.264/SVC e o codificador JSVM. Junto com a teoria para entender este tipo de codificação, também foram apresentadas várias propostas da literatura sobre a integração entre o vídeo escalável e as redes P2P.

4 – MÉTRICAS PARA AVALIAÇÃO DE QoV, QoS E QoE

Apesar dos avanços da tecnologia dos sistemas de comunicação e das técnicas de codificação, os dados de aplicações multimídia transmitidos, através de canais propensos a erros ou de fontes de codificação com perdas, podem sofrer distorções ou serem danificados no caminho desde o transmissor até o receptor [44], o que faz com que o conteúdo multimídia experimente uma degradação da qualidade percebida pelo usuário final. Para combater esses danos, é preciso medir esses erros e perdas usando indicadores de qualidade apropriados [45].

4.1 - MÉTRICAS DE QUALIDADE DE VÍDEO

Em geral, os métodos de avaliação da qualidade de vídeo (QoV) podem ser divididos em duas categorias: métodos de avaliação objetiva e métodos de avaliação subjetiva [44, 45]. Intuitivamente, podemos dizer que o melhor juiz da qualidade de um vídeo é o ser humano mesmo, por isso, a avaliação subjetiva reflete melhor a "verdadeira" qualidade percebida. Mas, a qualidade subjetiva usa métodos complexos, consome muito tempo, consequentemente é cara, às vezes não é prática ou não é factível, tem que ser feita com muito cuidado para obter resultados significativos, e não pode ser usada em ambientes que precisem de processamento em tempo real. A avaliação subjetiva pode ser feita com referência completa (*Full Reference*) ou Sem Referência (*No Reference*).

A avaliação objetiva não depende do ponto de vista de um humano, e sim de avaliar analiticamente o sinal do vídeo. Os métodos utilizados para avaliação objetiva possuem um algoritmo que mensura a qualidade da imagem, baseado na comparação automatizada da imagem de origem com a imagem processada (ou transmitida). Em algumas situações alguns métodos objetivos podem substituir com sucesso o uso de avaliação subjetiva. A avaliação objetiva pode considerar uma abordagem por "referência", e neste caso não é adequada para o monitoramento em tempo real porque precisa que o vídeo original (ou parte dele) e o vídeo recebido estejam disponíveis. E as avaliações sem referência não dependem da informação da origem, e os resultados obtidos podem ser confusos ou não conclusivos.

A avaliação objetiva pode ser dividida (ou feita) em três formas: *Full Reference*, *No Reference* e *Reduced Reference* [56]. Considerando que trabalharemos com avaliação objetiva, detalharemos a seguir essas 3 (três) formas.

Full Reference: Requer que o vídeo original (referência) e o vídeo recebido estejam disponíveis, e permite comparações subjetivas e objetivas detalhadas. Pode não ser muito prática para aplicações em tempo real, mas pode ser usada para extrair, avaliar, e comparar parâmetros de QoE e QoS em qualquer nível (de aplicação ou de rede) de uma forma *offline*.

Reduced Reference*:* O mesmo conjunto de parâmetros do vídeo original e do vídeo recebido é derivado e comparado. Por exemplo, a QoE no nível de aplicação pode ser obtida usando o HIQM (*Hybrid Image Quality Metric*) [44], e a QoS no nível da rede pode ser representada pelas variações da taxa de vazão e das perdas. Tais parâmetros, usualmente, têm as suas raízes nas pesquisas em *full reference* como um meio para resumir e interpretar os resultados.

No Reference: A informação da qualidade tem que ser extraída do vídeo recebido, porque o vídeo original não está disponível. Está é uma situação típica em cenários *online* onde o foco é exclusivamente a qualidade resultante como percebida pelo usuário final (avaliada através de observações e perguntas) ou pelo seu representativo (um algoritmo). No contexto de rede, métricas *no reference* usualmente carecem da possibilidade de discernir entre os problemas da qualidade derivados do vídeo de referência e das perturbações introduzidas pela rede.

Existem muitas formas de avaliar objetivamente a qualidade de um vídeo, mas nenhuma é perfeitamente precisa e adequada para aplicações em tempo real devido à complexidade da avaliação da qualidade. Habitualmente, isto tem sido feito com medidas como *Signal-to-Noise Ratio* (SNR), *Bit Error Rate* (BER) ou *Peak Signal-to-Noise Ratio* (PSNR). Mas, já foi mostrado que essas medidas nem sempre correlacionam bem com a qualidade percebida pelo usuário final. Assim, se usarmos uma dessas formas de medir a qualidade, é bom conferir os resultados usando outro método de avaliação de qualidade [45].

No nosso caso, a avaliação da qualidade vai ser feita usando o PSNR, e vamos reforçar os resultados usando o *Structure Similiarity Index Metric* (SSIM). O PSNR e o SSIM serão descritos a seguir.

4.1.1 - Peak Signal-to-Noise Ratio - PSNR

Em uma imagem, a cor de cada pixel pode ser vista como um ponto em um espaço tridimensional (chamado espaço de cores). Dessa forma, a cor pode ser descrita como uma combinação de três componentes (por exemplo, as componentes vermelha, verde, e

azul, se usado o espaço de cores RGB, ou componentes de luminância, de crominância vermelha, e de crominância azul, se usado o espaço de cores YCrCb ou YUV). Cada um desses componentes pode ser codificado em Nb *bits*. Geralmente, o PSNR é calculado sobre um componente só, então pode haver diversas formas do PSNR (por exemplo, o Y-PSNR é o PSNR calculado para o componente de luminância se usado o espaço de cores YUV). O PSNR é derivado do *Mean Squared Error* (MSE) em relação ao máximo valor possível de luminância [47].

Em geral, dado um quadro do vídeo de referência (vídeo original) e do vídeo codificado/reconstruído (vídeo recebido) com uma resolução de *mxn* pixels, para qualquer componente Y, o Y-PSNR é calculado segundo a equação 4.1, sendo o MSE dado pela equação 4.2 [47].

$$Y - PSNR[dB] = 10\log_{10}\left(\frac{L^2}{MSE}\right) \qquad (4.1)$$

$$MSE = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\|X(i,j) - Y(i,j)\|^2 \qquad (4.2)$$

Onde *X(i,j)* é o valor do componente Y para o pixel *(i,j)* no vídeo original, *Y*(*i,j*) é o valor do componente Y do pixel *(i,j)* no vídeo codificado/reconstruído; *L*= 2^{Nb}-1 (255, quando *Nb*=8), é o intervalo de valores para cada pixel na componente Y. O resultado do PSNR é um único número em decibéis.

No nosso trabalho, vamos focar no Y-PSNR por simplicidade, porque a componente da luminância é perceptivelmente dominante, e porque alguns testes preliminares mostraram que o PSNR calculado nas outras componentes (Cr e Cb) tem um comportamento similar ao Y-PSNR [48].

4.1.2 - Structure Similarity Index Metric (SSIM)

O SSIM é mais complexo que o PSNR [48]. A idéia do SSIM é representar melhor os erros que são perceptíveis e incômodos para os usuários. O índice não é uma diferença direta dos valores das componentes, mas é uma composição ponderada da diferença da luminância, a diferença do contraste e a diferença na estrutura (i.e., na organização geral) da imagem.

As imagens são altamente estruturadas. Por isto, para avaliar a qualidade precisamos medir, além das variações dos pixels, as distorções introduzidas na codificação ou na transmissão da imagem. O primeiro passo, para fazer isto, é distinguir as estruturas na cena. A luminância da superfície de um objeto é produto da iluminação e da reflectância, mas a sua estrutura é independente da iluminação. Portanto a informação estrutural de uma imagem é definida como o conjunto de atributos que formam a estrutura dos objetos representados na cena, apesar da luminância média e do contraste médio da imagem. Como estas duas características podem variar dentro da imagem, estas são consideradas localmente [49].

Sejam x e y dois sinais não negativos das imagens, os quais foram alinhados um com o outro (e.g., arranjos espaciais extraídos de cada imagem). Se considerarmos que um dos sinais tem qualidade perfeita, então a medida da similaridade pode servir como uma medida quantitativa da qualidade do outro sinal [49].

A tarefa de calcular o SSIM é dividida em três etapas de comparação: luminância, contraste e estrutura [49].

Primeiro, a luminância é comparada usando as equações (4.3), (4.4) e (4.5) [49].

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \qquad (4.3)$$

Onde

$$\mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad (4.4)$$

$$\mu_y = \frac{1}{n}\sum_{i=1}^{n} y_i \qquad (4.5)$$

μ_x e μ_y são os valores médios da luminância de cada uma das imagens, e,

$$C_1 = (K_1 L)^2 \qquad (4.6)$$

é uma constante incluída para evitar instabilidade quando $\mu_x^2 + \mu_y^2$ é muito próximo de zero, com $K_1 << 1$, e L é o intervalo de valores para os pixels (255 para uma imagem com uma escala de cinza de 8-*bits*).

A equação para comparar o contraste, (4.7) [49], tem a mesma forma que a equação para comparar a luminância:

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (4.7)$$

Mas, neste caso são usados os desvios padrões (σ_x e σ_y) como estimadores dos contrastes dos sinais, equações (4.8) e (4.9). E C_2 (5.10) [49] é uma constante com $K_2 << 1$.

$$\sigma_x = \left(\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)^2\right)^{\frac{1}{2}} \quad (4.8)$$

$$\sigma_y = \left(\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \mu_y)^2\right)^{\frac{1}{2}} \quad (4.9)$$

$$C_2 = (K_2 L)^2 \quad (4.10)$$

Depois de ter as comparações de luminância e contraste, se procede a fazer a comparação de estrutura. Para fazer essa comparação é usada a equação (4.11).

$$s(x,y) = \frac{2\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \quad (4.11)$$

E σ_{xy} pode ser calculado com a equação (4.12):

$$\sigma_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y) \quad (4.12)$$

Finalmente, comparando as três equações de comparação (4.3), (4.7), e (4.11) temos o SSIM, equação (4.13) [49]:

$$SSIM(x,y) = [l(x,y)]^{\alpha}[c(x,y)]^{\beta}[s(x,y)]^{\gamma} \quad (4.13)$$

Onde, $\alpha > 0$, $\beta > 0$ e $\gamma > 0$ são parâmetros para ajustar a importância relativa dos três componentes; tipicamente, $\alpha=\beta=\gamma=1$, e $C_3 = \frac{C_2}{2}$. Assim, o SSIM pode ser calculado usando a equação (4.14).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4.14)$$

O valor do SSIM vária entre -1 e 1, e, quanto mais perto de 1 melhor é a qualidade da imagem, mas degradações aceitáveis implicam um SSIM > 0.8 [48].
Na figura 4.1 é mostrado o funcionamento do SSIM.

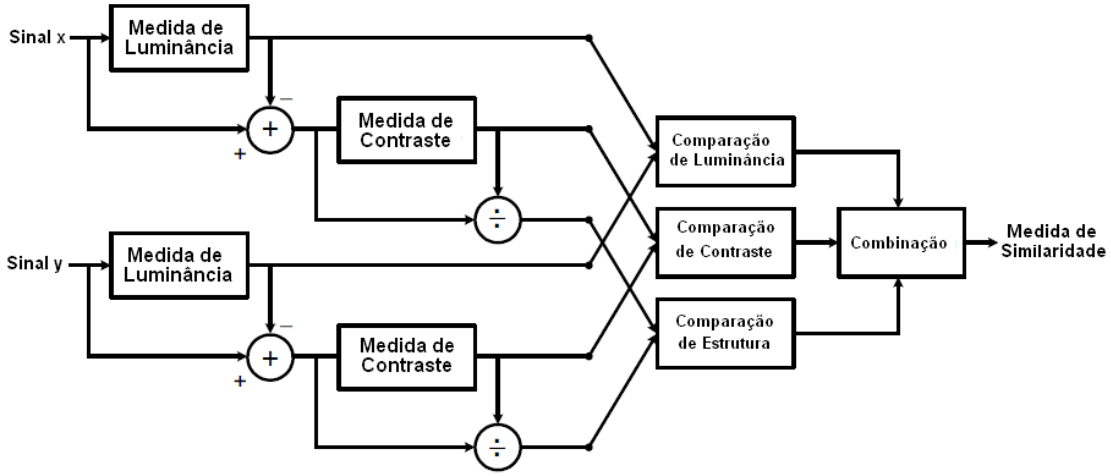

Figura 4.1: SSIM (Fonte: [49]).

4.2 - MÉTRICAS DE QoS

Durante o percurso de um pacote da fonte até o destinatário final podem ocorrer muitas coisas, levando em conta o fato de que um pacote passa por roteadores, switches, e por outros elementos presentes na rede [46]. Além do anterior, também existem as mudanças experimentadas na rede, devido às condições dos usuários que a estão usando, ao tráfego presente, as aplicações que estão rodando, etc.
A QoS é uma forma de avaliação objetiva. A QoS se baseia em parâmetros de rede que são mensuráveis, como o atraso, o *jitter*, a perda de pacotes e a variação da largura de banda. Esses parâmetros são as principais dificuldades enfrentadas na distribuição de vídeo em uma rede. Estes aspectos que influenciam a QoS são explicados a seguir.

4.2.1 - Atraso

Quando um pacote é transmitido, ao sair do sistema final (origem), para chegar a outro sistema final (destino), ele passa por uma série de roteadores, isso faz com que ele sofra diferentes tipos de atraso ao longo desse caminho. Dentre os atrasos sofridos tem-se o atraso de processamento nodal, o atraso de fila, o atraso de transmissão e o atraso de propagação; juntos, esses atrasos, ao longo do caminho, se acumulam para ter como resultado um atraso total.

O atraso de processamento está caracterizado pelo tempo requerido para a verificação do cabeçalho, e a determinação do direcionamento do pacote. Além disso,

também pode incluir outros fatores, como o tempo requerido para a verificação de erros de *bits* existentes no pacote que ocorreram durante a transmissão. Uma questão importante a ser considerada é que o atraso de processamento depende das capacidades de processamento presentes nos elementos do sistema.

O atraso de fila do pacote ocorrerá quando precisa esperar para ser transmitido no enlace. O atraso na fila dependerá da quantidade de outros pacotes que se encontram na fila esperando para serem transmitidos. Caso a fila esteja vazia e nenhum pacote esteja sendo transmitido naquele determinado momento, o atraso de fila será considerado zero. Mas, se o pacote encontra tráfego pesado (congestionamento na rede) e se há muitos pacotes esperando na fila para serem transmitidos, o atraso de fila será longo.

O atraso de transmissão dependerá do tamanho do pacote em *bits* e da velocidade na qual será transmitido esse pacote. Se o tamanho do pacote for dividido pela velocidade de transmissão tem-se a quantidade de tempo necessária para transmitir todos os *bits* do pacote para o enlace. Por exemplo, para um pacote de tamanho *l* e para um enlace com uma velocidade *r*, o atraso de transmissão será de *l/r*. Este será o tempo requerido para transmitir todos os *bits* do pacote para o enlace.

O atraso de propagação de um enlace é a distância entre dois roteadores dividida pela velocidade de propagação. Isto é, o atraso de propagação é *d*/*v*, onde *d* é a distância entre os dois pontos e *v* é a velocidade de transmissão do enlace. A velocidade de propagação do *bit* é igual à velocidade de propagação do enlace, que pode ser igual à velocidade da luz no caso da fibra ótica.

Assim, o atraso total poderia ser expresso como mostrado na equação (4.15) [46]:

$$A_{Total}=A_{Processamento}+A_{Fila}+A_{Transmissão}+A_{Propagação} \qquad (4.15)$$

Onde:

A_{Total} é atraso total, $A_{Processamento}$ é o atraso de processamento, A_{Fila}é o atraso de fila, $A_{Transmissão}$ é o atraso de transmissão, e $A_{Propagação}$ é o atraso de propagação.

4.2.2 - *Jitter*

O *jitter* pode ser definido como a flutuação do atraso entre pacotes, decorrente de atrasos da rede. O *jitter* é importante para as aplicações executadas em rede, cuja

operação adequada depende, de alguma forma, da garantia de que as informações, os pacotes de vídeo, sejam processadas em períodos de tempo bem definidos.

Na RFC 3393 da IETF [50] o *jitter* é definido como uma variação instantânea do atraso dos pacotes (IPDV - *Instantaneous Packet Delay Variation*). O IPDV de um par de pacotes, dentro de um *stream* de pacotes, é definido como a diferença do atraso dos dois pacotes, no tempo de chegada ao seu destino, ignorando qualquer pacote perdido.

Dada uma sequência de pacotes transmitidos nos tempos t(1), t(2), t(3),... t(n) e recebidos nos tempos t'(1), t'(2), t'(3),... t'(n). A sequência de atrasos é d(1), d(2), d(3),... d(n), onde $d(i) = t'(i) - t(i)$ e $d(i) \geq 0$. O IPDV, ou *jitter* como foi definido pela IETF, é a sequência d(2) - d(1), d(3) - d(2),... d(n) - d(n-1).

4.2.3 - Perda de pacotes

A perda de pacotes é um dos indicadores de qualidade na transmissão de pacotes em redes IP e tem influência na qualidade de serviço, podendo ocorrer pelo estouro de *buffers* em roteadores e *switches*, em decorrência de situações de congestionamento. Quando se tem a utilização dos protocolos UDP e RTP, esses pacotes não podem ser retransmitidos e a própria retransmissão não é tolerável em aplicações de vídeo em tempo real. Por outro lado, quando se usa o serviço de transferência confiável de dados do protocolo TCP, o qual fornece uma conexão confiável utilizando os seus algoritmos baseados em *timeout* e em retransmissão, uma perda simples de um pacote faz com que o TCP diminua a sua taxa de transmissão com alguma das suas técnicas para evitar congestionamentos ou de *slow start* [46]. Assim, com o TCP se teriam problemas para manter uma taxa alta de vazão de dados sobre um caminho com perdas, o que se veria refletido na QoS.

4.2.4 - Variação da largura de banda

Dependendo das condições da rede a largura de banda nos usuários finais pode variar. Além disso, tem que se considerar as outras aplicações, além da aplicação de *streaming* de vídeo, que também consomem largura de banda e estão rodando ao mesmo tempo. Além disso, em alguns algoritmos de escalonamento de pacotes e de seleção de vizinhos de redes P2P, a largura de banda disponível nos pares é utilizada para tomar decisões de enviar ou não um pacote. Para aceitar um vizinho, se um par lhe envia uma solicitação de vizinhança e tiver largura de banda disponível, ele pode aceitar essa solicitação. E para selecionar um par como vizinho, se um par tiver uma lista de vários pares candidatos para serem seus vizinhos, o par pode ordenar os pares de acordo com a

largura de banda disponível, e os pares que tiverem a maior largura de banda serão selecionados primeiro.

4.3 - AVALIAÇÃO DA QoE

Um desafio comum em qualquer rede para distribuir vídeo é a necessidade de garantir que o serviço forneça a qualidade mínima esperada pelos usuários finais. A qualidade de experiência (QoE) é o desempenho do sistema total considerando-se a perspectiva do usuário. A QoE é uma ferramenta importante no entendimento do ponto de vista dos usuários sobre os sistemas de distribuição de conteúdo [51, 52].

A QoE é basicamente uma medida subjetiva do desempenho no nível de serviço, do ponto de vista do usuário. Também é um indicador de quão bem o sistema está atingindo os seus objetivos. Para identificar fatores que desempenham um papel muito importante na QoE, alguns aspectos quantitativos específicos têm que ser considerados. Para serviços de distribuição de vídeo, o mais importante é a medida perceptual da qualidade do vídeo.

Podem existir muitos fatores que afetam a QoE, dependendo da fonte do fator, dentre eles temos:

- **Fatores do ambiente**: são exemplos de fatores de ambiente o ruído no ambiente, a iluminação do local, a resolução da tela (monitor ou TV) e as caixas de som usadas, as capacidades de computação do reprodutor, etc. Os fatores do ambiente são usualmente incontroláveis, e difíceis de medir em uma sessão de testes.
- **Fatores da fonte**: a fonte original do sinal do vídeo tem, obviamente, um forte impacto na qualidade global percebida. Por exemplo, o nível do som, o nível de luminância, e o movimento médio, têm um impacto na qualidade, especialmente, se juntados com outros fatores, como um *bitrate* de codificação muito baixo e/ou perdas de pacotes na rede.

  Os parâmetros da fonte do vídeo como a natureza da cena (por exemplo, quantidade de movimento, cor, contraste, tamanho da imagem, etc.), que dependem das características de uma sequência em particular que está sendo transmitida, podem ter também um impacto na percepção humana da qualidade do vídeo.

  Os parâmetros de compressão ou de codificação são os fatores mais importantes da fonte (MPEG-2, MPEG-4, H.264/AVC, H.264/SVC), incluindo a taxa de

amostragem, o número de bits por amostra, o *bitrate*, a taxa de quadros, os tipos de quadros codificados, o número de camadas, etc.

- **Fatores de distribuição** (**ou de rede**): a qualidade de serviço (QoS) medida na rede é um dos componentes mais importantes no projeto e manutenção, em geral. Tipicamente, incluem perda de pacotes, atraso, *jitter*, e fatores de largura de banda, mas não está claro como a maioria destes fatores afetam a qualidade global percebida, e, portanto, qual deles usar e como deve ser parametrizado. Além disso, há diferentes sensibilidades da qualidade com respeito a cada aplicação multimídia em particular.
- **Fatores do receptor**: além das técnicas de melhoramento da qualidade implementadas pelo transmissor (com a ciência do receptor), há um conjunto de procedimentos de melhora da qualidade os quais podem ser implementados somente no receptor.

Alguns exemplos são: *buffering*, encobrimento de perdas (inserção, interpolação e regeneração de dados perdidos), e melhoras no controle do congestionamento no caso de *streams* UDP.

Nas redes de distribuição de vídeo sobre redes P2P existem vários fatores essenciais que influenciam na qualidade percebida pelo usuário. Dentre eles temos [46]:

- Atraso de inicialização: esta métrica se baseia no tempo que o usuário tem que esperar antes de começar a reprodução do vídeo. Quanto menor for este tempo, melhor a qualidade da sessão.
- Atraso de troca de canal: quando se tem vários canais no sistema e o usuário quiser trocar de canal, o tempo que demora entre o usuário ordenar a troca de canal e começar a assistir o novo canal selecionado. Quanto menor o tempo, melhor para o usuário.
- Frequência da interrupção do serviço: Com que frequência o serviço de transmissão de vídeo é interrompido para o usuário.
- Qualidade da mídia: o objetivo é fornecer ao usuário a melhor qualidade da mídia, mas com um consumo razoável de largura de banda.
- Tamanho do *overlay*: nas redes P2P é importante observar o aproveitamento da largura de banda, a carga na fonte do vídeo, a qualidade da reprodução e a quantidade de sobrecarga de controle na rede, em função do tamanho no *overlay*.

- Número de mudanças de camada durante a reprodução: quando se trabalha com SVC, variações de camadas frequentes podem resultar em incômodo para os usuários que estão assistindo o vídeo com a menor qualidade. Por isto, é preciso medir o número de variações de camadas do SVC, como um indicador de qualidade do vídeo SVC. Um pequeno número de variações indica um melhor sistema de distribuição de vídeo.
- Camadas relativas recebidas: além do item anterior, o nível de camadas SVC recebidas por cada par durante a reprodução é importante. Devido a que cada par tem recursos locais diferentes e, por isto, somente pode receber certo intervalo de camadas SVC, não pode ser usado diretamente o número de camadas absolutas recebidas pelos pares para comparar o seu desempenho. Ao invés disso, o que se faz é calcular um número de camadas relativas recebidas para avaliar se um par está recebendo a maior qualidade que os seus recursos permitem.

Dependendo do tipo de rede sobre o qual vai ser distribuído o vídeo, os protocolos e os componentes dessa rede variam. Cada uma dessas redes tem os seus protocolos específicos, o que afetaria a QoE. Além dos componentes e dos protocolos, obviamente a qualidade do vídeo (se é analógico ou digital) afeta enormemente a eficiência na codificação e a QoE total.

4.3.1 - Mean Opinion Score (MOS)

Para pesquisas na área da QoE, o *Mean Opinion Score* (MOS) é amplamente escolhido como o resultado de testes subjetivos, e como referência para a modelagem de algoritmos de avaliação objetiva [53, 54].

A QoE pode ser avaliada com base em uma escala MOS (Mean Opinion Score), que envolve cinco valores possíveis, de acordo com a recomendação ITU-T P.800 [ITU800]: 5 = excelente, 4 = boa, 3 = razoável, 2 = pobre, 1 = ruim.

O MOS é a forma de avaliação de qualidade de experiência mais usada [51]. Mas precisa de condições especiais, tais como, uma sala escura, um ambiente calmo e equipamentos, e condições acústicas e visuais específicas. Além disso, as medidas demoram muito para serem feitas porque precisam de muitos sujeitos para que os resultados sejam válidos, do ponto de vista estatístico, isto somado ao tempo que leva a análise dos dados colhidos durante o processo de avaliação da qualidade [53].

4.4 - MAPEAMENTO DE QoS A QoE

Como visto anteriormente a avaliação da qualidade do vídeo pode ser feita de uma forma objetiva ou subjetiva. Métodos de teste subjetivos requerem uma perspectiva humana para avaliar a qualidade ou diferenças de qualidade entre duas imagens ou vídeos. Os métodos de teste objetivos são modelos matemáticos que estimam a qualidade do vídeo percebida por um usuário promédio.

Um dos objetivos quando se constrói uma aplicação sobre uma rede de comunicação é a satisfação do usuário final, o que é conhecido como um nível satisfatório de Qualidade de Experiência (QoE). Para uma aplicação multimídia o principal componente da QoE é a qualidade percebida, isto é, a qualidade do fluxo multimídia percebida assim que o fluxo é visto pelo usuário final, o que é claramente um conceito subjetivo.

A abordagem da QoE tem sido reconhecida como uma ferramenta importante para entender a percepção dos usuários de sistemas de distribuição de conteúdo, mas é muito pouco o que se conhece sobre a QoE dos usuários dos sistemas de TV P2P. As dificuldades por trás da avaliação subjetiva de sistemas P2P surgem pelo fato de que tais sistemas, geralmente, abrangem áreas geográficas muito extensas, envolvendo um grande número de pares. Levando isto em conta, o desempenho desses sistemas é consideravelmente afetado pelo comportamento imprevisível dos usuários e pelo estado das sub-redes que formam o sistema no total, fazendo com que sejam difíceis de modelar. Além disso, a avaliação subjetiva pode ser um processo caro e demorado, e o método não pode ser usado para monitorar a qualidade em tempo real [55].

A QoE liga a percepção, a experiência, e as expectativas de um usuário com respeito a uma aplicação e ao desempenho da rede, tipicamente expressos por parâmetros de QoS [56]. Relações quantitativas entre a QoE e a QoS são necessárias para poder construir mecanismos efetivos de controle sobre parâmetros mensuráveis de QoS.

Pesquisas sobre que parâmetros de rede são os mais sensíveis para o sistema, e como esses parâmetros afetam a QoE do usuário, têm sido feitas [55, 56, 57, 58, 59, 60].

As pesquisas feitas sobre a relação entre QoE e QoS têm mostrado que essa relação não é linear [57, 58, 59]. Em [56], Fiedler, Hossfeld e Tran-Gia propõem uma relação exponencial entre a QoE e a QoS. A sua hipótese é que existe uma interdependência exponencial entre a QoE e a QoS, e usando os dados dos resultados dos testes de MOS, encontram uma curva que se adapta melhor aos resultados, do que a

curva da relação logarítmica proposta em [57]. Esse trabalho está focado na avaliação da qualidade da voz dependendo da perda e do reordenamento dos pacotes, por separado (isto é, uma função exponencial para cada parâmetro), e a qualidade percebida da navegação web dependendo do tempo de sessão e da largura de banda, enquanto o nosso trabalho é sobre avaliação de qualidade de vídeo como vai ser descrito no capítulo 5. A relação proposta é mostrada na equação 4.16.

$$MOS = \alpha_1 e^{\alpha_2 x} + \alpha_3 \qquad (4.16)$$

Onde *x* pode ser a PER (*Packet Error Rate*) ou o *jitter*, no caso do serviço de VoIP, e a largura de banda ou o tempo ponderado de sessão (tempo que um usuário espera para receber os dados que pediu em uma página de Internet) no caso de navegação web. E os coeficientes α_i são calculados fazendo regressões não lineares.

Kim e Choi [58] fazem um estudo de um modelo de correlação entre a QoS e a QoE para a aferição da QoE de um serviço de IPTV. Eles consideram que a QoS é uma função de vários parâmetros de rede, tais como o atraso, o *jitter*, a perda de pacotes, o nível de *burst*, e a largura de banda. A QoE por sua vez é descrita como uma função exponencial da QoS, considerando nessa função a resolução da tela do usuário, o tipo de serviço assinado pelo usuário, e o tamanho do GOP do vídeo. O problema da abordagem desses autores é que eles definem que a QoS é diretamente proporcional às perdas, ao nível de *burst*, ao *jitter*, ao atraso, e ao mesmo tempo à largura de banda. Ou seja, quanto maior largura de banda, melhor QoS, mas quanto maiores as perdas, *jitter*, atraso e *burst* melhor QoS também. Outra questão a ser levada em conta é que, para os autores, quanto maior a QoS menor a QoE.

A QoS é definida como uma função das perdas (PER), do nível de *burst* (U), do *jitter* (J), do atraso dos pacotes (A) e da largura de banda (B), como mostrado na equação 4.17.

$$QoS(X) = K(\beta_1 PER + \beta_2 U + \beta_3 J + \beta_4 A + \beta_5 B) \qquad (4.17)$$

Onde o coeficiente β_i é o peso, definido pelo grau de importância relativa para a QoS de cada parâmetro no serviço de IPTV recomendados por organizações encarregadas de regular os padrões de qualidade (por exemplo, ITU-T, IETF, etc.), e a constante K é um fator determinante de QoS que depende da rede de acesso ao serviço de IPTV.

Definida a QoS, a correlação proposta entre a QoE e a QoS é dada pela equação 4.18.

$$MOS = Qr(1 - QoS(X))^{\frac{CQoS(X)}{R}} \qquad (4.18)$$

Onde *Qr* é um coeficiente que limita o valor do MOS de acordo com a resolução do terminal, QoS(X) foi definida na equação 4.17, *C* é um valor que depende do tipo de serviço assinado pelo usuário (por exemplo, premium) e R é determinado pela estrutura dos *frames* de acordo com o tamanho do GOP.

Em [51] os autores, fazendo um agrupamento (*clustering*) dos vídeos de acordo com o tipo de conteúdo (fundo, movimento dentro do vídeo, etc.), propõem um modelo de predição do MOS para o *streaming* de vídeo MPEG-4 sobre redes sem fio. As sequências de vídeo são classificadas e agrupadas em três categorias de acordo com uma análise das componentes principais (*Principal Component Analysis* – PCA) das características temporais e espaciais do vídeo: de movimento leve (*Slight Moviment – SM*), de movimento moderado (*Gentle Walk – GW*), e de movimento rápido (*Rapid Moviment – RM*). Depois de classificar e agrupar as sequências de vídeo, o passo seguinte é fazer uma regressão não linear para encontrar uma equação que se ajuste aos dados disponíveis da avaliação da qualidade das sequências de vídeo. Para gerar os dados, os autores combinaram diferentes taxas de *frames* na codificação com diferentes taxas de envio (no nível de aplicação) e com diferentes taxas de perdas (no nível de rede). Assim, eles lograram estimar o MOS levando em conta o conteúdo, parâmetros de rede e de aplicação. Os dados para a regressão foram gerados usando uma avaliação objetiva, neste caso o PSNR, e mapeando este PSNR em MOS.

A equação usada para predizer o MOS é mostrada a seguir.

$$MOS = \frac{\gamma_1 + \gamma_2 FR + \gamma_3 \ln SBR}{1 + \gamma_4 PER + \gamma_5 PER^2} \qquad (4.19)$$

Assim, para cada sequência de vídeo eles calcularam o PSNR, o mapearam em MOS (MOS obtido por mapeamento), e com uma regressão não-linear conseguiram uma equação para estimar o MOS (MOS predito) usando os parâmetros de *Frame Rate* (FR), *Sender Bitrate* (SBR) e *Packet Error Rate* (PER).

Em [60] os autores propõem um modelo para fazer *streaming* adaptativo de multimídia sobre IP baseados em métricas orientadas ao cliente. Definindo a QoE como uma relação causa-efeito entre métricas de rede e aspectos de qualidade tais como

distorções no vídeo ou no áudio ou qualquer parâmetro que possa afetar a percepção do usuário. Usando o método estatístico chamado de projeto de experimentos [61] (DOE, do inglês *Design of Experiments*) eles encontram uma relação causa-efeito entre respostas e fatores, derivada da modelagem empírica de dados experimentais. Seguindo o procedimento do DOE, a primeira etapa é selecionar os fatores relevantes que afetam a QoE. A segunda etapa é fazer uma análise mais profunda para estabelecer um modelo preciso. O foco principal é analisar o processo de transporte e decodificação do conteúdo digital, assim que for solicitado por um usuário médio, do ponto de vista de parâmetros controláveis (por exemplo, a taxa de bits do vídeo) e não controláveis (por exemplo, perdas, atraso e *jitter* dos pacotes).

A influência de cada fator na resposta é determinada por uma análise de Pareto [62], determinando depois a contribuição destes fatores na variabilidade do sistema fazendo uma análise ANOVA [62] (*Analysis of Variance*). Segundo os resultados, os fatores com maior influência na qualidade são a perda de pacotes, o atraso, o *jitter* e a taxa de codificação do vídeo, gerando a partir dos dados do experimento, e por meio de uma regressão, equações para a qualidade de experiência do vídeo ($QoE_{Vídeo}$) e a qualidade de experiência de áudio (QoE_{Audio}) é encontrada, equações 4.20 e 4.21:

$$QoE_{Video} = \delta_1 PER - \delta_2 A + \delta_3 J + \delta_4 CV + \delta_5 D^2 + \delta_6 D \cdot J + \delta_7 D \cdot CV \qquad (4.20)$$

$$QoE_{Audio} = \varphi_1 PER - \varphi_2 A + \varphi_3 J - \varphi_4 CV + \varphi_5 D^2 + \varphi_6 D \cdot J + \varphi_7 D \cdot CV \qquad (4.21)$$

Onde PER, J e A são as mesmas variáveis anteriormente definidas e CV é a taxa de codificação do vídeo.

Fazendo uma otimização multiobjetivo, para cada resposta da QoE_i (áudio e vídeo) uma função de desejo $d_i(QoE_i)$ determina um valor entre 0 e 1 para os possíveis valores da QoE_i, onde $d_i(QoE_i) = 0$ representa um valor totalmente não desejado, e $d_i(QoE_i) = 1$ representa um valor ideal na resposta. O valor D de desejo total é calculado usando uma média geométrica de cada um dos valores da função de desejo. Assim, D é calculado como mostrado na equação 4.22.

$$D = \{d_1(QoE_1) \cdot d_2(QoE_2) \dots d_n(QoE_n)\}^{\frac{1}{n}} \qquad (4.22)$$

O valor quantitativo D pode ser usado para fazer procedimentos de gerenciamento de serviços baseados em métricas de rede para controlar a qualidade

multimídia fornecida, modificando o valor da taxa de codificação do vídeo, um parâmetro controlável.

Assim, a QoE total pode ser definida usando a equação 4.23:

$$QoE = \sqrt{d_1(QoE_{Video}) \cdot d_2(QoE_{Audio})} \qquad (4.23)$$

Lembrando que a QoE está expressada como uma função desejo e o seu valor varia entre 0 e 1 neste caso específico.

Em [63], os autores quantificam os efeitos do *jitter* na qualidade perceptual do vídeo em uma rede 3G, fazendo uma modelagem da correlação entre a QoS e a QoE. Usando o algoritmo recomendado em [RFC 3393] para calcular o *jitter*, ou a variação do atraso dos pacotes (PDV) como é definida na recomendação. Observando o comportamento da avaliação feita pelo usuário com respeito à variação do *jitter*, os autores implementam uma ferramenta de estimação da QoE que mede parâmetros de rede no espaço do kernel, simultaneamente salvam as avaliações do usuário com respeito à percepção de um vídeo em andamento em tempo real em um telefone com o sistema operacional *Android*. Essa predição de QoE é usada para fazer o processo de seleção de rede. Com a informação obtida das redes disponíveis no ambiente de trabalho, o *framework* proposto em [60] toma decisões de *handover* para satisfazer a percepção do usuário final com respeito ao critério de "sempre estar conectado ao melhor" e ao de maximizar a QoE.

A equação 4.24 mostra a correlação entre a QoS e a QoE como definida pelos autores.

$$MOS = -\rho_1 J^{\rho_2} + \rho_3 \qquad (4.23)$$

Além do *jiter*, também foi estudado o efeito do *burst* sobre o MOS, chegando em uma equação semelhate àquela obtida anteriormente:

$$MOS = \mu_1 U^{-\mu_2} - \mu_3 \qquad (4.24)$$

4.5 - CONCLUSÃO

Neste capítulo foram apresentadas algumas métricas para avaliação de QoV, QoE e QoE que serão usadas como referências no desenvolvimento da nossa proposta. Também foram apresentados alguns fatores que afetam a aferição destas métricas.

Finalmente, foram apresentadas as abordagens, que consideramos mais importantes, para construir a nossa proposta para o mapeamento de QoS em QoE.

5 – DESCRIÇÃO DOS EXPERIMENTOS E PROPOSTA DE MAPEAMENTO QoS-QoE

O principal objetivo desta dissertação é propor uma relação quantitativa entre a QoS e a QoE na distribuição de vídeo escalável em redes P2P, bem como avaliar o impacto de parâmetros de rede sobre a qualidade de experiência (MOS). Para isto, foi preciso conhecer conceitos e características de codificação de vídeo escalável (capítulo 3) e redes P2P (capítulo 2), além de como fazer a avaliação da qualidade de serviço provido em um sistema de distribuição de vídeo deste tipo, e de como fazer o mapeamento dessa QoS em QoE (capítulo 4). Neste capítulo se apresenta o nosso trabalho propriamente dito, é aqui que propomos um mapeamento de QoS em QoE aplicável ao sistema considerado, bem como a influência de parâmetros de rede sobre o MOS. E também aqui avaliamos o desempenho da técnica de mapeamento proposta.

