Acesso interativo para aplicações P2P de streaming de vídeo *

Luiz J. H. Filho[1], Carlo K. da S. Rodrigues[2], Rosa M. M. Leão[1]

[1] Programa de Engenharia e Sistemas de Computação – COPPE/PESC
Universidade Federal do Rio de Janeiro – UFRJ
Rio de Janeiro – RJ – 21941-972 – CxP 68511 – Brasil

[2]Centro de Desenvolvimento de Sistemas - Departamento de Ciência e Tecnologia
Exército Brasileiro - QGEx - Bloco G - 2o. Piso - Brasília - DF - 70630-901 - Brasil

{ljhfilho,rosam}@land.ufrj.br, carlokleber@gmail.com

Abstract. *This article presents a new algorithm for video streaming based on the BitTorrent protocol. As main innovatives aspects, we have that the multimedia object chunks are categorized by their playing times and there is the deployment of distinct chunk-selection policies. The analysis and the validation are both carried out through simulations using workloads from a real multimedia server. The final results outline the efficiency of the new algorithm and, comparing to other works in the literature, show optimizations, for example, of up to 88%, 98% and 57% over the metrics number of available chunks at the playing time, start playing time and average waiting time, respectively.*

Resumo. *Este artigo propõe um novo algoritmo para streaming de vídeo baseado no protocolo BitTorrent. Sua concepção tem, como inovações principais, a implementação de prioridades para pedaços (unidades, blocos) do objeto multimídia e o emprego de diferentes políticas de seleção desses pedaços. A análise e a validação são feitas por meio de simulações usando cargas de um servidor multimídia real. Os resultados evidenciam a eficiência do novo algoritmo e, em relação a outros trabalhos da literatura, assinalam otimizações, por exemplo, de até 88%, 98% e 57% nas métricas número de pedaços não disponíveis no momento de exibição, tempo de início de exibição do objeto e tempo médio de espera por uma unidade de dados (pedaço, bloco), respectivamente.*

1. Introdução

Diversas propostas da literatura para aplicações de *streaming* de vídeo são baseadas na arquitetura cliente/servidor. No entanto, esta arquitetura possui escalabilidade limitada com o aumento do número de usuários. Para aumentar essa escalabilidade, duas importantes abordagens tem sido estudadas: *IP Multicast* e *Peer-to-Peer* (P2P).

A abordagem *IP Multicast* tem um servidor centralizado que realiza a transmissão de dados para os usuários por meio de canais *multicast*. A eficiência do uso da banda é conseguida porque um mesmo canal pode ser compartilhado por diversos usuários. Vários esquemas desse tipo já foram apresentados na literatura. Por exemplo, os protocolos de *Patching* [Hua et al. 1998, Rodrigues and Leão 2007] e *Stream Merging*

*Este trabalho é parcialmente financiado pelo CNPq e Faperj.

[Eager et al. 2000]. Conjuntamente a esses protocolos também é possível utilizar-se uma gerência de *buffer* local [Rodrigues et al. 2008]. O maior limitante dessa abordagem é o fato de que o serviço *multicast* nem sempre está habilitado de forma abrangente por toda a rede de comunicação a ser utilizada.

Na abordagem *Peer-to-Peer* (P2P) [Guo et al. 2003, Huang and Ross 2007a, Huang and Ross 2007b] também existe um servidor centralizado, mas apenas canais *unicast* são usualmente utilizados para transmissão dos dados e os próprios usuários, denominados *peers*, colaboram na transmissão dos objetos. O fato dos usuários também poderem realizar a transmissão de dados reduz substancialmente o número de requisições diretas ao servidor, aumentando a escalabilidade da arquitetura. Uma das maiores limitações dessa abordagem reside na incapacidade de garantir conexões estáveis e confiáveis entre os *peers*. O ambiente de rede é naturalmente dinâmico. Os requisitos de banda podem oscilar significativamente, pois os *peers* são livres para entrar/sair do sistema tão freqüentemente quanto desejarem.

Duas arquiteturas são usadas para a transmissão de vídeo P2P: arquitetura em árvore e arquitetura em malha [Liu et al. 2008]. A arquitetura em árvore possui uma grande desvantagem que é a construção e manutenção da árvore sobreposta (*overlay*) [Liu et al. 2008]. Portanto, optamos pela arquitetura baseada em malha. Consideramos o protocolo BitTorrent [Cohen 2003] pela sua eficiência, já mostrada em diversos trabalhos da literatura, para distribuição de conteúdo em redes P2P basedas em arquitetura em malha.

No BitTorrent o objeto a ser distribuído é dividido em pedaços (*chunks*, blocos ou unidades) que são transmitidos não seqüencialmente. Um *peer* pode recuperar diferentes pedaços do objeto a partir de diferentes *peers* simultaneamente. A velocidade com que um objeto é transferido pode então ser substancialmente maior que aquela atingida quando o objeto é recuperado a partir de apenas um único *peer* ou de um único servidor. Outras características importantes do BitTorrent são as suas estratégias de incentivo para reduzir os chamados *free-riders* e a sua política de seleção dos pedaços a serem recuperados. No entanto, apesar de sua eficiência para compartilhamento de arquivos, o protocolo BitTorrent não apresenta o mesmo desempenho quando usado para *streaming* de vídeo [Vlavianos et al. 2006]. Isso ocorre devido à sua política de seleção de pedaços, que prioriza o mais raro, fazendo com que um pedaço possa não estar disponível no momento da sua visualização.

A política de recuperação de pedaços seqüencial é uma possibilidade óbvia para aplicações de *streaming* onde usuários acessam seqüencialmente um objeto. No entanto, esta política prejudica uma das mais importantes propriedades do BitTorrent que é a eficiência em replicar os pedaços mais raros rapidamente. A política seqüencial também pode prejudicar, a depender do tipo de acesso dos usuários, a estratégia de incentivos. Por exemplo, considerando um acesso em que todos os usuários desejam ver a mesma informação desde o início até o seu final, um *peer* que acabou de entrar no enxame (*swarm*) poderá recuperar pedaços de todos os que entraram antes dele. Entretanto, devido à política seqüencial, ele não terá nada a oferecer.