5.1 - DESCRIÇÃO DA SIMULAÇÃO E DA ARQUITETURA USADA

5.1.1 - O simulador P2PTVSim

A rede P2P foi simulada usando o simulador de redes de distribuição P2P descrito no Apêndice II, P2PTVSim [92]. Optamos pela simulação porque ainda não existe um cliente P2P de código aberto que dê suporte à distribuição de vídeo codificado no padrão H.264/SVC. Além disso, não foi possível ter a quantidade necessária de computadores (por exemplo, 1000) disponíveis para fazer os experimentos. O simulador é bem conhecido no meio das redes P2P e para o propósito do trabalho é adequado. Optamos por este simulador porque a maioria dos simuladores de redes P2P disponíveis é de propósito geral (dentro de redes). Quer dizer, são usados para redes P2P, mas só 2 deles são usados para simular *streaming* de vídeo sobre redes P2P, SSSim e P2PTVSim.

Para a avaliação da qualidade de vídeo obtida durante o desenvolvimento dos experimentos foram usadas as métricas de qualidade objetiva PSNR e SSIM, descritas no Capítulo 3.

5.1.2 - Metodologia de avaliação

A figura 5.1 apresenta a metodologia usada nos experimentos. A parte de cima da figura mostra como podemos chegar em valores de MOS obtido, onde são necessárias simulações de distribuição de vídeo sobre redes P2P e um mapeamento de PSNR em MOS (feito pelo evalvid). A parte de baixo da figura mostra como podemos chegar em

valores de MOS predito, onde, partindo de valores de SBR, PER e FR que são inseridos em uma equação usada como preditor podemos predizer o valor do MOS.

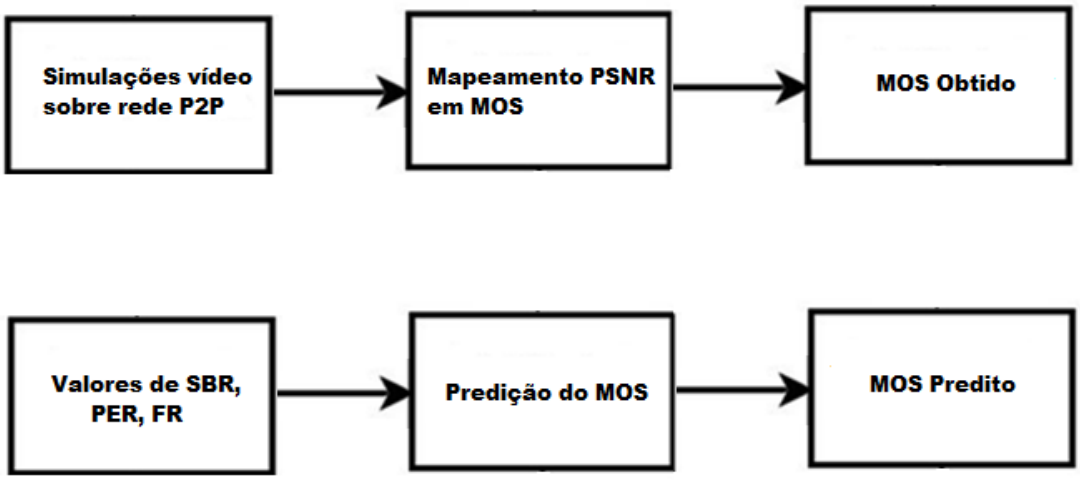

Figura 5.1: Metodologia do experimento.

5.1.3 - Arquitetura e experimentos para avaliação do MOS obtido

O diagrama de blocos mostrado na Figura 5.2 descreve como é feita a avaliação da qualidade no sistema de distribuição de VoD sobre redes P2P usado neste trabalho.

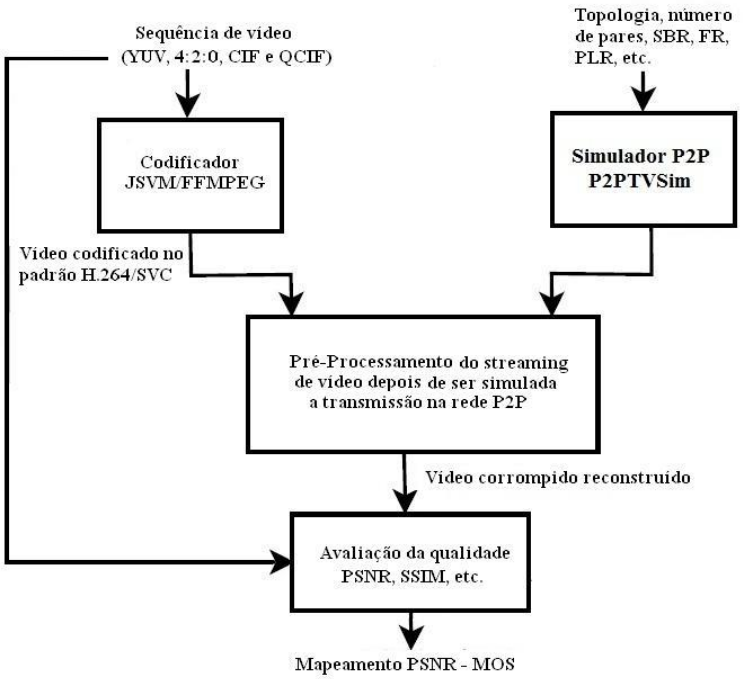

Figura 5.2: Diagrama de blocos do experimento (Adaptada de [48]).

5.1.3.1 - Primeiro Passo: Codificação e Classificação das Sequências de Vídeo

Com base na citada figura, o primeiro passo envolve codificar as sequências de vídeo, disponível nos formatos CIF e QCIF, com o *software* JSVM. Para esse fim, as sequências de vídeo foram baixadas da Internet no formato YUV, com uma razão de amostragem de 4:2:0, e uma resolução CIF (352x288). Em seguida, foram geradas as mesmas sequências de vídeo no formato QCIF (176x144) usando o *software* ffmpeg [74]. Essas sequências foram codificadas no padrão H.264/SVC usando os recursos de escalabilidade espacial, temporal e de qualidade do *software* JSVM, descrito no Capítulo 3, e disponível para descarga gratuitamente na Internet[3].

A codificação foi feita usando-se os três tipos de escalabilidade (espacial, temporal, e de qualidade), e uma escalabilidade de granularidade média (MGS). A sequência codificada contém: duas camadas espaciais, resoluções CIF e QCIF. Cinco camadas temporais, taxas de quadros de 1.875, 3.75, 7.5, 15 e 30 fps (*frames* por segundo). Três camadas de qualidade SNR (Q0, Q1 e Q2), e um GOP de 16. Combinando as diferentes escalabilidades, o vídeo codificado no padrão H.264/SVC, neste caso particular, pode ter até trinta camadas, sendo uma camada base e vinte e nove camadas de reforço. Um dos arquivos de configuração usado nos experimentos deste trabalho para fazer a codificação de vídeo escalável no JSVM pode ser visto no Apêndice I.

No arquivo de configuração podem ser vistos aspectos importantes, tais como o nome do arquivo de saída, a taxa de *frames*, o número de *frames* a serem codificados (que variam de acordo com a sequência usada), o tipo de granularidade usada, etc.

Além do arquivo de configuração geral, o JSVM precisa dos arquivos de configuração para cada camada, onde se define aspectos da resolução do vídeo, QCIF ou CIF neste caso, e os valores dos parâmetros de quantização. Um exemplo de arquivo de configuração de camada pode ser visto também no Apêndice I.

Depois da codificação com o JSVM, no arquivo do *BitStream* de cada vídeo podem ser vistas as diferentes camadas do vídeo escalável com as suas respectivas características, além da taxa de bits necessária para transmitir cada uma delas (*bitrate*). E, na coluna DTQ estão especificados os tipos de escalabilidade (D = espacial, T = temporal, e Q = qualidade):

[3] https://github.com/kierank/jsvm

Os vídeos também foram codificados usando o ffmpeg [76], para efeitos de comparação entre os padrões H.264/AVC e H.264/SVC para verificar que o vídeo escalável tem um melhor desempenho que o vídeo não escalável como mostraremos na Seção 5.2.1.

No desenvolvimento do trabalho, assumimos que o vídeo está totalmente codificado e disponível na fonte antes de ser enviado, pois, o trabalho é sobre distribuição de VoD, e não distribuição de vídeo ao vivo. Além disso, o codificador de vídeo escalável demora muito tempo no processo de codificação do conteúdo, o que faz com que a codificação ao vivo (em tempo real) não seja possível ainda. Assume-se também que o par já fez o processo de pedido do conteúdo e já está recebendo o vídeo na etapa de captura de dados do sistema. No total, foram usadas doze sequências de vídeo para os testes, sequências estas que são conhecidas no contexto de avaliação da qualidade de vídeo.

Esse conhecimento prévio permitiu que os vídeos fôssem classificados e agrupados em *clusters* pelas suas características temporais e espaciais, de acordo com o modelo de [51] em: sequências de movimento leve (*Slight Moviment – SM*), de movimento moderado (*Gentle Walk – GW*), e de movimento rápido (*Rapid Moviment – RM*). Na tabela 5.1 são descritos os vídeos usados neste trabalho e suas características.

Tabela 5.1:Características das sequências de vídeo.

Sequência	N° de *Frames*	Formato	Resolução	Grupo
Akiyo	300	YUV, 4:2:0	CIF, QCIF	SM
Bridge-Close	2000	YUV, 4:2:0	CIF, QCIF	SM
Carphone	382	YUV, 4:2:0	CIF, QCIF	GW
Coastguard	300	YUV, 4:2:0	CIF, QCIF	RM
Football	90	YUV, 4:2:0	CIF, QCIF	RM
Foreman	300	YUV, 4:2:0	CIF, QCIF	GW
Grandma	870	YUV, 4:2:0	CIF, QCIF	SM
Rugby	120	YUV, 4:2:0	CIF, QCIF	GW
Stefan	90	YUV, 4:2:0	CIF, QCIF	RM
Suzie	150	YUV, 4:2:0	CIF, QCIF	SM
Table-Tennis	300	YUV, 4:2:0	CIF, QCIF	GW
Tempete	260	YUV, 4:2:0	CIF, QCIF	RM

Os vídeos do grupo de movimento leve são sequências de vídeo com pequenas regiões de interesse, em movimento, sobre um fundo estático. Na figura 5.3 são mostrados *frames* deste tipo de vídeos (Akiyo, Bridge-Close, Grandma e Suzie).

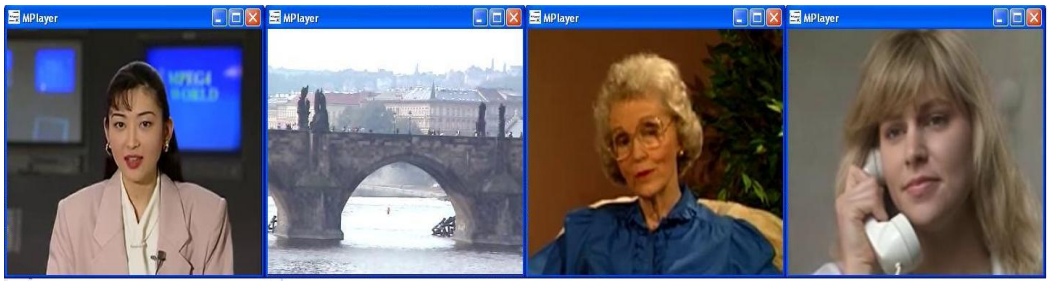

Figura 5.3: Vídeos da categoria SM.

Os vídeos do grupo de movimento moderado são sequências de vídeo de ângulo amplo, com movimento no conteúdo e no fundo. Na figura 5.4 são mostrados *frames* deste tipo de vídeos (Carphone, Foreman, Rugby e Stefan).

Figura 5.4: Vídeos da categoria GW.

Os vídeos do grupo de movimento rápido são sequências de vídeo com movimentos muito rápidos dos objetos de interesse. Na figura 5.5 são mostrados *frames* deste tipo de vídeos (Coastguard, Football, Tempete e Tennis-Table).

Figura 5.5: Vídeos da categoria RM.

O processo de classificação das sequências de vídeo pode ser automatizado, a partir da extração de características presentes nas sequências e do agrupamento (“clustering”) de sequências com características semelhantes. Nesta dissertaçao, a discussão e o detalhamento desse processo de classificação é indicado como trabalho futuro.

5.1.3.2. - Segundo passo: Simulação da Distribuição de Vídeo em Rede P2P

O passo seguinte é simular a distribuição do vídeo sobre uma rede P2P usando o simulador P2PTVSim. Este simulador precisa de um arquivo de configuração onde são especificados vários parâmetros da rede (por exemplo, o número de pares, o número e tamanho dos *chunks*, a probabilidade de perdas, o *jitter*, a largura de banda de *download* e de *upload* dos pares, a largura de banda de *upload* da fonte, o *bitrate* do vídeo, o escalonador de pacotes, etc.). Os parâmetros importantes e os valores usados neste trabalho são explicados a seguir.

O primeiro parâmetro é o número de pares, NumPeers, que foi fixado em 1000 para pensar em larga escala e ter um número considerável de pares dentro da rede. Junto com o número de pares tem que ser considerado o segundo parâmetro do tipo do *overlay*, que foi pensado para ser uma rede baseada em malha gerada de forma aleatória, com vinte vizinhos para cada par tendo as opções FULL, GNR, GNP, TREE e PREBUILT. no caso foi selecionado GNR porque com um determinado número de vizinhos é suficiente para garantir a qualidade ao par [84], quer dizer a rede é um grafo n-randômico de grau 20. O grau foi fixado em 20 porque, por causa do escalonador usado (LUc), um número menor do que 11 pode gerar mais perdas das esperadas [84]. Ou seja, se queremos "controlar" as perdas (PER) durante a simulação, com o parâmetro ProbChunkError, perdas adicionais fazem com que o valor seja maior do valor de PER que foi fixado. Assim, para ter uma folga e evitar esta situação, este número foi fixado em 20. Este valor pode ser calculado como Log_2N+1 (sendo N o número de pares da rede).

O terceiro parâmetro importante é o do escalonador (LATEST_BA), onde o par seleciona o último *chunk* útil que tem na sua janela e o envia a um vizinho que ele seleciona de uma forma ciente da largura de banda e que, dadas as circunstâncias de largura de banda dos pares da rede (a mesma para todos os pares), tem o mesmo desempenho do LATEST_RANDOM, onde o par seleciona o último *chunk* útil que tem na sua janela e o envia a um vizinho que ele seleciona de uma forma aleatória. A outra opção do simulador, CONFIG_PEER, permite configurar os pares no sentido que possam escolher o que vai ser feito primeiro: selecionar o *chunk* e depois o par alvo ou selecionar o par e depois o *chunk* a ser enviado, o que melhor for para o par usando uma variedade de diferentes estratégias de seleção.

Outro parâmetro de interesse é o tamanho dos *chunks*, onde usamos um *chunk* = um GOP, para fazer com que o sistema seja ciente do formato do conteúdo. Usar o

tamanho de um GOP é uma abordagem que melhora a qualidade do vídeo [48], pois, pela estrutura do vídeo, se perder um quadro I os outros *frames* (P e B) que dependem dele não poderão ser decodificados corretamente. Se forem enviados em *chunks* separados, se perder o *chunk* que contem o quadro I, o outro *chunk* com os quadros P e B será enviado desnecessariamente porque se recebido, sem o quadro I, não será possível decodificá-los.

O último parâmetro é a largura de banda dos pares e da fonte, 1 Mbps de *upload* para os pares e a fonte, e a largura de banda de *download* sendo muito maior para garantir que o gargalo seja no *link* de *upload* e não no de *download* [48, 84], a largura de banda é definida usando o parâmetro *PeerBandwidths* como explicado no Apêndice II.

Configurando o simulador com os valores dos parâmetros acima mencionados, cada simulação foi repetida cem vezes (sobre diferentes topologias aleatórias). Uma média dos resultados obtidos é feita para ter certa validade estatística. Os valores para os parâmetros de configuração do simulador, e parte deste trabalho, se basearam em [48], mas o trabalho desses autores é sobre *streaming* ao vivo e sobre o padrão H.264/AVC, a versão não escalável do padrão H.264.

Observa-se que apesar do contido na figura 5.1 poder dar a entender que as 1ª e 2ª fases podem ocorrer em paralelo, na verdade o importante é que os dois primeiros passos sejam feitos antes do terceiro passo sem importar a ordem, estão no mesmo nível de abstração na arquitetura do sistema mesmo que, para efeitos práticos, tenham sido tomados em sequência para levar uma ordem de execução das tarefas dentro dos experimentos.

5.1.3.3. - Terceiro passo: Pré-processamento do streaming de vídeo

O terceiro passo é o pré-processamento, para preparar o vídeo para avaliar a qualidade dele, neste passo é gerada a sequência de vídeo corrompida, para ser processada durante a etapa de avaliação da qualidade.

Como resultado das simulações o P2PTVSim gera vários arquivos com estatísticas da rede. Um desses arquivos contém uma lista de pacotes (*chunks*) perdidos. Esse arquivo é usado para remover os pacotes perdidos do *streaming* codificado como primeira etapa de pré-processamento do sistema. A segunda etapa do pré-processamento consiste em substituir cada *frame* perdido pelo *frame* imediatamente anterior. Assim, o vídeo de referência e o vídeo "recebido" terão o mesmo número de *frames*, critério

importante na avaliação do PSNR e do SSIM. A última etapa do pré-processamento é decodificar o vídeo para o formato “original” do vídeo YUV, 4:2:0 com uma resolução CIF, que é a maior resolução disponível.

5.1.3.4 - Quarto passo: avaliação da qualidade

A avaliação da qualidade é feita comparando o vídeo original e o vídeo decodificado usando as ferramentas para cálculo do PSNR e do SSIM do *framework* Evalvid [34]. Os valores de FR, SBR e PER vão sendo modificados, e, para cada combinação dos diferentes valores dos parâmetros de rede e de aplicação uma avaliação é feita. As sequências de vídeo com as diferentes combinações de parâmetros são mostradas na Tabela 5.2.

Os parâmetros FR, SBR e PER foram selecionados porque, de acordo com as configurações do simulador, são os parâmetros mais influentes na qualidade percebida. Esta afirmação é feita porque a largura de banda configurada para os pares é suficiente para receber o vídeo sem problemas e, além disso, não flutua durante a transmissão do vídeo. Outros parâmetros importantes são a taxa de abandono dos pares (*churn rate*) e o *jitter*, mas uma vez formada a rede os pares não a abandonam. Isto é estabelecido no arquivo de configuração, e, no caso do *jitter*, um *chunk* depois de sua chegada no par tem que esperar um tempo para ser reproduzido (*PlayoutDelay*). Esse tempo é suficientemente grande para fazer com que o *jitter* não importe, fazendo com que a ordem de chegada dos pacotes também não importe. Portanto, as variáveis usadas na modelagem da predição de MOS vão ser FR, SBR e PER.

Tabela 5.2: Combinações dos parâmetros das sequências de vídeo.

Tipo	Sequência	FR (fps)	SBR (kbps)	PER
SM	Akiyo	1.875, 3.75, 7.5, 15, 30	48, 128, 192, 256, 512	0.01, 0.05, 0.1, 0.15, 0.2
	Bridge-Close			
	Grandma			
	Suzie			
GW	Carphone		96, 192, 320, 480, 576	
	Foreman			
	Rugby			
	Table-Tennis			
RM	Coastguard		128, 288, 512, 640, 1024, 1440	
	Football			
	Stefan			
	Tempete			

Os valores da perda de pacotes (PER) estão no intervalo 0.01 a 0.2 (1% até 20%) com incrementos de 0.05. Valores maiores do que 20% nas perdas dos pacotes geram uma qualidade do vídeo muito ruim e não há diferença entre o MOS desses valores maiores que 20% (por exemplo, não há diferença entre a qualidade de um vídeo distribuído em uma rede com perdas de 30%, e um vídeo distribuído em uma rede com perdas de 40%). O mapeamento para MOS de valores acima de 20% nas perdas é ruim para todos. Aproveitando a codificação escalável, e as configurações usadas para codificar as sequências de vídeo, podem ser usadas cinco taxas de *frames* (FR) 1.875, 3.75, 7.5, 15 e 30 fps. Usando o arquivo de *BitRate*, que pode ser gerado a partir do vídeo escalável com a ferramenta JSVM, foram escolhidas as diferentes taxas do vídeo (SBR).

Como não se têm valores de MOS subjetivos disponíveis, esses valores de MOS são obtidos objetivamente com a ferramenta de mapeamento de PSNR em MOS do Evalvid, o qual se baseia na Tabela 5.3.

Tabela 5.3: Mapeamento de PSNR a MOS obtido

MOS	PSNR (dB)
5 (excelente)	>37
4 (boa)	31 – 36.9
3 (razoável)	25 – 30.9
2 (pobre)	20 – 24.9
1 (ruim)	<19.9

5.1.4 - Comparação H.264/SVC x H.264/AVC relativa ao MOS obtido

As primeiras simulações foram feitas para verificar o desempenho do vídeo escalável sobre o vídeo não escalável e justificar a escolha do padrão H.264/SVC ao invés do padrão H.264/AVC, ou qualquer padrão não escalável anterior a esse.

Assim, a presente seção trata de uma comparação da qualidade objetiva, com base nos padrões citados e utilizando as métricas PSNR e SSIM.

Inicialmente, nas figuras 5.6 e 5.7 podem ser vistos os desempenhos dos padrões H.264/AVC e H.264/SVC em termos do PSNR, variando um parâmetro do simulador (o *PlayoutDelay*, ver apêndice I). Quanto maior o *PlayoutDelay*, melhores são os resultados em termos de qualidade, mas também se tem maior tempo de execução do simulador.

Nas figuras 5.8 e 5.9 podem ser vistos os desempenhos dos padrões H.264/AVC e H.264/SVC em termos do SSIM. Com as mesmas variações feitas para a aferição do PSNR, o SSIM confirma o que já tinha mostrado o PSNR: que o padrão H.264/SVC tem um melhor desempenho que o padrão H.264/AVC. Portanto, as simulações

executadas a partir desta parte do trabalho usarão somente vídeo escalável (H.264/SVC). Além disso, inicialmente optamos por selecionar um valor de *PlayoutDelay* de 18s, mas teve que ser reduzido a 14s porque incrementou o tempo de simulação consideravelmente, valor que oferece uma qualidade relativamente boa de acordo ao mapeamento de PSNR em MOS apresentado na Tabela 5.3 e os gráficos desta seção.

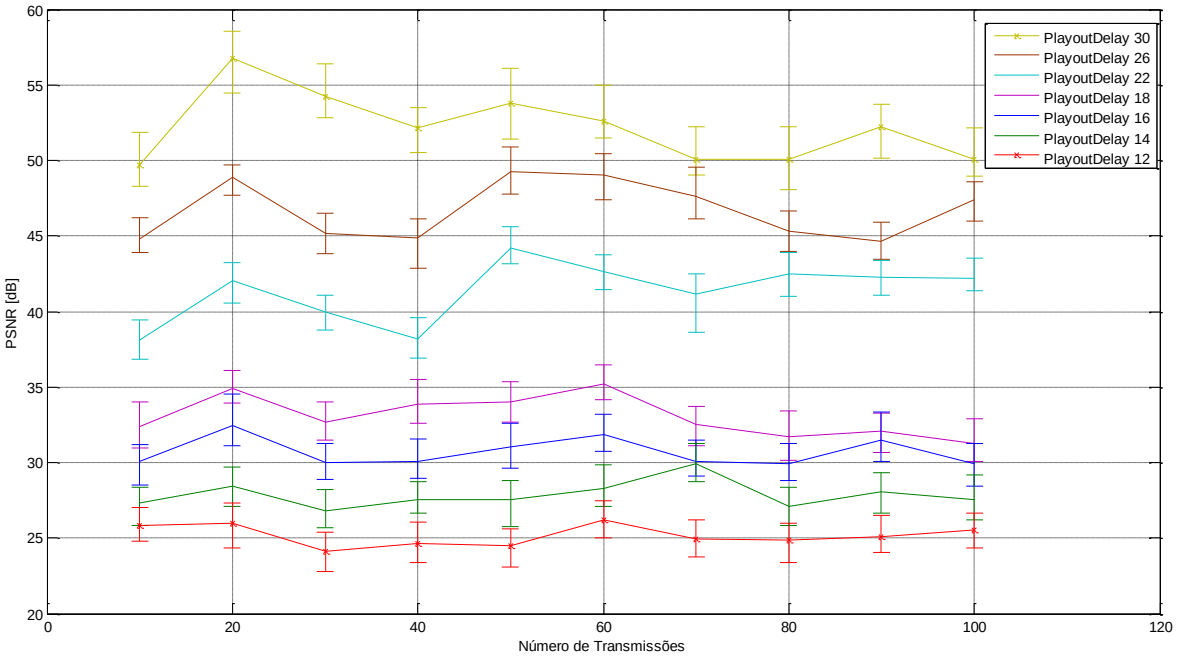

Figura 5.6: PSNR vídeo H.264/AVC

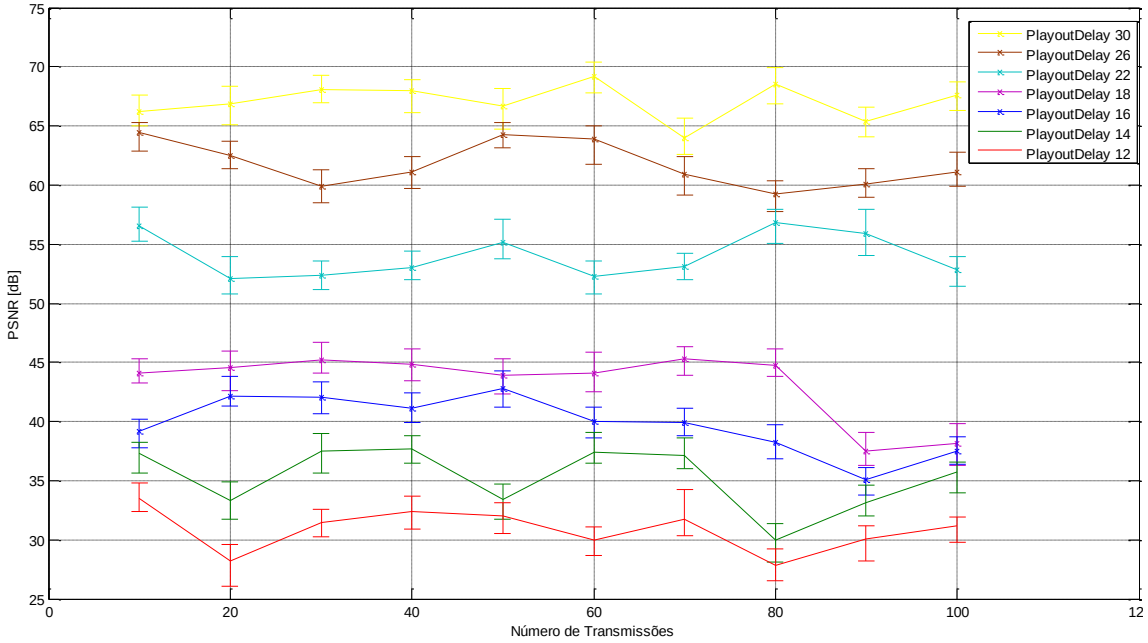

Figura 5.7: PSNR vídeo H.264/SVC

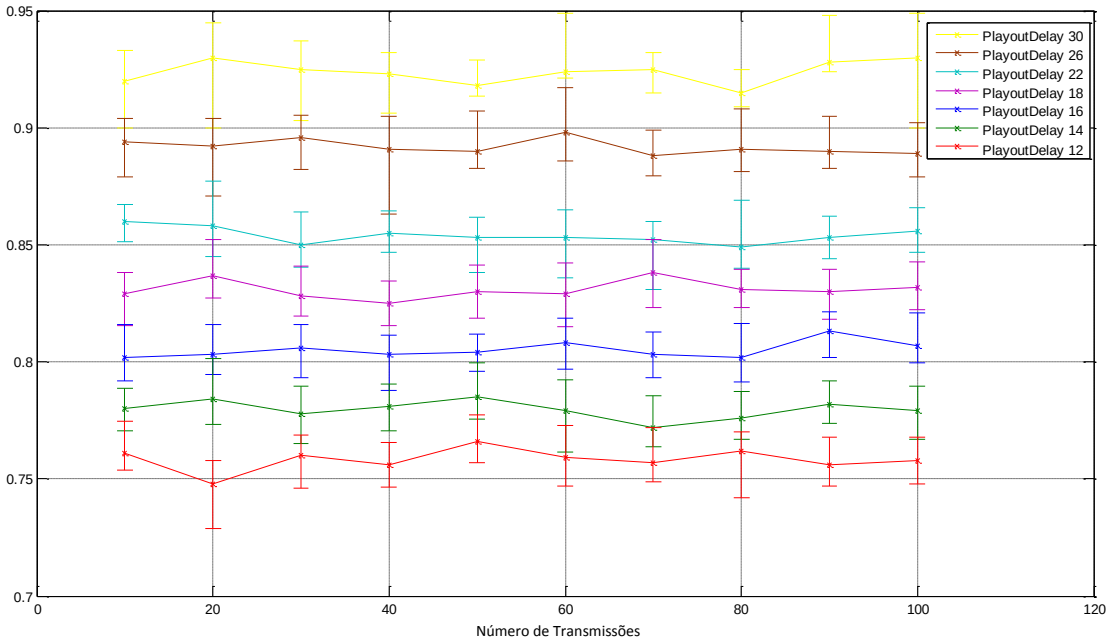

Figura 5.8: SSIM vídeo H.264/AVC

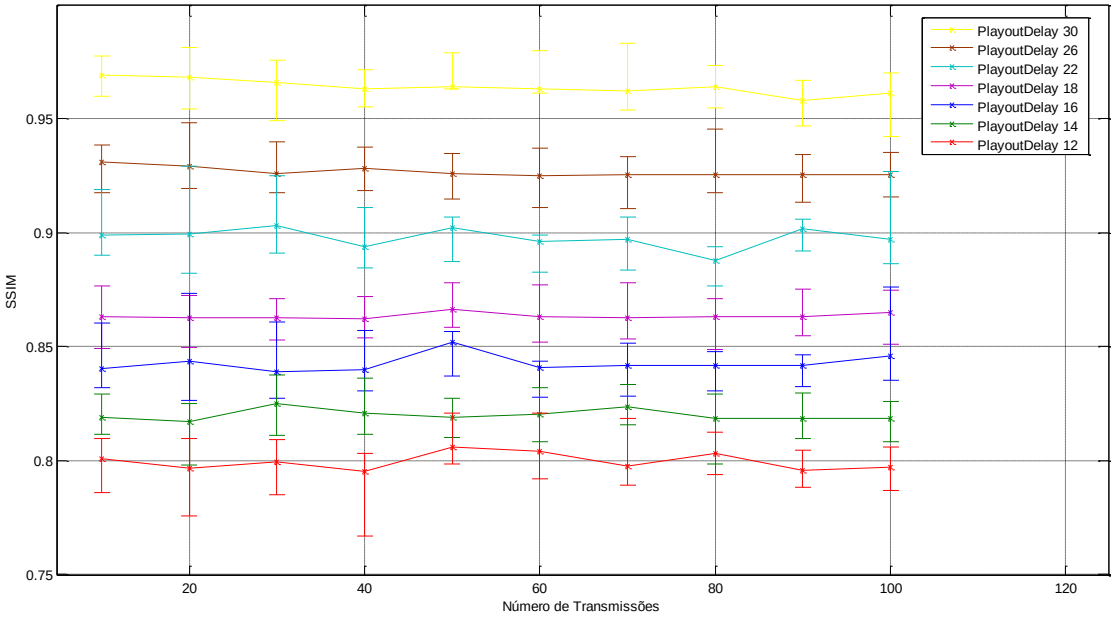

Figura 5.9: SSIM vídeo H.264/SVC

Os resultados mostram que, de fato, o H.264/SVC tem um melhor desempenho que o H.264/AVC tomando como referência de comparação os valores de PSNR e de SSIM atingidos por cada um dos padrões durante os experimentos.

5.2 - MAPEAMENTO DE QoS EM QoE PARA PREDIÇÃO DE MOS

5.2.1 - Metodologia

Para fazer o mapeamento de QoS em QoE propomos uma equação de baixa complexidade, mas de resultados muito bons, a qual serve como preditor de MOS.

Para chegar a uma proposta de equação, é preciso analisar o que acontece com cada um dos parâmetros em separado, fazer uma regressão não linear, porque já foi comprovado que essa relação entre QoS e QoE não é linear [56], ver a forma da equação obtida para cada parâmetro, e depois combinar as equações em uma só para assim obter um mapeador de QoS em QoE.

Com os dados de MOS gerados e partindo da proposta de [51], onde os fatores que mais influenciam a percepção da qualidade de vídeo são a taxa de *frames* (FR – *Frame Rate)*, a perda dos pacotes (PER – Packet Error Rate) e o *bitrate* do vídeo (SBR – *Sender BitRate*), é possível fazer uma regressão não linear usando as ferramentas *nlintool* e *polyfit* do *software* MATLAB™, e as diferentes ferramentas que ele oferece para esses casos, para chegar a uma equação que relacione o valor do MOS com os parâmetros mencionados, isto é, expressar o MOS como uma função do FR, a PER e o SBR, MOS= *f*(FR,SBR,PER).

O MATLAB usa o algoritmo de Levenberg-Marquardt descrito em [75] para fazer as regressões não lineares. Este algoritmo precisa de entrada X (a variável independente), y (resposta, variável independente), *fun* (função pré-definida pelo usuário que quer que seja ajustada) e um vetor *beta0* que contém um chute inicial dos valores dos coeficientes da função *fun*. Maiores informações sobre a ferramenta de regressão não linear do MATLAB podem ser encontradas em [76].

Os resultados das simulações e a equação derivada da regressão não linear feita com o MATLAB™, são apresentados na Seção 5.2.

5.2.2 - Predição de MOS e análise dos resultados

Nesta Seção serão apresentados os mapeamentos realizados com base nas simulações e nas regressões não lineares acima descritas.

Na figura 5.10 pode ser visto o comportamento do MOS com respeito às diferentes taxas de vídeo (SBR). Os valores das taxas de vídeo para as classes de vídeo (SM, GW, RM) foram selecionados a partir dos arquivos de *BitStream* de cada um dos vídeos. Cabe notar que, coincidencialmente, os vídeos escaláveis apresentam *bitrates* similares às dos vídeos na sua mesma categoria, em termos de *bitrates* necessários para

transmitir a camada base e as camadas de reforço. Além disso, a taxa de bits necessária para transmitir uma camada de uma sequência de vídeo do grupo SM é menor do que a dos vídeos dos grupos GW e RM, e a taxa de bits necessária para transmitir um vídeo do grupo GW é menor do que para um vídeo do grupo RM.

Na realidade deve existir um equilíbrio entre qualidade e SBR. Quanto maior o SBR melhor a qualidade, mas um valor de SBR muito alto pode levar ao aumento do tamanho dos pacotes e, portanto, ao aumento das perdas dos pacotes na rede.

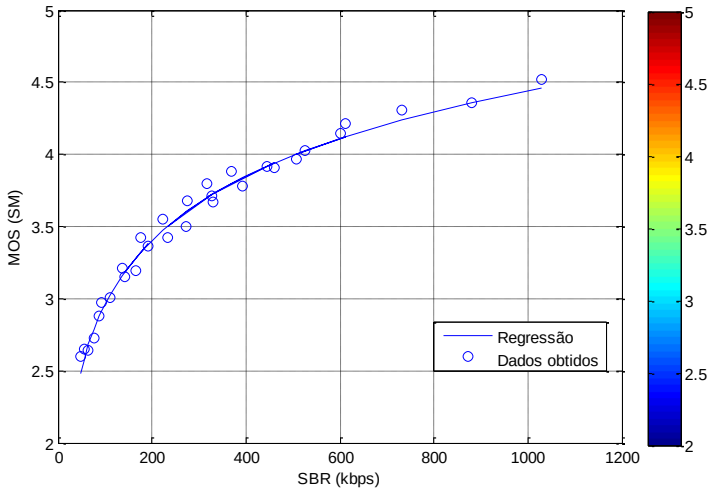

Figura 5.10: MOS em função da SBR.

As regressões mostraram que a equação para expressar o MOS em termos da SBR é da forma mostrada na equação 5.2:

$$MOS(SBR) = a_1 + b_1 Ln(SBR) \qquad (5.1)$$

Onde a_i e b_i são os coeficientes gerados pela regressão, e Ln é o logaritmo natural.

Na figura 5.11 são apresentados os resultados do MOS com respeito aos valores de FR.

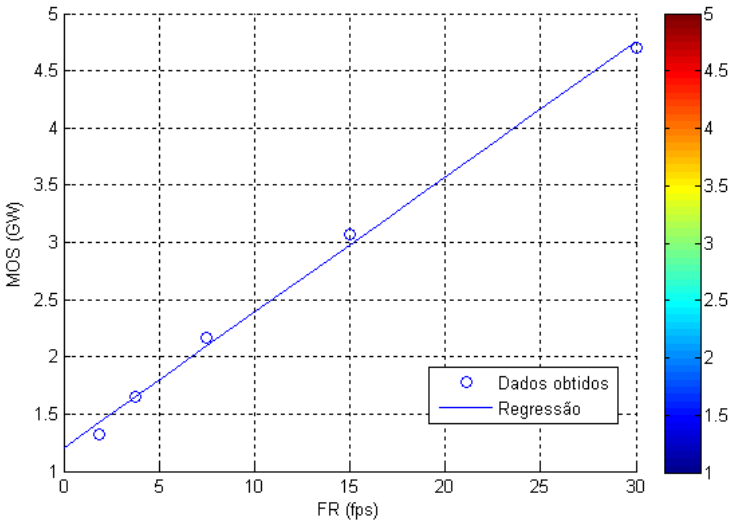

Figura 5.11: MOS em função da FR.