Diversas são as propostas de adaptação do BitTorrent para suportar aplicações de *streaming* de vídeo (por exemplo, [Vlavianos et al. 2006, Zhou et al. 2007,

Shah and Pâris 2007, Carlsson and Eager 2007]). Todas foram elaboradas para tornar a transmissão eficiente quando usuários acessam seqüencialmente um vídeo pré-armazenado em tempo real. A proposta de [Shah and Pâris 2007] usa uma janela deslizante que contém os pedaços mais próximos de serem tocados. A política para recuperação dos pedaços dentro da janela é a do pedaço mais raro primeiro (*rarest first*), que é a mesma política do BitTorrent. A diferença para o BitTorrent é que a busca do pedaço mais raro é restrita à janela deslizante. Em [Carlsson and Eager 2007] são definidas duas políticas probabilísticas para recuperação dos pedaços. Na primeira um pedaço é selecionado segundo uma distribuição Bernoulli: com probabilidade p os pedaços são recuperados seqüencialmente e com probabilidade $(1 - p)$ é recuperado o pedaço mais raro. Na segunda política é usada uma distribuição *Zipf* para selecionar o pedaço a ser recuperado. Os trabalhos de [Vlavianos et al. 2006, Zhou et al. 2007] dividem os pedaços em conjuntos com prioridades diferentes. A seleção de um pedaço depende da probabilidade atribuída ao conjunto ao qual ele pertence e de uma determinada política usada para seleção (por exemplo, o pedaço mais raro do conjunto).

A principal contribuição deste artigo é a apresentação de uma proposta para aplicações de *streaming* de vídeo visando tornar a recuperação eficiente quando o acesso dos usuários é interativo. O acesso interativo é fundamental, por exemplo, para aplicações de ensino a distância [de Souza e Silva et al. 2006]. Usamos a idéia de classificar os pedaços de acordo com a prioridade em relação ao momento de exibição [Vlavianos et al. 2006, Zhou et al. 2007], em conjunto com um modelo de comportamento do usuário [de Vielmond et al. 2007]. Desta forma, pedaços são recuperados de acordo com a sua prioridade e usando diferentes políticas: o pedaço mais raro, o pedaço previsto pelo modelo, o próximo pedaço a ser tocado. Avaliamos a nossa proposta por meio de simulações com cargas geradas a partir de *logs* de comportamento de usuário, coletados do sistema RIO (*Random I/O System*) [Netto et al. 2005]. Diferentes métricas de performance são consideradas e análises competitivas são feitas com outras propostas da literatura. Os resultados indicam otimizações de até uma ordem de grandeza nas métricas consideradas.

O restante deste texto tem a seguinte organização. Na Seção 2 citamos os trabalhos em que nos baseamos e que usamos nas análises competitivas. Descrevemos a nossa proposta na Seção 3 e apresentamos os resultados na Seção 4. As conclusões e trabalhos futuros estão na Seção 5.

2. Trabalhos Relacionados

Nesta seção revisamos brevemente três trabalhos da literatura que estão relacionados ao nosso. O primeiro refere-se ao protocolo BitTorrent [Cohen 2003], e constitui-se um fundamento para o entendimento deste texto. Já o segundo [Vlavianos et al. 2006] e o terceiro [Zhou et al. 2007] trabalhos têm suas relevâncias destacadas por serem protocolos recentemente desenvolvidos para *streaming* de vídeo e por serem baseados no protocolo BitTorrent. Esses protocolos são utilizados na análise competitiva que fazemos mais à frente na seção de avaliação de performance.

2.1. Protocolo BitTorrent

No protocolo BitTorrent existem dois tipos de *peers*: *leechers* e *seeds*. *Leechers* são *peers* que não têm todos os pedaços do objeto, enquanto *seeds* são *peers* que têm todos

os pedaços do objeto. *Seeds* apenas transmitem, enquanto *leechers* são transmissores e receptores.

Um enxame de *peers* é um conjunto de *peers* que participam da transmissão e recepção de um mesmo objeto. Cada enxame é controlado por um processo centralizado denominado *tracker*. Para pertencer a um enxame, é necessário que o *leecher* contacte o *tracker*. Este, por sua vez, passa ao *leecher* uma lista de *peers* que têm o objeto desejado. O *leecher* então seleciona um subconjunto da lista e inicia requisições para estabelecer conexões TCP bidirecionais com os membros, denominados vizinhos, do subconjunto selecionado.

O BitTorrent possui uma importante estratégia de incentivo denominada *tit-for-tat* para evitar o *free-riding*, isto é, a condição de que os *peers* ajam de forma egoísta e utilizem o enxame apenas para receber pedaços sem transmitir para nenhum outro *peer*. Sob essa estratégia, cada *peer* transmite tipicamente para os *k peers* que recentemente lhe permitiram as melhores taxas de recepção, muito embora ele possa receber solicitações de mais de *k peers* interessados em receber pedaços do objeto. Todo esse processo descrito é denominado de *política de seleção do vizinho*.

A recusa para transmitir para certos *peers*, intrínseca à *política de seleção do vizinho*, é chamada de *choking*. Também existe o processo chamado de *optimistic unchoking*, que permite aos *peers* reservarem parte de sua banda para realizar transmissões para *peers* selecionados aleatoriamente, independentemente de sua contribuição ao enxame. Os processos de *choking* e *unchoking* são realizados em intervalos regulares de tempo, denominados *rechoking intervals*.

Ainda, para aumentar a eficiência do enxame, o protocolo BitTorrent utiliza a política do mais raro para selecionar a ordem em que os pedaços do objeto serão recuperados. Os pedaços que são primeiramente solicitados são aqueles que aparecem em menor número entre os vizinhos, ou seja, são os pedaços menos replicados (mais raros). Outra característica importante, que aumenta a rapidez com que um pedaço seja recuperado, é que cada pedaço é dividido em sub-pedaços e os sub-pedaços podem ser solicitados a *peers* diferentes. Todo esse processo é denominado de *política de seleção do pedaço*.

Em síntese, o protocolo BitTorrent possui um baixo *overhead* de controle, é escalável, eficiente e fácil de implementar. A replicação de conteúdo é o seu objetivo principal. Relembramos que, para aplicação desse protocolo em sistemas de *streaming* de vídeo, a *política de seleção do pedaço* utilizada não é, no entanto, adequada. Em sistemas de *streaming*, cada pedaço deve ser recebido dentro de um certo limite de tempo, senão ele é descartado. Como os pedaços do objeto não são solicitados em seqüência e sim baseados na sua *raridade*, podem ocorrer diversas falhas na visualização do objeto.

2.2. Protocolo BiToS

A proposta BiToS [Vlavianos et al. 2006] modifica a proposta do BitTorrent original no que tange exclusivamente à *política de seleção do pedaço*, conforme explicamos a seguir. Os pedaços a serem solicitados aos vizinhos são, para efeito de definição de prioridade, classificados em três conjuntos: *recebidos*, *alta prioridade* e *baixa prioridade*. Os pedaços do primeiro conjunto são aqueles que já foram recebidos. Os do segundo conjunto são aqueles que ainda não foram recuperados e que estão próximos de serem visualizados pelo usuário. Quando um pedaço não foi recuperado e já passou o tempo

para que seja visualizado pelo usuário, diz-se que o pedaço foi perdido. Finalmente, os pedaços do último conjunto são aqueles que ainda não foram recuperados e cujo instante de reprodução não está próximo.

Cada *peer* solicita com probabilidade p um pedaço do conjunto de *alta prioridade*, e com probabilidade $(1 - p)$ um pedaço do conjunto de *baixa prioridade*. A probabilidade p busca estabelecer um equilíbrio entre o que precisa ser visualizado imediatamente e a recuperação de pedaços mais raros. Esta probabilidade pode ser ajustada dinamicamente em função das condições de operação do sistema. A política de seleção do pedaço mais raro é utilizada independentemente do conjunto considerado, sendo que se dois pedaços são igualmente raros, então deve-se optar por aquele que esteja mais próximo de ser visualizado. Por último, os pedaços que estão no conjunto *recebidos* são aqueles que são efetivamente lidos pela aplicação e que serão compartilhados com os outros *peers*.

2.3. Protocolo de Zhou-Chiu-Lui

Assim como o protocolo BiToS, a proposta de Zhou-Chiu-Lui [Zhou et al. 2007] modifica o BitTorrent original no que se refere exclusivamente à sua *política de seleção do pedaço*. Esta proposta é muito semelhante ao BiToS. Os pedaços a serem solicitados aos vizinhos são, classificados em dois conjuntos: *alta prioridade* e *baixa prioridade*. Os pedaços do primeiro conjunto são aqueles que estão próximos de serem visualizados pelo usuário. Os pedaços do segundo conjunto são aqueles que serão necessários apenas em instantes futuros.

A operação do sistema se dá da forma descrita a seguir. Existe uma probabilidade p associada à solicitação de pedaços do conjunto de *alta prioridade*, e uma probabilidade $(1 - p)$ associada à solicitação de pedaços do conjunto de *baixa prioridade*. A diferença com relação ao BiToS é que uma política gulosa é usada para solicitar os pedaços do conjunto de *alta prioridade*, enquanto que a política do mais raro é utilizada para o conjunto de *baixa prioridade*. Na política gulosa os pedaços são solicitados em ordem seqüencial.

3. Algoritmo Proposto

A nossa proposta é baseada no protocolo BitTorrent. Diferentemente dos trabalhos de [Vlavianos et al. 2006, Zhou et al. 2007], que voltam-se para um serviço de *streaming* de vídeo onde usuários assistem seqüencialmente a um vídeo, nossa proposta foi elaborada para um serviço onde os usuários podem realizar ações interativas como pausar, saltar para frente, saltar para trás, etc. A idéia do algoritmo é usar o modelo de [de Vielmond et al. 2007] em conjunto com uma política que estabelece prioridades para seleção de pedaços [Vlavianos et al. 2006, Zhou et al. 2007].

3.1. Modelo de Vielmond-Leão-Silva

O modelo de [de Vielmond et al. 2007] é um modelo de Markov Oculto (HMM) hierárquico para emular o comportamento de usuários acessando um servidor de ensino a distância. Este modelo baseou-se em uma aplicação real de ensino a distância, onde alunos do curso do CEDERJ [de Souza e Silva et al. 2006] assistem a vídeo-aulas previamente gravadas e sincronizadas com transparências [de Souza e Silva et al. 2009] por meio do servidor RIO [Netto et al. 2005]. A estrutura hierárquica possui propriedades interessantes: a complexidade da fase de treinamento é menor que o HMM convencional, as dependências de curto prazo são capturadas pela cadeia da hierarquia inferior, e

a dinâmica de longo prazo é governada pela cadeia de Markov oculta (a hierarquia superior). A cadeia de Markov oculta governa a dinâmica de uma sessão de usuário, e os estados ocultos capturam a dependência das ações do usuário dentro do contexto de uma transparência. Sendo assim, dentro de um estado oculto temos a dinâmica das ações do usuário.

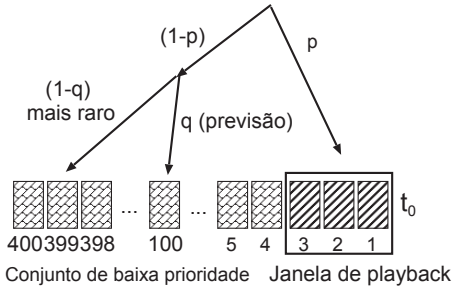

Figura 1. Algoritmo Proposto.

3.2. Descrição do algoritmo

Em nossa proposta, a *política de seleção do vizinho* é a mesma do protocolo BitTorrent. Apenas a *política de seleção do pedaço* é alterada, conforme explicado a seguir. Classificamos os pedaços que não foram recuperados de acordo com duas prioridades: pedaços de alta prioridade (mais próximos de serem tocados) e pedaços de baixa prioridade. Chamamos de *janela de playback* ao conjunto dos m pedaços de alta prioridade. Enquanto que os demais pedaços formam o conjunto de baixa prioridade, de tamanho $T - m$, onde T é o tamanho total do objeto.

Consideramos duas probabilidades: p e q. A probabilidade p determina de qual conjunto recuperar, da janela de *playback* (i.e., conjunto de alta prioridade) ou do conjunto de baixa prioridade. Já a probabilidade q determina que política deve ser usada para recuperar os pedaços do conjunto de baixa prioridade: pedaço mais raro ou pedaço baseado no modelo do usuário [de Vielmond et al. 2007]. Nosso algoritmo possui ainda duas variantes: podemos recuperar pedaços do conjunto de alta prioridade utilizando a política do pedaço mais raro, variante *Previsão Mais Raro*, ou utilizarmos a política gulosa, variante *Previsão Seqüencial*.