Os valores de FR são tomados do arquivo de configuração da codificação do vídeo escalável. Esses valores são os valores de fps disponíveis na qualidade temporal do vídeo. Da figura pode ser vista que uma linha reta pode modelar esta relação entre MOS e FR, relação mostrada na equação 5.3:

$$MOS(FR) = a_2 + b_2 FR \qquad (5.2)$$

Na figura 5.12 pode ser visto a variação do MOS em função da perda dos pacotes (PER).

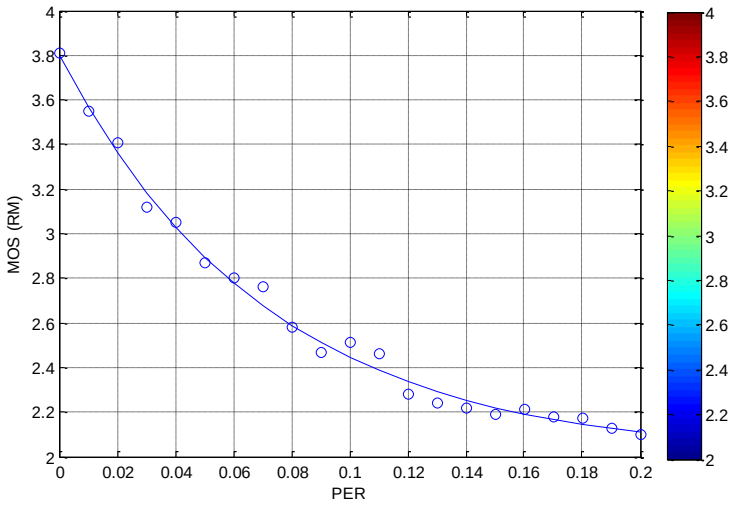

Figura 5.12: MOS em função da PER.

As regressões mostraram que a curva que se ajusta aos valores de MOS com respeito à perda dos pacotes é exponencial. A equação 5.4 mostra a forma da curva que melhor se ajusta aos valores de MOS:

$$MOS(PER) = a_3 e^{b_3 PER} \qquad (5.3)$$

Assim, de 5.2, 5.3 e 5.4 se propõe que a relação entre MOS e FR, SBR e PER seja da forma mostrada na equação 5.5:

$$MOS(FR, SBR, PER) = [a + bFR + cLn(SBR)]e^{dPER} \qquad (5.4)$$

A questão agora é calcular os coeficientes a, b, c e d da equação 5.4. Os valores desses coeficientes variam de acordo com o tipo de vídeo (SM, GW, RM), mas a equação continua sendo a mesma. Esses coeficientes obtidos são mostrados na Tabela 5.4.

Uma questão importante na análise é ver, usando o coeficiente de determinação (R^2) e a raíz do erro quadrático médio (RMSE, do inglês *root mean square error*), o quão bons são esses coeficientes e essa equação na hora de prognosticar o MOS (linhas R^2 e RSME da Tabela 5.4). É bom saber que, quanto mais perto esteja o valor de R^2 de 1, melhor se ajustam os coeficientes à predição, e valores de R^2 maiores do que 80% (0.8) representam um bom ajuste.

Tabela 5.4: Coeficientes obtidos e avaliação do desempenho da equação

Coeficiente	SM	GW	RM
a	3.10	2.89	2.37
b	-0.0017	-0.0012	-0.0011
c	0.133	0.1495	0.18
d	-1.86	-2.031	-2.55
R^2	93.25%	90.18%	87.46%
rmse	0.086	0.241	0.273

A figura 5.13 mostra o gráfico do MOS como função do FR e do SBR, isto é fixando o valor da PER em 0 (sem perdas na rede).

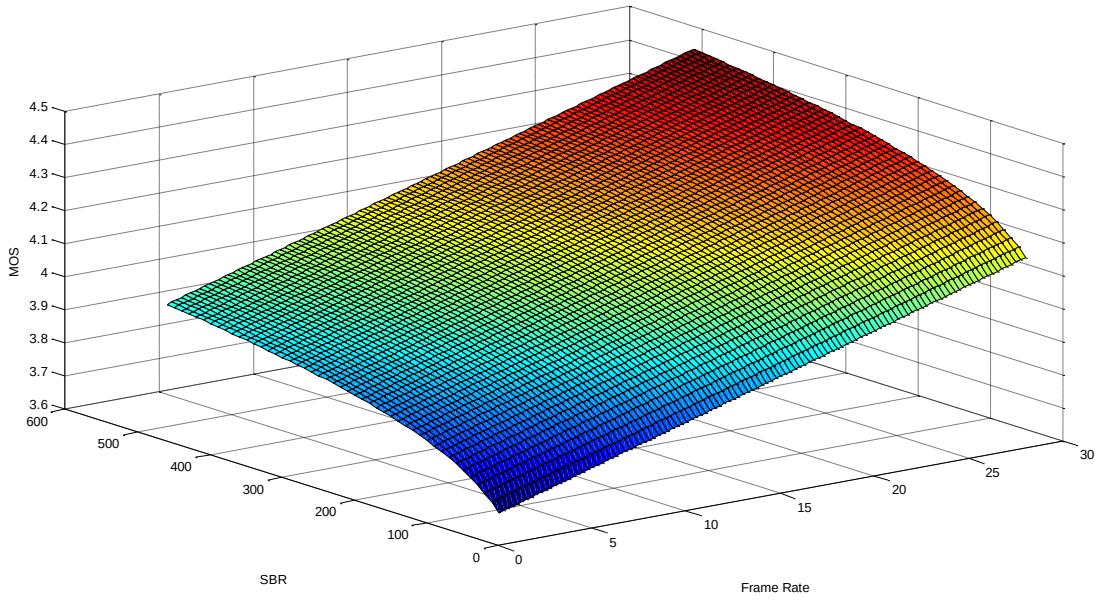

Figura 5.13: MOS como função do FR e do SBR (PER constante)

5.2.3 - Avaliação do impacto de parâmetros de rede sobre o MOS

Uma análise da variância foi feita para estudar o impacto sobre o MOS de cada um dos parâmetros (SBR, FR, PER) da equação de MOS descrita em 5.4. Na Tabela 5.5 são mostrados esses resultados. A análise é feita para cada parâmetro individualmente, para a combinação de dois parâmetros e, finalmente, para a combinação dos três parâmetros. A análise foi feita usando a ferramenta anovan do MATLAB, que faz o *n-way* ANOVA, no caso deste trabalho *three-way* ANOVA.

A coluna DF corresponde ao grau de liberdade (do inglês *Degree of Freedom*) que tem o modelo para cada fonte de variação do MOS; a coluna F-Value apresenta a estatística F (*F-Statistic*), que indica se o parâmetro tem uma influência significativa sobre o MOS; e a coluna P-Value é a coluna do valor p, o qual é derivado da função de distribuição acumulada (CDF, do inglês *Cumulative Distributed Function*) de F. Quanto menor o valor de p maior a influência do(s) parâmetro(s) no valor do MOS. Valores de p menores do que 0.05 (5%) significam que o parâmetro ou a combinação dos parâmetros influencia de uma forma relevante no resultado do MOS (em 95%).

A análise da variância mostra que o parâmetro mais influente no resultado do MOS é a SBR (apresenta o menor valor de p). Além disso, na combinação dos parâmetros tem que se ter cuidado ao combinar SBR e PER, pois deve existir um ponto de equilíbrio, um certo *trade-off* para um melhor desempenho.

Tabela 5.5: Resultados do three-way ANOVA

FONTE	DF	F-Value	P-Value
SBR	14	11.37	0.025
FR	14	4.15	0.067
PER	14	3.93	0.053
SBR*FR	196	6.01	0.062
SBR*PER	196	2.41	0.018
FR*PER	196	1.98	0.094
SBR*FR*PER	2744	5.28	0.013

Por causa dos valores limitados de taxa de *frames* (só cinco valores de FR usados no arquivo de configuração do codificador de vídeo), e como uma das análises foi o *three-way* ANOVA fatorial equilibrado, onde cada grupo tem o mesmo tamanho de amostra, os tamanhos de amostra dos outros parâmetros (SBR e PER) também foram limitados para facilitar a obtenção de resultados, pois no caso de desequilíbrio, o ajuste dos dados e os experimentos são mais complexos.

5.2.4 - Comparação entre valores obtidos e preditos de MOS

A Figura 5.14 mostra o gráfico dos valores obtidos para MOS usando a equação contra os valores obtidos fazendo a medição.

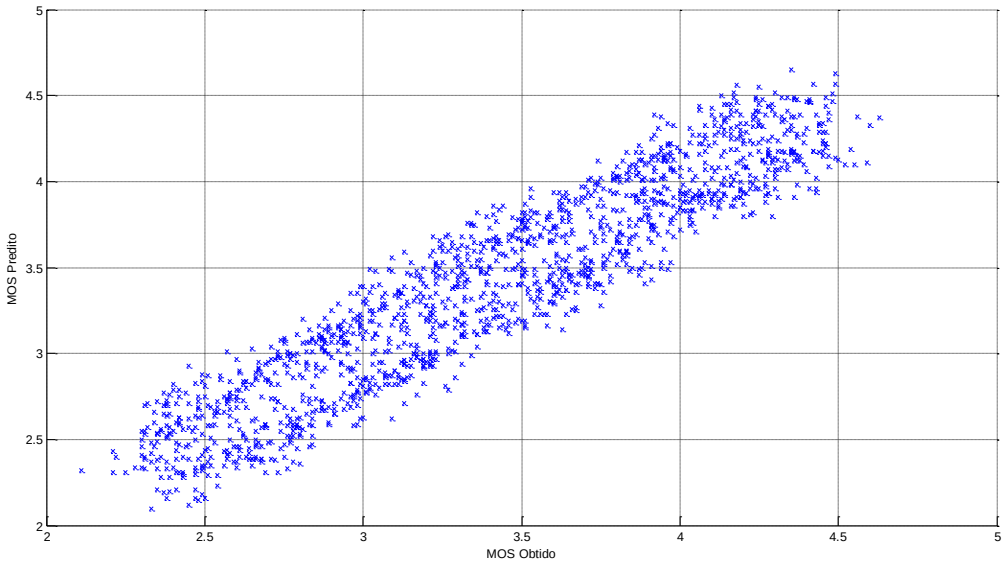

Figura 5.14: MOS predito vs MOS obtido.

A figura mostra que o desempenho do preditor de MOS é muito bom, esse desempenho é avaliado comparando os dados obtidos nas simulações e os dados preditos. Da análise podemos concluir que para o caso de SM a correlação foi melhor que para os outros dois casos (GW e RM). Outra coisa que foi observada foi o fato de que o tipo de vídeos RM apresenta uma maior degradação da qualidade respeito às variações dos valores dos parâmetros.

5.2.5 - Impacto de parâmetros de QoS sobre a QoV

A Figura 5.15 mostra o impacto dos três parâmetros de QoS (SBR, FR, PER) sobre a qualidade dos vídeos para os três tipos de conteúdo (SM, GW, RM). Os resultados com as maiores variações do MOS foram obtidas para os vídeos da categoria RM. O intervalo de variação do valor do MOS para os vídeos é mais ou menos o mesmo, mas com uma media diferente, o que mostra uma maior influência do parâmetro FR sobre o MOS do que o SBR e o PER para os vídeos de SM. No caso dos vídeos de GW, acontece quase a mesma coisa, mas a maior influência é do PER. Para os vídeos de RM, além de ter os piores resultados, quanto ao valor médio do MOS, tem-se também maiores intervalos de variação do valor do MOS, com o FR sendo o parâmetro que mais afeta o valor do MOS.

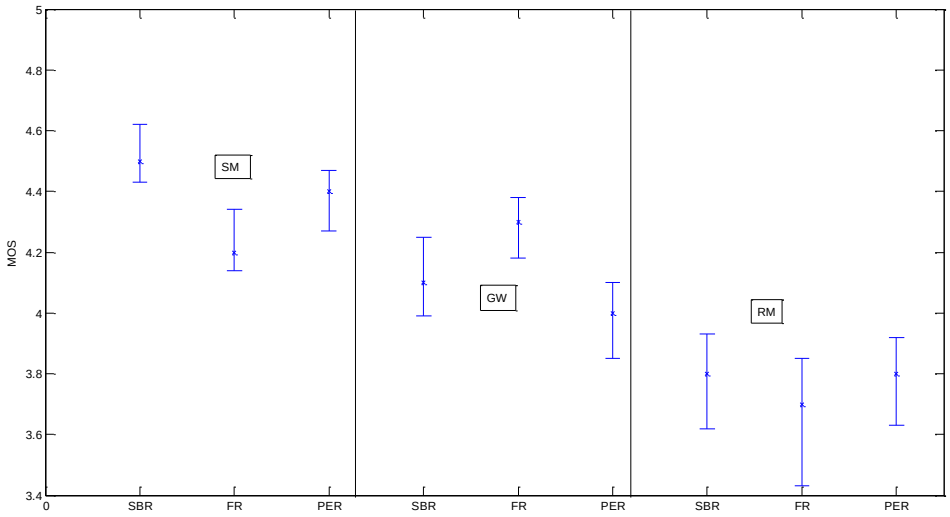

Figura 5.15: Efeitos significativos do SBR, FR e PER sobre o MOS.

Em termos comparativos, é possível afirmar que, com relação à proposta de [51], os resultados obtidos são melhores, tanto em valores de MOS quanto em valores de QoV medidos usando PSNR. Isto era o que se esperava, pois os experimentos daqueles autores foram feitos sobre uma arquitetura Cliente/Servidor e uma das vantagens de combinar SVC e P2P é, além de aliviar a carga no servidor com a rede P2P, melhorar a qualidade percebida pelos usuários usando o vídeo codificado em camadas.

5.3 - ESBOÇO DE PROPOSTA DE ESQUEMA ADAPTATIVO

É possível propor um esquema adaptativo baseado no preditor mostrado na equação 5.4. A equação 5.4 pode ser reescrita, para encontrar o valor do parâmetro SBR, como:

$$SBR = e^{\frac{MOSe^{-dPER}-a-bFR}{c}} \qquad (5.5)$$

Assim, com o valor do SBR é possível selecionar uma camada de vídeo específica usando o arquivo de *bitrate* gerado pelo codificador de vídeo escalável (H.264/SVC) JSVM, pelo fato de que cada uma das camadas tem um valor de *bitrate* associado. Isto pode ser visto no exemplo de arquivo de bitrate apresentado na Figura 5.16.

Contained Layers:
==========

Layer	Resolution	Framerate	Bitrate	MinBitrate	DTQ
0	176x144	1.8750	47.00	47.00	(0,0,0)
1	176x144	3.7500	54.80	54.80	(0,1,0)
2	176x144	7.5000	65.20	65.20	(0,2,0)
3	176x144	15.0000	76.10	76.10	(0,3,0)
4	176x144	30.0000	87.80	87.80	(0,4,0)
5	176x144	1.8750	92.50		(0,0,1)
6	176x144	3.7500	111.00		(0,1,1)
7	176x144	7.5000	136.80		(0,2,1)
8	176x144	15.0000	164.70		(0,3,1)
9	176x144	30.0000	192.50		(0,4,1)
10	176x144	1.8750	142.60		(0,0,2)
11	176x144	3.7500	175.10		(0,1,2)
12	176x144	7.5000	221.80		(0,2,2)
13	176x144	15.0000	275.20		(0,3,2)
14	176x144	30.0000	328.30		(0,4,2)
15	352x288	1.8750	232.30	136.70	(1,0,0)
16	352x288	3.7500	272.40	152.10	(1,1,0)
17	352x288	7.5000	329.50	172.90	(1,2,0)
18	352x288	15.0000	393.60	194.50	(1,3,0)
19	352x288	30.0000	460.10	219.60	(1,4,0)
20	352x288	1.8750	317.40		(1,0,1)
21	352x288	3.7500	370.00		(1,1,1)
22	352x288	7.5000	444.00		(1,2,1)
23	352x288	15.0000	526.80		(1,3,1)
24	352x288	30.0000	611.50		(1,4,1)
25	352x288	1.8750	506.40		(1,0,2)
26	352x288	3.7500	600.20		(1,1,2)
27	352x288	7.5000	730.50		(1,2,2)
28	352x288	15.0000	880.60		(1,3,2)
29	352x288	30.0000	1030.00		(1,4,2)

Figura 5.16: Arquivo de *bitrate* de um vídeo escalável.

A coluna *bitrate* representa o *bitrate* que uma camada com um FR, uma resolução e uma qualidade determinada precisa para ser transmitida. Portanto, é possível determinar qual camada transmitir (junto com as camadas subjacentes à dita camada) usando um valor de *bitrate* específico.

Neste artigo, o valor do SBR depende também do tipo do vídeo (SM, GW, RM), pois, para cada tipo de vídeo os valores dos coeficientes usados para calcular o SBR variam.

O algoritmo usado para este propósito é descrito a seguir.

O usuário começa o processo solicitando o vídeo e enviando um valor de MOS requerido (é recomendado um valor de MOS maior do que 3 para uma QoE, no mínimo, aceitável). Neste ponto, como mostraram os experimentos, valores de MOS maiores ou iguais a 3 precisam, aproximadamente, de um FR maior ou igual 15. Portanto, um valor de MOS tem um valor de FR associado, e, pelas configurações do codificador, os únicos valores de FR maiores ou iguais a 15 são 15 e 30.

As perdas inicialmente tem um valor de 0% e, à medida que a transmissão esteja sendo realizada, podem ser obtidas estatísticas da rede para mudar o valor das perdas (PER) e atualizar o valor do SBR. Este valor de perdas também pode ser obtido por medio de um *feedback* do par que esteja recebendo o vídeo.

O valor do SBR define o FR também, pois se houver poucas perdas na rede, podem ser obtidos valores maiores de SBR, valores estes que definem as camadas a serem transmitidas, ou seja, camadas que têm associados valores específicos de FR para melhorar a qualidade percebida, ou valores maiores de resolução, embora o FR sempre terá prioridade.

Assim, com um MOS desejado pelo usuário e definindo perdas de 0%, é possível calcular um valor de SBR inicial, digamos SBR_0, para decidir qual camada do vídeo vai ser enviada para o par que esteja recebendo o vídeo.

O esboço do algoritmo para o esquema adaptativo é apresentado a seguir:

Algoritmo 1 Esquema Adaptativo

Obter: 0≥ MOS ≥ 5 //Recomendado MOS ≥ 3, valor dado pelo usuário.
Definir: PER_0=0
FR_0=15
N=10
Calcular $SBR_0 = e^{\frac{MOSe^{-dPER}-a-bFR}{c}}$

Selecionar: Camada[0] com *bitrate* ≤ SBR_0

Enviar: Camada[0]
t = duração do vídeo/N // define cada quanto é feita a atualização do valor de SBR e do MOS
Enquanto: reprodução do vídeo esteja sendo feita
Cada t Fazer
Obter PERi // PER do par i que está recebendo o vídeo
Obter MOSi // Atualizar o valor de MOS, se o usuário deseja aumentar o diminuir.
Calcular SBRi
Se SBRi ≤ *bitrate*[camada base]
Enviar: Camada[base]
Senão Se SBRi ≥ *bitrate*[maior camada]
Enviar: Camada[maior]
Senão
Selecionar: Camada[i] com *bitrate* ≤ SBRi
Enviar: Camada[i]
Fim Se
Fim Enquanto

Fim Algoritmo 1

Como uma observação, fica o fato de que o SBR também depende do tipo de vídeo a ser transmitido (SM, GW ou RM). Assim, para cada tipo de vídeo existem valores diferentes de SBR e valores limites, mostrados na Tabela 5.2.

A Figura 5.17 mostra o diagrama conceitual do funcionamento do esquema adaptativo proposto. A figura mostra uma parte da rede em malha, onde vários pares

estão enviando um vídeo para outro par. O esquema é de baixa complexidade, pois o preditor precisa processar poucos dados (calcular o SBR baseado no PER e no MOS desejado pelo par que está recebendo o vídeo) para tomar decisões sobre quais camadas do vídeo enviar. Diferentemente dos casos de esquemas adaptativos mostrados no Capítulo 4, o esquema não está centrado no receptor, pois este só envia informações aos pares que enviam o vídeo, e são os remetentes quem decidem qual camada vão enviar. Outra diferença é o uso do simulador P2PTVsim anteriormente mencionado, pois aquele usado na maioria dos casos anteriormente expostos é o NS-2.

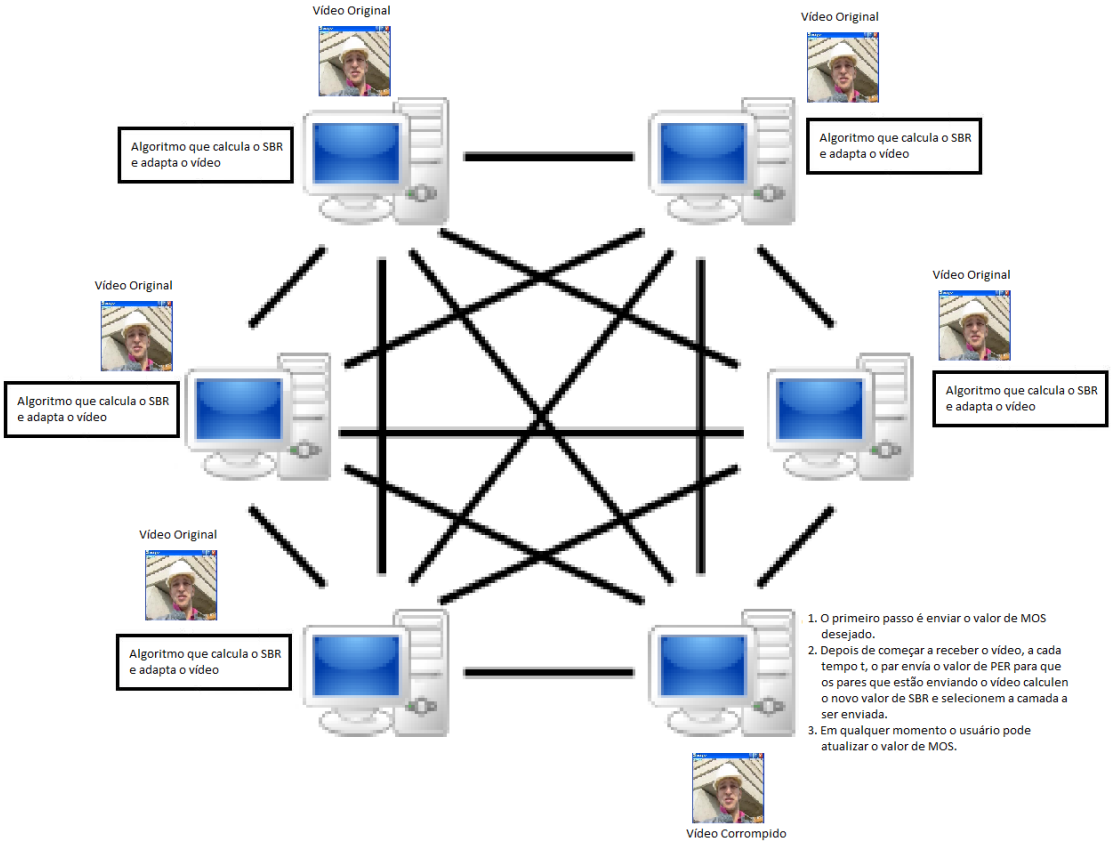

Figura 5.17: Diagrama conceitual do esquema adaptativo.

A Figura 5.18 mostra a adaptação do SBR sendo feita, onde se mostra que o SBR varia de acordo com os parâmetros que o algoritmo usa para calcular este valor e, de acordo com este, selecionar o valor adequado no arquivo de *bitrate* do vídeo codificado. Cabe resaltar que os valores do SBR estão limitados pelos valores contidos no arquivo de *bitrate* e, quando o SBR tem um valor maior ou menor do que o limite estabelecido no arquivo de *bitrate*, o algoritmo trunca esse valor ao máximo ou ao mínimo valor possível.

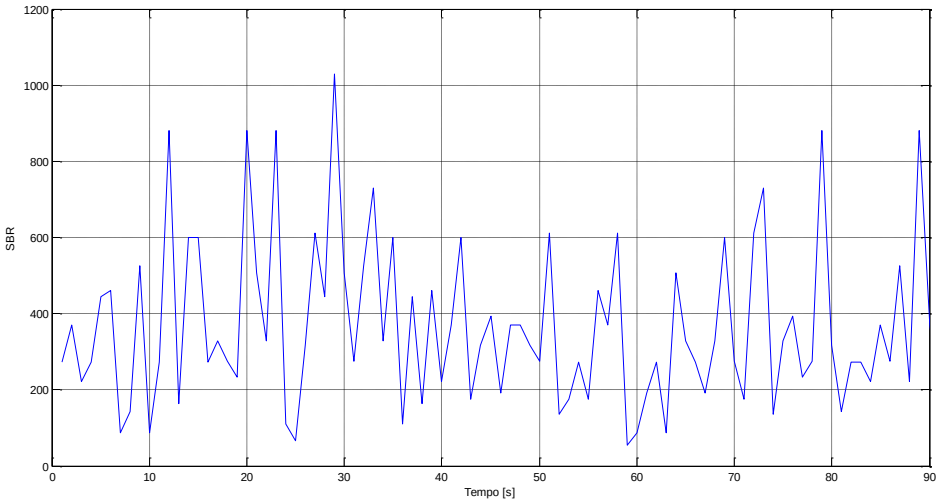

Figura 5.18: Adaptação do *bitrate*.

A Figura 5.19 mostra o valor do MOS com o esquema e sem o esquema adaptativo proposto nesta dissertação. Pode ser vista a maior variação da avaliação do MOS obtido sem o esquema em relação ao MOS com o esquema. Como o MOS é definido pelo usuário, e no caso o valor do MOS foi predefinido para ser 3.5, a adaptação tenta fazer com que o valor do MOS seja sempre 3.5 (ou esteja o mais perto possível) e adapta o *streaming* para tentar obter sempre este valor.

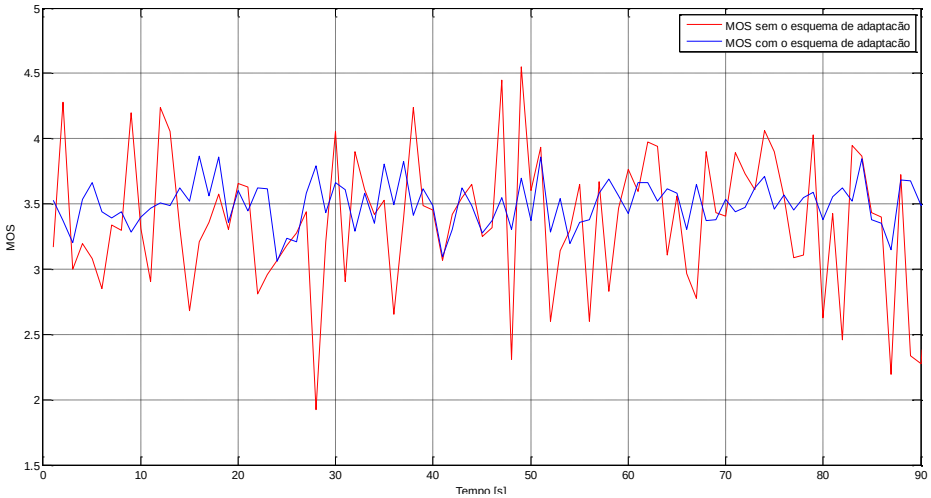

Figura 5.19: Adaptação do MOS.

5.4 - CONCLUSÃO

O presente capítulo tratou os aspectos referentes à descrição dos experimentos e à proposta de mapeamento QoS-QoE. Dentro da descrição dos experimentos foi necessário falar sobre as simulações e sobre a arquitetura proposta, além da metodologia adotada para a avaliação do MOS. Outro assunto importante tratado neste capítulo é o mapeamento de QoS em QoE, como é feita a predição do MOS e a avaliação do

impacto dos parâmetros de rede selecionados (SBR, PER e FR) sobre o MOS. Finalmente foi proposto um esboço para um esquema adaptativo baseado nos resultados obtidos.

6 – CONCLUSÕES E TRABALHOS FUTUROS

Esta dissertação teve por objetivo o tratamento de aspectos relativos à qualidade de serviço e de experiência em redes do tipo P2P, quando utilizadas para transmissão de vídeo escalável. Para esse fim, foi primeiramente feita uma revisão bibliográfica, de forma a facilitar o entendimento e o aprofundamento das questões aqui tratadas.

De início, e para melhor aproveitar a característica de adaptabilidade do vídeo escalável, buscamos apresentar e discutir as suas características e possibilidades, vantagens e benefícios, especialmente as escalabilidades espacial, temporal e de qualidade, ao lado de métricas de avaliação da qualidade de vídeo.

Em seguida, as redes P2P e os sistemas de distribuição de VoD que utilizam essas redes (os sistemas P2PVoD) foram apresentados. Foi apresentada uma classificação para tais sistemas, segundo o grau de descentralização da rede P2P, a estrutura da rede P2P e o mecanismo de descoberta de conteúdo. Mostramos as estruturas baseadas em árvore e malha, apontando as vantagens e desvantagens das duas estruturas e a complexidade de implementar uma estrutura híbrida árvore-malha.

Foram então abordados os sistemas e propostas que tratam da utilização de vídeo escalável sob demanda em redes P2P, tendo sido ponto comum na maioria dessas propostas o adequado "casamento" entre essas duas áreas. Foram também apontados fatores chave no desenho dos sistemas P2PVoD e alguns mecanismos-chave para melhorar o desempenho desses sistemas.

Para completar esse arcabouço e focar de forma específica no problema tratado, a questão do inter-relacionamento entre QoS (*Quality of Service*) e QoE(*Quality of Experience*) foi estudada com base em diversas propostas relativas ao assunto, ainda que sem o foco predominante de vídeo escalável em redes P2P, face ao que foi localizado na literatura. Verificou-se a importância de avaliar o impacto de parâmetros de rede e de aplicação na qualidade percebida pelo usuário final em um sistema de distribuição VoD sobre uma rede P2P, com o vídeo sendo codificado no padrão H.264/SVC, escolhido considerando a sua característica de adaptabilidade.

Após levantamento daquelas propostas, propusemos um mapeamento QoS-QoE, visando encontrar uma relação entre a qualidade de serviço (definida por métricas de rede e de aplicação) e a qualidade subjetiva (estimada pelo MOS – *Mean Opinion Score*) em sistemas de distribuição de vídeo escalável sob demanda em redes P2P.

Em seguida, foi feita uma avaliação objetiva do desempenho da distribuição de vídeo

sobre redes P2P (usando o PSNR), a qual foi mapeada na provável QoE vivida pelo usuário a partir da QoS provida pela rede, estimada considerando parâmetros de aplicação (SBR e FR) e de rede (PER).

Com base nos resultados dos testes e em uma análise estatística dos dados obtidos, logrou-se produzir um preditor de QoE, explicitado por meio do valor do MOS de um sistema de distribuição de vídeo sob demanda sobre redes P2P.

A metodologia usada mostrou-se adequada, permitindo avaliar de forma consistente o desempenho do preditor proposto, com base em simulação a eventos discretos utilizando ferramenta escolhida após avaliação de um conjunto de ferramentas de simulação de sistemas P2P (Apêndice II). Tal preditor apresentou resultados considerados muito bons, sob o ponto de vista da análise estatística, tomando por base o coeficiente de determinação (R^2) e a raiz do erro quadrático médio (RMSE). E comparando com o vídeo original.

Com base na equação encontrada, tornamos possível, para o sistema considerado e as sequências de vídeo utilizadas, o mapeamento de QoS em QoE, isto é, a partir de métricas objetivas e parâmetros quantificáveis, pode ser inferida uma métrica subjetiva como é o MOS.

A contribuição mais importante deste trabalho consiste em um preditor de MOS em um sistema de distribuição de vídeo escalável sobre uma rede P2P. Algo que, até o momento, não tinha sido feito para essa codificação em conjunto com a arquitetura de rede considerada (ou ao menos foi o que mostrou o levantamento bibliográfico feito ao longo do desenvolvimento do nosso trabalho).

Experimentos adicionais podem ser realizados para tentar melhorar o desempenho do preditor, ou seja, para obter um maior volume de dados para ajustar melhor os coeficientes da equação e, se possível, melhorar o rendimento.

Como trabalho futuro propomos implementar o esquema adaptativo proposto em alguma das linguagens de programação disponíveis, bem como avaliar e propor técnicas de classificação de sequências de vídeo. Outro trabalho futuro consiste em implementar o esquema adaptativo em um sistema real, como por exemplo o Goalbit que é de código aberto, mas ainda não fornece suporte à distribuição de vídeo escalável. Outro trabalho consistiria em fazer a avaliação da qualidade sobre redes heterogêneas.

Por fim, visando validar e aperfeiçoar o trabalho com dados de MOS obtidos de forma subjetiva, sugere-se realizar experimentos com usuários reais para comparar com os valores de MOS gerados pelo preditor.

REFERÊNCIAS

[1]. Y. Ding, J. Liu, D. Wang, and H. Jiang, "Peer-to-peer video-on-demand with scalable video coding," *Computer Communications*, vol. 33, pp. 1589-1597, 2010.

[2]. Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, pp. 18-28, 2008.

[3]. L. Yu, X. Liao, H. Jin, and W. Jiang, "Integrated buffering schemes for P2P VoD services," *Peer-to-Peer Networking and Applications*, vol. 4, pp. 63-74, 2010.

[4]. Y. He and Y. Liu, "VOVO: VCR-Oriented Video-on-Demand in Large-Scale Peer-to-Peer Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, pp. 528-539, 2009.

[5]. Y. Huang, T.Z.J. Fu, D.-M. Chiu, J.C.S. Lui and C. Huang, "Challenges, design and analysis of a large-scale P2P-VoD system," *Proceedings of the ACM SIGCOMM 2008 conference on Data communication - SIGCOMM '08*, p. 375, 2008.

[6]. K. Huguenin, A.-M. Kermarrec, V. Rai and M. Van Steen, "Designing a tit-for-tat based peer-to-peer video-on-demand system," *Proceedings of the 20th international workshop on Network and operating systems support for digital audio and video - NOSSDAV '10*, p. 93, 2010.

[7]. X. Qiu, C. Wu, X. Lin and F. Lau, "InstantLeap: fast neighbor discovery in P2P VoD streaming," *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, ACM, p. 19–24, 2009.

[8]. T.T. Do, K. a Hua and M. a Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*, Vol.3, pp. 1467-1472, 2004.

[9]. A. Bhattacharya, Z. Yang, and S. Zhang, "Temporal-DHT and Its Application in P2P-VoD Systems," *2010 IEEE International Symposium on Multimedia*, pp. 81-88, 2010.

[10]. D. Wang and C.K. Yeo, "Exploring Locality of Reference in P2P VoD Systems," *2010 IEEE Global Telecommunications Conference, GLOBECOM '10*, pp. 1-6, 2010.

[11]. D. Xie, B. Qian, Y. Peng and T. Chen, "A Model of Job Scheduling with Deadline for Video-on-Demand System," *2009 International Conference on Web Information Systems and Mining*, pp. 661-668, 2009.

[12]. B. Cai, Z. Luo and Y. Luo, "A Special Topology for Files Pre-Fetch in Large Scale Video on Demand," *2010 International Conference on Biomedical Engineering and Computer Science*, pp. 1-4, 2010.

[13]. J.-H. Roh and S.-H. Jin, "Video-on-Demand Streaming in P2P Environment," *2007 IEEE International Symposium on Consumer Electronics*, pp. 1-5, 2007.

[14]. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," *Proceedings of the ACM SIGCOMM 2001 conference on Data communication - SIGCOMM '01*, pp. 149–160, 2001.

[15]. Z. Lu, S. Zhang, J. Wu, W. Fu and Y. Zhong, "Design and Implementation of a Novel P2P-Based VoD System Using Media File Segments Selecting Algorithm," *7th IEEE International Conference on Computer and Information Technology (CIT 2007)*, pp. 599-604, 2007.

[16]. U.C. Kozat, O. Harmanci, S. Kanumuri, M.U. Demircin and M.R. Civanlar, "Peer Assisted Video Streaming With Supply-Demand-Based Cache Optimization," *IEEE Transactions on Multimedia*, vol. 11, pp. 494-508, 2009.

[17]. A. Wu, L. Yuan, X. Liu and K. Liu, "A P2P Architecture for Large-scale VoD Service," *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007)*, pp. 841-846, 2007.

[18]. Z. Lian-ying and Z. Xiao, "The research of VoD system performance based on CDN and P2P technologies," *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, vol. 4, pp. 385-388, 2010.

[19]. S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," ACM Computing Surveys, vol. 36, pp. 335-371, 2004.

[20]. M. Zhang and B. Feng, "A P2P VoD system using dynamic priority," *2009 IEEE 9th Malaysia International Conference on Communications (MICC)*, pp. 518-523, 2009.

[21]. Q. Gu, "A schedule algorithm of peer-to-peer VoD system," *International Conference On Computer and Communication Technologies in Agriculture Engineering, CCTAE'10*, p. 583–586, 2010.

[22]. S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena and P.R. Rodriguez, "Is high-quality VoD feasible using P2P swarming?," *Proceedings of the 16th international conference on World Wide Web, ACM*, pp. 903–912, 2007.

[23]. R.P. Leal, E.P. Martín and J.A. Cachinero, “Internet TV Broadcast: What Next?,” *2009 Fourth International Conference on Digital Telecommunications*, pp. 71-74, 2009.