A Figura 1 ilustra o funcionamento das variantes. Na variante *Previsão Seqüencial*, os pedaços são recuperados seqüencialmente com probabilidade p da janela de *playback*. Na variante *Previsão Mais Raro*, o pedaço mais raro é recuperado com probabilidade p da janela de *playback*. Para o conjunto de baixa prioridade, escolhido com probabilidade $(1 - p)$, com probabilidade q são recuperados pedaços seguindo o modelo do usuário parametrizado com *logs* reais, e com probabilidade $(1 - q)$ é recuperado o pedaço mais raro. Chamamos de *janela de previsão* ao conjunto de pedaços que são recuperados de acordo com o modelo do usuário. O tamanho da janela de previsão depende do tempo médio que o usuário permanece em *play*. Note que a única diferença entre as variantes *Previsão Seqüencial* e *Previsão Mais Raro* é a política de seleção de pedaços para a janela de *playback*.

A Figura 2 mostra um exemplo do funcionamento da variante *Previsão Seqüencial* em três instantes de tempo. As ações do usuário e as ações obtidas do modelo estão

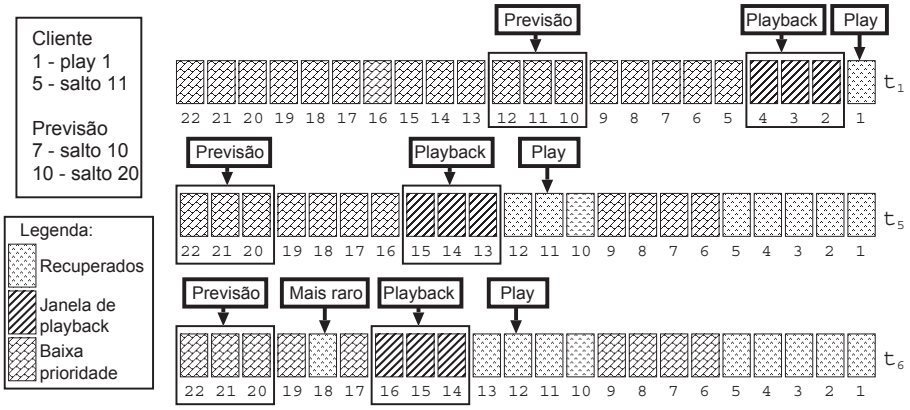

Figura 2. Propostas: Previsão Seqüencial e Previsão Mais Raro.

descritas no quadro à esquerda da figura. As ações do usuário são: no instante 1 – *play no pedaço 1*; e no instante 5 – *salto para o pedaço 11*. As ações provenientes do modelo são: no instante 7 – *salto para o pedaço 10*; e no instante 10 – *salto para o pedaço* 20. No instante t_1, o usuário inicia a visualização do objeto. Sua janela de *playback* são os pedaços 2, 3 e 4 que serão recuperados seqüencialmente com probabilidade p. Com probabilidade q serão recuperados os pedaços 10, 11 e 12 da janela de previsão, pois, segundo o *log* de ações gerado pelo modelo do usuário, no instante t_7 o usuário executará um salto para o pedaço 10. Neste exemplo definimos a janela de previsão igual a janela de *playback*.

Ainda, na Figura 2, no instante de tempo t_5, o usuário faz um salto para o pedaço 11. Note que os pedaços 10, 11 e 12 já haviam sido recuperados anteriormente, pois faziam parte da janela de previsão no instante t_1, logo a nova janela de *playback* é composta pelos pedaços 13, 14 e 15. A nova janela de previsão é composta pelos pedaços 20, 21 e 22, pois um caminho amostral do modelo indica um salto para o pedaço 20 no instante t_{10}. Em t_6, com probabilidade $(1 - q)$ já foi recuperado o pedaço 18 (política do pedaço mais raro) e com probabilidade p foi recuperado o pedaço 13 da janela de *playback*.

4. Resultados

Nesta seção avaliamos a nossa proposta para distribuição de vídeo com interatividade. Também realizamos análises competitivas com o protocolo BitTorrent e dois outros algoritmos da literatura: *BiToS* e *Zhou-Chiu-Lui*.

4.1. Métricas

Em nossos experimentos, utilizamos cinco métricas para avaliação de performance: *número médio de pedaços ausentes*; *tempo para iniciar a reprodução*; *tempo médio de espera*; *taxa de download* e *taxa de upload*. Comentamos a seguir sobre as mesmas.

A métrica número médio de pedaços ausentes é definida como o número de pedaços que não se encontram no *buffer* do usuário no momento da sua reprodução. A principal característica dessa métrica é demonstrar a continuidade da reprodução. A sua média foi estimada da seguinte forma: $PA = \sum_{i=1}^{N} (PA_i/N)$, onde PA_i são os pedaços ausentes do usuário i e N é o total de usuários do enxame. A métrica tempo para iniciar a reprodução é definida como o tempo que o usuário tem que esperar pela chegada

do primeiro pedaço. Sua média é estimada da seguinte forma: $TI = \sum_{i=1}^{N} (TI_i/N)$, onde TI_i é o tempo relativo ao usuário i. A métrica tempo médio de espera é o tempo necessário para a retomada da reprodução após uma interrupção causada pela falta de pedaço no *buffer* do usuário. A sua média é igual a: $TE = \sum_{i=1}^{N} (TE_i/N)$, onde TE_i é o tempo médio relativo ao usuário i. A taxa de *download* refere-se ao número de pedaços recebidos pelo *peer* em um intervalo de tempo δ_t. Por fim a taxa de *upload* refere-se ao número de pedaços enviado pelo *peer* em um intervalo de tempo δ_t.

Assim percebe-se que as três primeiras métricas medem a qualidade com que o objeto é reproduzido, quantificando o número e o tempo médio de interrupções durante a visualização do objeto. Já as duas últimas métricas medem o desempenho quanto a eficiência na distribuição do objeto.

4.2. Cargas

Usamos dois tipos de cargas para avaliar as propostas: cargas reais e cargas sintéticas (geradas pelo modelo de [de Vielmond et al. 2007]). As cargas reais são os *logs* de comportamento de usuários do servidor multimídia RIO [Netto et al. 2005], utilizado no curso do CEDERJ [de Souza e Silva et al. 2006]. Um usuário do sistema RIO pode executar as seguintes ações interativas: *Play*, *Stop*, *Pause*, Salto para frente e Salto para trás. Para gerar as cargas sintéticas, usamos 391 *logs* reais com sessões entre 20 e 30 minutos.

As cargas são classificadas em função do nível de interatividade (I = número médio de requisições por sessão) da seguinte forma: alta ($15 < I < 40$); média ($5 < I < 16$); baixa ($0 < I < 6$) e mista ($0 < I < 40$). Ainda, para garantir um significativo espectro de análise, escolhemos cargas estatisticamente diferentes entre si. As variáveis usadas para denotar as estatísticas das cargas, reais e sintéticas, estão na Tabela 1, e seus valores correspondentes estão na Tabela 2. A partir dessas tabelas, podemos notar que o nível de interatividade da carga sintética é entre 10% e 15% maior ou menor que o nível de interatividade da carga real. Já a diferença para o tamanho médio do segmento é sempre abaixo de 3%, exceto para carga mista. A carga sintética relacionada à carga mista, por esta ser uma carga heterogênea, apresenta uma diferença maior com relação à carga real.

Tabela 1. definição das variáveis

Variáveis	Definição
N	Número total de requisições
I	Número médio de requisições por sessão
L	Tamanho médio do segmento, medido em segundos
$Std(L)$	Desvio padrão de L
$Coef(L)$	Coeficiente de variação de L

4.3. Experimentos

Os resultados de simulação são obtidos usando a ferramenta Tangram-II [de Souza e Silva et al. 2009]. Salvo informado diferentemente, consideraremos um único cenário para todas as simulações, com os seguintes parâmetros: tamanho do objeto igual a $1800\,$s, número de *seeds* igual a 1, a capacidade dos usuários e do *seed* é igual a $100\,kbytes/s$. Para a nossa proposta, a probabilidade q é igual a 0.50, a

Tabela 2. Estatísticas

Estatística	Nível de interatividade							
	Alta		Média		Baixa		Mista	
	Real	Previsão	Real	Previsão	Real	Previsão	Real	Previsão
N	1752	1582	1205	1287	388	454	3346	2541
I	24.01	21.68	9.80	10.46	1.99	2.33	8.56	6.50
L	26.03	26.80	61.40	61.70	260.65	260.70	75.54	106.47
$Std(L)$	29	33	68	65	260	263	77	107
$Coef(L)$	1.14	1.23	1.11	1.05	1.00	1.00	1.02	1.00

probabilidade p é igual a 0.80 e o tamanho do conjunto de alta prioridade m é igual a 8% do tamanho do objeto. A janela de previsão é igual a L, o tamanho médio do segmento. No modelo usamos a carga real para simular o comportamento do usuário e a carga sintética para a previsão do comportamento do usuário.

Organizamos a apresentação dos resultados em três seções distintas. A Seção 4.3.1 dedica-se aos resultados provenientes da análise competitiva entre as propostas utilizando três tipos de cargas: alta, média e baixa. Na Seção 4.3.2 são realizados experimentos para avaliar o desempenho das propostas variando o tamanho do conjunto de alta prioridade (parâmetro m) e é avaliado o desempenho das propostas para valores diferentes da probabilidade p. Finalmente, a Seção 4.3.3 analisa o comportamento dos algoritmos com o aumento da população considerando uma carga mista.

4.3.1. Análise Competitiva entre as Propostas

Na Figura 3, os resultados das métricas número médio de pedaços ausentes, tempo para iniciar a reprodução e tempo médio de espera são expressos em percentual do tamanho do objeto. As Figuras 3(a) e 4(a) resumem os resultados obtidos para a carga alta. Observa-se que, para o número médio de pedaços ausentes, o menor valor foi obtido pelo protocolo BitTorrent (8.64 pedaços), enquanto que o maior valor foi o da proposta *Zhou-Chiu-Lui* (57.06 pedaços). O algoritmo (variante) *Previsão Mais Raro* foi o que obteve resultado mais próximo ao do BitTorrent com média de 0.5% de pedaços ausentes. Entretanto, na métrica tempo médio de espera para iniciar a reprodução, o protocolo BitTorrent obteve o maior tempo (100.08 s), enquanto que o *BiToS* o menor (1.8 s). Este mesmo comportamento se repetiu para a métrica tempo médio de espera, no qual o BitTorrent manteve o maior tempo (69.48 s) e o menor tempo foi obtido pela *Previsão Seqüencial* (10.44 s).

Por meio dessas métricas podemos observar que no BitTorrent ocorrem poucas interrupções. Entretanto, estas interrupções duram muito tempo, pois quando o pedaço a ser visualizado é recuperado, já houve tempo para que outros pedaços subseqüentes fossem recebidos. Outra característica observada é que o retardo para iniciar a exibição pode ser alto, pois depende da *raridade* do primeiro pedaço solicitado pelo usuário. Já os protocolos *Previsão Seqüencial*, *BiToS* e *Zhou-Chiu-Lui* apresentam outro comportamento: um número maior de interrupções, mas o tempo médio de espera para reiniciar a reprodução é pequeno assim como o tempo para recebimento do primeiro pedaço solici-

tado pelo usuário. Este resultado é conseqüência do algoritmo de seleção dos pedaços que atribui uma prioridade maior para a recuperação dos pedaços próximos a serem tocados. O algoritmo que oferece a melhor qualidade para o usuário é a variante *Previsão Mais Raro*, pois apresenta poucas interrupções e cada uma delas tem duração da ordem de 1 s.