[24]. X. Zhang, J. Liu, B. Li and T. Yum, CoolStreaming/DONet: “A data-driven overlay network for efficient live media streaming,” *IEEE INFOCOM’05*, vol. 3, pp. 2102–2111, 2005.

[25]. BitTorrent Homepage http://www.bittorrent.com Acessado em 30/04/2013.

[26]. L. Kai, S. Xiao, Q. Zhigang, P. Lingjuan and L. Hui, “Peers’ Segment Replacement Strategies in P2P VoD System Based On Client-Side Segmented Cache,” *2009 WRI International Conference on Communications and Mobile Computing*, Jan. 2009, pp. 26-33.

[27]. M. Zhang, Y. Xiong, Q. Zhang, L. Sun and S. Yang, “Optimizing the throughput of data-driven peer-to-peer streaming,” *IEEE Transactions on Parallel and Distributed systems*, vol. 20, Jan. 2008, p. 97–110.

[28]. Goalbit Homepage http://goalbit.sourceforge.net/ Acessado em 30/04/2013.

[29]. Joost Homepage http://www.joost.com/ Acessado em 30/04/2013.

[30]. Gridmedia Homepage http://igridmedia.sourceforge.net/ acessado em 30/04/2013.

[31]. CoolStreaming Homepage http://www.coolstreaming.us/ acessado em 30/04/2013.

[32]. PPLive Homepage http://www.pplive.com acessado em 30/04/2013.

[33]. Tribler Homepage http://www.tribler.org acessado em 30/04/2013.

[34]. Evalvid homepage http://www.tkn.tu-berlin.de/research/evalvid/cif.html. Acessado em 30/04/2013

[35]. H. Sohn, H. Yoo, W. De Neve, C. S. Kim and Y. M. Ro, “Full-Reference Video Quality Metric for Fully Scalable and Mobile SVC Content”, IEEE Transactions on Broadcasting, Vol. 56, 2010.

[36]. Skype: Peer-To-Peer Telephone Software http://www.skype.com acessado em 30/04/2013

[37]. X. Zhua, T. Schierl, T. Wiegand and B. Giroda, “Video Multicast over Wireless Mesh Networks with Scalable Video Coding (SVC)”, NSF Grant CCR-0325639. 2008.

[38]. H. Schwarz, D. Marpe and T. Wiegand, Member, IEEE, “Overview of the Scalable Video Coding Extension of the H.264/AVC Standard”, *IEEE*

Transactions On Circuits And Systems For Video Technology, Vol. 17, No. 9, 2007.

[39]. Gnutella Homepage. http://www.gnutellaforums.com/ acessado em 30/04/2013.

[40]. M. Wien, H. Schwarz and T. Oelbaum, "Performance Analysis of SVC", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 17, No. 9, 2007.

[41]. H. Schwarz and T. Wiegand, "The Scalable Video Coding Amendment of H.264/AVC Standard", Heinrich-Hertz-Institut, 2008.

[42]. H. Schwarz and M. Wien, "*The Scalable Video Coding Extension of the H.264/AVC Standard*, IEEE Signal Processing Magazine, Institute of Electrical and Electronic Engineers, Vol. 25, No. 2, pp. 135, 2008.

[43]. Z. Avramova and D. D. Vleeschauwer, "Comparison of simulcast and scalable video coding in terms of the required capacity in an IPTV network," *Packet Video*, pp. 113-122, 2007.

[44]. U. Engelke and H.-J. Zepernick, "Perceptual-based Quality Metrics for Image and Video Services: A Survey," *2007 Next Generation Internet Networks*, pp. 190-197, 2007.

[45]. L. Lu and X. Lu, "Quality assessing of video over a packet network," *Second Workshop on Digital Media and its Application in Museum & Heritages (DMAMH 2007)*, pp. 365-369, 2007.

[46]. J. F. Kurose and K. W. Ross, "Redes de Computadores e a Internet – Uma Abordagem top down", Person Addison Wesley, 3ª edição. 2006.

[47]. Y. Wang, "Survey of Objective Video Quality Measurements,*", EMC Corporation Hopkinton*, MA, Vol. 1748, 2006.

[48]. C. Kiraly, L. Abeni and R. Lo Cigno, "Effects of P2P Streaming on Video Quality," *Communications (ICC), 2010*, no. 214412, pp. 2-6, 2010.

[49]. Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity.," *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, vol. 13, no. 4, pp. 600-12, 2004.

[50]. RFC 3393, 2002. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)

[51]. A. Khan and L. Sun, “QoE-driven adaptation scheme for video applications over wireless networks,” IET Communications, Vol. 4, No.2 Special Issue on "Video communications over wireless networks", pp. 1337 - 1347, 2010.

[52]. DSL Forum Technical Work WT-126. Video services quality of experience (QoE) requirements and mechanisms, 2007.

[53]. J. Xu, L. Xing, A. Perkis and Y. Jiang, “On the Properties of Mean Opinion Scores for Quality of Experience Management,” *2011 IEEE International Symposium on Multimedia*, pp. 500-505, 2011.

[54]. F. Ribeiro, D. Florencio, C. Zhang and M. Seltzer, "CrowdMOS: An approach for crowdsourcing mean opinion score studies," *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on* , vol., no., pp.2416-2419,2011

[55]. F. Agboma, M. Smy and A. Liotta, “QoE analysis of a peer-to-peer television system,” *Proc. of IADIS Telecommunications Networks and Systems*, pp. 114-119, 2008.

[56]. M. Fiedler, T. Hossfeld and P. Tran-Gia, “A generic quantitative relationship between quality of experience and quality of service,” Network, IEEE, vol. 24, no. 2, pp. 36–41, 2010.

[57]. S. Khirman and P. Henriksen, “Relationship between Quality-of-service and Quality-of-Experience for Public Internet Service,” *Proc. of the 3rd Workshop on Passive and Active Measurement*, 2002.

[58]. H. Kim and S. Choi, “A study on a QoS/QoE Correlation Model for QoE Evaluation on IPTV Service,” *Proc. of The 12th International Conference on Advanced communication technology, ICACT'10*, pp. 1377-1382, 2010.

[59]. L. Li-yuan, Z. Wen-an and S. Jun-de, "The Research of Quality of Experience Evaluation Method in Pervasive Computing Environment," *Pervasive Computing and Applications, 2006 1st International Symposium on* , vol., no., pp.178-182, 2006

[60]. D. Lopez et al., “Adaptive multimedia streaming over IP based on customer oriented metrics”, *Proc. of The seventh IEEE International Symposium on Computer Networks, ISCN’06*, pp. 185–191, 2006.

[61]. K. Yang and B. El-Haik: Design for Six Sigma McGraw-Hill. ISBN 0071412085. 2003.

[62]. National Institute of Standards and Tecnology (NIST). e-Handbook of Statistical Methods. [Online]. http://www.itl.nist.gov/div898/handbook/ Acessado em 30/04/2013.

[63]. S. Ickin et al., "The effects of Packet Delay Variation on the perceptual quality of video," *IEEE 35th Conference on Local Computer Networks (LCN)*, vol., no., pp.663-668, 2010.

[64]. M. Mushtaq and T. Ahmed. "Smooth Video Delivery for SVC Based Media Streaming Over P2P Networks," *IEEE 5th Consumer Communications and Networking Conference, CCNC'08*, pages 447-451, 2008

[65]. G. Zhang and C. Yuan. "Self-adaptive Peer-to-Peer streaming for heterogeneous networks using Scalable Video Coding," *IEEE 12th International Conference on Communication Technology*, pp. 1390-1393, 2010.

[66]. Y. Sánchez, T. Schierl, C. Hellge, and T. Wiegand, "P2P group communication using Scalable Video Coding," *17th IEEE International Conference on Image Processing ICIP'10*, pp.4445-4448, 2010.

[67]. F. de A. L. Fuentes, "Adaptive Mechanism for P2P Video Streaming Using SVC and MDC," *International Conference on Complex, Intelligent and Software Intensive Systems CISIS'10*, pp.457-462, 2010.

[68]. PlanetLab Homepage: https://www.planet-lab.org/ acessado em 30/04/2013

[69]. R. P. Nunes, R. S. Cruz and M. S. Nunes, "Scalable Video Distribution in Peer-to-Peer Architecture," *Proceedings of the 10a Conferência sobre Redes de Computadores CRC'10*, pp. 95-100, 2010.

[70]. U. Abbasi, M. Mushtaq and T. Ahmed, "Delivering scalable video coding using P2P Small-World based push-pull mechanism," *Global Information Infrastructure Symposium, GIIS '09*, pp.1-7, 2009.

[71]. M. Zhang, J-G. Luo, L. Zhao and S. Yang, "A Peer-to-Peer Network for Live Media Streaming - Using a Push-Pull Approach". *Proceedings of ACM Multimedia 2005*, short paper, 2005.

[72]. M. Abboud, T. Zinner, K. Pussep, S. Al-Sabea and R. Steinmetz, "On the impact of quality adaptation in SVC-based P2P video-on-demand systems". *Proceedings of the second annual ACM conference on Multimedia systems*, pp. 223-232, 2011.

[73]. A. Abdelhalim and T. Ahmed, H. Walid-Khaled and S. Matsuoka, "Using Bittorrent and SVC for efficient video sharing and streaming". *2012 IEEE Symposium on Computers and Communications (ISCC)*, pp. 537-543, 2012.

[74]. N. Ramzan, E. Quacchio, T. Zgaljic, S. Asioli, L. Celetto, E. Izquierdo and F. Rovati, "Peer-to-peer streaming of scalable video in future Internet applications". *IEEE Communications Magazine*, vol.49, no.3, pp.128-135, 2011.

[75]. FFmpeg homepage http://www.ffmpeg.org/ Acessado em 30/04/2013

[76]. G. A. F. Seber and C. J. Wild. *Nonlinear Regression*. Hoboken, NJ: Wiley-Interscience, 2003.

[77]. MATLAB nlinfit http://www.mathworks.com/help/toolbox/stats/nlinfit.html Acessado em 30/04/2013

[78]. D. Zhang, X. Chen and H. Yang, "State of the art and challenges on peer-to-peer simulators," *Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing, WiCOM'09*, 2009.

[79]. S. Naicken, A. Basu, B. Livingston and S. Rodhetbhai, "Towards yet another peer-to-peer simulator", *Proc. of The 4th International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs)* pp. 1-10, 2011.

[80]. R. Bhardwaj, V. Dixit and A. K. Upadhyay, "An Overview on Tools for Peer to Peer Network Simulation," *International Journal of Computer Applications*, vol. 1, no. 19, pp. 74-81, 2010.

[81]. L. Abeni, L. Kiraly and R. Lo Cigno, "SSSim: a Simple and Scalable Simulator for P2P Streaming Systems," *Proc. of IEEE Computer-Aided Modeling and Design of Communication Links and Networks CAMAD'09,* 2009.

[82]. R. Lo Cigno, A. Russo and D. Carra, "On some fundamental properties of P2P push/pull protocols," *Second International Conference on Communications and Electronics, ICCE 2008.*, pp.67-73, 2008.

[83]. L. Bracciale, F. Lo Piccolo, D. Luzzi, S. Salsano, G. Bianchi and N. Blefari-Melazzi, "A push-based scheduling algorithm for large scale P2P live streaming," *4th International Telecommunication Networking Workshop on QoS in Multiservice IP Networks, IT-NEWS 2008.*, pp.1-7, 2008.

[84]. L. Abeni, C. Kiraly and R. Lo Cigno, "On the Optimal Scheduling of Streaming Applications in Unstructured Meshes," *Proceedings of the 8th International IFIP-TC 6 Networking Conference, NETWORKING '09*, pp. 117-130, 2009.

[85]. A. Ghanbari, H. R. Rabiee, M. Khansari and M. Salehi, "PPM - A Hybrid Push-Pull Mesh-Based Peer-to-Peer Live Video Streaming Protocol," *21st International Conference on Computer Communications and Networks (ICCCN)*, pp.1-8, 2012.

[86]. P. Giacomazzi and A. Poli, "Push-pull techniques in peer-to-peer video streaming systems with tree/forest topology," *2010 International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp.89-95, 2010.

[87]. M. Zhang, J. G. Luo, L. Zhao and S. Q. Yang, "A peer-to-peer network for live media streaming - Using a push-pull approach," *in Proceedings of the ACM Multimedia'05*, 2005.

[88]. N. Magharei and R. Rejaie, "PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming," *Networking, IEEE/ACM Transactions on* , vol.17, no.4, pp.1052 - 1065, 2009.

[89]. M. Zhou and J. Liu, "Tree-assisted gossiping for overlay video," *Multimedia Tools Applications*, vol.29, issue 3, pp. 211-232. 2006.

[90]. A. J. Ganesh, A. Kermarrec and L. Massoulie, "Peer-to-peer membership management for gossip-based protocols," *IEEE Transactions on Computers*, vol.52, no.2, pp.139-149, 2003.

[91]. H. Jiang and S. Jin, "Exploiting dynamic querying like flooding techniques in unstructured peer-to-peer networks," *13th IEEE International Conference on Network Protocols, ICNP 2005*, pp. 6-9, 2005.

[92]. P2PTVSim http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim Acessado em 30/04/2013.

[93]. F. G. Marmol and G. M. Perez, "State of the Art in Trust and Reputation Models in P2P networks", In: Shen, X. et al., (Editores) *Handbook of Peer to Peer Networking*. USA: Springerlink, 2010. pp. 761-784.

[94]. A. Samreen ans S. Hussain, "Trust management and incentive mechanism for P2P networks: Survey to cope challenges," *IEEE International Multitopic Conference 2008, INMIC 2008*, pp. 301-306, 2008.

[95]. J. Kang, G. Goh and K. Chung, "A New Inter-Layer Prediction Scheme for Spatial Scalability with Different Frame Rates", *Proceedings of the 2009 ACM Symposium on Applied Computing*, pp. 1779-1783, 2009.

APÊNDICE I - FERRAMENTAS DE SIMULAÇÃO E PROCESSAMENTO DE DADOS.

I.1 – SIMULADORES DE REDES P2P

Para avaliar o desempenho de uma rede P2P podem ser usados métodos analíticos, medições em um sistema real ou por médio de simulações.

Usar uma abordagem analítica precisaria de um modelo matemático do sistema P2P, porém, esses sistemas tendem a serem complexos. Além disso, muitas simplificações tem que ser assumidas para poder gerar o modelo. E, por causa dessas simplificações, qualquer resultado obtido teria uma aplicabilidade limitada [78] [79].

Uma alternativa à abordagem analítica é fazer medições em um sistema real, já que fazer experimentos em sistemas reais é muito arriscado e caro, e consome muito tempo e recursos de *hardware* e de administração [79] [80]. Um sistema P2P tem que ser escalável e suportar um grande número de usuários. Por exemplo, seria muito difícil dispor de 1000 computadores/pares em um laboratório.

O interesse dos pesquisadores/cientistas pelos *frameworks* para simular redes P2P tem crescido nos últimos anos, para verificar a confiabilidade da tecnologia e dos métodos, antes de serem testados no mundo real [80]. Os simuladores não sofrem dos problemas mencionados anteriormente na análise matemática e nos experimentos com sistemas reais [81]. A implementação na simulação tem um custo menor do que um experimento em larga escala, e significativamente menos recursos são consumidos. Também, o modelo ainda pode ser mais simples do que um modelo matemático e pode compartilhar similaridades no código de implementação com nós P2P reais [80].

Se desenhados cuidadosamente, os simuladores podem ser ferramentas muito eficientes e muito populares, para pesquisar e validar as aplicações nas diferentes tecnologias de P2P [80].

Atualmente existem muitos simuladores na literatura, que abrangem desde modelagem de redes no nível dos pacotes até os que se concentram puramente na rede *overlay*. Além dos simuladores usados para P2P, existem outros simuladores de redes em geral que também são usados para P2P. Entre os simuladores, de rede P2P e de redes em geral, temos [78] [79] [80]:

- P2PTVSim
- SSSim

- OverSim
- FreePastry
- SimPastry
- JavaSim
- P2PSim
- GnutellaSim
- Query Cycle
- ProtoPeer
- PeerSim
- Overlay Weaver
- DHTSim
- PlanetSim
- NS-3 (de propósito geral)
- Omnet++ (de propósito geral)

A maioria dos simuladores anteriormente mencionados são simuladores de propósito geral (dentro de redes), outros de propósito geral (dentro de redes P2P), e só 2 deles são usados para simular *streaming* de vídeo sobre redes P2P (SSSim e P2PTVSim) [82].

No trabalho de [81] são mostradas as vantagens do SSSim com respeito a P2PTVSim, falando de consumo de memória, número de pares da rede, tempo, etc. No artigo aparece como melhor o desempenho do SSSim, mas assumindo muitas simplificações que para o nosso caso seriam inadequadas. Não se pode assumir que todos os clientes estão sincronizados, que a largura de banda de *upload* é a mesma para todos os nós, e que a largura de banda de *download* é infinita (mesma capacidade de largura de banda nos clientes anularia uma parte do nosso pré-suposto que é a heterogeneidade dos pares**).**

Pelas razões expostas anteriormente, optamos por escolher o P2PTVSim, para o desenvolvimento do nosso trabalho. Um simulador que está focado em avaliar os efeitos de várias características e defeitos da rede sobre o desempenho dos escalonadores.

I.2 - Peer-to-Peer TV Simulator – P2PTVSim

O P2PTVSim foi inicialmente desenvolvido pelo Politécnico Di Torino, mas desde a sua criação ele tem evoluído a ponto de se tornar um simulador mais geral no qual vários parceiros do projeto têm contribuído. Este simula o fluxo de *chunks* através de

uma rede pares interconectados. O P2PTVSim faz parte do projeto NAPA-WINE[4] (***N**etwork-**A**ware **P**2P-TV **A**pplication over **W**ise **N**etworks*).

O P2PTVSim está escrito na linguagem de programação C++, linguagem padronizada pela ISO/IEC/ANSI. Trabalha sobre qualquer sistema operativo, mas o desenvolvimento e os testes são feitos ou em Visual Studio, no Windows, ou em gcc, no Linux. Não há dependências particulares, somente das bibliotecas padrões de C.

O P2PTVSim tem um arquivo de configuração, chamado de *config.txt* e localizado na pasta onde foi instalado o simulador, para configurar o comportamento do simulador. Além disso, é possível usar argumentos desde a linha de comandos para serem adicionados à lista de parâmetros ou para sobrescrever os argumentos que já estão no arquivo.

Há um grande número de parâmetros de configuração, e o completo entendimento dos efeitos de todos eles precisam um conhecimento profundo do desenho e da implementação do simulador. A maioria dos parâmetros tem associado um comentário de explicação incluído no arquivo de configuração de exemplo fornecido com o programa. Alguns dos parâmetros mais importantes são os seguintes:

- *FilenamePrefix:* Pode ser usado para direcionar todos os arquivos de saída do simulador para uma pasta específica (a qual já deve existir) ou para prefixar eles com algum *string* de texto que identifica uma simulação em particular.
- *PeerType*: provavelmente será LATEST_BA, LATEST_RANDOM, ou CONFIG_PEER. Este parâmetro estabelece a subclasse particular do *Peer* a ser usado no *overlay*. A principal diferença entre os vários pares é a estratégia do escalonador dos *chunks* que eles usam para selecionar um *chunk* e um *peer* alvo quando têm largura de banda de *upload* livre. O simulador usa o método *push* para a distribuição de *chunks* na rede, mesmo que haja pouca diferença entre *push* e *pull* em um ambiente onde cada par tem (potencialmente) conhecimento completo dos estados dos outros pares. Assim, no valor de LATEST_BA o par seleciona o último *chunk* útil que tem na sua janela e o envia a um vizinho que ele seleciona de uma forma "ciente" da largura de banda. Especificamente, a probabilidade de selecionar um dos vizinhos é proporcional à largura de banda de *upload* do vizinho. Os pares do tipo LATEST_RAMDOM, do outro lado, simplesmente selecionam um vizinho aleatoriamente como alvo para o último

[4] http://www.napa-wine.eu

chunk. Os pares do tipo CONFIG_PEERS, são mais gerais e são configurados usando a *string* de configuração fornecida no parâmetro *ConfigPeerConfig dentro do arquivo de configuração. Este tipo de par pode selecionar ou o chunk primeiro e depois o par alvo, ou o par alvo primeiro e o chunk adequado depois, usando uma variedade de diferentes estratégias de seleção. No arquivo de exemplo de configuração fornecido (config.txt*) e no código fonte em *configpeer.h e em configpeer.cpp pode se ter uma completa explicação deste tipo de pares.*

- *OverlaySeed*: Este argumento estabelece o número de sementes (*seeds*) para gerar a rede *overlay* aleatória e para as decisões de subseqüentes de distribuição dos *chunks* (e.g., quando selecionar um par para lhe enviar um *chunk* em particular) se nenhuma semente de distribuição explicita tem sido especificada (ver parâmetro a seguir).
- *DiffusionSeed:* Estabelece um número aleatório de sementes separados para uso somente na distribuição dos *chunks*.
- *NumPeers: Estabelece o número de pares na rede overlay (sem incluir a fonte).*
- *OverlayType: determina como os pares estão interconectados no overlay. Os valores possíveis são: FULL, GNR, GNP, TREE, e PREBUILT. O primeiro valor leva a uma rede totalmente conectada, onde cada par está conectado a todos os outros pares participantes da rede. Em um overlay GNR cada par seleciona aleatoriamente um número Degree (próximo parâmetro a ser descrito) de vizinhos para se conectar. Um overlay GNP é similar a um overlay GNR, mas no caso do GNP o número de vizinhos é selecionado usando uma distribuição de Poisson. Um overlay TREE é um grafo direcionado, com a fonte como raiz (ver estrutura de árvore). Um overlay PREBUILT lê arquivos de configuração especificados pelos parâmetros InputDegreeFile* e *InputLinkFile para reusar um overlay existente, e.g., um gerado em uma simulação previa.*
- *Degree: este parâmetro estabelece o número de vizinhos aleatórios que cada par do overlay seleciona, quando se usa um overlay GNR ou GNP. Desde que cada par seleciona este número de vizinhos o número total de enlaces par-par é de NumPeers * Degree, e se temos enlaces bidirecionais, o grau do grafo não direcionado é de 2 * Degree.*

- *NumChunks: Estabelece o número de chunks que serão distribuídos através da rede antes de que os arquivos estadísticos de saída sejam gerados.*
- *PeerBandwidths: este parâmetro define um conjunto de diferentes classes de larguras de banda dos pares, e estabelece a largura de banda de upload e de download para cada classe. O valor especial INF pode ser usado para indicar infinito. Dentro de cada classe, as larguras de banda dos pares são uniformemente distribuídas sobre o valor central específicado com uma variação fracionária especificada por um parâmetro adicional BandwidthFractVariation. Por exemplo, se PeerBandwidths fosse*

  {(0.25, 5, INF}, {0.75, 0, 1}}

  O resultado seria que 25% dos pares teriam uma largura de banda de *upload* aproximadamente 5 Mbps e uma largura de banda de *download* infinita, enquano que o 75% restante teria zero largura de banda de *upload* e uma largura de banda de *download* de 1 Mbps. Se *BandwidthFractVariation fosse 0.1, então haveria, pelo menos, um 10% de variação das larguras de banda a partir do valor médio especificado.*
- *SourceUploadBW*: este parâmetro determina a largura de banda de *upload* da fonte, o qual sempre é o par 0. Em todos os outros aspectos, a fonte é um par normal de classe 0, isto é, da primeira classe especificada pelo parâmetro *PeerBandwidths*. Se *SourceUploadBW* não for especificado, a largura de banda de *upload* da fonte é gerada aleatoriamente da mesma maneira como para todos os outros pares da classe 0.
- *PlayoutDelay: indica o máximo número de segundos decorridos desde que um chunk está disponível na fonte até que chega até um par, e se o chunk tem que ser colocado na janela do par. Os Chunks que levam mais tempo do que isso para chegar são registrados como "tendo chegado", mas não são armazenados para posterior distribuição. Um valor muito grande (por exemplo, 10 segundos) vai aumentar significativamente o tempo de execução do simulador.*
- *ChunkDelayCounters:* Cada par mantém um histograma dos atrasos dos *chunks* que recebe a partir de quando o *chunk* se tornou disponível na fonte. Este parâmetro define quantos contadores destes, isto é, barras do histograma, são armazenados em cada par. As barras abrangem o intervalo de 0 até

PlayoutDelay, com a última barra registrando qualquer chegada tarde. Pore exemplo, se o PlayoutDelay é 5 segundos e ChunkDelayCounters é 10, as barras serão para atrasos de 0-0.5, 0.5-1, 1-1.5, etc. até 4.5-infinito.

Depois de ter configurado a rede P2P, segundo os parâmetros anteriormente mencionados, e depois de rodar a simulação, o simulador gera uns arquivos de saída. Esses arquivos são os que servem para analisar o desempenho da rede. Esses arquivos são mencionados e descritos a seguir:

- *ConfigDump.out:* o conjunto completo de propriedades definidas pelo arquivo config.txt e /ou quaisquer parâmetros de linha de comando. Isto é principalmente apenas para referência, para que você possa ver quais parâmetros você usou para produzir um determinado resultado, mas também pode ser renomeado como *config.txt* e usado para regerar um conjunto de resultados.
- *Degree.out: uma linha por par, dando o seu grau, além de sua largura de banda de upload e download.*
- *Links.out: uma linha que resume as propriedades da rede overlay, seguido de uma linda por cada enlace da rede overlay. Dando a sua fonte e os números dos pares de destino.*
- *Upload_rate_peer.out:* uma linha para cada par, dando o seu número, largura de banda de *upload* e classe (ou seja, qual das tuplas no parâmetro *PeerBandwidths* foi usado para gerá-la).
- *Absolute_time_arrival.out: este arquivo é uma matriz com dimensão (NUM_CHUNKS x NUM_PEERS), com uma linha para cada chunk. Cada elemento (i, j) é o tempo absoluto em que um chunk dado i chegou ao par j. Um valor de -1 significa que o chunk não conseguiu chegar a esse par. Coletando os dados para este arquivo é proibitivamente caro na memória para simulações de grande porte, ele é feito apenas se os parâmetros ChunkArrivalsFile ou ChunkDelaysFile são dados. Caso contrário, o arquivo DelayHistograms.out pode ser usado para sintetizar os atrasos dos chunks através de toda a execução.*
- *Chunk_delay.out: são os mesmos dados do arquivo previamente explicado, exceto que são expressados em termos do atraso desde que o chunk se tornou disponível na fonte até que foi entregue no par específico. Tal como o arquivo*

anterior somente pode ser gerado se os parâmetros ChunkArrivalsFile ou ChunkDelaysFile são dados.

- *Out_of_sequence.out: uma linha para cada par, incluindo a fonte, especificando quantos dos chunks que receberam estavam "fora da seqüência". Um chunk é considerado fora da sequência se não chega imediatamente após do chunk com o número de identificação de um a menos do que o seu.*
- *Total_Chunk_Losses_per_peer.out: u*ma linha por cada par, especificando quantos *chunks* dos *NumChunks* não conseguiu receber.
- *Throughput_peer.out: uma linha por cada enlace no overlay, especificando o par fonte, o par de destino e o número de chunks que foram enviados nesse enlace.*
- *Upload_total_peer.out: uma linha por cada par, especificando o número total de chunks que cada par enviou para os seus vizinhos.*
- *DelayHistograms.out: este arquivo fornece o estado final dos contadores de atraso dos chunks (ver o parâmetro ChunkDelayCounters acima) para cada par. A primeira linha do arquivo dá o número de contadores em cada par, o número de pares, e a largura de cada barra do histograma em segundos. O restante do arquivo é formado por duas linhas por cada par: a primeira linha é o identificador do par, a sua classe e quantos chunks ele perdeu e a segunda linha é a de todos os contadores atuais.*

Informação adicional sobre o simulador P2PTVSim pode ser encontrada no site http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim.

O arquivo script com o as configurações do P2PTVSim é mostrado a seguir:

```
NUM_SEEDS=100              # number of random graphs to generate
for the test. Note that MAX values will be averaged amond runs
with different seeds
SEEDs=`seq 1 1 $NUM_SEEDS`
#SEEDs=1

NUM_PEERSs=1000          # number of peers, including the source
DEGREEs=20

MEANBWs=1.0             # mean bandwidth. 1:critical regime;
<1:overload regime >1:undeload regime
PEERSCLASS1s=0        # ratio of peers in the first class
MEANCLASS1s=1.0 # BW of peers in the first class.
PEERSCLASS2s=1
MEANCLASS2s=1.0
VARIANCERATEs=0
NUM_CHUNKSs=1000         # number of chunks
```

```
FIRST_CHUNK=0.001 #a small initial transient
LAST_CHUNK=0.9
SCHED_ORDERs="C_P"
CSCHEDs="LUc" # chunk scheduling alogorithms used
PSCHEDs="RUp" # peer scheduling alogorithms used
PSNR_EVERY=10
#PSNR_CODER='-f h264 -vcodec libx264 -vpre
PSNR_CODER='$SVCENCODER -pf ../SVC_Akiyo/Akiyo.cfg '
PLAYOUT_DEADLINEs="12 14 16 18 22 26 30"
```

I.3 - CODIFICADOR DE VÍDEO ESCALÁVEL JSVM

O JSVM (*Joint Scalable Video Model*) é um software de referência para codificação de vídeo escalável projetado pela JVT (*Joint Video Team*), pela ISO/IEC MPEG (Moving Pictures Experts Group) e pela ITU-T *Video Coding Experts Group* (VCEG), disponível em [42].

Ele é escrito em C++ e seu código fonte é aberto, além disso, no pacote de software do JSVM podem ser encontradas bibliotecas e arquivos executáveis.

As bibliotecas fornecidas pelo JSVM são:

- *H264AVCCommonLibStatic*: fornece classes que são usadas tanto pelo codificador, quanto pelo decodificador.
- *H264AVCEncoderLibStatic*: fornece classes que são usadas somente pelo codificador. Isso inclui as classes para estimativa de movimento, modo de decisão, e codificação de entropia.
- *H264AVCDecoderLibStatic*: possui classes que são usadas somente pelo decodificador. Isso inclui as classes para decodificação de entropia e análise de bit-stream.
- *AvcRewriterLibStatic*: compartilha os mesmo arquivos contidos em H264AVCDecoderLibStatic, mas compilado com um compilador definido, SHARP_AVC_REWRITE_OUTPUT: Ele fornece bibliotecas que são usadas somente pelo AvcRewriter.
- *H264AVCVideoIoLibStatic*: fornece classes para leitura e escrita nas unidades NAL no formato de *byte-stream*, bem como classes para leitura e escrita de dados brutos de vídeo.

Os arquivos executáveis fornecidos pelo JSVM são:

- *DownConvertStatic*: pode ser usado para aumentar ou diminuir a amostragem da sequência de vídeo.
- *H264AVCEncoderLibTestStatic*: pode ser usado para gerar camadas únicas AVC

ou *bitstream* SVC.

- *H264AVCDecoderLibTestStatic*: pode ser usado para decodificar AVC ou *bitstream* SVC e reconstruir a sequência bruta do vídeo.
- *AVCRewriter*: pode ser usado para reescrever um bit-stream SVC para AVC quando o avc_rewrite_flag estiver presente na camada de reforço.
- *BitStreamExtractorStatic*: pode ser usado para extração de sub-bitstream com uma baixa resolução temporal/espacial e/ou taxas de bits do bit-stream escalável global.
- *QualityLevelAssignerStatic*: pode ser usado para geração de bit-stream com informações adicionais de qualidade na camada dado um determinado bit-stream escalável. Além disso, as informações adicionais de qualidade tanto de entrada quanto de saída são idênticas.
- *MCTFPreProcessor*: esta ferramenta é utilizada para pré filtragem das sequências de imagens.
- *PSNRStatic*: pode ser usado para mensurar o PSNR entre duas sequências brutas de vídeo. Além disso, pode ser usado para mensurar o fluxo de bits de um dado *bitstream*.
- *FixedQPEncoderStatic*: pode ser usado controlando a codificação e ajustar o fluxo de bits do *bitstream* gerado.
- *SIPAnalyser*: pode ser usado para fazer a predição seletiva entre as camadas.

YUVCompareStatic: pode ser usado para encontrar incompatibilidade na codificação/decodificação.

A seguir, é mostrado o arquivo de configuração de extensão do JSVM.

```
# Scalable H.264/AVC Extension Configuration File

#============================== GENERAL ==============================
OutputFile              Tennis-Table.264   # Bitstream file
FrameRate               30          # Maximum frame rate [Hz]
FramesToBeEncoded       300        # Number of frames (at input frame
rate)
MaxDelay                1200.0      # Maximum structural delay [ms]
NonRequiredEnable       0           # NonRequiredSEI enable (0:disable,
1:enable)
CgsSnrRefinement        1           # SNR refinement as 1: MGS; 0: CGS
EncodeKeyPictures       1           # Key pics at T=0 (0:none, 1:MGS,
2:all)
MGSControl              1   # ME/MC for non-key pictures in MGS layers
                            # (0:std, 1:ME with EL, 2:ME+MC with EL)



GOPSize                 16          # GOP Size (at maximum frame rate)
```

```
IntraPeriod             16          # Intra Period
NumberReferenceFrames   1           # Number of reference pictures
BaseLayerMode           1           # Base layer mode (0,1:AVC
compatible, 2:AVC w subseq SEI)

#==================== MOTION SEARCH =========================
SearchMode              4                       # Search mode
(0:BlockSearch, 4:FastSearch)
SearchFuncFullPel       0                       # Search function full
pel
                                                #   (0:SAD, 1:SSE,
2:HADAMARD, 3:SAD-YUV)
SearchFuncSubPel        0                       # Search function sub
pel
                                                #   (0:SAD, 1:SSE,
2:HADAMARD)
SearchRange             16                      # Search range (Full
Pel)
BiPredIter              2                       # Max iterations for
bi-pred search
IterSearchRange         2                       # Search range for
iterations (0: normal)

#====================== LOOP FILTER ==========================
LoopFilterDisable       0                       # Loop filter idc (0:
on, 1: off, 2:
                                                #   on except for slice
boundaries)
LoopFilterAlphaC0Offset 0                       # AlphaOffset(-6..+6):
valid range
LoopFilterBetaOffset    0                       # BetaOffset (-6..+6):
valid range

#======================== LAYER DEFINITION ====================
NumLayers            6                      # Number of layers
LayerCfg             /Tennis/QCIFlayer0.cfg # Layer configuration file
LayerCfg             /Tennis/QCIFlayer1.cfg # Layer configuration file
LayerCfg             /Tennis/QCIFlayer2.cfg # Layer configuration file
LayerCfg             /Tennis/CIFlayer3.cfg  # Layer configuration file
LayerCfg             /Tennis/CIFlayer4.cfg  # Layer configuration file
LayerCfg             /Tennis/CIFlayer5.cfg  # Layer configuration file

PreAndSuffixUnitEnable  1                       # Add prefix and suffix
unit (0: off, 1: on) shall always be on in SVC contexts (i.e. when
there are FGS/CGS/spatial enhancement layers)
MMCOBaseEnable          1                       # MMCO for base
representation (0: off, 1: on)
TLNestingFlag           1                       # Sets the temporal
level nesting flag (0: off, 1: on)
TL0DepRepIdxSeiEnable   1                       # Sends dependency
representation index sei (0: off, 1: on)

#========================== HRD ===================================
EnableNalHRD            0
EnableVclHRD            0
```

Outro arquivo importante é o arquivo de configuração de camadas usado pelo JSVM.

```
# Layer Configuration File

#========================== INPUT/OUTPUT ==========================
SourceWidth     176                     # Input  frame width
SourceHeight    144                     # Input  frame height
FrameRateIn     30                      # Input  frame rate [Hz]
FrameRateOut    30                      # Output frame rate [Hz]
InputFile       /Tennis/table_qcif.yuv  # Input  file
#ReconFile       rec_layer0.yuv         # Reconstructed file
SymbolMode      0                       # 0=CAVLC, 1=CABAC

#============================== CODING ==============================
MaxDeltaQP      0                       # Max. absolute delta QP
QP              34.0                    # Quantization parameters

#============================== CONTROL ==============================
MeQP0     34.00  # QP for motion estimation/mode decision (stage 0)
MeQP1     34.00  # QP for motion estimation / mode decision (stage 1)
MeQP2     34.00  # QP for motion estimation / mode decision (stage 2)
MeQP3     34.00  # QP for motion estimation / mode decision (stage 3)
MeQP4     34.00  # QP for motion estimation / mode decision (stage 4)
MeQP5     34.00  # QP for motion estimation / mode decision (stage 5)

InterLayerPred  0   # Inter-layer Prediction (0:no, 1:yes, 2:adaptive)
```

I.4 - MATLAB

MATLAB é uma linguagem de alto nível e um ambiente interativo de computação numérica, visualização e programação. O uso do MATLAB facilita a análise de dados, o desenvolvimento de algoritmos e a criação de modelos e aplicações. A linguagem, ferramentas e funções matemáticas incorporadas permitem encontrar uma solução mais rápida que com o uso de planilhas de cálculo ou de linguagens de programação tradicionais, como C++ ou Java.

Particularmente, no desenvolvimento deste trabalho foram fundamentais duas ferramentas do MATLAB: anovan (para o ANOVA, *Analysis of Variance*) e nlintool (para as regressões não lineares).

A ferramenta anovan (*y*, *group*) faz uma análise *multiway* da variância (*n-way* ANOVA) para testar os efeitos de múltiplos fatores na média no vetor de resultados *y*. O teste feito compara a variância explicada por fatores à esquerda sobre a variância que não pode ser explicada. Os fatores e os níveis de fatores da observação em *y* são especificados pelo *array* de células *group*. Cada uma das células do no array de células *group* contém uma lista de níveis de fatores que identificam as observações em *y* com respeito a um dos

fatores. Informações e documentação adicional podem ser encontradas em http://www.mathworks.com/help/stats/anovan.html.