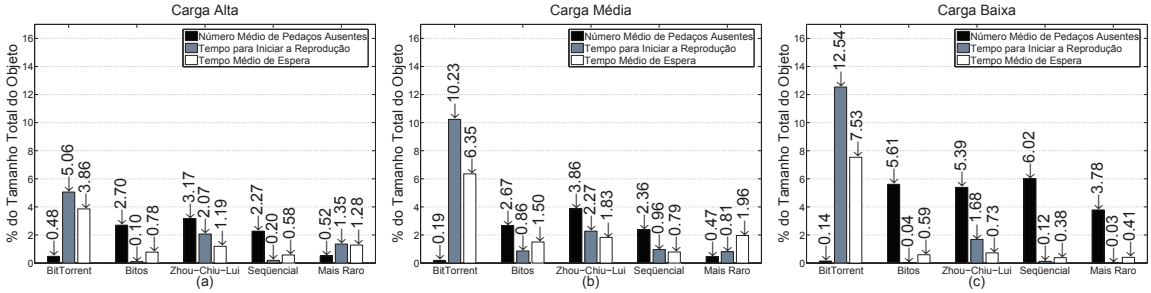

Figura 3. Análise competitiva entre as propostas: (a) Carga Alta; (b) Carga Média; (c) Carga Baixa.

A Figura 4(a) apresenta as taxas médias de *download* e *upload*. Podemos observar que, para as cargas alta e média, os algoritmos (variantes) *Previsão Mais Raro* e BitTorrent, seguidos do *Previsão Seqüencial*, conseguiram as maiores taxas, o que demonstra a eficiência desses algoritmos na recuperação dos pedaços. Esta métrica está diretamente relacionada à rapidez com que o algoritmo *espalha* os pedaços entre os usuários. A variante *Previsão Mais Raro* apresenta desempenho similar ao BitTorrent, pois associa uma política de recuperação do pedaço mais raro com uma política de recuperação de pedaços que é baseada na previsão. O protocolo *BiToS*, apesar de também usar a política de seleção do pedaço mais raro, não apresenta o mesmo desempenho. Uma possível explicação para este comportamento é que quando os pedaços possuem a mesma *raridade*, sempre é recuperado aquele que está mais próximo do ponto de *play* do usuário, o que acaba diminuindo a diversidade dos pedaços recuperados. Para carga baixa, a variante *Previsão Mais Raro* não apresenta o mesmo desempenho que para outras cargas. Este resultado pode ser explicado porque, para carga baixa, o número de saltos é menor, o tamanho da janela de previsão é maior e nesta janela os pacotes são recuperados em sequência. Estes fatores diminuem a rapidez na replicação dos pedaços.

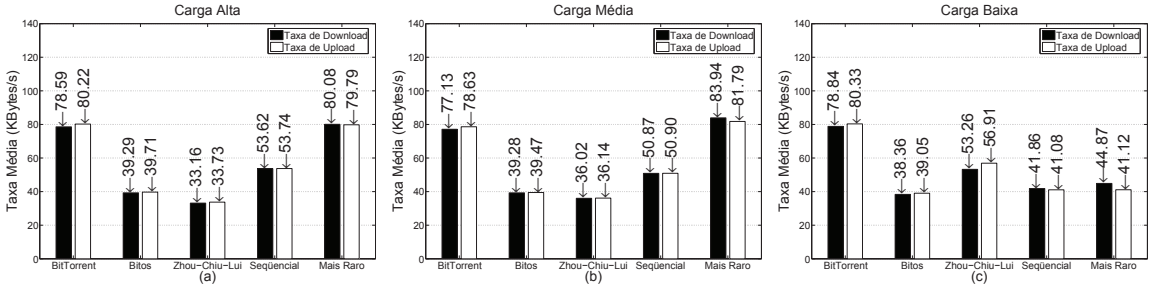

Figura 4. Análise competitiva entre as propostas: (a) Carga Alta; (b) Carga Média; (c) Carga Baixa.

Na Figura 3(b) e 4(b) são apresentados os resultados obtidos para a carga média. Estes resultados são semelhantes àqueles observados para carga alta. Notamos que o BitTorrent apresenta aumento significativo nos tempos médios de espera e para iniciar

a reprodução. Já a variante *Previsão Mais Raro* melhorou seu desempenho no número de ausências e no tempo para iniciar a reprodução, mantendo a maior taxa de *download* e *upload*. As propostas *Previsão Seqüencial*, *BiToS* e *Zhou-Chiu-Lui* apresentaram uma pequena diminuição na qualidade oferecida ao usuário.

Na Figura 3(c) e 4(c) são apresentados os resultados obtidos para a carga baixa. O desempenho piorou para todas as propostas em relação as métricas número de pedaços ausentes, taxa de *download* e *upload*. Isto se deve ao fato de que, para esta carga, o usuário assiste a trechos mais longos seqüencialmente e, portanto, aumenta a probabilidade de ocorrer uma falta de um pedaço.

Na Figura 5 ilustramos o comportamento de cada um dos usuários de carga alta. Observa-se que a variante *Previsão Mais Raro* é a que apresenta menor variabilidade entre os usuários com relação às métricas estimadas.

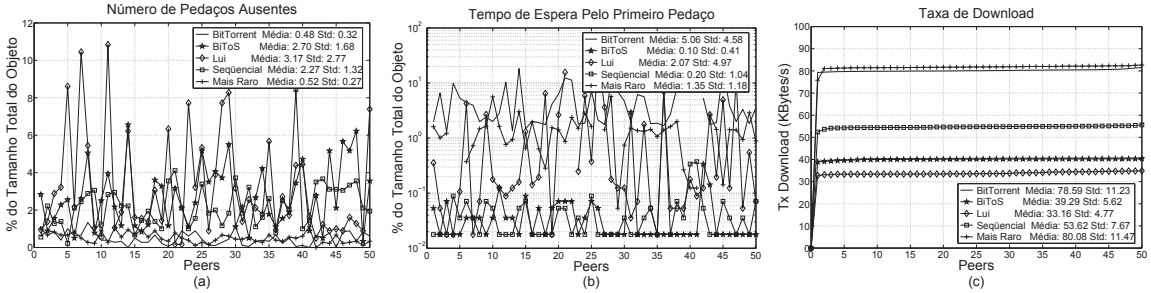

Figura 5. Análise competitiva entre as propostas para a carga alta: (a) Número de Pedaços Ausentes; (b) Tempo Inicial de Visualização; (c) Taxa de *Download*.