O nlintool é a ferramenta para fazer regressões interativas não lineares. É a interface gráfica de usuário (GUI) da função nlinfit (`X`, `Y`, *`modelfun`*, *`beta0`*) que dá como resultado um vetor com os coeficientes estimados para uma regressão não linear das respostas em Y usando os preditores contidos em X usando o modelo especificado em *modelfun*. Os coeficientes são estimados usando a estimação iterativa dos mínimos quadrados, com valores iniciais especificados por *beta0*. Informações e documentação adicional podem ser encontradas em http://www.mathworks.com/help/stats/nlintool.html.

I.5 - VARIÁVEIS A SETAR

Os experimentos foram feitos em um computador com dois sistemas operativos, uma distribuição Linux, Ubuntu 10.04, para gerar os dados usando o simulador P2PTVSim e uma distribuição Windows XP para rodar o MATLAB e fazer o processamento dos dados.

Para rodar o simulador, na pasta onde está o diretorio raíz do simulador, tem que ser setadas algumas variáveis para criar corretamente os *paths* que o simulador usará. Estes caminhos são:

export PERL5LIB=$PWD/:$PERL5LIB
export FFDIR=$PWD/ffmpeg
export FFMPEG=$FFDIR/ffmpeg
export PSNR_TOOLS=$PWD/psnr-tools
export PSNR_TMPPREFIX=/dev/shm/psnr/
export JSVMDIR=$PWD/jsvm/bin
export SVCENCODER=$JSVMDIR/./H264AVCEncoderLibTestStatic
export PSNR_TOOLS=$PWD/psnr-tools
export RAW_VIDEO=$PWD/akiyo_cif.yuv
export PSNR_TMPPREFIX=/dev/shm/psnr/
export SVCplayer=/home/valmiro/src_1.12/Mplayer/mplayer

I.6 - RESULTADOS

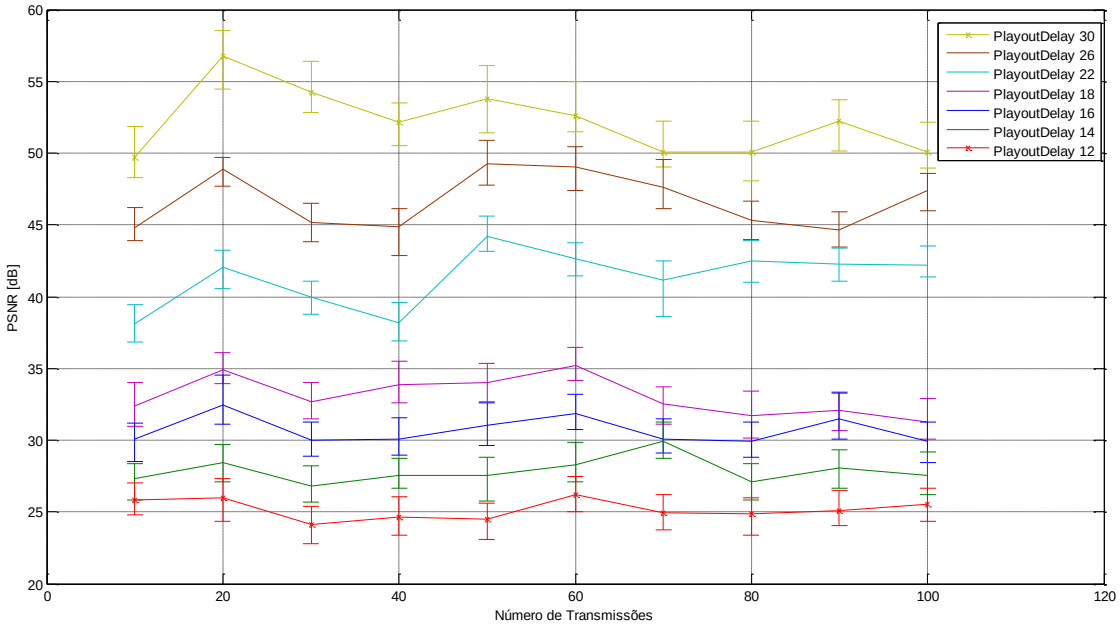

Figura 1. PSNR vídeo H.264/AVC.

Valor do PSNR para os vídeos codificados no formato H.264/AVC com os diferentes valores do parâmetro Playout_Delay (PD).

PD30	PD26	PD22	PD18	PD16	PD14	PD12
49.73	44.78	38.11	32.38	30.05	27.33	25.85
56.75	48.91	42.01	34.9	32.44	28.47	26.02
54.27	45.2	39.97	32.64	30.02	26.83	24.14
52.13	44.86	38.19	33.86	30.05	27.54	24.62
53.8	49.25	44.23	33.98	31.02	27.52	24.53
52.61	49.03	42.62	35.17	31.89	28.26	26.19
50.09	47.64	41.13	32.51	30.06	29.93	24.93
50.05	45.33	42.47	31.73	29.96	27.12	24.86
52.23	44.68	42.25	32.11	31.47	28.07	25.13
50.11	47.42	42.19	31.25	29.91	27.51	25.51

Desvio padrão de acordo ao parâmetro do simulador Playout_Delay (PD).

ebPD30l	ebPD30u	ebPD26l	ebPD26u	ebPD22l	ebPD22u	ebPD18l
1.47	2.1	0.9	1.4	1.25	1.33	1.43
2.3	1.8	1.24	0.8	1.43	1.19	0.97
1.48	2.11	1.36	1.33	1.16	1.12	1.12
1.63	1.35	2.01	1.26	1.31	1.43	1.28
2.36	2.27	1.48	1.62	1.07	1.42	1.33
1.1	2.38	1.65	1.45	1.14	1.17	1.04
1.04	2.15	1.48	1.12	2.48	1.36	1.37
1.96	2.18	1.35	1.35	1.43	1.14	1.56
2.1	1.5	1.25	1.22	1.19	0.9	1.47
1.15	2.05	1.43	1.18	0.78	1.34	1.18
ebPD18u	ebPD16l	ebPD16u	ebPD14l	ebPD14u	ebPD12l	ebPD12u
1.65	1.52	1.15	1.47	1.03	1.02	1.16
1.21	1.31	2.08	1.35	1.21	1.68	1.32
1.39	1.14	1.23	1.16	1.36	1.32	1.24
1.66	1.11	1.51	0.89	1.19	1.24	1.43
1.34	1.42	1.55	1.73	1.26	1.47	1.11
1.28	1.16	1.3	1.13	1.59	1.2	1.27
1.24	0.92	1.41	1.22	1.33	1.19	1.25
1.72	1.17	1.27	1.27	1.21	1.51	1.13
1.14	1.37	1.85	1.41	1.25	1.09	1.39
1.62	1.44	1.35	1.32	1.67	1.16	1.18

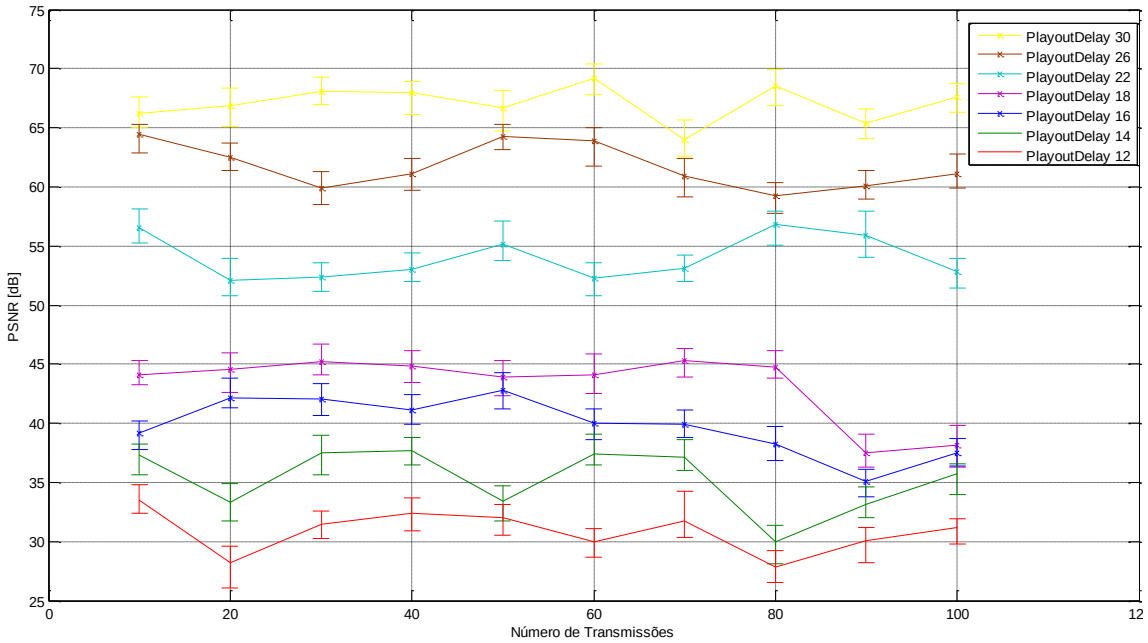

Figura 2: PSNR vídeo H.264/SVC

Valor do PSNR para os vídeos codificados no formato H.264/SVC com os diferentes valores de Playout_Delay (PD).

PD30	PD26	PD22	PD18	PD16	PD14	PD12
66.21	64.42	56.53	44.09	39.18	37.31	33.54
66.83	62.49	52.13	44.52	42.15	33.28	28.16
68.05	59.9	52.37	45.18	42.02	37.53	31.46
67.96	61.15	53.01	44.83	41.17	37.67	32.37
66.72	64.31	55.18	43.91	42.83	33.45	32.05
69.17	63.87	52.24	44.07	40.05	37.42	29.97
64.03	60.93	53.15	45.32	39.92	37.14	31.78
68.52	59.24	56.83	44.76	38.21	29.95	27.83
65.41	60.08	55.93	37.52	35.04	33.16	30.03
67.61	61.12	52.88	38.16	37.53	35.71	31.14

Desvio padrão dos valores de PSNR para cada valor de Playout_Delay (PD).

ebPD30l	ebPD30u	ebPD26l	ebPD26u	ebPD22l	ebPD22u	ebPD18l
1.18	1.4	1.53	0.87	1.28	1.64	0.82
1.73	1.5	1.07	1.21	1.36	1.8	1.94
1.12	1.28	1.38	1.36	1.18	1.21	1.12
1.81	0.96	1.42	1.25	1.03	1.43	1.42
1.96	1.46	1.19	1.02	1.4	1.95	1.6
1.33	1.25	2.15	1.14	1.47	1.36	1.56
1.46	1.63	1.77	1.48	1.13	1.12	1.37
1.67	1.44	1.49	1.13	1.75	1.15	0.91
1.35	1.18	1.1	1.29	1.86	2.01	1.25
1.29	1.13	1.22	1.68	1.41	1.11	1.88
ebPD18u	ebPD16l	ebPD16u	ebPD14l	ebPD14u	ebPD12l	ebPD12u
1.23	1.4	1.02	1.71	0.95	1.15	1.25
1.41	0.8	1.68	1.52	1.66	2.08	1.43
1.55	1.33	1.32	1.85	1.47	1.23	1.16
1.3	1.26	1.24	1.21	1.11	1.51	1.31
1.38	1.62	1.47	1.74	1.28	1.55	1.07
1.79	1.45	1.2	0.96	1.68	1.3	1.14
1.01	1.12	1.19	1.14	1.5	1.41	2.48
1.34	1.35	1.51	1.82	1.39	1.27	1.43
1.52	1.22	1.09	1.13	1.45	1.85	1.19
1.65	1.18	1.16	1.74	0.87	1.35	0.78

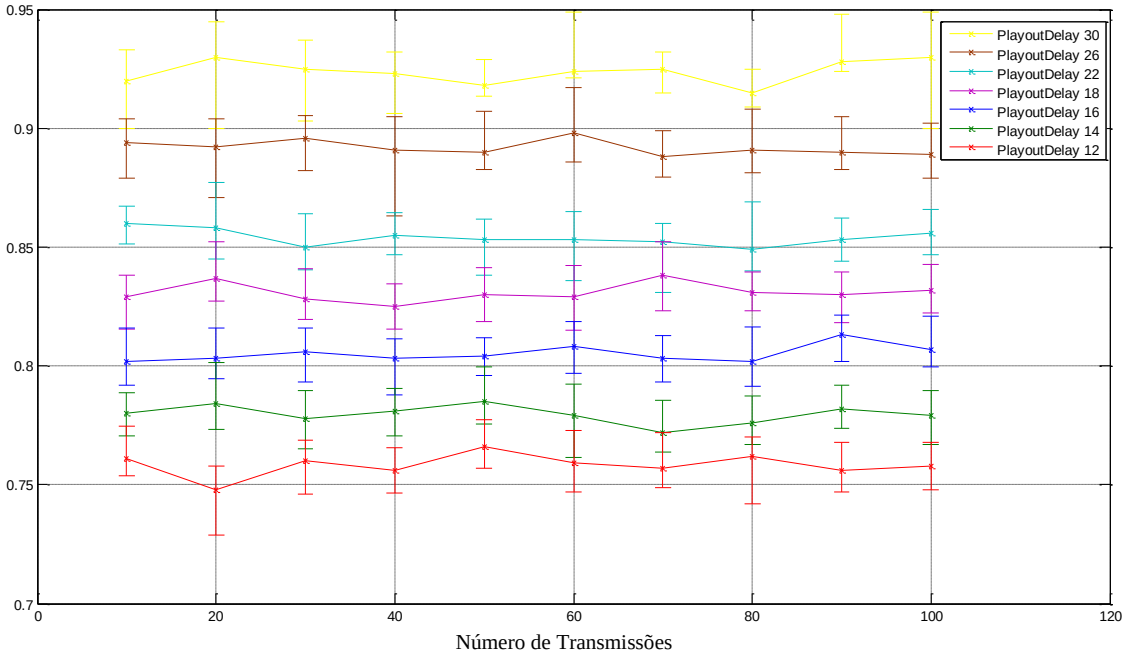

Figura 3: SSIM vídeo H.264/AVC.

Valor do SSIM para os vídeos codificados no formato H.264/AVC com os diferentes valores de Playout_Delay (PD).

PD30	PD26	PD22	PD18	PD156	PD14	PD12
0.92	0.894	0.86	0.829	0.802	0.78	0.761
0.93	0.892	0.858	0.837	0.803	0.784	0.748
0.925	0.85	0.85	0.828	0.806	0.778	0.76
0.923	0.896	0.855	0.825	0.803	0.781	0.756
0.918	0.891	0.853	0.83	0.804	0.785	0.766
0.924	0.89	0.853	0.829	0.808	0.779	0.759
0.925	0.898	0.852	0.838	0.803	0.772	0.757
0.915	0.888	0.849	0.831	0.802	0.776	0.762
0.928	0.891	0.853	0.83	0.813	0.782	0.756
0.93	0.89	0.856	0.832	0.807	0.779	0.758

Desvio padrão dos valores de PSNR para cada valor de Playout_Delay (PD).

ebPD30l	ebPD30u	ebPD26l	ebPD26u	ebPD22l	ebPD22u	ebPD18l
0.02	0.013	0.015	0.01	0.0085	0.0074	0.0135
0.03	0.015	0.021	0.012	0.013	0.019	0.0097
0.022	0.012	0.014	0.0094	0.0098	0.014	0.00864
0.017	0.009	0.028	0.014	0.0083	0.0095	0.00945
0.0045	0.011	0.0075	0.017	0.015	0.0089	0.0115
0.0028	0.025	0.012	0.019	0.017	0.0121	0.014
0.01	0.007	0.0084	0.011	0.021	0.008	0.015
0.006	0.01	0.0096	0.017	0.0092	0.02	0.0079
0.0042	0.02	0.0075	0.015	0.0088	0.009	0.012
0.03	0.019	0.01	0.013	0.0091	0.01	0.0098
ebPD18u	ebPD16l	ebPD16u	ebPD14l	ebPD14u	ebPD12l	ebPD12u
0.0091	0.0101	0.014	0.0094	0.00868	0.0074	0.0135
0.0152	0.0082	0.0131	0.0108	0.0172	0.019	0.0097
0.0128	0.0126	0.00977	0.013	0.0118	0.014	0.00864
0.00953	0.0154	0.00861	0.0106	0.0095	0.0095	0.00945
0.0112	0.00825	0.00795	0.00937	0.0148	0.0089	0.0115
0.0133	0.0113	0.0108	0.0174	0.0133	0.0121	0.014
0.0141	0.00983	0.00962	0.00814	0.0136	0.008	0.015
0.00847	0.0108	0.0145	0.009	0.0112	0.02	0.0079
0.00964	0.0111	0.0084	0.0083	0.0097	0.009	0.012
0.0105	0.00763	0.0138	0.0123	0.0105	0.01	0.0098

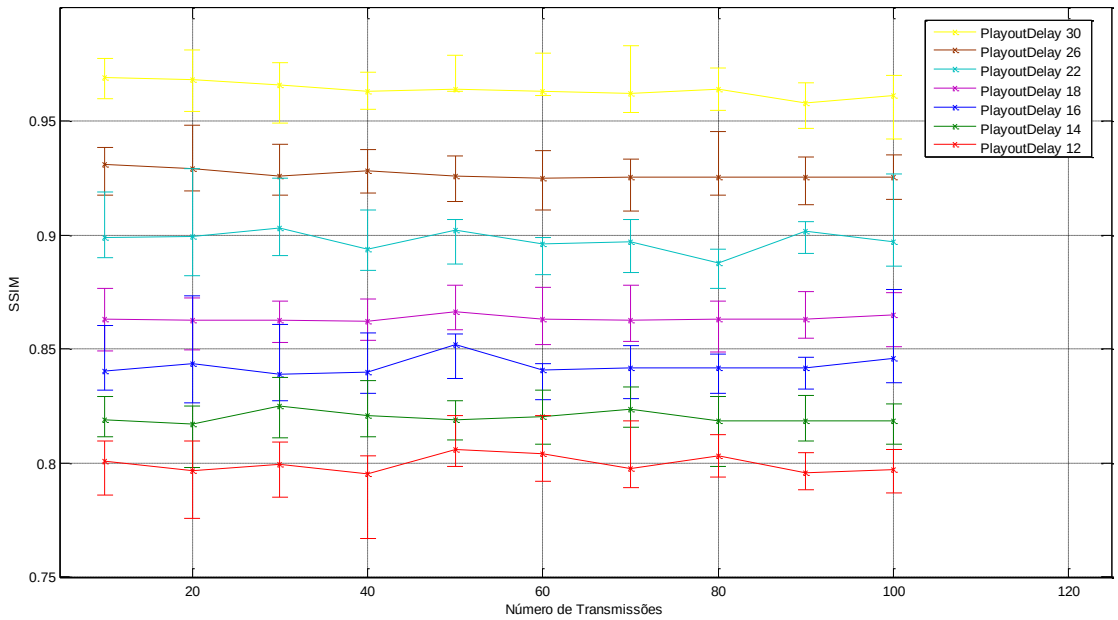

Figura 4: SSIM vídeo H.264/SVC

Valor do SSIM para os vídeos codificados no formato H.264/SVC com os diferentes valores de Playout_Delay (PD).

PD30	PD26	PD22	PD18	PD16	PD14	PD12
0.969	0.931	0.899	0.863	0.8405	0.819	0.801
0.968	0.929	0.8993	0.8628	0.8435	0.817	0.7968
0.966	0.926	0.903	0.8625	0.839	0.825	0.7993
0.963	0.928	0.894	0.8624	0.84	0.821	0.7951
0.964	0.926	0.9023	0.8665	0.858	0.819	0.806
0.963	0.925	0.896	0.863	0.841	0.8205	0.804
0.962	0.9255	0.897	0.8628	0.8417	0.8237	0.7976
0.964	0.9252	0.888	0.8632	0.8416	0.81843	0.8033
0.958	0.9254	0.9015	0.863	0.842	0.81856	0.7957
0.961	0.9253	0.897	0.865	0.846	0.81852	0.7971

Desvio padrão dos valores de PSNR para cada valor de Playout_Delay (PD).

ebPD30l	ebPD30u	ebPD26l	ebPD26u	ebPD22l	ebPD22u	ebPD18l
0.0088	0.0082	0.0135	0.0074	0.0087	0.019	0.014
0.016	0.0122	0.0097	0.019	0.0172	0.026	0.0131
0.017	0.0093	0.00864	0.014	0.0118	0.023	0.0098
0.0081	0.0084	0.00945	0.0095	0.0092	0.017	0.00861
0.0098	0.018	0.0115	0.0089	0.014	0.0045	0.0082
0.0013	0.015	0.014	0.0121	0.0138	0.0033	0.0108
0.0079	0.018	0.015	0.008	0.0131	0.0096	0.0097
0.0091	0.0089	0.0079	0.02	0.011	0.0063	0.0139
0.0131	0.0083	0.012	0.009	0.0093	0.0047	0.0083
0.017	0.0093	0.0098	0.01	0.0102	0.024	0.0135
ebPD18u	ebPD16l	ebPD16u	ebPD14l	ebPD14u	ebPD12l	ebPD12u
0.0135	0.0085	0.02	0.0073	0.0099	0.015	0.0088
0.0097	0.0172	0.03	0.021	0.0084	0.018	0.014
0.00859	0.0118	0.0195	0.015	0.0118	0.014	0.0091
0.0088	0.0091	0.017	0.0093	0.015	0.027	0.0086
0.0122	0.0147	0.0045	0.0091	0.0081	0.0073	0.013
0.0137	0.0129	0.0028	0.012	0.0113	0.012	0.02
0.015	0.014	0.011	0.077	0.0101	0.0087	0.018
0.0082	0.0108	0.006	0.021	0.0108	0.0096	0.0089
0.012	0.0094	0.0042	0.009	0.012	0.0074	0.0091
0.0097	0.011	0.03	0.012	0.008	0.011	0.0087

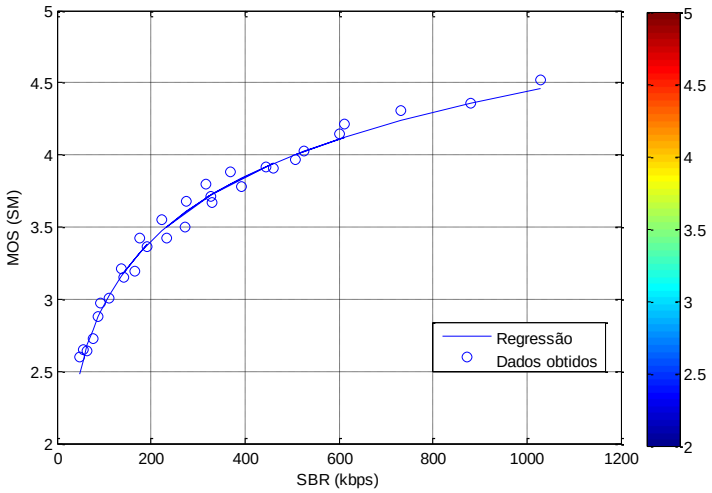

Figura 5: MOS em função da SBR.

Valores do MOS com a variação do SBR.

SBR(kbps)	MOS
47	2.6
54.8	2.65
65.2	2.64
76.1	2.73
87.8	2.88
92.5	2.97
111	3.01
136.8	3.21
164.7	3.19
192.5	3.36
142.6	3.15
175.1	3.42
221.8	3.55
275.2	3.68
328.3	3.71
232.3	3.42
272.4	3.5
329.5	3.67
393.6	3.78
460.1	3.91
317.4	3.8
370	3.88
444	3.92
526.6	4.03
611.5	4.21
506.4	3.97
600.2	4.15
730.5	4.31
880.6	4.36
1030	4.52

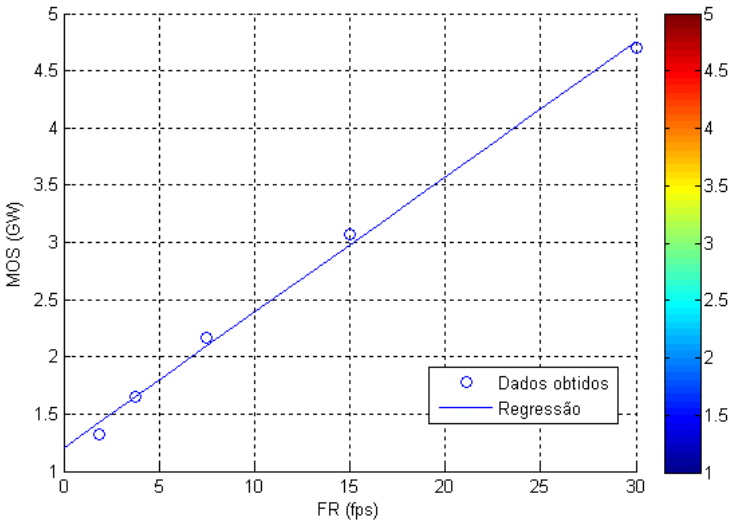

Figura 6: MOS em função da FR.

Valores do MOS com a variação do FR.

FR	MOS
1.875	1.36
3.75	1.71
7.5	2.2
15	3.04
30	4.62

Observação: os valores de FR foram limitados a cinco valores só: 1.875, 3.75, 7.5, 15 e 30 fps, por causa da codificação e dos arquivos de configuração usados para codificar o vídeo.

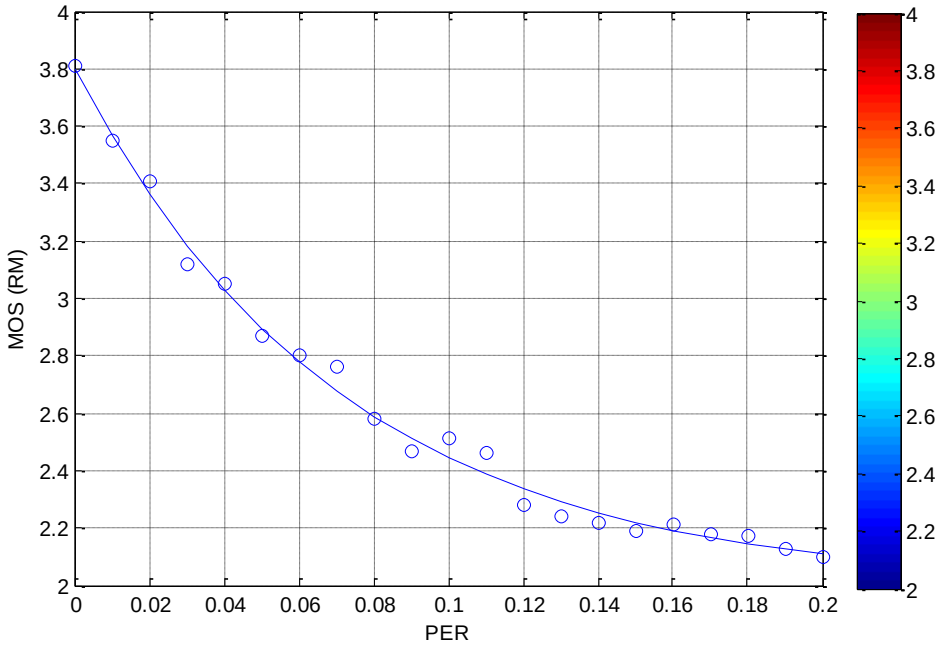

Figura 7: MOS em função da PER.

Valores do MOS com a variação do PER.

PER	MOS
0	3.9
0.01	3.85
0.02	3.81
0.03	3.65
0.04	3.6
0.05	3.48
0.06	3.37
0.07	3.19
0.08	3.1
0.09	3.03
0.1	2.97
0.11	2.86
0.12	2.8
0.13	2.74
0.14	2.72
0.15	2.64
0.16	2.6
0.17	2.5
0.18	2.35
0.19	2.23
0.2	2.01

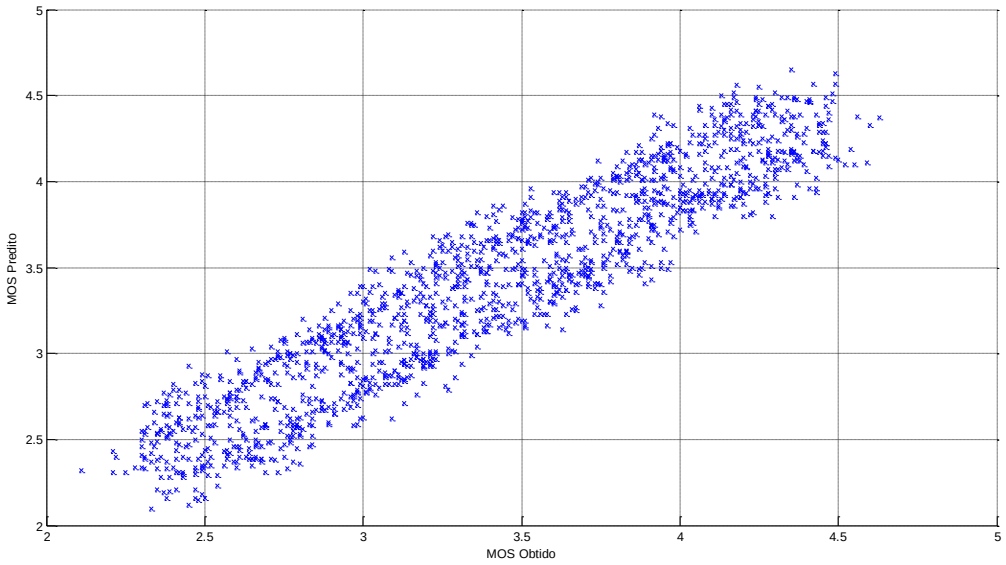

Figura 8: MOS predito vs MOS obtido.

MOS obtido (MOSob) VS MOS predito (MOSpr).

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.27	3.879	2.60	2.371	4.13	4.323
2.54	2.233	2.90	3.073	3.45	3.144
2.38	2.711	4.37	4.134	3.63	3.871
2.60	2.966	2.44	2.768	4.01	3.877
3.86	3.909	3.54	3.820	2.76	3.012
3.70	3.446	4.26	4.316	4.63	4.371
3.03	3.226	3.63	3.792	2.83	2.463
3.86	4.102	2.60	2.342	3.88	3.947
3.56	3.235	4.41	4.476	4.25	4.550
3.30	3.393	2.79	3.110	2.49	2.336
3.58	3.161	3.87	3.464	3.65	3.770
3.83	4.170	3.61	3.365	3.96	3.867
4.02	3.941	3.28	3.621	3.19	2.934
3.73	3.993	2.69	2.582	3.86	4.043
2.74	3.077	3.22	2.966	4.00	3.720
2.78	2.373	3.30	3.189	3.26	3.471
2.92	3.291	4.26	4.125	4.27	3.918
3.75	3.397	3.93	3.688	2.79	2.452
2.58	2.681	2.84	2.664	2.49	2.877
2.47	2.571	3.64	3.653	2.63	2.401
4.22	3.808	4.34	4.408	3.80	3.525
3.43	3.656	4.06	3.948	4.25	4.328
3.48	3.178				

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.46	4.218	2.58	2.844	3.13	3.194
4.02	3.892	3.36	3.542	3.23	3.473
2.93	2.942	4.24	4.484	2.25	2.305
3.83	3.574	3.38	3.352	3.76	3.865
3.22	2.932	2.57	2.441	3.60	3.387
3.49	3.732	3.15	3.528	3.51	3.818
3.95	3.874	4.33	4.355	3.13	2.964
3.85	3.920	2.51	2.704	4.11	4.326
3.74	3.860	3.16	3.308	4.25	4.235
3.38	3.152	3.48	3.293	4.43	3.955
4.42	4.575	2.93	3.170	2.33	2.483
2.76	2.641	3.96	4.340	3.30	3.445
2.77	2.498	3.87	4.131	3.80	4.027
3.38	3.617	3.39	3.419	3.88	3.500
3.10	3.215	2.66	2.551	3.94	3.812
4.19	4.281	4.36	3.914	3.64	3.481
4.39	4.280	2.98	3.193	3.13	3.191
2.31	2.392	3.39	3.800	3.81	3.591
3.22	3.278	3.54	3.630	3.52	3.660
3.27	3.027	4.47	4.093	4.22	4.159
3.67	3.868	4.11	3.805	3.51	3.189
3.37	3.468	2.97	2.910	2.99	3.391
2.35	2.486	2.91	2.654	2.78	2.554
2.93	3.138	3.55	3.420	2.49	2.640
4.19	3.911	3.93	3.937	4.06	4.422
3.31	3.170	3.19	2.954	2.73	3.012
3.07	2.825	2.35	2.533	2.36	2.551
2.41	2.597	3.35	3.654	3.71	3.936
3.67	3.896	2.69	2.315	2.51	2.369
4.00	3.849	2.77	2.478	4.23	4.390
3.23	3.407	2.66	2.502	3.36	3.252
3.96	3.705	2.60	2.458	3.26	2.807
3.51	3.787	2.89	2.688	3.84	4.058
3.86	3.918	3.94	4.379	2.66	2.888
2.99	3.229	2.79	2.621	3.71	3.500
3.95	3.877	3.26	3.682	4.47	4.151
2.38	2.375	2.73	2.716	2.88	3.207
3.90	3.667	4.35	4.379	2.75	3.088
2.58	2.774	4.05	3.707	3.23	2.971
2.98	2.768	2.80	2.570	3.91	3.719
2.74	2.448	3.39	3.469	4.15	4.445
2.38	2.700	2.68	2.390	3.39	3.662
3.05	3.168	3.14	3.285	3.63	3.478
2.88	2.924	3.09	2.618	3.91	3.428
2.87	2.727	4.10	3.900	2.57	3.007

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.08	4.275	4.36	4.122	3.02	3.253
3.06	3.335	3.36	3.824	3.57	3.731
4.36	4.169	3.17	2.762	2.37	2.189
4.37	3.978	2.95	3.070	3.41	3.610
4.42	4.184	3.42	3.398	4.35	4.161
3.52	3.735	4.31	3.907	3.96	3.486
4.44	4.288	4.28	4.071	2.11	2.324
2.36	2.564	3.29	2.857	3.94	3.527
3.00	2.634	2.45	2.925	2.41	2.709
2.41	2.477	3.25	3.301	3.34	3.321
3.39	3.400	4.45	4.339	4.19	4.342
2.96	3.347	3.61	3.584	2.63	2.840
3.74	3.408	2.75	3.078	2.42	2.752
4.19	4.387	2.37	2.772	3.32	3.314
3.71	3.414	3.67	3.363	3.07	3.270
3.57	3.234	3.66	3.259	3.02	3.359
2.71	2.743	3.21	2.927	4.08	3.879
4.59	4.109	4.29	3.802	3.09	3.560
3.92	4.266	2.53	2.484	3.14	3.119
2.65	3.029	3.33	3.621	3.23	3.535
3.68	3.613	3.11	3.228	3.09	3.478
3.95	4.020	2.57	2.709	4.18	4.564
3.14	3.466	3.35	3.748	2.68	2.391
3.75	4.015	3.31	2.940	4.45	4.154
4.16	4.315	3.53	3.826	2.93	3.144
3.79	4.054	3.94	3.495	4.06	4.164
2.82	2.955	3.46	3.195	3.10	2.898
4.32	4.488	3.53	3.907	2.87	3.088
4.24	4.407	2.95	3.126	2.40	2.788
2.31	2.333	2.47	2.338	2.40	2.549
3.81	4.031	3.18	3.304	3.31	3.681
4.52	4.098	4.24	4.379	4.41	4.464
2.80	2.528	3.19	2.833	4.36	4.477
3.56	3.693	4.12	3.887	3.32	3.100
2.33	2.369	3.23	3.515	2.50	2.157
4.50	4.132	2.85	2.866	3.21	3.566
3.27	3.598	2.76	2.577	3.52	3.348
4.14	4.361	4.14	4.176	2.89	2.598
2.43	2.313	3.72	3.467	2.47	2.163
3.00	3.302	4.45	4.180	3.04	3.483
4.28	3.933	4.03	4.067	2.53	2.592
2.84	2.815	2.22	2.402	2.55	2.849
3.51	3.164	3.08	3.085	3.51	3.874
3.46	3.164	2.59	2.461	4.21	4.094
2.88	3.063	3.21	3.194	2.81	3.062

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.41	3.963	3.40	3.503	2.78	2.567
3.64	3.938	2.79	2.571	2.36	2.396
2.92	3.023	3.97	4.223	4.35	4.184
3.30	3.541	3.71	3.386	4.03	3.834
3.28	3.655	2.60	2.611	3.64	3.365
3.85	3.497	4.46	4.438	3.98	4.333
2.86	3.165	3.34	3.763	4.49	4.630
2.84	3.082	3.53	3.965	4.35	4.305
2.42	2.625	2.45	2.392	2.63	2.498
2.39	2.197	3.07	3.349	2.87	2.685
3.05	2.947	2.88	3.083	2.65	2.354
3.61	3.686	4.25	4.452	4.09	3.930
4.02	3.920	2.50	2.753	3.78	3.455
4.04	3.827	3.05	2.902	2.97	2.761
3.31	3.633	3.95	3.973	4.19	4.260
3.92	3.869	3.98	3.856	3.59	3.658
3.82	4.124	2.78	2.590	2.69	2.945
4.43	4.024	4.37	4.278	4.38	4.148
2.51	2.874	3.09	2.874	3.50	3.615
3.80	3.681	3.40	3.295	2.82	3.013
3.86	4.167	4.45	4.243	4.03	3.796
2.60	2.751	4.30	4.022	3.36	3.509
3.73	3.678	3.34	3.752	3.08	3.491
3.67	3.270	4.03	3.779	4.10	4.426
3.75	3.284	2.45	2.574	2.35	2.211
4.17	3.856	3.24	3.622	3.27	3.687
3.24	3.664	3.60	3.339	3.19	3.328
3.91	4.253	2.37	2.696	2.99	3.124
4.49	4.142	3.35	3.600	2.98	2.783
3.41	3.208	3.01	2.781	3.63	3.839
4.54	4.188	4.39	4.097	3.59	3.300
4.32	4.178	4.31	4.128	2.61	2.556
3.32	3.415	4.04	4.270	4.46	4.492
2.48	2.149	3.67	3.402	2.49	2.843
2.34	2.602	2.76	2.886	2.76	2.992
3.55	3.327	3.89	3.534	4.29	4.446
3.60	3.292	4.24	3.962	2.66	2.616
4.16	4.453	3.19	2.925	4.00	4.180
3.59	3.511	2.46	2.545	4.44	4.344
4.20	3.872	3.47	3.492	3.20	3.284
3.38	3.268	3.62	3.836	3.37	3.333
3.47	3.785	4.19	4.120	2.67	2.482
4.10	3.866	4.36	4.379	3.69	3.884
4.25	4.409	2.61	2.896	3.33	3.764
3.63	3.905	3.22	3.123	4.30	4.055

Mob	Mpr	Mob	Mpr	Mob	Mpr
3.11	3.333	2.99	3.184	4.28	4.435
2.97	2.751	4.02	4.146	3.41	3.862
3.82	3.841	3.20	3.185	3.44	3.578
2.89	2.591	4.00	3.848	4.43	4.391
3.06	3.290	4.20	3.801	3.71	3.443
4.16	3.863	2.45	2.708	3.71	3.450
2.84	3.093	2.78	2.988	3.73	3.382
4.35	4.654	3.95	3.611	4.17	4.494
2.87	2.688	3.63	3.144	2.78	2.864
3.83	3.918	3.51	3.148	2.57	2.389
2.73	2.976	2.76	2.332	4.25	3.957
3.82	3.687	3.29	3.070	4.06	4.271
2.75	2.396	3.87	4.210	2.47	2.639
3.47	3.688	3.42	3.766	2.84	2.474
4.45	4.217	3.44	3.169	4.18	4.046
3.65	3.797	3.74	3.409	3.66	3.463
4.18	4.464	3.41	3.808	2.72	2.971
3.87	3.775	2.52	2.405	3.65	3.435
2.80	2.517	4.21	3.934	3.30	3.570
3.01	3.324	3.64	3.228	2.92	3.075
3.98	3.913	3.11	2.861	3.37	3.547
2.70	2.848	2.50	2.808	3.82	3.553
3.92	3.747	2.61	2.463	4.28	3.927
2.97	2.580	3.31	3.235	3.46	3.660
4.25	4.335	2.79	2.952	3.42	3.680
4.30	4.141	3.41	3.620	4.06	4.219
3.84	3.779	2.74	2.602	2.42	2.472
2.58	2.742	4.15	4.367	2.60	2.751
4.29	4.262	3.83	3.519	2.83	2.585
3.13	2.850	2.65	2.394	4.03	4.311
3.63	3.676	4.18	3.913	4.09	4.207
4.34	4.103	4.14	4.353	2.83	3.136
3.74	4.001	2.32	2.710	2.48	2.393
4.56	4.382	4.04	3.758	2.61	2.790
3.34	3.509	3.34	3.682	3.58	3.776
3.21	3.010	3.24	3.502	3.89	3.410
4.16	4.385	3.62	3.938	2.72	2.380
3.90	3.804	2.77	2.520	3.73	3.949
3.21	2.995	4.27	4.109	2.98	2.948
2.91	2.647	4.24	4.494	4.17	4.519
3.97	3.532	3.84	3.674	3.62	3.885
4.43	3.940	4.13	3.902	3.55	3.817
4.27	4.027	4.01	3.980	4.30	4.391
2.47	2.831	3.43	3.642	3.44	3.856

Mob	Mpr	Mob	Mpr	Mob	Mpr
2.57	2.739	4.30	4.391	3.43	3.146
2.54	2.661	3.44	3.856	2.61	2.400
2.35	2.383	2.57	2.739	3.84	4.092
4.55	4.101	2.54	2.661	2.96	2.835
3.44	3.578	2.35	2.383	4.20	3.944
4.43	4.391	4.55	4.101	4.27	4.386
3.71	3.443	3.13	2.840	4.36	4.493
3.71	3.450	3.11	2.884	3.17	3.452
3.73	3.382	4.00	4.026	3.41	3.669
4.17	4.494	2.91	2.681	3.84	3.477
2.78	2.864	3.59	3.711	3.72	4.015
2.57	2.389	2.91	2.706	2.39	2.282
4.25	3.957	3.56	3.363	3.91	3.655
4.06	4.271	3.23	2.998	2.96	2.826
2.47	2.639	4.44	4.168	3.75	3.735
2.84	2.474	3.55	3.275	4.34	4.070
4.18	4.046	2.99	2.621	3.07	2.894
3.66	3.463	3.97	4.149	3.94	3.891
2.72	2.971	3.23	3.189	4.18	4.321
3.65	3.435	3.39	3.127	3.89	3.814
3.30	3.570	2.75	2.694	2.66	2.457
2.92	3.075	2.97	2.731	2.71	3.038
3.37	3.547	3.46	3.123	2.98	3.224
3.82	3.553	2.54	2.658	2.49	2.532
4.28	3.927	2.55	2.874	3.77	3.554
3.46	3.660	2.31	2.532	2.99	3.158
3.42	3.680	4.11	4.269	4.25	4.350
4.06	4.219	2.49	2.319	2.62	2.839
2.42	2.472	3.91	3.746	3.25	3.598
2.60	2.751	3.69	3.971	2.77	3.004
2.83	2.585	3.63	3.548	2.49	2.181
4.03	4.311	2.87	3.026	3.60	3.732
4.09	4.207	2.67	2.903	3.83	3.605
2.83	3.136	3.31	3.175	3.02	2.831
2.48	2.393	2.36	2.280	3.21	2.995
2.61	2.790	4.35	4.193	2.38	2.637
3.58	3.776	2.42	2.792	2.47	2.214
3.89	3.410	3.70	3.965	4.37	4.154
2.72	2.380	3.84	4.021	2.42	2.601
3.73	3.949	4.30	4.072	3.55	3.368
2.98	2.948	4.40	4.062	4.21	4.337
4.17	4.519	3.11	3.355	4.10	4.025
3.62	3.885	3.91	3.740	2.73	2.308
3.55	3.817	4.16	4.192	3.87	4.054

Mob	Mpr	Mob	Mpr	Mob	Mpr
3.62	3.367	2.79	3.000	3.32	3.325
4.13	4.500	3.20	2.954	3.51	3.472
3.36	3.039	2.43	2.279	4.32	4.078
3.37	3.496	2.93	2.738	2.94	3.111
3.58	3.799	3.42	3.343	2.56	2.749
4.14	4.271	3.13	2.941	3.52	3.771
3.26	3.179	3.64	3.913	2.51	2.453
3.17	2.981	3.46	3.692	2.39	2.758
2.58	2.422	3.99	4.200	2.82	2.937
3.32	3.391	2.56	2.355	2.93	2.951
2.35	2.723	3.23	3.442	3.00	2.846
4.28	4.156	2.58	2.503	4.07	3.855
2.82	3.069	2.81	2.765	2.41	2.388
2.69	2.848	3.64	3.368	3.67	3.252
3.71	3.447	3.90	4.115	2.98	3.031
2.33	2.418	2.75	2.473	4.35	4.330
4.29	4.418	3.21	3.098	2.30	2.497
3.28	3.018	4.05	4.030	3.18	3.350
2.80	2.356	3.03	3.390	3.70	3.467
4.08	4.334	3.22	3.684	3.60	3.660
3.88	4.134	2.95	3.057	2.30	2.408
4.20	4.086	3.47	3.257	3.88	3.928
2.60	2.727	2.43	2.301	3.19	3.413
3.80	3.649	3.30	3.611	4.39	4.230
4.38	4.393	4.03	3.947	4.12	4.287
2.67	2.908	3.66	3.703	4.25	4.178
2.45	2.119	4.16	3.975	3.73	3.415
3.17	3.479	3.73	3.648	2.47	2.297
4.06	4.437	4.22	4.044	2.93	3.092
3.63	3.769	3.49	3.200	3.70	3.909
4.20	4.224	4.49	4.572	2.64	2.389
4.46	4.217	2.98	2.702	3.18	3.445
3.83	4.051	3.01	2.862	4.06	3.910
3.79	3.843	4.37	4.264	3.30	3.406
3.37	3.592	4.47	4.395	2.47	2.315
3.02	3.489	3.42	3.657	2.39	2.528
2.76	2.930	2.62	2.730	3.63	3.402
4.13	3.995	2.34	2.459	3.23	3.083
2.74	2.982	2.84	2.524	3.83	3.599
3.66	3.421	2.81	2.861	3.01	3.290
3.41	3.353	3.31	3.109	3.08	2.869
3.59	3.455	2.38	2.732	3.12	2.950
3.07	2.874	2.70	2.484	3.93	4.164
3.60	3.940	2.33	2.104	3.24	3.445
2.48	2.725	3.21	3.215	4.47	4.361

Mob	Mpr	Mob	Mpr	Mob	Mpr
3.03	3.094	3.15	2.972	2.95	2.709
3.94	4.201	4.11	3.916	3.92	4.012
4.35	4.169	4.12	3.850	2.81	3.202
4.08	3.994	3.08	3.061	3.69	3.914
3.25	3.525	3.59	3.304	2.43	2.403
4.04	4.012	2.88	3.136	3.27	3.319
2.51	2.294	3.50	3.450	4.28	4.168
3.98	3.677	2.87	3.098	2.73	2.987
2.30	2.338	3.79	4.019	3.89	3.960
3.02	2.787	2.35	2.326	2.84	3.012
3.12	3.271	3.18	2.961	3.91	4.150
2.93	3.101	4.29	4.110	3.29	3.392
3.13	2.963	3.95	4.144	4.14	3.979
4.32	4.350	3.45	3.485	3.06	2.946
2.70	2.953	3.97	4.217	2.37	2.648
4.12	3.860	2.70	2.838	3.88	4.010
3.76	3.656	2.72	2.740	3.72	3.544
2.87	2.719	3.60	3.466	3.69	3.460
3.31	3.104	2.31	2.402	2.21	2.427
4.60	4.335	4.23	4.219	3.38	3.135
3.31	3.233	3.39	3.468	2.60	2.788
3.74	3.981	2.67	2.950	4.24	4.018
3.14	3.299	3.97	3.784	2.72	2.683
3.72	3.917	3.16	3.382	4.22	3.996
3.89	4.022	3.46	3.287	3.01	2.770
2.38	2.522	3.61	3.736	2.73	2.723
2.39	2.653	3.78	3.949	3.54	3.413
4.14	3.931	3.97	4.070	3.81	3.645
2.68	2.365	2.96	2.903	4.12	4.078
4.29	4.127	2.90	2.980	2.30	2.554
3.00	2.812	3.00	3.085	3.92	3.723
3.84	3.644	2.79	2.574	2.71	2.818
2.93	2.876	3.63	3.327	2.96	2.691
4.14	4.025	2.77	2.972	3.52	3.377
3.44	3.217	3.92	4.147	3.97	3.764
2.54	2.689	2.96	3.164	2.85	2.737
2.58	2.577	3.61	3.530	3.03	2.862
2.71	2.389	3.28	3.379	3.56	3.324
3.04	2.755	3.95	4.148	2.45	2.622
2.53	2.638	2.64	2.416	3.81	3.761
4.22	3.964	3.64	3.834	3.42	3.256
4.21	4.083	3.53	3.793	3.14	3.153
2.84	2.579	3.33	3.548	2.85	2.941
4.15	4.082	4.08	4.184	3.53	3.679
2.99	2.739	3.86	3.879	3.29	3.202

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.46	4.196	3.55	3.755	3.42	3.217
3.82	3.568	2.85	2.672	3.17	2.983
4.28	4.146	2.68	2.880	3.80	4.034
3.07	3.292	2.84	2.620	2.94	3.047
2.57	2.412	3.97	4.079	3.16	3.002
2.77	2.623	3.05	2.805	3.18	3.231
3.22	3.231	2.95	2.920	4.40	4.310
2.50	2.346	3.04	3.024	3.62	3.428
4.36	4.184	2.45	2.470	3.70	3.557
2.38	2.326	3.39	3.127	2.76	2.515
3.59	3.799	3.22	3.263	3.59	3.646
3.03	2.940	3.76	3.698	2.81	2.731
4.26	4.124	3.37	3.638	4.38	4.121
3.22	3.484	3.83	3.590	3.36	3.108
3.26	3.440	4.24	4.406	3.68	3.409
3.13	2.940	3.19	3.005	3.39	3.203
2.58	2.643	3.76	3.792	3.56	3.676
3.65	3.817	3.81	3.698	3.50	3.760
3.19	2.974	3.10	2.900	3.41	3.309
3.24	3.156	3.83	3.589	3.96	4.208
3.96	3.956	2.86	3.115	2.47	2.504
2.68	2.456	2.66	2.849	3.36	3.108
2.78	2.524	3.17	3.048	3.49	3.405
3.12	2.855	4.34	4.296	3.11	3.358
3.49	3.674	3.46	3.403	3.23	3.015
2.30	2.460	4.08	4.071	3.76	3.495
2.32	2.567	2.56	2.660	4.19	3.947
3.85	3.684	4.00	3.927	3.18	3.431
3.17	3.030	2.89	3.125	3.58	3.418
4.11	4.231	3.12	3.354	3.91	3.903
2.52	2.515	3.21	2.961	2.77	3.001
2.53	2.661	2.61	2.886	3.14	2.928
3.54	3.231	3.07	2.856	4.02	4.164
3.08	3.024	2.94	3.192	3.19	2.970
2.90	3.101	4.09	3.946	3.69	3.476
3.75	3.740	2.62	2.819	2.96	2.746
3.57	3.540	4.04	3.900	3.50	3.535
2.99	2.820	4.11	4.360	4.17	3.900
2.99	3.154	2.92	3.159	3.49	3.778
3.14	2.860	2.97	3.198	4.19	3.914
4.02	3.831	2.83	3.067	4.30	4.525
2.65	2.611	2.66	2.386	4.19	4.296
3.77	3.749	3.19	2.925	2.65	2.822
3.72	3.711	2.69	2.588	3.65	3.454
3.52	3.650	3.89	3.887	3.04	2.821

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.12	3.947	4.41	4.472	3.59	3.426
3.39	3.166	3.76	3.573	2.40	2.335
3.72	3.488	3.04	3.105	3.59	3.433
4.48	4.506	4.28	4.154	3.89	3.930
3.33	3.426	4.21	4.285	2.42	2.629
4.27	3.989	3.03	3.193	3.61	3.353
3.66	3.716	2.98	2.858	4.20	4.237
2.63	2.838	3.96	4.156	2.79	2.595
3.51	3.731	2.41	2.440	3.84	3.737
3.41	3.263	2.39	2.542	2.38	2.155
3.76	3.521	4.11	4.060	3.22	3.154
2.86	3.087	4.04	3.907	3.94	3.930
3.52	3.638	3.87	3.609	3.91	4.132
2.88	3.018	2.54	2.290	2.68	2.747
2.89	3.148	3.23	3.512	3.20	3.395
4.23	4.321	3.91	4.014	3.05	3.261
3.01	3.127	2.43	2.563	4.18	4.190
3.64	3.496	2.59	2.366	2.80	2.629
3.11	3.368	3.65	3.456	3.40	3.510
4.33	4.174	3.86	3.749	3.81	4.001
2.88	3.017	3.64	3.541	3.93	4.127
4.18	4.250	3.51	3.490	3.32	3.182
3.29	3.393	4.28	4.436	2.38	2.510
3.12	3.140	4.46	4.293	3.03	2.998
3.31	3.479	3.47	3.678	3.40	3.662
3.11	3.039	3.90	3.805	3.32	3.234
3.32	3.131	4.10	4.083	3.81	3.596
2.44	2.462	4.16	4.254	2.52	2.348
3.61	3.390	3.07	3.007	2.89	3.001
4.14	3.915	2.82	2.655	2.71	2.485
3.37	3.186	3.23	3.352	3.02	2.800
3.15	3.054	3.79	3.930	3.04	3.115
3.50	3.407	3.29	3.145	2.63	2.800
2.71	2.974	4.16	4.025	3.28	3.480
4.14	4.298	4.16	4.283	3.32	3.466
3.69	3.606	2.98	2.794	4.40	4.474
4.00	3.901	3.31	3.461	3.48	3.592
2.52	2.691	3.80	3.833	3.75	3.904
3.17	2.995	3.50	3.661	3.21	3.334
2.31	2.548	3.29	3.560	3.35	3.135
2.89	3.156	4.16	4.417	3.54	3.465
3.95	4.039	4.05	3.919	3.91	3.739
3.80	4.015	2.72	2.889	4.19	3.965
3.56	3.382	4.37	4.479	3.16	2.998
3.06	3.270	4.32	4.382	2.21	2.307

Mob	Mpr	Mob	Mpr	Mob	Mpr
3.50	3.459	3.21	2.944	3.65	3.567
3.72	3.531	3.13	3.235	3.29	3.281
3.59	3.823	4.28	4.193	3.20	3.277
2.28	2.338	2.48	2.633	3.22	3.596
2.80	2.877	2.57	2.726	2.48	2.692
2.52	2.577	2.87	2.902	2.51	2.575
2.88	3.008	3.79	3.613	2.97	3.124
2.94	2.684	2.68	2.565	2.64	2.446
3.98	3.794	4.31	4.242	3.79	4.012
3.49	3.673	3.31	3.108	3.97	4.165
2.94	3.074	2.66	2.468	2.78	2.681
3.92	4.056	4.28	4.241	4.33	4.278
2.87	3.041	3.50	3.491	3.13	2.940
2.39	2.573	3.60	3.297	2.61	2.441
3.99	3.817	2.65	2.389	4.08	4.115
2.96	2.922	2.41	2.306	4.31	4.068
3.70	3.521	4.10	3.897	3.50	3.720
3.60	3.398	3.51	3.742	3.78	3.577
3.42	3.636	3.70	3.490	3.95	4.141
3.87	3.889	3.69	3.533	4.34	4.495
3.49	3.522	3.43	3.305	2.79	2.582
3.72	3.798	3.95	4.117	3.95	4.124
3.40	3.165	3.37	3.264	3.44	3.276
3.33	3.583	3.30	3.425	3.70	3.622
3.48	3.497	2.94	3.067	3.33	3.431
3.98	4.056	2.92	3.151	3.50	3.819
3.62	3.497	3.11	2.821	3.54	3.485
3.92	3.902	3.10	2.817	3.13	3.388
2.49	2.408	3.34	2.991	3.43	3.583
2.48	2.749	3.58	3.227	4.13	4.147
3.15	2.934	3.94	4.020	3.01	2.848
2.63	2.694	2.96	3.248	4.19	3.998
2.85	2.916	2.62	2.519	3.89	4.103
3.26	3.274	3.13	2.707	4.06	3.877
3.20	3.210	3.90	3.849	2.38	2.577
4.23	4.360	2.69	2.567	4.18	4.114
2.58	2.765	4.03	3.749	3.30	3.512
2.87	3.080	3.37	3.478	2.48	2.683
3.10	3.214	3.93	4.311	3.90	3.771
2.66	2.863	3.92	4.392	3.74	4.124
3.48	3.724	3.80	3.659	2.77	2.599
3.54	3.666	3.71	4.011	3.27	2.789
2.83	2.715	2.81	2.556	2.82	2.906
4.26	4.481	3.08	2.819	2.60	2.688
4.44	4.329	4.14	4.452	3.12	3.171

Mob	Mpr	Mob	Mpr	Mob	Mpr
4.19	3.948	3.98	4.071	4.48	4.469
3.60	3.725	3.10	3.336	3.88	4.168
2.83	2.742	3.93	3.881	2.90	3.253
4.10	3.897	3.02	3.189	4.30	3.960
4.00	3.951	4.13	4.394	3.26	3.179
4.06	3.919	2.42	2.591	3.17	3.331
2.56	2.428	3.60	3.735	3.29	3.013
2.75	2.942	3.03	3.251	4.22	4.243
3.94	4.176	3.76	3.531	3.05	3.399
3.65	3.890	2.62	2.529	2.98	3.349
4.07	3.932	4.07	3.916	2.66	2.396
4.04	3.835	2.35	2.538	4.23	3.816
4.01	3.934	2.54	2.720	3.83	4.012
3.31	3.517	4.20	4.141	3.09	3.167
4.18	4.407	3.98	4.128	3.25	3.633
4.07	4.220	4.45	4.290	3.13	3.587
3.63	3.451	2.41	2.207	3.00	3.389
3.64	3.686	2.99	3.211	4.17	4.006
3.44	3.393	4.11	3.926	3.96	4.220
4.16	3.980	3.86	4.074	3.13	3.057
2.69	2.585	3.88	4.125	4.10	4.395
3.75	3.769	2.42	2.312	2.40	2.821
3.80	3.939	3.62	3.859	3.25	3.164
3.76	3.959	3.86	3.686	3.15	2.870
4.00	4.082	3.21	3.244	2.31	2.701
2.50	2.438	4.31	4.178	3.80	4.087
3.88	3.783	3.81	3.708	2.87	2.669
3.92	4.084	2.67	2.917	3.27	2.996
4.26	4.254	2.74	2.550	3.71	3.836
3.85	4.103	3.49	3.396	4.23	3.992
2.58	2.750	3.78	3.719	2.88	2.669
3.50	3.773	4.03	3.881	2.88	3.057
3.92	3.945	2.34	2.659	3.78	3.425
3.43	3.221	2.75	3.057	3.42	3.543
4.33	4.123	2.85	3.061	2.73	2.821
3.45	3.570	4.27	3.973	3.68	3.466
3.50	3.781	3.61	3.570	3.43	3.844
3.96	4.118	3.19	3.500	3.12	3.351
4.42	4.185	4.01	3.835	2.98	2.594
2.91	3.112	2.49	2.355	3.76	3.363
3.47	3.609	3.40	3.727	3.15	3.459
3.88	4.013	3.91	3.505	2.94	2.769
2.95	3.000	4.03	4.235	3.89	3.851
3.53	3.320	3.54	3.299	3.87	3.495
4.08	4.250	4.05	3.924	3.50	3.187

APÊNDICE II - Artigo publicado no "III Congreso Internacional de Computación y Telecomunicaciones" - COMTEL 2011

Distribuição de Vídeo sob Demanda sobre redes P2P

Valmiro José Rangel Galvis[1], Paulo Roberto de Lira Gondim[1]
1 Universidade de Brasília
FT - Departamento de Engenharia Elétrica
rodatam2@gmail.com, pgondim@unb.br

Resumo

Este documento apresenta um overview sobre a distribuição de Vídeo sob Demanda sobre redes Peer-to-Peer (P2PVoD). Nos últimos anos o mundo tem dado muita atenção para pesquisas sobre o uso das redes P2P, de distribuição de conteúdo, para distribuir vídeo ao vivo (Live Streaming) e sob demanda (VoD). Esse tipo de redes promete ser mais econômico e com maior escalabilidade do que as redes baseadas na arquitetura Cliente/Servidor tradicional, além de solucionar o problema do gargalo do lado do servidor quando muitos usuários querem acessar a um mesmo conteúdo ao mesmo tempo. O trabalho relatado neste artigo é focado na distribuição de vídeo sob demanda nas redes P2P, levando em conta que o serviço de VoD é o serviço mais importante nas aplicações de multimídia. Mas, mesmo focado no VoD, o artigo ainda tenta comparar os sistemas de VoD com os sistemas de Live Streaming desde o ponto de vista da sincronização dos clientes, dos comportamentos dos usuários e as características do Streaming de cada sistema. São apresentadas as arquiteturas, as estruturas, os mecanismos de descoberta de conteúdo, os mecanismos de detecção e recuperação de falhas, e o gerenciamento da rede. Além disso, também serão apresentadas as principais técnicas de busca e publicação de conteúdo entre os pares, e as considerações essenciais para o desenho ótimo de sistemas P2PVoD.

Palavras chave: *VoD, redes P2P, arquitetura, estrutura, gerenciamento, overlay*

Abstract

This document introduces an overview about Video-on-Demand over Peer-to-Peer networks (P2PVoD). In recent years the researchers all over the world has given much attention to research about the use of P2P networks, file distribution networks, to distribute live streaming video and video on demand. Such networks promise to be more economical and scalable than networks based on the traditional architecture

Client / Server, and troubleshoot the server-side bottleneck when many users want to access the same content at the same time. The work reported in this paper focuses on on-demand video distribution in P2P networks, taking into consideration that the VoD service is the most important service in multimedia applications. But even focused on VoD, this article also tries to compare systems of Live Streaming with systems of VoD from the viewpoint of users' synchronization and behaviors, and the characteristics of each one of the streamings of the two systems (Live Streaming e VoD). We talk about the architectures, the structures, the mechanism of content discovery, the mechanism of failure detection and recovery, and the network management. Besides, the main techniques of content discovery and publishing between peers, and the essentials issues for the optimal design of P2PVoD systems, will be introduced too.

Keywords: *VoD, P2P network, architectures, structure, management, overlay*

1. Introdução

As redes P2P têm evoluído como um paradigma promissório no compartilhamento de conteúdos em larga escala [1]. Devido à sobrecarga do lado do servidor em sistemas tradicionais baseados na arquitetura *Unicast* Cliente/Servidor, onde se experimentam gargalos assim que o número de clientes aumenta na rede, tendo perdas na largura de banda e na velocidade do serviço. As redes P2P surgem como uma alternativa para aliviar a carga no servidor nos sistemas de distribuição de conteúdo. Soluções alternativas como o IP *Multicast* e as redes de distribuição de conteúdo (CDN, do inglês *Content Distribution Network*), mas o IP *Multicast* é muito complexo para se implementar e precisa de mudanças nos roteadores da rede [13] e as CDN's são muito caras, em março de 2008 o custo da largura de banda de Youtube foi calculado em U$1,000,000 aproximadamente [10].

Existem duas grandes categorias para classificar os sistemas de distribuição de vídeo sobre redes P2P: de *Live Streaming* (vídeo ao vivo) e *Video-on-Demand*. Neste artigo esses sistemas são chamados de P2PLive e P2PVoD [2]. O foco deste artigo é estudar os sistemas P2PVoD. Mas mesmo assim são mencionados sistemas P2PLive, para efeitos de esclarecimentos e comparações que ajudam a melhorar o entendimento da nossa abordagem. Os sistemas P2PVoD são classificados de acordo com o grau de descentralização da rede P2P, à estrutura da rede P2P e ao mecanismo de descoberta de conteúdo.

O restante deste artigo está organizado da seguinte forma. Na seção 2 se apresentam os trabalhos relacionados ao trabalho feito neste artigo. Na seção 3 são apresentadas as características essenciais da distribuição de vídeo sobre redes P2P e são comparadas as duas grandes categorias: os sistemas de distribuição de vídeo ao vivo e sob demanda. Na seção 4 são apresentadas diversas formas de classificar as redes de distribuição de vídeo sob demanda sobre redes P2P. Na seção 5 são consideradas algumas questões de desenho e de melhoras nos sistemas P2PVoD. Finalmente na seção 6 são apresentadas as conclusões e são sugeridos trabalhos futuros.

2. Trabalhos relacionados

Alguns *surveys* foram publicados até aqui sobre o assunto tratado neste artigo [2, 5]. Em [2], Y. Liu, Y. Guo e C. Liang fizeram um *survey* sobre os sistemas de streaming de vídeo sobre redes P2P, tanto para transmissão ao vivo quanto para vídeo sob demanda. Classificaram os sistemas de P2PVoD de acordo com estrutura da rede em árvore, *cache-and-relay* e malha. Em [5], os autores mostraram os desafios, o desenho e a análise de um sistema P2P de larga escala, mas não é uma classificação desses sistemas como tal. Eles só apontaram os aspectos mais importantes no desenho de um sistema P2PVoD e os aspectos a medir para avaliar o desempenho desse tipo de sistema (comportamento e satisfação dos usuários e a eficiência do método de replicação do conteúdo).

3. Distribuição de vídeo Sobre redes P2P

As redes P2P têm evoluído como um paradigma promissório no compartilhamento de conteúdos em larga escala [1]. Devido à sobrecarga do lado do servidor em sistemas tradicionais baseados na arquitetura Unicast Cliente/Servidor, onde se experimentam gargalos assim que o número de clientes aumenta na rede, tendo perdas na largura de banda e na velocidade do serviço [2, 5, 13] apresentando problemas para sistemas de larga escala, onde se precisa capacidade de atendimento para um grande número de usuários. As redes P2P aliviam a carga no servidor usando os recursos disponíveis nos clientes, tais como a largura de banda do *uplink* e a capacidade do buffer e de armazenamento. Assim, a diferença da arquitetura Cliente/Servidor, quanto maior o número de participantes da rede mais recursos terá a rede P2P para o compartilhamento de arquivos entre os pares (neste artigo o uso dos termos nós, pares, clientes e usuários é indiferente) atraindo um grande número de usuários na Internet [2].

Duas propostas, além do uso das redes P2P, para solucionar o problema dos gargalos da

arquitetura Cliente/Servidor são o IP *Multicast* e as CDN's. Mas, o IP *Multicast* na camada de rede é muito complexo para ser implementado [13, 17] e as CDN's, que distribuem a carga da rede entre vários servidores espalhados geograficamente, são muito custosas [13]. Uma proposta híbrida consiste em combinar, em um sistema de distribuição de vídeo só, as tecnologias P2P e CDN aproveitando as vantagens individuais de cada uma delas para oferecer estabilidade e confiabilidade. A CDN é usada para garantir QoS aos clientes e a rede P2P é usada para intercambiar o conteúdo de vídeo entre os usuários e aliviar o número de requisições que chega aos servidores da CDN, aproveitando a escalabilidade e o baixo custo desse tipo de redes [16, 18].

No escopo das redes P2P o que se faz é construir uma rede sobreposta (*Overlay Network*) na camada de aplicação, independentemente da camada de rede subjacente, entre os nós participantes no sistema [2]. A topologia, a estrutura, e o grau de descentralização, e o roteamento e os mecanismos de localização que são usados para a troca de mensagens e de conteúdo são essenciais na operação do sistema, porque afetam a tolerância a falhas, a auto-sustentabilidade, a adaptação a falhas, o desempenho, a escalabilidade e a segurança da rede [19].

Na distribuição de vídeo sobre redes P2P podem se distinguir duas grandes categorias. O *Live Streaming* (Vídeo ao Vivo) e o VoD (Vídeo sob Demanda) [2, 5]. A principal diferença entre a transmissão de Vídeo ao Vivo e a de Vídeo sob Demanda é a sincronização dos clientes. No vídeo ao vivo os clientes estão sincronizados com respeito ao ponto de reprodução. Enquanto no vídeo sob demanda os clientes podem estar observando ou não à mesma parte do vídeo, assim o ponto de reprodução pode ou não variar entre os pares. Outra diferença entre esses sistemas é o que acontece com a entrada de um par novo. No P2PLive *Streaming* o cliente começa assistir o vídeo a partir do momento da entrada no sistema, sem importar o tempo de reprodução, enquanto nos sistemas P2PVoD o cliente decide a partir de que ponto ele vai assistir o vídeo. Além disso, o atraso fim-a-fim é mais importante nos sistemas P2PLive do que nos sistemas P2PVoD, os sistemas P2PVoD são mais tolerantes a atrasos porque o vídeo é pré-gravado e não precisam mostrar ao usuário essa sensação de que o vídeo é ao vivo, chamada "*liveness*" [8, 15]. A resposta às mudanças na transmissão do vídeo também é diferente para cada sistema. Por exemplo, se experimentarem uma redução da QoS, os usuários dos sistemas P2PVoD podem deixar o sistema com maior freqüência porque têm a opção de assistir o vídeo depois, enquanto os usuários dos sistemas P2PLive não têm outra escolha. Esse conceito da freqüência de entrada e saída dos

pares em qualquer momento é conhecido como volatilidade dos pares (*Peer Churn*) [2, 16].

A seguir, vai se falar sobre a distribuição de vídeo sob demanda sobre redes P2P (P2PVoD) e as diferentes formas de classificar este tipo de sistemas, de acordo com certas características como a estrutura da rede, que é o foco deste artigo.

4. Distribuição de vídeo sob demanda sobre redes P2P

Já foram feitos muitos estudos e pesquisas sobre os sistemas P2PLive, mas com as diferenças fundamentais entre os sistemas P2PLive e P2PVoD estabelecidas, pode-se dizer que o escopo usado no desenho dos sistemas P2PLive é difícil de aplicar no desenho dos sistemas P2PVoD [8] e os sistemas que já suportam P2PLive tem que mudar muitas coisas para se adaptar ao VoD, que é o serviço mais importante de aplicações multimídia [3] e um fator fundamental para atrair os consumidores para os serviços de IPTV [2].

As redes P2PVoD podem ser classificadas de acordo com:

- O grau de descentralização da rede P2P
- A estrutura da rede P2P
- O mecanismo de descoberta de conteúdo

4.1. Classificação pelo grau de descentralização da rede

Uma primeira forma de classificar os sistemas de distribuição de P2PVoD, como redes P2P que são, é pelo grau de descentralização da arquitetura da rede. Essas arquiteturas podem ser: totalmente descentralizadas, parcialmente centralizadas e híbridas descentralizadas [19].

Nas arquiteturas totalmente descentralizadas todos os nós da rede têm as mesmas funções na rede, agindo como clientes e servidores ao mesmo tempo sem se ter um controle centralizado das suas atividades. Não existe uma hierarquia na rede [15].

Nas arquiteturas parcialmente centralizadas o fundamento é o mesmo das totalmente descentralizadas. Mas, alguns dos nós assumem funções mais importantes dentro da rede atuando como índices locais centrais para compartilhamento de arquivos dos pares locais. Esses nós são chamados de Super Nós ou Super Pares [20, 19, 5] e estes nós não constituem pontos localizados de falha na rede P2P porque eles são escolhidos dinamicamente e, no caso de falha, a rede automaticamente os substituirá por outros nós.

Nas arquiteturas híbridas descentralizadas há um servidor central (*tracker*) que facilita a

interação entre os pares mantendo diretórios de metadados, que descrevem o conteúdo armazenado em cada par [3, 5, 21]. E as interações para a obtenção dos conteúdos podem ser entre pares somente ou com o servidor, mas o recomendado é que um par só procure conteúdo no servidor como último recurso. Mas, uma desvantagem desta arquitetura é que tem um ponto único de falha localizado (o servidor central) [8, 2] que tipicamente faz com que o sistema seja inerentemente não escalável e vulnerável à censura, a falhas técnicas e a ataques maliciosos [19].

4.2. Classificação de acordo com a estrutura da rede

Um sistema P2PVoD pode ter uma estrutura baseada em árvore de *Multicast*, baseada em malha ou uma estrutura híbrida, uma combinação entre árvore e malha [19].

4.2.1. Estrutura baseada em árvore de *multicast* na camada de aplicação

Os sistemas P2P baseados na estrutura de árvore foram desenhados originalmente para funcionar como o IP *Multicast* na camada de aplicação, independente do suporte da camada de rede subjacente, onde a fonte do vídeo é a raiz da árvore. Os usuários na rede sobreposta estão sincronizados e recebem o conteúdo na ordem que a fonte o envia, e o desafio nesse tipo de estrutura é como arranjar os usuários assíncronos dos sistemas P2PVoD eficientemente e oferecer suporte a operações de VCR (avançar ou atrasar o vídeo, pular a outra parte do vídeo, etc.) [2].

A estrutura de árvore pode ser de árvore simples ou de múltiplas árvores, mas o de árvore simples não é muito utilizado pelos fatos de que os nós folha (nós finais da árvore) não contribuem com seus recursos à rede, de que na hora que um nó falha todos os seus nós filhos ficam sem o streaming e de que o processo de recuperação da falha não é o suficientemente rápido para lidar com a volatilidade dos pares, tornando o sistema ineficiente [2].

Para lidar com os diferentes problemas da estrutura de árvore simples foram estudadas aplicações de árvore de *multicast* [2, 8, 4, 11, 13]. Onde o servidor é a raiz, mas os pares são arranjados em subárvores no sistema de acordo com seus tempos de chegada em sessões ou gerações.

Em [2], o escopo dos autores é que os usuários são agrupados em sessões de acordo com um tempo limiar T, usuários que chegarem ao sistema dentro desse tempo limiar constituem uma sessão. Junto com o servidor, clientes que pertencem à mesma sessão formam uma árvore de multicast na camada de aplicação, que é chamada de árvore base. O servidor transmite o vídeo normalmente e quando um novo nó chega ao sistema

ele pega o streaming de um dos nós que já faz parte da árvore base, enquanto a parte que perdeu do vídeo ele obtém de um dos nós da árvore que já o tem armazenado no cachê.

Outro esquema estudado pelos autores em [2], é o *Cache-and-Relay* onde, além do tempo de chegada, é utilizado o tamanho da janela deslizante do cachê dos participantes para agrupá-los em clusters. O usuário "A" que chega primeiro ao sistema obtém o vídeo diretamente do servidor e armazena o conteúdo no cachê usando uma janela deslizante, centrada no seu ponto de reprodução, e encaminha o vídeo para os pares que cheguem depois ao sistema, cuja parte de interesse do vídeo esteja ainda dentro da janela deslizante de A, esses usuários de forma similar a "A" encaminham o vídeo para outros usuários, formando, assim, um cluster. Um usuário "B" que chegar fora dessa janela é forçado a obter o vídeo diretamente do servidor e a formar outro cluster de forma similar a "A", junto com os usuários que chegarem depois dele.