4.3.2. Avaliação dos parâmetros m e p

O objetivo deste experimento é avaliar o desempenho das propostas variando os parâmetros m (tamanho da janela de *playback*) e p (probabilidade de escolha de um pedaço da janela de *playback*). Inicialmente variamos o parâmetro m e fixamos $p = 0.8$. Consideramos os seguintes valores para m: 2%, 4%, 6%, 8% e 10% do tamanho do objeto e usuários com alta interatividade (carga alta). Os resultados estão na Figura 6. Não podemos observar qualquer variação significativa nos resultados obtidos quando variamos o valor de m. Além disso, nenhum dos valores de m apresentou resultados significativamente melhores para todas as métricas. Portanto, a escolha do valor de m pode ser baseada, por exemplo, na métrica considerada mais importante.

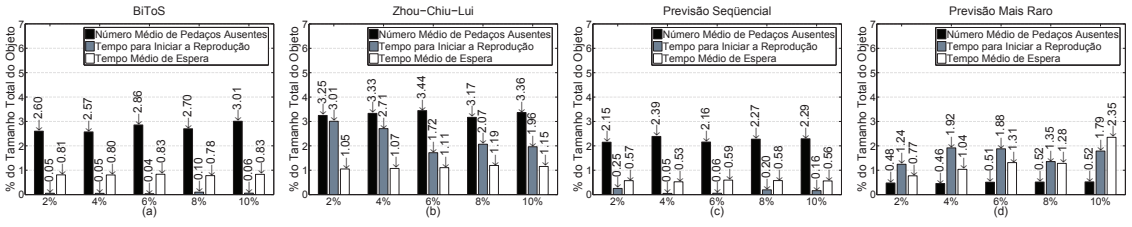

Figura 6. Avaliação do parâmetro m: (a) BiToS; (b) Zhou-Chiu-Lui; (c) Previsão Seqüencial; (d) Previsão Mais Raro.

Em seguida avaliamos a sensibilidade das propostas com relação à probabilidade p. Consideramos os seguintes valores para p: 0.70, 0.80 e 0.90, fixamos m igual a 8% e supomos usuários de alta interatividade. Na Figura 7 são apresentados os resultados desse experimento. Analisando os resultados podemos observar que, assim como para o parâmetro m, não existe grande variabilidade nos valores obtidos para as métricas.

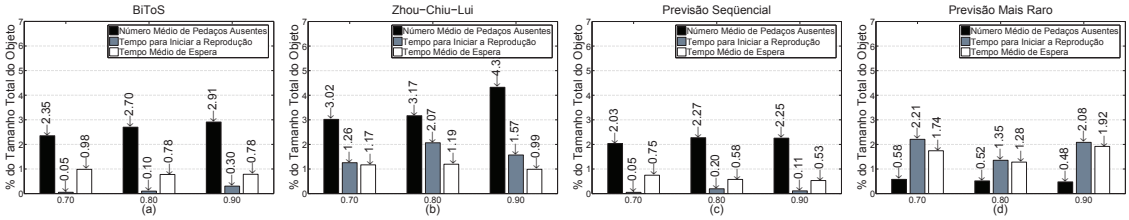

Figura 7. Avaliação do parâmetro p: (a) BiToS; (b) Zhou-Chiu-Lui; (c) Previsão Seqüencial; (d) Previsão Mais Raro.

4.3.3. Experimento Variando o Tamanho da População

Nestes experimentos variamos o tamanho da população de 50 até 200 *peers*. Usamos uma carga mista, ou seja, composta por usuários com alta, média e baixa interatividade.

Calculamos, para cada uma das métricas, o ganho ou a perda percentual em relação ao BitTorrent. O gráfico da Figura 8(a) ilustra o comportamento da métrica número médio de pedaços ausentes com o aumento da população. Essa é a única métrica para a qual o BitTorrent apresenta desempenho significativamente melhor que as outras propostas, exceto para a variante *Previsão Mais Raro*, onde o ganho não é significativo. Podemos observar que o ganho com relação ao BitTorrent diminui com o aumento da população para as propostas *BiToS*, *Zhou-Chiu-Lui* e *Previsão Seqüencial*. Já para a proposta *Previsão mais raro* não existe alteração significativa com o aumento da população.

Nas Figuras 8(b) e 8(c), observamos os ganhos obtidos por todas as propostas para as métricas tempo para iniciar a reprodução e tempo médio de espera. Não existe variação significativa do ganho com o aumento da população, exceto para a variante *Previsão mais raro*, onde o ganho diminui com o aumento da população.

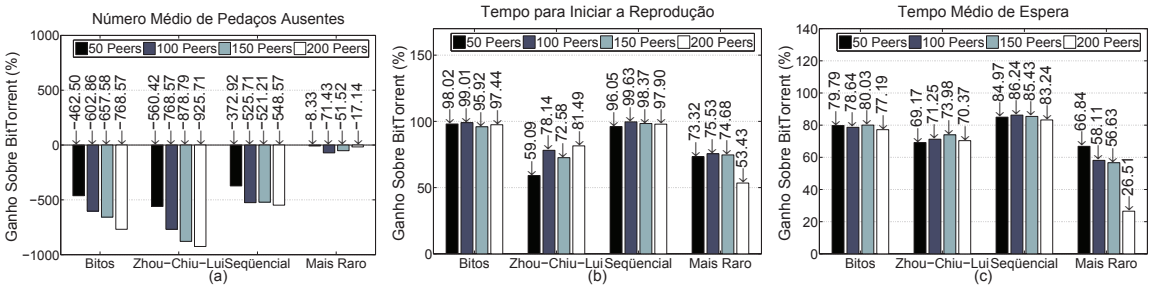

Figura 8. Avaliação da População: (a) Número Médio de Pedaços Ausentes; (b) Tempo para Iniciar a Reprodução; (c) Tempo Médio de Espera.

De forma a avaliar a utilização da banda do *seed*, calculamos a distribuição da banda, conforme ilustrado na Figura 9. Observamos que o BitTorrent e a variante *Previsão mais raro* são aqueles onde a banda do *seed* é menos utilizada, indicando que esses

protocolos são os mais escaláveis, pois usam menos a banda do *seed* e apresentam as maiores taxas de *download* e *upload* para os *peers*.