Em [8], os autores agrupam os pares em gerações e em sessões. O vídeo é dividido em blocos de recuperação Blocos-R (*Retrieval Block*), onde cada bloco é equivalente a uma unidade de tempo de reprodução. Os Blocos-R são numerados de acordo com sua posição com respeito ao tempo no vídeo, de 1 até o comprimento total do vídeo. Os nós participantes têm que contribuir ao sistema com um espaço na memória (buffer) para armazenar em cachê o conteúdo mais recente do streaming de vídeo que estão recebendo. Cada cliente pode servir outros clientes que chegarem depois dele, entre o seu tempo de chegada (tchegada) e o tempo de chegada mais o tamanho do buffer (TBuffer), ou seja no intervalo [tchegada, tchegada+TBuffer]. Os clientes que tiverem o mesmo grupo de Blocos-R armazenados em cachê são agrupados em gerações. Os clientes dessas gerações, excluindo o servidor, formam uma sessão de vídeo. Uma sessão é terminada quando nenhum dos clientes, pertencentes a essa sessão, tem o primeiro Bloco-R do vídeo e se um cliente quer se unir ao sistema tem que começar uma nova.

Em [4], o servidor começa uma sessão de transmissão de broadcast do vídeo a cada m minutos. Assim, os pares que entrarem no sistema desde o inicio da transmissão até o minuto m formam a sessão 1. Do mesmo jeito, os pares que cheguem depois ao sistema formam as sessões 2, 3, etc. Em cada sessão, os pares que cheguem primeiro ao sistema são os filhos diretos do servidor e os pares que cheguem depois da ocupação total da largura de banda do servidor são redirecionados para se converterem em descendentes dos que chegaram primeiro. Para reforçar o esquema de transmissão do vídeo, o

servidor envia uma lista aleatória de alguns dos participantes a cada usuário que está entrando ao sistema. Quando um usuário entra tarde em uma sessão e perde a parte inicial do vídeo ele seleciona alguns dos nós da lista aleatória, que o servidor lhe enviou, para baixar a parte inicial do vídeo que ele ainda não assistiu.

Em [11], os autores dividem a árvore em camadas. Cada nível da árvore é uma camada e, além do mais, usam o conceito de múltiplos pais para dotar o sistema de maior robustez, contra a falha de um nó, do que os sistemas com árvores de um pai só. As camadas são divididas com respeito ao segmento de vídeo que os pares de cada camada estejam assistindo, portanto a camada *i* é a camada dos pares que estão assistindo o segmento *i* do vídeo. Cada par que quer entrar no sistema contata o servidor central da rede para que este lhe indique os nós candidatos a pai dele na camada imediatamente acima da camada na qual ele quer entrar.

4.2.2. Estrutura baseada em malha

Os sistemas baseados na estrutura de malha têm mostrado um melhor desempenho do que os sistemas baseados na estrutura de árvore, e a maioria das pesquisas atuais sobre o VoD são baseadas nesta estrutura [12]. Eles têm uma robustez intrínseca, uma boa resposta à volatilidade dos pares, adaptabilidade aos ambientes dinâmicos e um uso eficiente da largura de banda disponível nos pares. Mas, como esses sistemas não têm uma estrutura bem definida, a descoberta do conteúdo e o roteamento das mensagens são mais complicados [23].

Atualmente existem várias propostas sobre sistemas de P2PVoD que usam a estrutura de malha para a construção da rede sobreposta na camada de aplicação [1, 5, 6, 7]. Onde não existe a relação pai/filho como na estrutura de árvore, mas sim, de vizinhos ou relações de vizinhança entre os pares, aliás, o problema da seleção de vizinhos é um problema fundamental na construção da rede sobreposta que influi diretamente na qualidade do serviço do sistema.

Para os autores em [1], cada par que quer se unir à rede tem que contatar um *tracker* que lhe envia uma lista dos pares interessados no mesmo vídeo. O novo par seleciona aleatoriamente entre 8 e 10 pares dessa lista, para estabelecer parcerias. Com as parcerias estabelecidas o par fica pronto para compartilhar largura de banda e trocar informações sobre a disponibilidade dos dados, e através de um escalonador procurar os segmentos de vídeo que precisa. Esse esquema de construção da rede é chamado de *Data-Driven Overlay Network (DONet)* [24] ou *Swarming* [27, 22].

Em [5], o escopo dos autores é similar ao de [1], mas com a diferença de que quando um par começa a assistir a um vídeo, ele informa ao *tracker* que esta retransmitindo esse vídeo e também quando não tem mais o vídeo armazenado no cachê e vai parar de retransmitir. Quando um par quer começar a assistir um vídeo, ele contata o *tracker* para encontrar pares que tenham esse vídeo. E é assim que se faz a seleção dos vizinhos.

Analisando os diferentes sistemas estudados se percebeu que na construção da rede sobreposta se usa o método de enxames, o que muda é a forma como os pares selecionam os seus vizinhos no enxame. Os usuários que querem entrar na rede contatam um *tracker* que, de acordo com o segmento de vídeo de interesse ou a um vídeo de interesse, responde com uma lista de candidatos vizinhos. O problema agora é como selecionar os vizinhos dessa lista de uma forma eficiente e como manter as relações de troca de dados entre os vizinhos.

Em [6], os autores criam vários enxames de acordo com o ponto de reprodução de cada par. Agrupando os pares de acordo com o ponto de reprodução garantem uma troca de segmentos de interesse entre os pares do mesmo enxame. Mas, para incentivar a troca de segmentos com pares dos enxames que estejam mais na frente, desde o ponto de vista do ponto de reprodução, o servidor puxa segmentos que os pares que estejam mais na frente precisem para os pares mais atrasados, com a finalidade de dotar a esses pares com segmentos de interesse para trocar com os usuários mais avançados do sistema, que já têm a primeira parte do vídeo assistida armazenada no seu cachê.

Para oferecer melhor suporte às operações de VCR, em [7] criam o InstantLeap (pulo instantâneo). Neste sistema, o vídeo é dividido em m segmentos. Um usuário faz parte do grupo *i* se o seu ponto atual de reprodução cai dentro do segmento *i* do vídeo. Os pares dos grupos *i*-1, *i* e *i*+1 provavelmente têm conteúdo de buffer que se sobreponha, sendo, assim, possíveis pares provedores entre si. Um par do grupo *i* tem duas listas de vizinhos: uma lista que é um subconjunto do seu grupo *i* e dos grupos adjacentes *i*-1 e *i*+1 para descarga e troca de blocos do vídeo; a segunda lista inclui pares que não estejam nos grupo *i*-1, *i* e *i*+1, para assim, no caso do par precisar pular do seu ponto de reprodução para outro, ele pode usar um nó dessa lista como um atalho para alcançar essa outra parte do vídeo.

4.2.3. Estrutura híbrida (Árvore - Malha)

São poucos os sistemas que usam essa estrutura por causa da complexidade. Um

exemplo desses sistemas é o TAG [3]. O TAG usa combina as estruturas de árvore e malha para o escalonamento dos dados de mídia em ambientes de VoD. O TAG usa principalmente a estrutura de malha para entregar os dados, mas a seqüência é determinada pela estrutura de árvore. Este tipo de arquitetura aumenta a dificuldade da implementação do sistema, e ainda sofre com a descontinuidade na reprodução do vídeo.

4.3. Classificação de acordo com o mecanismo de descoberta do conteúdo

Sem exceção, todos os sistemas P2P estão baseados nos seguintes métodos para a publicação e a busca de conteúdo entre os pares da rede [3]:

- Tabelas Hash Distribuídas (DHT)
- Índices centralizados (*Trackers*)
- Gossip/Flooding

4.3.1. Mecanismos baseados em DHT

Os sistemas que usam o mecanismo do DHT [10, 11, 12] utilizam uma função *hash* para atribuir uma chave a um arquivo e aos nós do sistema de acordo com um parâmetro único, em uma operação que é chamada de *hashing*. No caso dos arquivos pode ser pelo nome e no caso dos nós pode ser pelo endereço IP. Depois de realizar o processo de *hashing*, cada nó do sistema tem que cuidar de um determinado número de chaves, uma forma de distribuir as chaves é o de dividir o número de chaves (*k*) pelo número de nós (*n*), assim cada nó tem que cuidar de *k/n* chaves. Assim, quando um nó for procurar por um vídeo em especial no sistema, procura o nó que têm que cuidar da chave associada a esse vídeo. Quando um nó abandona a rede as chaves têm que ser redistribuídas entre os nós, portanto este método de DHT não é recomendado para sistemas onde os pares apresentem uma volatilidade alta, entram e saem do sistema freqüentemente, porque precisariam muitas atualizações o que poderá trazer sobrecarga de mensagens de controle e de atualização. Uma proposta para lidar com essa dinamicidade do sistema é um método de atualizações preguiçosas chamado de t-DHT (*Temporal DHT*), descrito em [9].

Em [10], D. Wang e C.K. Yeo usam um roteamento DHT para criar atalhos entre os pares na rede para diminuir o tempo de busca de um segmento de vídeo.

Em [11], os autores usando uma estrutura baseada em DHT, chamada anel *CHORD* [14], propõem um sistema de múltiplas camadas, onde cada camada é um anel lógico *CHORD*. Unindo as diferentes camadas formam uma estrutura cilíndrica. As camadas

são definidas de acordo com os segmentos de interesse dos pares do sistema, pares interessados no mesmo conteúdo são agrupados em uma mesma camada. O vídeo é transmitido de camada em camada, a partir da camada mais próxima ao servidor (primeiros segmentos do vídeo), mas um nó filho de uma camada inferior pode pegar os segmentos de vídeo de diferentes pais na camada superior (múltiplos pais).

As estruturas baseadas em anéis *CHORD* são as mais difundidas, entre os sistemas que usam o DHT. As variações entre os sistemas dependem do número de anéis e de como é mantida e gerenciada a estrutura e de como é feita a busca do conteúdo entre os participantes.

4.3.2. Mecanismos baseados em índices centralizados (*Trackers*)

Os mecanismos baseados em índices centralizados têm um *tracker*, que é um servidor central onde estão armazenados os dados ou estão indexados os pares que têm o conteúdo de interesse que está se procurando. Os pares do sistema periodicamente informam ao *tracker* quais partes do vídeo possuem enviando o mapa do seu buffer (*Buffer Map*) e mensagens de que estão ativos no sistema. Assim, quando um usuário precisar de um conteúdo, envia uma requisição para o *tracker* que, baseado na informação do conteúdo disponível nos pares, redireciona o usuário para o par ou os pares que possuem o conteúdo de interesse, já depende do usuário selecionar o par que possa lhe servir melhor o conteúdo.

Exemplos de sistemas que usam esse mecanismo já foram descritos anteriormente em 4.2.2.

4.3.3. Mecanismos baseados em *Gossip/Flooding*

Esse tipo de mecanismo é ótimo para estruturas que não têm uma topologia definida (malha). Além de ser o mecanismo que apresenta maior descentralização, porque não precisa de um servidor central para buscar conteúdo na rede, mas o uso de um *tracker* pode ser opcional para dotar ao sistema de maior robustez para lidar com falhas e volatilidade dos nós. Um par em busca de um conteúdo começa a trocar o mapa do seu buffer com os seus vizinhos para saber quais partes do vídeo os seus vizinhos possuem e para informar quais partes ele possui, esse mecanismo é conhecido como método baseado na fofoca (*gossip*). Quando um par está interessado em um determinado *chunk* do vídeo que os seus vizinhos não possuem, ele começa a espalhar mensagens na rede, enviando uma mensagem de que está procurando por esse *chunk* para os seus vizinhos, que também enviam essa mensagem para os seus vizinhos e assim por diante. Mas tem

que tomar cuidado para limitar a replicação da mensagem e não ocasionar sobrecarga na rede.

Um exemplo claro do funcionamento do mecanismo de fofoca pode ser encontrado em [3].

Os mecanismos expostos para a descoberta de conteúdo na rede também são os mesmos usados para o gerenciamento e a manutenção da rede P2P sobreposta.

5. Considerações no desenho de redes de distribuição de vídeo sob demanda sobre redes P2P

No desenho de um sistema P2PVoD têm que ser considerados vários fatores. Em [3] propõem considerar o tamanho dos segmentos (o tamanho de cada *chunk* no qual o vídeo é dividido para a sua transmissão). A estratégia usada para replicar os segmentos na rede, com a finalidade de garantir a maior disponibilidade dos segmentos entre os pares, para satisfazer as demandas de visualização dos usuários sem incorrer com uma sobrecarga excessiva no sistema. A seleção de *chunks* também é um fator importante, como são escalonados para serem recebidos e para serem descartados do buffer, com técnicas já conhecidas na literatura como FIFO (*First In, First Out*), LRU (Least Recently Used), LRS (*Least Reusable First*), EDF (*Excessive Duplicated First*), LSSF (*Least Server Supplementary First*) [26]. E, por último, tratam o tema da estratégia de transmissão dos *chunks* para realizar o *download* dos *chunks* de uma forma eficiente.

Além das questões anteriormente mencionadas, em [8] são considerados fatores essenciais, tais como a entrada de um nó na rede. A entrada do nó tem que ser o mais rápida possível. A localização e recuperação de falhas também têm que ser rápidas e imperceptíveis para os usuários. O sistema também tem que ter um manejo adequado das petições assíncronas dos clientes e um overhead de controle baixo.

Outra forma de melhorar o desempenho dos sistemas de P2PVoD, desde o ponto de vista das operações de VCR, é o uso de uma técnica chamada *prefetching*, que consiste em antecipar os segmentos a serem reproduzidos no tocador de vídeo. Tem se mostrado que os sistemas que implementam o *prefetching* podem obter um desempenho muito melhor, com respeito a aliviar a carga do servidor, do que os sistemas que não [12]. O *prefetching* pode ser feito de uma forma seqüencial, aleatória, procurando os *chunks* mais raros primeiro, estabelecendo relações entre os segmentos de um vídeo (intra-vídeo) [4] ou estabelecendo relações entre vídeos (inter-vídeo) [10].

A diferença dos sistemas de P2PLive e de compartilhamento de arquivos, os pares dos sistemas P2PVoD tendem a ser mais egoístas. Um par só se interessa por conteúdo além

do seu ponto de reprodução e só pode ajudar aos pares que se encontram atrasados [3]. Uma proposta interessante para lidar com este problema é a de [6], que usando o mecanismo de incentivo ao compartilhamento dos recursos do BitTorrent [22, 25], um dos sistemas mais populares do mundo para o compartilhamento e distribuição de arquivos, conhecido como *Tit-for-tat*, incentiva aos clientes para a troca de arquivos dotando aos pares mais atrasados com segmentos de interesse, para os pares mais avançados no sistema.

6. Conclusões e trabalho futuro

Neste artigo se apresentou uma classificação que não tinha sido feita antes dos sistemas P2PVoD, mesmo assim que tenham sido feitos vários estudos e pesquisas sobre esse tipo de sistemas, o nosso escopo é totalmente diferente, e logrou-se classificar os sistemas P2PVoD segundo o grau de descentralização da rede P2P, a estrutura da rede P2P e o mecanismo de descoberta de conteúdo.

Mostraram-se as estruturas baseadas em árvore e malha, apontando as vantagens e desvantagens das duas estruturas e a complexidade de implementar uma estrutura híbrida árvore - malha.

Apontamos também fatores chave no desenho dos sistemas P2PVoD e alguns mecanismos para melhorar o desempenho desses sistemas como o prefetching e o incentivo à interação entre os pares.

Como apenas estamos começando o trabalho, não se tem resultados experimentais, mas o resultado do artigo é satisfatório como um primeiro passo no estudo dos sistemas P2PVoD.

Uma proposta para trabalho futuro é adicionar mais uma forma de classificar os sistemas, os que oferecem suporte à transmissão de vídeo escalável (H.264/SVC, MPEG2, etc.) e os que não. Além disso, estudar como são usados os mecanismos, de descoberta de conteúdo, expostos em 4.3 para o gerenciamento e a manutenção da rede P2P sobreposta. Outra proposta é a de estudar os diferentes sistemas existentes atualmente e classificá-los de acordo com a taxonomia proposta neste artigo. Estudar com mais profundidade os algoritmos usados na construção, no gerenciamento e na manutenção rede. A maioria dos sistemas estudados neste artigo foi desenhada para operar com um canal só, faltaria testar o que acontece ou que soluções podem ser propostas no caso de um ambiente de múltiplos canais. E finalmente também pode ser viável a criação do nosso próprio *framework* para um sistema de P2PVoD.

O trabalho futuro vai ser desenvolvido usando o GoalBit [28], uma plataforma de transmissão de conteúdo de vídeo sobre redes P2P de código aberto disponível para vários sistemas operativos, vários formatos de vídeo e que segue o padrão PPSP (*Peer-to-Peer Streaming Protocol*) da IETF (*Internet Engineering Task Force)*.

REFERÊNCIAS

[1]. Y. Ding, J. Liu, D. Wang, e H. Jiang, "Peer-to-peer video-on-demand with scalable video coding," *Computer Communications*, vol. 33, Sep. 2010, pp. 1589-1597.

[2]. Y. Liu, Y. Guo, e C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, Jan. 2008, pp. 18-28.

[3]. L. Yu, X. Liao, H. Jin, e W. Jiang, "Integrated buffering schemes for P2P VoD services," *Peer-to-Peer Networking and Applications*, vol. 4, Nov. 2010, pp. 63-74.

[4]. Y. He e Y. Liu, "VOVO: VCR-Oriented Video-on-Demand in Large-Scale Peer-to-Peer Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, Abr. 2009, pp. 528-539.

[5]. Y. Huang, T.Z.J. Fu, D.-M. Chiu, J.C.S. Lui, e C. Huang, "Challenges, design and analysis of a large-scale p2p-vod system," *Proceedings of the ACM SIGCOMM 2008 conference on Data communication - SIGCOMM '08*, 2008, p. 375.

[6]. K. Huguenin, A.-M. Kermarrec, V. Rai, e M. Van Steen, "Designing a tit-for-tat based peer-to-peer video-on-demand system," *Proceedings of the 20th international workshop on Network and operating systems support for digital audio and video - NOSSDAV '10*, 2010, p. 93.

[7]. X. Qiu, C. Wu, X. Lin, e F. Lau, "InstantLeap: fast neighbor discovery in P2P VoD streaming," *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, ACM, 2009, p. 19–24.

[8]. T.T. Do, K. a Hua, e M. a Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*, 2004, Vol.3, pp. 1467-1472.

[9]. A. Bhattacharya, Z. Yang, and S. Zhang, "Temporal-DHT and Its Application in P2P-VoD Systems," *2010 IEEE International Symposium on Multimedia*, Dez. 2010, pp. 81-88.

[10]. D. Wang e C.K. Yeo, “Exploring Locality of Reference in P2P VoD Systems,” *2010 IEEE Global Telecommunications Conference, GLOBECOM ‘10,* 2010, pp. 1-6.

[11]. D. Xie, B. Qian, Y. Peng, e T. Chen, “A Model of Job Scheduling with Deadline for Video-on-Demand System,” *2009 International Conference on Web Information Systems and Mining,* Nov. 2009, pp. 661-668.

[12]. B. Cai, Z. Luo, e Y. Luo, “A Special Topology for Files Pre-Fetch in Large Scale Video on Demand,” *2010 International Conference on Biomedical Engineering and Computer Science,* Abr. 2010, pp. 1-4.

[13]. J.-H. Roh e S.-H. Jin, “Video-on-Demand Streaming in P2P Environment,” *2007 IEEE International Symposium on Consumer Electronics,* Jun. 2007, pp. 1-5.

[14]. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, e H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for Internet applications,” *Proceedings of the ACM SIGCOMM 2001 conference on Data communication - SIGCOMM ’01*, 2001, pp. 149–160

[15]. Z. Lu, S. Zhang, J. Wu, W. Fu, e Y. Zhong, “Design and Implementation of a Novel P2P-Based VOD System Using Media File Segments Selecting Algorithm,” *7th IEEE International Conference on Computer and Information Technology (CIT 2007),* Oct. 2007, pp. 599-604.

[16]. U.C. Kozat, O. Harmanci, S. Kanumuri, M.U. Demircin, e M.R. Civanlar, “Peer Assisted Video Streaming With Supply-Demand-Based Cache Optimization,” *IEEE Transactions on Multimedia*, vol. 11, Apr. 2009, pp. 494-508.

[17]. A. Wu, L. Yuan, X. Liu, e K. Liu, “A P2P Architecture for Large-scale VoD Service,” *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007),* Jul. 2007, pp. 841-846.

[18]. Z. Lian-ying e Z. Xiao, “The research of VoD system performance based on CDN and P2P technologies,” *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE),* vol. 4, Fev. 2010, pp. 385-388.

[19]. S. Androutsellis-Theotokis e D. Spinellis, “A survey of peer-to-peer content distribution technologies,” ACM Computing Surveys, vol. 36, Dez. 2004, pp. 335-371.

[20]. M. Zhang e B. Feng, “A P2P VoD system using dynamic priority,” *2009 IEEE 9th Malaysia International Conference on Communications (MICC)*, Dez. 2009, pp. 518-523.

[21]. Q. Gu, “A schedule algorithm of peer-to-peer VoD system,” *Computer and Communication Technologies in Agriculture Engineering (CCTAE), 2010 International Conference On, IEEE*, 2010, p. 583–586.

[22]. S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, e P.R. Rodriguez, “Is high-quality VoD feasible using P2P swarming?,” *Proceedings of the 16th international conference on World Wide Web, ACM*, 2007, pp. 903–912.

[23]. R.P. Leal, E.P. Martín, e J.Á. Cachinero, “Internet TV Broadcast: What Next?,” *2009 Fourth International Conference on Digital Telecommunications*, Jul. 2009, pp. 71-74.

[24]. X. Zhang, J. Liu, B. Li, e T. Yum, CoolStreaming/DONet: “A data-driven overlay network for efficient live media streaming,” *IEEE INFOCOM’05*, vol. 3, 2005, pp. 2102–2111.

[25]. BitTorrent Homepage *http://www.bittorrent.com* acessado em 30/07/2011

[26]. L. Kai, S. Xiao, Q. Zhigang, P. Lingjuan, e L. Hui, “Peers’ Segment Replacement Strategies in P2P Vod System Based On Client-Side Segmented Cache,” *2009 WRI International Conference on Communications and Mobile Computing*, Jan. 2009, pp. 26-33.

[27]. M. Zhang, Y. Xiong, Q. Zhang, L. Sun, e S. Yang, “Optimizing the throughput of data-driven peer-to-peer streaming,” *IEEE Transactions on Parallel and Distributed systems*, vol. 20, Jan. 2008, p. 97–110.

[28]. http://goalbit.sourceforge.net

APÊNDICE III – Resumo Publicado no 31º. Simpósio Brasileiro de Redes de Computadores/Workshop de Redes P2P+

Avaliação da QoE na Distribuição de Vídeo Escalável sobre Redes *Peer-to-peer*

Valmiro José Rangel Galvis[1], Paulo Roberto de Lira Gondim[1]

[1] Faculdade de Tecnologia - Universidade de Brasília – UnB
Brasília – D.F. - Brasil.

{valmiro, pgondim}@unb.br

Abstract. *This paper presents a predictor of the quality of experience (QoE) as perceived by a user receiving a scalable video distributed over a peer-to-peer (P2P) network. The scheme is based on a quantitative relationship between the measured quality of service using network parameters and the quality as perceived by the end-user. That is, the mapping of the quality of service (QoS) into quality of experience (QoE). According to the experiments and measures, the relationship predicts the Mean Opinion Score (MOS) accurately.*

Resumo*. Neste artigo apresenta-se um preditor de qualidade de experiência (QoE, do inglês Quality of Experience) para estimar a qualidade de experiência tal qual estaria sendo percebida por um usuário recebendo um vídeo escalável distribuído por meio de uma rede P2P. O esquema é baseado em uma relação quantitativa entre a qualidade aferida a partir de parâmetros de rede e a qualidade como percebida pelo usuário final. Isto é, o mapeamento da qualidade de serviço (QoS, do inglês Quality of Service) em qualidade de experiência (QoE). De acordo aos experimentos realizados, a relação prediz de uma forma precisa o MOS (do inglês, Mean Opinion Score).*

1 – Introdução

A utilização de redes *Peer-to-Peer* (P2P) para "streaming" de vídeo na Internet tem recebido atenção considerável de pesquisadores da área, permitindo aos computadores funcionar como um meio de armazenamento distribuído, contribuindo, buscando e obtendo conteúdo digital [1].

Se por um lado as arquiteturas de redes P2P solucionam problemas existentes em arquiteturas cliente-servidor, a transmissão de vídeo em redes P2P apresenta grandes

desafios, considerando a demanda, a capacidade limitada de "upload" dos pares, a heterogeneidade de receptores e as variações em termos de disponibilidade de banda.

Para tratar a citada variabilidade de banda inerente a sistemas P2P, as tecnologias de transmissão e de codificação devem suportar adaptação de taxas de transmissão. Mais que isso, se tecnologias de codificação de vídeo tradicionais (como MPEG2) são utilizadas, torna-se difícil tratar diferentes tipos e capacidades de receptores, variando desde telefones móveis até displays de alta definição (televisores LCD, por exemplo).

Neste sentido, o tratamento da heterogeneidade dos nós e das diferentes taxas de bits dentro de uma rede *peer-to-peer*, pode ser beneficiada pelo emprego de vídeo escalável, em que tem sido considerado o padrão de codificação de vídeo H.264/SVC *(Scalable Video Coding).* Sendo o vídeo escalável e codificado em camadas, o usuário pode receber a camada base e buscar apresentá-la; por outro lado, se a rede e o usuário tem os recursos para tratar camadas de reforço, será possível melhorar a qualidade percebida.

Neste contexto, um dos problemas que vem sendo objeto de investigação recente ([3], [6],[7],[10]) é o do mapeamento entre a qualidade de serviço (QoS) oferecida pela rede e a qualidade de experiência (QoE) percebida pelo usuário. A necessidade de tal mapeamento decorre da dificuldade de estimar, de forma automatizada e em tempo real, a experiência que está sendo vivida pelo usuário final. Mais que isso, é necessário avaliar o impacto de parâmetros de QoS sobre a qualidade de vídeo (QoV) e por conseguinte sobre a QoE, bem como buscar estabelecer as adaptações que forem necessárias em termos de taxa de envio de vídeo, de forma compatível com as capacidades dos receptores e a disponibilidade de recursos de rede.

Assim, este artigo avalia o impacto de parâmetros de rede e de aplicação na qualidade percebida pelo usuário final em um sistema de distribuição VoD sobre uma rede P2P, com o vídeo sendo codificado no padrão H.264/SVC.

Sua organização é a seguinte: na seção 2 são discutidos trabalhos relacionados. A seção 3 trata dos experimentos realizados e do relacionamento entre parâmetros de QoS e a QoE. A seção 4, por fim, delineia conclusões obtidas e aponta trabalhos futuros.

2 - Trabalhos Relacionados

Pesquisas sobre a importância relativa de parâmetros de rede para a QoE do usuário têm sido feitas ([2-5], [7]). As pesquisas feitas sobre a relação entre QoE e QoS têm mostrado que essa relação não é linear [2, 3, 4].

Kim e Choi [3] fazem um estudo de um modelo de correlação entre a QoS e a QoE para a aferição da QoE de um serviço de IPTV. Consideram que a QoS é uma função de vários parâmetros de rede e descrevem a QoE como uma função exponencial da QoS, considerando nessa função a resolução da tela do usuário, o tipo de serviço assinado pelo usuário, e o tamanho do GOP do vídeo.

Em [6] os autores, após um agrupamento dos vídeos de acordo com o tipo de conteúdo (fundo, movimento dentro do vídeo, etc.), propõem um modelo de predição do MOS para o *streaming* de vídeo MPEG-4 sobre redes sem fio. Após classificar e agrupar as sequências de vídeo, é feita uma regressão linear para encontrar uma equação que se ajuste aos dados disponíveis da avaliação da qualidade das sequências de vídeo, permitindo estimar o MOS levando em conta o conteúdo, parâmetros de rede e de aplicação. Os dados para a regressão linear foram gerados usando uma avaliação objetiva, neste caso o PSNR, e mapeando este PSNR em MOS.

Em [7], Fiedler et al. propõem uma relação genérica quantitativa entre a QoE e a QoS. A sua proposta é que existe uma interdependência exponencial entre a QoE e a QoS, e usando os dados dos resultados dos testes de MOS, encontram uma curva que se adapta melhor aos resultados, do que a curva da relação logarítmica proposta em [2].

3 - Mapeamento de QoS em QoE

As dificuldades por trás da avaliação subjetiva de sistemas P2P surgem pelo fato de que tais sistemas, geralmente, abrangem áreas geográficas muito extensas, envolvendo um grande número de pares (e de usuários), sendo demorado e de custo elevado atender à necessidade de participação humana. Mais que isso, substituir a avaliação realizada por seres humanos por processos automatizados é, especialmente no caso de avaliação da qualidade de vídeo, uma tarefa difícil.

A avaliação da qualidade de vídeo no sistema de distribuição de VoD aqui tratado considerou formatos CIF e QCIF, e o *software* JSVM (*Joint Scalable Video Model*). Os testes consideraram camadas espaciais (duas), temporais (seis) e de qualidade SNR (duas); escalabilidade de granularidade media (MGS); taxas: 1.875, 3.75, 7.5, 15 e 30 fps; GOP de 16.

Com os vídeos codificados, simulou-se a distribuição do vídeo sobre uma rede P2P usando o simulador P2PTVSim, e a avaliação da qualidade foi feita usando Evalvid [11]. Trabalhos continuam sendo feitos tratando SSIM e VQM.

Os vídeos foram agrupados pelas suas características temporais e espaciais [6] em: sequências de movimento leve (*Slight Moviment – SM*), figura 2; de movimento moderado (*Gentle Walk – GW*), e de movimento rápido (*Rapid Moviment – RM*).

Fig. 1. Vídeos da categoria SM

Usando o arquivo de *BitRate*, foram escolhidas as diferentes taxas do vídeo (SBR). Nos experimentos foram usados três vídeos de cada tipo (SM, GW, RM) para treinamento e um vídeo de cada tipo para os testes (quatro vídeos no total para cada tipo).

4 - Modelo de predição de QoE

Para fazer o mapeamento de QoS em QoE se propõe uma equação de baixa complexidade, que serve como preditor de MOS, obtida por regressão não linear. Como não se têm valores de MOS subjetivos disponíveis, esses valores de MOS são obtidos objetivamente com a ferramenta de mapeamento de PSNR em MOS do Evalvid.

As regressões mostraram que a equação para expressar o MOS em termos da SBR é da forma mostrada na equação 1, e a figura 2 mostra o comportamento do MOS em função do SBR:

$$MOS(SBR) = a + bLn(SBR) \qquad (1)$$

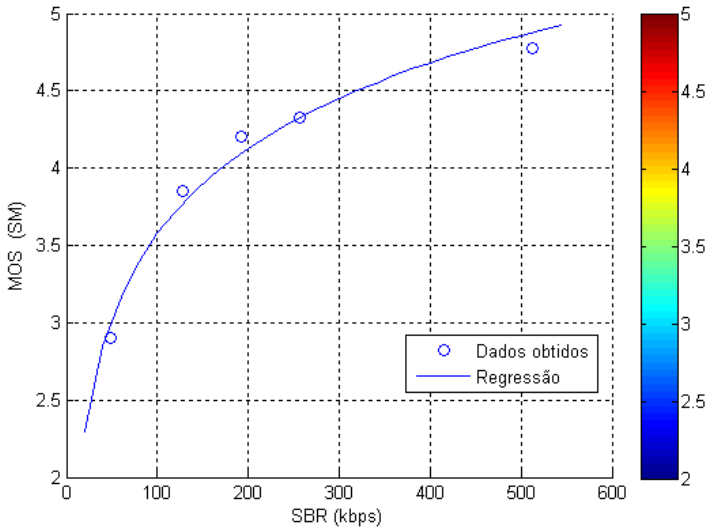

Figura 2. MOS em função da SBR para vídeos do tipo SM.

Os valores de FR são tomados do arquivo de configuração da codificação do vídeo escalável. Esses valores são os valores de fps disponíveis na qualidade temporal do vídeo. Da figura pode ser vista que uma linha reta pode modelar esta relação entre MOS e FR, relação mostrada na equação 2, e a figura 6 mostra o comportamento do MOS em função do FR:

$$MOS(FR) = a + bFR \qquad (2)$$

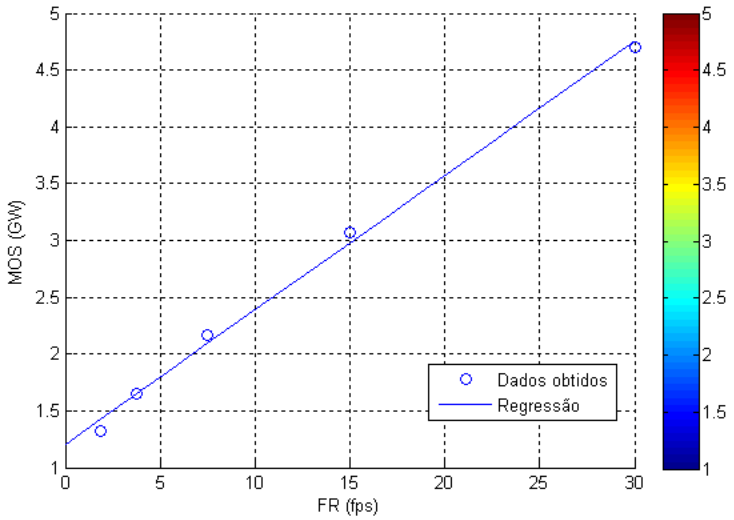

Figura 3. MOS em função da FR para vídeos do tipo GW.

Modelagem adicional, de natureza similar às modelagens apresentadas acima, é feita para expressar o MOS em função da PER ("Packet Error Rate"). Por fim, uma expressão para o MOS em função dos parâmetros SBR, FR e PER se encontra descrita em [12].

A figura 3 apresenta os valores de MOS (escala de 1 a 5) em função da taxa de *frames* FR e da taxa de envio SBR, supondo perda de *frames* nula, em que se obtém valores de MOS situados entre 3,6 e 4,5 a partir dos valores de FR e SBR considerados.

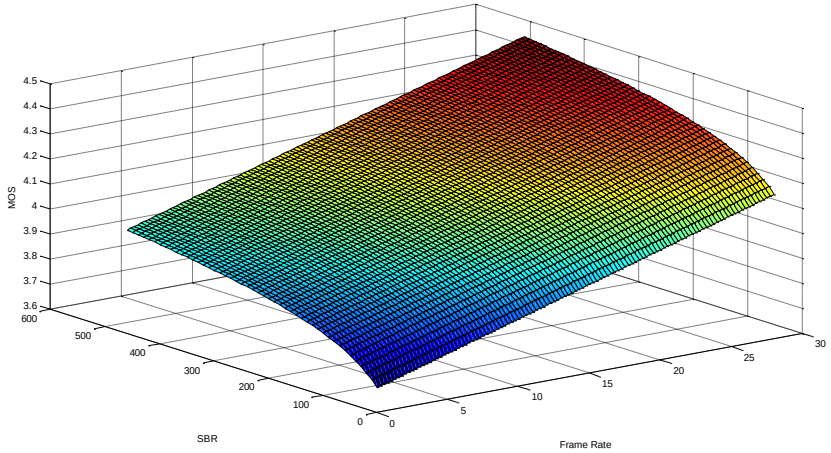

Figura 3. MOS como função do FR e do SBR (PER nula).

O modelo é estendido em [12], permitindo confirmar a possibilidade concreta de

mapear a influencia de parâmetros de rede em uma medida da QoE, explicitada pelos valores de MOS mapeados a partir dos valores de PSNR obtidos. Dessa forma, abre-se o caminho para trabalhos complementares que, a partir do mapeamento realizado, possam levar à definição adequada de taxas de envio de vídeo, conforme as possibilidades da rede e do receptor, proporcionando assim melhores experiências aos usuários, bem como uma melhor utilização dos recursos relativos à infraestrutura de rede.

Estudos adicionais foram realizados, visando avaliar a influência de parâmetros relativos à escalabilidade de vídeo, bem como a análise estatística (incluindo análise de variância) dos resultados experimentais obtidos.

5 - Conclusões e trabalhos futuros

Neste artigo, com base nos resultados dos testes e uma análise estatística dos dados obtidos na transmissão de vídeo escalável, logrou-se predizer o valor do MOS de um sistema de distribuição de vídeo sob demanda sobre redes P2P, à base de regressão não linear. Um mapeamento de QoS em QoE pode então ser obtido, isto é, a partir de métricas objetivas e parâmetros quantificáveis, pode ser inferida uma métrica subjetiva como é o MOS.

Um trabalho futuro é o de implementar o esquema em um sistema real, como o Goalbit que é de código aberto, mas ainda não brinda suporte à distribuição de vídeo escalável. Outro trabalho é fazer a avaliação da qualidade sobre redes heterogêneas, considerando diferentes condições de largura de banda, resoluções de tela e capacidades de processamento dos pares da rede.

REFERÊNCIAS

[1]. 1. M. Zhang e B. Feng, "A P2P VoD system using dynamic priority," *2009 IEEE 9th Malaysia International Conference on Communications (MICC)*, pp. 518-523, 2009.

[2]. S. Khirman and P. Henriksen, "Relationship between Quality-of-service and Quality-of-Experience for Public Internet Service," *Proc. of the 3rd Workshop on Passive and Active Measurement*, 2002.

[3]. H. Kim and S. Choi, "A study on a QoS/QoE correlation model for QoE evaluation on IPTV service", *Proc. of The 12th International Conference on Advanced communication technology, ICACT'10*, pp. 1377-1382, 2010.

[4]. L. Li-yuan et al., "The Research of Quality of Experience Evaluation Method in Pervasive Computing Environment," *Pervasive Computing and Application, 1st International Symposium on* , vol., no., pp.178-182, 2006.

[5]. D. Lopez et al., "Adaptive multimedia streaming over IP based on customer oriented metrics," *Proc. of The seventh IEEE International Symposium on Computer Networks, ISCN'06*, pp. 185–191, 2006.

[6]. A. Khan and L. Sun, "QoE-driven adaptation scheme for video applications over wireless networks," IET Communications, Vol. 4, No.2 Special Issue on "Video communications over wireless networks", pp. 1337 - 1347, 2010.

[7]. M. Fiedler et al., "A generic quantitative relationship between quality of experience and quality of service," Network, IEEE, v. 24, no. 2, pp. 36–41, 2010.

[8]. M. Mushtaq and T. Ahmed. "Smooth Video Delivery for SVC Based Media Streaming Over P2P Networks," *IEEE 5th Consumer Communications and Networking Conference, CCNC'08*, pages 447-451, 2008

[9]. Sánchez, Y. et al., "P2P group communication using Scalable Video Coding," *17th IEEE International Conference on Image Processing ICIP'10*, pp.4445-4448, 2010.

[10]. C. Kiraly et al., "Effects of P2P Streaming on Video Quality," *Communications (ICC), 2010*, no. 214412, pp. 2-6, 2010.

[11]. Evalvid homepage http://www.tkn.tu-berlin.de/research/evalvid/cif.html. Acessado em 02/10/2012.

[12]. Galvis, V. R., "Transmissão de Vídeo Escalável em Redes P2P", dissertação de mestrado, Universidade de Brasília, 2013.

APÊNDICE IV – Artigo submetido ao Elsevier, Visual Communication and Image Representation Journal.

QoE Assessment in Scalable Video Distribution Over Peer-to-Peer Networks

Valmiro José Rangel Galvis, Paulo Roberto de Lira Gondim

Faculdade de Tecnologia - Universidade de Brasília – UnB
Brasília – D.F. - Brasil.

{valmiro, pgondim}@unb.br

Abstract. *Scalable video has been considered a promising option for promoting adaptation to the highly variable conditions of bandwidth in video distribution systems, and peer-to-peer (P2P) networks have received crescent attention due to the possibility of solving the bottlenecks found in Client/Server architectures. This paper presents a scheme to estimate the quality of experience (QoE) as perceived by a user receiving a scalable video distributed over a P2P network. The scheme is based on a quantitative relationship between the measured quality of service (QoS) using network parameters and the quality as perceived by the end-user. According to the experiments and measures, the relationship predicts the Mean Opinion Score (MOS) accurately.*

Keywords: QoE, QoS, H.264/SVC, MOS, P2P

1. Introduction

The distribution of video is one of the most relevant applications for the Internet, arousing great interest by the users, while involving requirements difficult to be met. Among such requirements, are included the high bandwidth commonly required and the need for traffic handling with time constraints (real time), subject to conditions and variability of network resources.

The use of peer-to-peer (P2P) networks for video streaming on the Internet has received considerable attention from researchers (e.g., [6, 18, 20]), allowing personal computers to function in a coordinated manner, as a distributed storage medium by contributing, searching, and obtaining digital content [15]. Thus, P2P architectures allow the sharing of video chunks through a direct exchange between users of the network, instead of requiring the intermediation or support of a server or a centralized entity, allowing achieve scalability of bandwidth resources, path redundancy and network capacity for self-organization.

If on one hand the architectures of P2P networks allow to solve many existing problems in client-server architectures, video streaming on P2P networks still presents challenges, considering the growing demand, limited upload capacity of peers, the heterogeneity of receivers and, in particular, large variations in the availability of bandwidth.

To address the aforementioned bandwidth variability inherent to P2P systems, transmission and coding technologies must support adaptation of transmission rates, according to the available bandwidth. Moreover, if traditional video coding technologies (such as MPEG2 or, even more recently, H.264/AVC) is used, it becomes difficult to handle different types and capacities of receivers, ranging from mobile phones to high-definition displays (LCD televisions, for example).

In this sense, the treatment of heterogeneous nodes and different bitrates within a peer-to-peer network can be benefited by the use of scalable video, where has been considered the standard H.264/SVC (Scalable Video Coding), as an extension of the standard H.264/AVC (Advanced Video Coding).

The said extension has emerged as a promising option for allowing a user to receive part of the stream according to their processing capacity, resolution of the receiver, and capabilities of the network infrastructure on which it stands. Since the scalable video is encoded in layers, the user can receive the base layer and start to play the video; on the other hand, if the network and the user have available resources to treat other layers, called enhancement layers, it is possible to improve the quality perceived.

In this context, one of the problems that has been the subject of recent researchs [1, 4, 7, 30] is the mapping between the quality of service (QoS) offered by the network and the quality of experience (QoE) perceived by the user. The need for such mapping is justified for several reasons, among them the difficulty of estimating, in an automated way and in real-time, the experience that is being lived by the end user. More than that, it is necessary to assess the impact of QoS parameters on video quality (QoV) and therefore on the QoE, and seek to establish the necessary adjustments in terms of sending rate of video sequence (or layers of the same), consistent with the capabilities of the receivers and the availability of network resources.

So, we are interested in the assessment of changes in perceived quality depending on changes in network and application parameters. That is, find a relationship between the network and application parameters and quality delivered to the end user, which allows

to predict and to quantify the value of perceived quality for, from this relationship, to develop a technique for adaptive streaming of the content or of the transmission rates.
Moreover, the lack of adequate scalable video encoders to work with live streaming made us to choose to study a system based on video on demand (VoD). Thus, this paper evaluates the impact of network and application parameters in the quality perceived by the end user in a video on demand distribution over a P2P network, with the video being encoded in the standard H.264/SVC.
The organization of this paper is as follows: section 2 presents aspects related to scalable video and P2P networks; in section 3 are discussed related work; section 4 discusses the experiments performed and the relationship between parameters used for QoS and QoE, based in nonlinear regression. Section 5, finally, outlines the conclusions and future work.

2. Scalable Video Coding and P2P Networks

This section aims to present basic concepts about scalable video and P2P networks.

2.1. Scalable Video Coding

The scalability of video is provided based on a structure of layered video coding, from which a base layer can be increased by one or more layers of refinement or improvement (enhancement layers). Among the scalable video standards, there is the standard H.264/SVC (Scalable Video Coding), which is an extension of H.264/AVC (Advanced Video Coding). The most common types of video scalability are the temporal scalability, the spatial scalability, and the scalability SNR (Signal-to-Noise Ratio) or quality scalability, as described below [10]:

- Temporal scalability: the frame rate of the enhancement layers is larger compared with the frame rates of the lower layers, including larger than the frame rate of the base layer.
- Spatial Scalability: the enhancement layers have a resolution equal or greater than the base layer or if compared to other lower layers: the spatial scalability is the possibility to transmit images with different resolutions.
- Quality Scalability (SNR): layers with spatial and temporal resolution remain the same and only the layers of quality are enhanced. The higher the SNR, the higher the quality of the image produced.

For coding the videos used in this article, we used the medium granularity scalability (MGS) as well as the temporal and spatial scalability combined.

Some advantages that scalable video coding offers, compared with non-scalable video are listed below:

- In the case of simulcast, multiple versions of the same video will be available to be served to different users with different processing, bandwidth and storage capabilities. Obviously, these different versions have a higher degree of redundancy, and even if they have different bitrates all streams represent the same content. The scalable video coding strives to reduce this redundancy and, therefore, may produce a video stream that requires significantly less storage space than the sum of all versions of a simulcast video stream.
- Scalable video streams can be encoded with different bitrates. So, you can increase the options of choosing a bitrate from a broad set of possible values.
- Management of different versions of the same video bitrate is avoided. With the scalable video one bitstream can serve to multiple users with different needs. As a result the adjustment of the bitrate is simplified. To adjust there is no need to swap between two separate bitstreams, because this operation can be done using the same video stream. This convenience improves the flexibility of the video stream and increases the system robustness against transmission link failures.
- Encoding the video with a layered structure allows to associate different degrees of priority to each layer. The base layer of a scalable video stream contains information that is essential for the playback of the video. Thus, the base layer is the most important part of the video stream. So, the higher the order of the enhancement layer the lower its priority.

  The advantage of this layered structure is that specific parts of the video can be prioritized. In a network with limited bandwidth that prioritization allows to prefer data packets that are essential for video playback. Therefore, in cases where there is not enough bandwidth to receive the complete video, you can offer at least one low quality version of the video.
- Especially for P2P networks, scalable video coding offers another advantage that is related to the advantages mentioned above. Generally, all peers in a network do not form a homogeneous group, differing by the available bandwidth or the computing capabilities. Thus, in the case of the simulcast, the video would require different bitrates and therefore also need different streams. This would

lead to the problem of having to break the overlay in different subgroups where each subgroup shares only one version of the video stream [12, 13].

2.2. VoD over Peer-to-Peer Networks

P2P networks have evolved as a promising paradigm for distributing video in large scale, in a manner more efficient than the traditional architectures [1]. Due to overload in the server side in systems based in the unicast architecture Client/Server, where you experience bottlenecks when the number of clients increases in the network, having losses in bandwidth and in the speed of the service, P2P networks arise as an alternative to relieve the load on the server in content delivery systems.

We can see, as pointed out in [33], that most of the video streaming systems over P2P networks (e.g., PPLive, SopCast, TV Ants, UUSee, CoolStreaming) do not use video scalability yet (thus serving only one version of the video stream for all pairs), and therefore have limited support for heterogeneous pairs.

Thus, it is expected that the SVC codification, if used in systems P2PVoD (peer-to-peer video-on-demand), will be capable of providing an efficient adaptation to heterogeneous resources and dynamic behavior of the network, and allows participation of the peers in content distribution in large scale [1].

3. Related Work

Two categories are considered here. The first one, talking specifically about the use of scalable vídeo over P2P networks, and, the second one, talking about issues related to the mapping of QoS on QoE. In these two categories, we can observe the lack of work involving mapping QoS on QoE in P2P networks using scalable video, which shows an aspect of originality in the proposal presented here.

Within the first category, in [18], the authors use SVC to ensure smooth delivery of the video content between peers of the network choosing pairs that contribute, with different parts of the video, according to the offered QoS. The video is encoded to provide only temporal scalability (7.5, 15 and 30fps), has a QCIF resolution (176x144) and only one option of SNR scalability. The most important layer of the video (the base layer) is obtained from the pair with the best QoS offered. The QoS offered is calculated using the bandwidth that the pair could contribute during the video transmission. The bandwidth available between the transmitter and receiver is estimated by the RTT (Round Trip Time). Simulations for test the performance of the system where made with the quality of a scalable video streaming sent over a peer-to-peer network with a

similar topology of a Gnutella network [21]. There was an improvement in the throughput rate of the video in the pairs and a better video quality received. The assessment of the quality was made using the PSNR (Peak Signal-to-Noise Ratio) metric.

In [19], the authors show the advantages of the use of scalable video coding with a P2P network, and propose an adaptive streaming over a P2P network based on the topology of BitTorrent [22], for heterogeneous networks. The quality is measured using the PSNR, being shown, by simulations, the adjustment of the bitstream with respect to networks of different bandwidths subject to fluctuations.

Nunes et al. [25] implement a prototype for the distribution of scalable video using a prioritized sliding window algorithm choosing the proper parts that are already stored in each pair and reserve a download window with the size of a chunk that prioritizes the base layer of the scalable video. The effectiveness of the algorithm is evaluated comparing with the standard selection algorithm of parts of BitTorrent from the point of view of the time of download and upload and delay in the delivery of live video. Finally, another work of this first category is presented in [20].

Additionally, researches on the relationship between QoS and QoE have been made [1-6], showing initially that this relationship is not linear [3, 4, 5]. In [1], Fiedler et al. proposed an exponential relationship between QoE and QoS, and using the data of test results of MOS (Mean Opinion Score), found a curve that best fits those results, than the logarithmic relationship proposed in [3].

Kim and Choi [4] make a study of a correlation model between QoS and QoE to measure the QoE of an IPTV service. QoS is defined as a function of the loss (PER), burst level (U), jitter (J), packet delay (A) and the bandwidth (B), as shown in equation 1.

$$QoS(X) = K(\beta_1 PER + \beta_2 U + \beta_3 J + \beta_4 A + \beta_5 B) \qquad (1)$$

Where the coefficient β_i is a weight defined by the relative importance of each QoS parameter for the IPTV service, recommended by the organizations responsible for regulating the quality standards (e.g., ITU-T, IETF, etc.). And the constant K is a factor that determinates the QoS depending on the network access technology to the IPTV service.

With the QoS defined by equation 1, the proposed correlation between QoE and QoS is given by equation 2.

$$MOS = Qr(1 - QoS(X))^{\frac{CQoS(X)}{R}} \qquad (2)$$

Where Qr is a factor that limits the MOS value according to the resolution of the terminal, C is a value that depends on the type of service signed by the user (for example, premium), and R is determined by the structure of the frame according to the GOP (Group of Pictures) size.

In [6], the authors propose a model for adaptive multimedia streaming over IP based on customer oriented metrics. Based on experimental design techniques, find a relationship obtained using empirical modeling of experimental data, considering controllable parameters (eg, the bit rate of the video) and uncontrollable (eg, loss, delay and jitter). The influence of each factor on the response is determined by a Pareto analysis, determining after the contribution of these factors in the variability of the system by doing an ANOVA (Analysis of Variance). According to the results, the factors with the greatest influence on the quality are packet loss, delay, jitter and video coding rate, generating from the experimental data, and by means of a nonlinear regression, an equation to estimate the quality of experience or video (QoEVídeo) and the quality of experience of the audio (QoEAudio) is found (equations 3 and 4).

$$QoE_{Video} = \delta_1 PER - \delta_2 A + \delta_3 J + \delta_4 CV + \delta_5 D^2 + \delta_6 D \cdot J + \delta_7 D \cdot CV \qquad (3)$$

$$QoE_{Audio} = \varphi_1 PER - \varphi_2 A + \varphi_3 J - \varphi_4 CV + \varphi_5 D^2 + \varphi_6 D \cdot J + \varphi_7 D \cdot CV \qquad (4)$$

Where PER, J and A are the same variables as previously defined, and CV is the encoding rate of the video.

Making a multi-objective optimization for each response QoE_i (audio and video) a function of desire $d_i(QoE_i)$ determines a value between 0 and 1 for the possible values of QoE_i, where $d_i(QoE_i) = 0$ represents a value totally unwanted and $d_i(QoE_i) = 1$ is an optimal value in the response. The overall desire value (D) is calculated using a geometric mean of all the values of the desires function. Thus, D is calculated as shown in Equation 5.

$$D = \{d_1(QoE_1) \cdot d_2(QoE_2) \ldots d_n(QoE_n)\}^{\frac{1}{n}} \qquad (5)$$

The quantitative value D can be used to make management procedures of services based on network metrics to control the multimedia quality provided by modifying the value of the encoding rate of the video, which is a controllable parameter.

Thus the overall QoE can be defined using equation 6:

$$QoE = \sqrt{d_1(QoE_{Video}) \cdot d_2(QoE_{Audio})} \qquad (6)$$

Recalling that the QoE is expressed as a desire function and its value ranges from 0 to 1 in this case.

In [7], the authors propose a prediction model of MOS for MPEG-4 sequences over a wireless network. After classify and group the video sequences is made a nonlinear regression to fit simulation data to an equation that describes the behavior of the MOS. The authors combined different frame rates in coding with different send rates (application level) and with different loss rates (network level) thus succeeded in estimating MOS considering content, network and application parameters. The equation used to predict the MOS is shown below.

$$MOS = \frac{\gamma_1 + \gamma_2 FR + \gamma_3 \ln SBR}{1 + \gamma_4 PER + \gamma_5 PER^2} \qquad (7)$$

Thus, for each video sequence they calculated the PSNR, then mapped it into MOS values (MOS obtained), and with a nonlinear regression achieved an equation to estimate the MOS (MOS predicted) using the parameters of Frame Rate (FR), Sender bitrate (SBR) and Packet Error Rate (PER).

In [8], the authors quantify the effects of jitter on the perceptual quality of video over a 3G network, proposing a modeling of the correlation between QoS and QoE. Using the algorithm recommended in [9] to calculate the jitter, the authors implement a tool for estimating the QoE from network parameters, also considering user ratings in real time on a device with an Android operating system. This prediction of QoE is used to make the selection process of the network that best serves the end user.

Equation 8 shows the correlation between QoS and QoE as defined by the authors.

$$MOS = -\rho_1 J^{\rho_2} + \rho_3 \qquad (8)$$

Besides the jitter was also studied the effect of burst on the MOS, coming in a equation similar to that obtained previously:

$$MOS = \mu_1 J^{-\mu_2} - \mu_3 \qquad (9)$$

4. Mapping QoS into QoE

4.1 Initial considerations

The assessing of the quality of a video can be done in an objective or a subjective way. Subjective testing methods require a human perspective to evaluate quality or quality

differences between two images or videos. Methods of objective tests are mathematical models that estimate the perceived video quality by an average user.

One of the goals when building an application on a communication network is the user satisfaction, which can lead to a satisfactory quality of experience. For a multimedia application the main component of the QoE is the perceived quality, that is, the perceived quality of the multimedia stream, as the stream is viewed by the end user, which is clearly a subjective concept.

The difficulties behind the subjective evaluation of P2P systems arise from the fact that such systems usually cover very large geographical areas, involving a large number of peers. Taking this into account, the performance of these systems is significantly affected by the unpredictable behavior of the users and the status of sub-networks that make up the whole network, making them difficult to model. In addition to that, the subjective evaluation can be a costly and a time consuming process, and the method cannot be used to monitor quality in real time [28].

4.2 Architecture for testing

The architecture considered is based on the use of discrete event simulation. The block diagram shown below describes how the assessment of quality is calculated in the distribution system for VoD over P2P networks, used in this work.

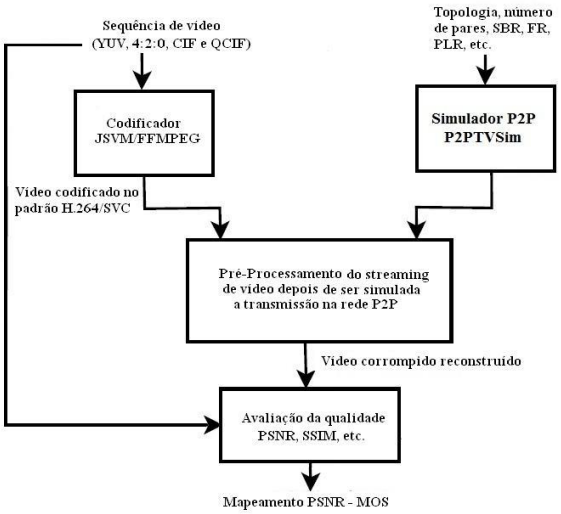

Fig. 1. Scenario for testing.

The first step is to encode the video stream, which is in YUV (4:2:0), available in CIF (176x144) and QCIF (352x288), with the JSVM software. The encoding is done using three types of scalability (spatial, temporal and quality), and a medium grain scalability (MGS). The encoded sequence contains: two spatial layers, CIF and QCIF resolutions.

Five temporal layers, frame rates of 1.875, 3.75, 7.5, 15 and 30 fps (frames per second). Three layers of quality SNR (Q0, Q1 and Q2) and a GOP of 16. Combining the different scalabilities, the video encoded in the H.264/SVC standard in this particular case, can be up to thirty layers, one base layer and twenty nine enhancement layers.

With the videos coded and classified, the next step is to simulate the video distribution over a P2P network using the simulator P2PTVSim. This simulator needs a configuration file where are specified the network parameters (for example, the number of pairs, the number and size of the chunks, the probability of loss, jitter, download and upload bandwidth of the pairs the upload bandwidth of the source, the bitrate of the video packet scheduler, etc.). The most important parameters and their values used in this work will be explained below.

The first parameter is the number of pairs which is the parameter NumPeers which was set in 1000 to think in a large scale network and to have a considerable number of peers. Along with the number of pairs we have to consider the second parameter which is the overlay type. It was set to be a network based on a randomly generated mesh with twenty neighbors for each pair. So, the option GNR was selected because a certain number of neighbors is enough to ensure the quality for each peer [33], i.e., the network is a random graph with 1000 peers and a degree of 20.

The third important parameter is the scheduler (LATEST_BA), where the peer selects the last useful chunk that in its window and sends it to a neighbor that selects in a bandwidth aware way and, given the circumstances of bandwidth of the peers (the same for all peers), has the same performance of the LATEST_RANDOM scheduler, where the peer selects the last useful chunk in its window and sends it to a neighbor that selects randomly. The other option, CONFIG_PEER, allows you to configure the peer in order to choose what will be done first: select the chunk to be sent and then the peer or select the peer and then the chunk to be sent.

Finally, we have the last two parameters of interest, the size of the chunks where a chunk has the same size of a GOP, to make the system to be aware of the format of the content. Using a chunk size equal to the size of a GOP is an approach that improves the video quality [30]. The last parameter is the bandwidth (PeerBandwidths) of the peers and the source, 1 Mbps for upload to the peer and the source, and the download bandwidth being much larger to ensure that the bottleneck is in the upload link and not in the download link [30, 33].

Setting the simulation with the values of the above mentioned, each simulation was repeated one hundred times (on different random topologies), and an average of the data is done to make certain statistical validity. The values for the configuration parameters of the simulator, and part of this work is based on [30], but the work of these authors is about live streaming and with the H.264/AVC standard, non-scalable version of the standard H.264.

As a result of the simulations, P2PTVSim generates several files with network statistics. One of these files contains a list of lost packets (chunks). This file is used to remove the lost packets streaming encoded as part of the preprocessing. The second stage of preprocessing consists in replacing each lost frame immediately by the preceding frame, so the reference video and the "received" video will have the same number of frames, an important criterion when assessing SSIM and PSNR. The last step of the preprocessing is to decode the video in the "original" format YUV (4:2:0) and with a CIF resolution, which is the highest resolution available.

The evaluation of the quality is done by comparing the original video and the received video using the framework Evalvid [31] to calculate the PSNR and the SSIM. The values of FR (Frame Rate), SBR (Sender Bit Rate) and PER (Packet Error Rate) were modified, and, for each combination of values of different network and application parameters an assessment is made. The parameters FR, SBR and PER were selected because of their influence on the perceived quality showed by an ANOVA analysis. Therefore, these parameters are used to modeling the MOS prediction, in order to define an equation that relates them to an estimate of the quality of the experience of the end user, that is, an equation that has the form MOS = f (FR, SBR, PER).

4.3 Videos and parameters values

The videos used were encoded in the standard H.264/SVC using the JSVM software with temporal, spatial and SNR scalabilities, with a GOP size of 16, MGS refinement, and am AVC base layer mode. We considered twelve video sequences well known in the area of evaluation of video quality.

The videos have been classified and grouped by their temporal and spatial characteristics according to [7]: slight movement (SM), gentle walk (GW) and rapid movement (RM). Table 1 shows the videos used and the values of the parameters (SBR, FR and PER).

Table 2. Videos and parameters values.

Class	Sequence	FR (fps)	SBR (kbps)	PER
SM	Akiyo	1.875, 3.75, 7.5, 15, 30	48, 128, 192, 256, 512	0.01, 0.05, 0.1, 0.15, 0.2
	Bridge-Close			
	Grandma			
	Suzie			
GW	Carphone		96, 192, 320, 480, 576	
	Foreman			
	Rugby			
	Table-Tennis			
RM	Coastguard		128, 288, 512, 640, 1024, 1440	
	Football			
	Stefan			

SM video sequences have regions of interest with little movement, e.g., a quiet person with a static background and moving only his lips and eyes, and sometimes hands, that is, the video only has those regions of interest. The sequences of the GW class have a continuous change of scene in order, for example, a video call. The RM sequences are characterized by having fast movements of objects within the video with the background having a generally uniform color, for example, a football game. Examples for each category are shown in figure 2.

Fig. 2. Examples of the videos for each category (SM, GW, RM).

We choose working with video on demand because of the difficulty of having a H.264/SVC video encoder suitable to work with live streaming.

Next, we are going to talk about the parameters.

The values of packet loss (PER) is between 1% and 20% in increments of 0.05. Losses greater than 20% of packets generate a very bad quality of the video and there is no difference between these MOS for values greater than 20% (e.g., there is no difference between the quality of a video distributed on a network with losses 30%, and a video distributed on a network with losses of 40%). The MOS mapped for values of packet loss above 20% is bad for all the cases.

Taking advantage of the scalable video coding, and the settings used to encode the video sequences, we can use five frame rates (FR): 1.875, 3.75, 7.5, 15 and 30 fps. Using the bitrate file, which can be generated from the scalable video with the JSVM tool, were chosen the different rates of the video (SBR). In the experiments were used three videos of each type (SM, GW, RM) for training and a video of each type for testing (four videos in total for each type).

4.4 QoE model prediction

To do the mapping of QoS into QoE we propose an equation of low complexitythat serves as a predictor of MOS. This equation is obtained by nonlinear regression using the tools nlintool and polyfit of MATLAB™. As we do not have subjective MOS values available, these MOS values are obtained objectively with the mapping tool of PSNR into MOS of the Evalvid that use the values presented in Table 3.

Tabela 3. Mapping PSNR into MOS

MOS	PSNR (dB)
5 (Excellent)	>37
4 (Good)	31 – 36.9
3 (Fair)	25 – 30.9
2 (Poor)	20 – 24.9
1 (Bad)	<19.9

The regressions showed that the equation to express the MOS in terms SBR is as shown below:

$$MOS(SBR) = a_1 + b_1 ln(SBR) \qquad (10)$$

The FR values are taken from the configuration file of the scalable video coding. These values are the values of fps available on the temporal scalability of the videos. The relationship between the MOS and the FR is shown in equation 11:

$$MOS(FR) = a_2 + b_2 FR \qquad (11)$$

The regressions showed that the curve that better fits to the MOS values for packets loss is exponentially. Equation 12 shows the relationship between MOS and PER.

$$MOS(PER) = a_3 \exp(b_3 PER) \qquad (12)$$

It is possible to propose a predictor of MOS, using the equations 11, 12 and 13, that relates QoS with QoE. The estimation of QoE is then proposed as being based on equation 13:

$$MOS(FR, SBR, PER) = [a + bFR + cLn(SBR)]e^{dPER} \qquad (13)$$

The coefficients a, b, c, d will vary according to the type of video (SM, GW, RM) while maintaining the same equation. To obtain these coefficients we performed a nonlinear regression, based on the Levenberg-Marquardt algorithm and considering training sets and rating based on sequences described in Section 4.3.

Evaluating the appropriateness of the predictor is based on the coefficient of determination (R^2) and the the root mean square error. The value of R^2 can be calculated using:

$$R^2 = \frac{\sum_1^n (\hat{y}_i - \bar{y})^2}{\sum_1^n (\hat{y}_i - \bar{y})^2 + \sum_1^n (\hat{y}_i - y_i)^2} \qquad (14)$$

Where $\hat{y}_i$ is the estimated (predicted) value, $\bar{y}$ is the average value of the observations and y_i is the observed value. The R^2 value is between 0 and 1.

The RMSE is a measure of the difference between a value predicted by a model and the observed value and is given by:

$$RMSE = \sqrt{\frac{\sum_1^n (\hat{y}_i - y_i)^2}{n}} \qquad (15)$$

The values obtained for the considered parameters are shown in table 4, in which we can observe a good fit of the prediction model, based on the values of R^2 and RMSE, but with a better suitability for the SM sequences.

Tabela 4. Values of the Coefficients, R^2 and RMSE.

Coeficiente	**SM**	**GW**	**RM**
A	3.10	2.89	2.37
B	-0.0017	-0.0012	-0.0011
C	0.133	0.1495	0.18
D	-1.86	-2.031	-2.55
R^2	93.25%	90.18%	87.46%
RMSE	0.086	0.241	0.273

An analysis of variance (three-way ANOVA) was performed to study the impact on the MOS of each of the parameters (SBR, FR, PER) of the equation, showing that the most influential parameter in the outcome of the MOS is the SBR. For reasons of space in this article, more details can be found in [32].

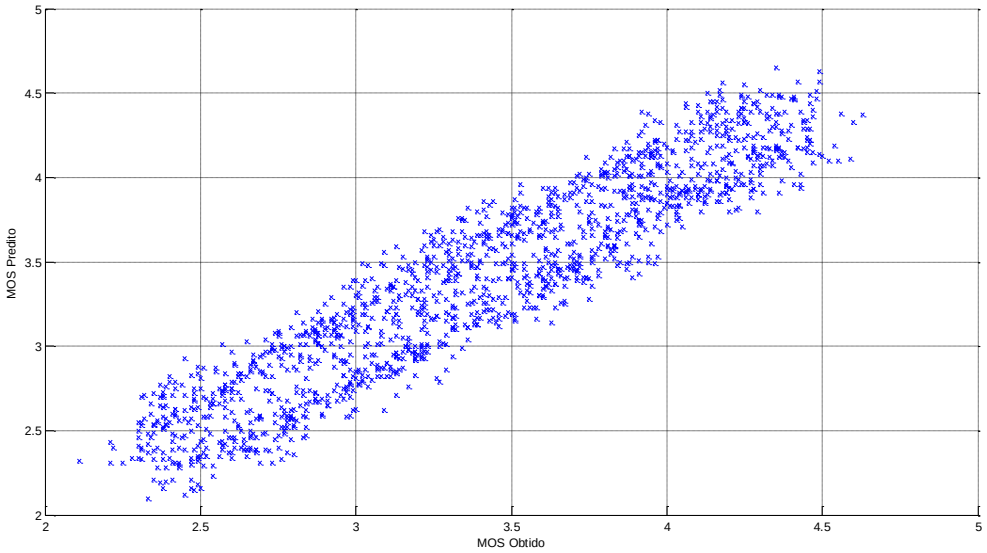

Figura 8. Predicted MOS vs Obtained MOS.

Figure 8 shows a graph that compares the MOS values predicted using the equation 4 with the values obtained by measuring the MOS. Figure 9 shows the impact of the three QoS parameters (SBR, FR, PER) on the quality of the videos for the three content types (SM, GW, RM). The results with the largest variations of MOS were obtained for the category RM.

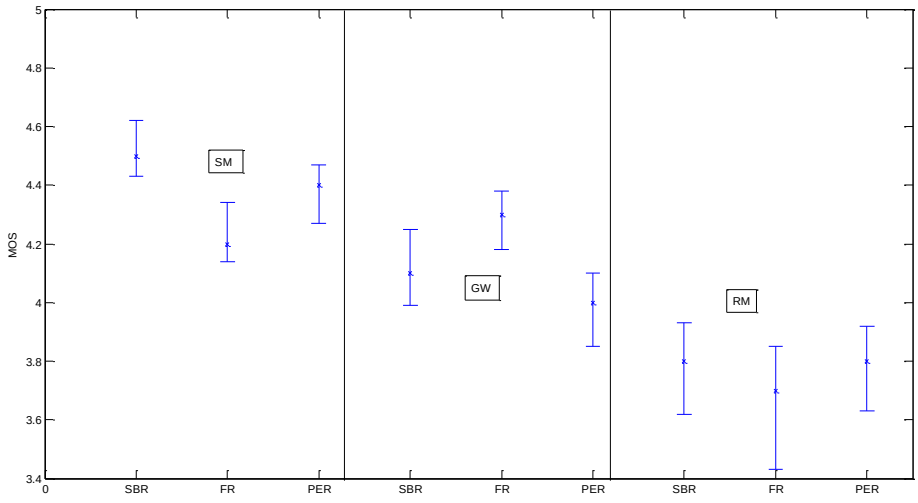

Fig. 9. Effects of SBR, FR and PER on the MOS.

5. Conclusions

Based on the systems studied in this paper we propose a mapping QoS-QoE relationship between quality of service (defined by metrics of network and application) and

subjective quality (estimated by MOS - Mean Opinion Score) for scalable video-on-demand distribution systems over P2P networks.

An objective assessment of the performance of video distribution over P2P networks (using PSNR) was made, which was mapped in the probable QoE experienced by the user from the QoS provided by the network, estimated considering application (SBR and FR) and network (PER) parameters.

Based on the found equation, it has been possible, for the system considered and the used sequences, mapping QoS into QoE, that is, from objective metrics and quantifiable parameters can be inferred a subjective metric as the MOS.

Additional experiments can be performed to try to improve the performance of the predictor, i.e., obtain a greater volume of data to better adjust the coefficients of the equation and to improve the yield.

6. Acknowledgements

The authors would like to thank to CAPES and Universidade de Brasilia for the adequate support.

REFERÊNCIAS

[1]. M. Fiedler et al., “A generic quantitative relationship between quality of experience and quality of service,” Network, IEEE, v. 24, no. 2, pp. 36–41, 2010.

[2]. F. Agboma et al., “QoE analysis of a peer-to-peer television system,” *Proc. of IADIS Telecommunications Networks and Systems*, pp. 114-119, 2008.

[3]. S. Khirman and P. Henriksen, “Relationship between Quality-of-service and Quality-of-Experience for Public Internet Service,” *Proc. of the 3rd Workshop on Passive and Active Measurement*, 2002.

[4]. H. Kim and S. Choi, “A study on a QoS/QoE correlation model for QoE evaluation on IPTV service”, *Proc. of The 12th International Conference on Advanced communication technology, ICACT'10,* pp. 1377-1382, 2010.

[5]. L. Li-yuan et al., "The Research of Quality of Experience Evaluation Method in Pervasive Computing Environment," *Pervasive Computing and Application, 1st International Symposium on* , vol., no., pp.178-182, 2006.

[6]. D. Lopez et al., “Adaptive multimedia streaming over IP based on customer oriented metrics,” *Proc. of The seventh IEEE International Symposium on Computer Networks, ISCN’06*, pp. 185–191, 2006.

[7]. A. Khan and L. Sun, “QoE-driven adaptation scheme for video applications over wireless networks,” IET Communications, Vol. 4, No.2 Special Issue on "Video communications over wireless networks", pp. 1337 - 1347, 2010.

[8]. Ickin, S. et al., "The effects of Packet Delay Variation on the perceptual quality of video," *IEEE 35th Conference on Local Computer Networks (LCN)*, vol. 4, no. 12, pp.663-668, 2010.

[9]. IETF, RFC 3393, 2002. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)

[10]. P. Thomas and K. George, “Scalable Video Streaming Traffic Delivery in IP/UMTS Networking Environments”, Journal of Multimedia, Vol. 2, no. 2, 2007.

[11]. Z. Avramova and D. D. Vleeschauwer, “Comparison of simulcast and scalable video coding in terms of the required capacity in an IPTV network,” *Packet Video*, pp. 113-122, 2007.

[12]. Heiko Schwarz et al., “Overview of the Scalable Video Coding Extension of the H.264/AVC Standard”, *IEEE Transactions On Circuits And Systems For Video Technology*, Vol. 17, No. 9, 2007.

[13]. Heiko Schwarz et al., “The Scalable Video Coding Amendment of H.264/AVC Standard”, Heinrich-Hertz-Institut, 2008.

[14]. S. Androutsellis-Theotokis e D. Spinellis, “A survey of peer-to-peer content distribution technologies,” ACM Computing Surveys, vol. 36, pp. 335-371, 2004.

[15]. M. Zhang e B. Feng, “A P2P VoD system using dynamic priority,” *2009 IEEE 9th*

[16]. J.-H. Roh e S.-H. Jin, “Video-on-Demand Streaming in P2P Environment,” *2007 IEEE International Symposium on Consumer Electronics*, pp. 1-5, 2007.

[17]. D. Wang e C.K. Yeo, “Exploring Locality of Reference in P2P VoD Systems,” *IEEE Global Telecommunications Conference, GLOBECOM ‘10*, pp. 1-6, 2010.

[18]. M. Mushtaq and T. Ahmed. “Smooth Video Delivery for SVC Based Media Streaming Over P2P Networks,” *IEEE 5th Consumer Communications and Networking Conference, CCNC’08*, pages 447-451, 2008

[19]. G. Zhang and C. Yuan. “Self-adaptive Peer-to-Peer streaming for heterogeneous networks using Scalable Video Coding,” *IEEE 12th International Conference on Communication Technology*, pp. 1390-1393, 2010.

[20]. Sánchez, Y. et al., "P2P group communication using Scalable Video Coding," *17th IEEE International Conference on Image Processing ICIP'10*, pp.4445-4448, 2010.

[21]. Gnutella Homepage. http://www.gnutellaforums.com/ acessado em 18/05/2013.

[22]. BitTorrent Homepage http://www.bittorrent.com acessado em 18/05/2013.

[23]. de Asis Lopez Fuentes, F.; , "Adaptive Mechanism for P2P Video Streaming Using SVC and MDC," *International Conference on Complex, Intelligent and Software Intensive Systems CISIS'10*, pp.457-462, 2010.

[24]. PlanetLab Homepage: https://www.planet-lab.org/ acessado em 18/05/2013.

[25]. R. P. Nunes et al., "Scalable Video Distribution in Peer-to-Peer Architecture," *Proceedings of the 10a Conferência sobre Redes de Computadores CRC'10*, pp. 95-100, 2010.

[26]. Abbasi, U. et al., "Delivering scalable video coding using P2P Small-World based push-pull mechanism," *Global Information Infrastructure Symposium, GIIS '09*, pp.1-7, 2009.

[27]. Zhang, M. et al., " A Peer-to-Peer Network for Live Media Streaming - Using a Push-Pull Approach". *Proceedings of ACM Multimedia 2005*, short paper, 2005.

[28]. F. Agboma et al., "QoE analysis of a peer-to-peer television system," *Proc. of IADIS Telecommunications Networks and Systems*, pp. 114-119, 2008.

[29]. P2PTVSim homepage http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim. Acessado em 18/05/2013.

[30]. C. Kiraly et al., "Effects of P2P Streaming on Video Quality," *Communications (ICC), 2010*, no. 214412, pp. 2-6, 2010.

[31]. Evalvid homepage http://www.tkn.tu-berlin.de/research/evalvid/cif.html Acessado em 018/05/2013.

[32]. Galvis, V. R., "Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P)", dissertação de mestrado, Universidade de Brasília, 2013.

[33]. L. Abeni et al., "On the Optimal Scheduling of Streaming Applications in Unstructured Meshes," *Proceedings of the 8th International IFIP-TC 6 Networking Conference, NETWORKING '09*, pp. 117-130, 2009.