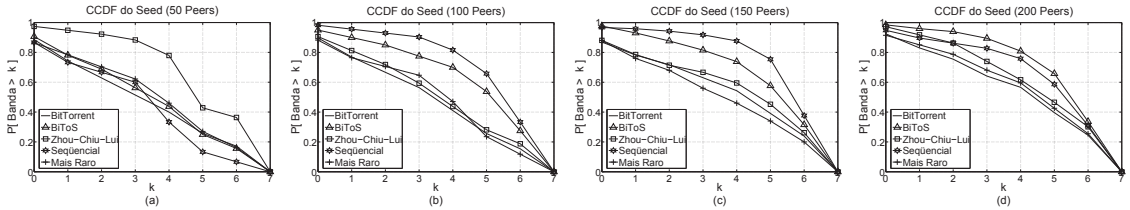

Figura 9. CCDF do Seed: (a) 50 peers; (b) 100 peers; (c) 150 peers; (d) 200 peers.

5. Conclusões e trabalhos futuros

Neste trabalho propomos um novo algoritmo de arquitetura P2P para a transmissão de vídeo pré-armazenado, considerando usuários capazes de realizar ações interativas. O novo algoritmo baseia-se na priorização dos pedaços do objeto multimídia a ser transmitido e em um modelo de comportamento do usuário. Nosso algoritmo possui duas variantes: *Previsão Seqüencial* e *Previsão Mais Raro*. Realizamos simulações com cargas obtidas de um servidor real, além de comparações com outras propostas da literatura. Dentre os resultados obtidos, destacamos:

- A variante *Previsão Mais Raro* oferece a melhor qualidade para o usuário, pois apresenta o melhor compromisso entre as métricas número médio de interrupções e tempo médio de espera para visualizar um pedaço do objeto multimídia;
- Na variante *Previsão Mais Raro*, o número médio de interrupções é semelhante ao obtido para o BitTorrent (que apresentou o melhor resultado para esta métrica);
- Contrariamente ao BitTorrent, que apresentou os maiores valores para o tempo médio de espera para visualizar um pedaço, os tempos obtidos para a variante *Previsão Mais Raro* são similares àqueles obtidos pelas as propostas da literatura com melhor desempenho;
- Na variante *Previsão Mais Raro*, as taxas de *upload* e *download* são semelhantes às obtidas para o BitTorrent (que foram as maiores), revelando a sua eficiência em distribuir rapidamente um conteúdo;
- A variante *Previsão Seqüencial* apresentou resultados ligeiramente superiores ao BiToS, que mostrou-se, dentre as propostas da literatura, o mais adequado para *streaming* de vídeo.

Como trabalhos futuros, pretendemos considerar cenários com populações maiores e uma maior variação dos parâmetros do novo algoritmo proposto, a saber: número de pedaços do conjunto de alta prioridade (m) e probabilidade de seleção (p).

Referências

Carlsson, N. and Eager, D. L. (2007). Peer-assisted On-demand Streaming of Stored Media using BitTorrent-like Protocols. In *IFIP/TC6 Networking 2007*, Atlanta, Georgia, USA.

Cohen, B. (2003). Incentives Build Robustness in Bittorrent. In *First Workshop on Economics of Peer-to-peer Systems*, Berkeley, USA.

de Souza e Silva, E., Figueiredo, D., and Leão, R. M. M. (2009). The TANGRAM-II Integrated Modeling Environment for Computer Systems and Networks. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):45–65.

de Souza e Silva, E., Leão, R. M. M., Santos, A. D., Azevedo, J. A., and Netto, B. C. M. (2006). Multimedia Supporting Tools for the CEDERJ - Distance Learning Initiative applied to the Computer Systems Course. In *22th ICDE World Conference on Distance Education*, pages 1–11, Rio de Janeiro, RJ, Brasil.

de Vielmond, C. C. L. B., Leão, R. M. M., and de Souza e Silva, E. (2007). Um Modelo HMM Hierárquico para Usuários Interativos Acessando um Servidor Multimídia. In *XXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 469–482, Belém, PA, Brasil.

Eager, D., Vernon, M., and Zahorjan, J. (2000). Bandwidth Skimming: A Technique for Cost-Effective Video-on-Demand. In *Proc. IS&T/SPIE MMCN*, pages 206–215, San Jose, CA, USA.

Guo, Y., Suh, K., Kurose, J., and Towsley, D. (2003). P2cast: Peer-to-Peer Patching Scheme for VoD Service. In *Proc. 12th World Wide Web Conference*, pages 301–309, Budapest, Hungary.

Hua, K. A., Cai, Y., and Sheu, S. (1998). Patching: A Multicast Technique For True Video-on-Demand Services. In *Proc. 6th ACM Int'l Multimedia Conference (ACM MULTIMEDIA)*, pages 191–200, Bristol, UK.

Huang, J. L. and Ross, K. (2007a). Can Internet VoD be profitable? In *Proc. of ACM SIGCOM*, Kyoto, Japão.

Huang, J. L. and Ross, K. (2007b). Peer-assisted VoD: Making Internet Video Distribution Cheap. In *Proc. of IPTPS'07*, Redmond, WA, USA.

Liu, Y., Guo, Y., and Liangy, C. (2008). A survey on peer-to-peer video streaming systems. *Peer-to-Peer Networking and Applications*, 1(1):18–28.

Netto, B. C. M., Azevedo, J. A., e Silva, E. A. S., and Leão, R. M. M. (2005). Servidor Multimídia RIO em Ensino à Distância. In *Proc. 6th International Free Software Forum*, Porto Alegre, RS, Brasil.

Rodrigues, C. K. S., Filho, L. J. H., and Leão, R. M. M. (2008). On Scalable Interactive Video-On-Demand Services. *European Journal of Scientific Research*, 21(4):662–686.

Rodrigues, C. K. S. and Leão, R. M. M. (2007). Bandwidth usage distribution of multimedia servers using patching. *Computer Networks*, 51(3):569–587.

Shah, P. and Pâris, J.-F. (2007). Peer-to-peer multimedia streaming using BitTorrent. In *IEEE International Performance, Computing and Communications Conference (IPCCC)*, pages 340–347, New Orleans, LA, USA.

Vlavianos, A., Iliofotou, M., and Faloutsos, M. (2006). BiToS: Enhancing BitTorrent for supporting streaming applications. In 9^{th} *IEEE Global Internet Symposium*, Barcelona, Spain.

Zhou, Y., Chiu, D. M., and Lui, J. C. S. (2007). A simple model for analyzing P2P streaming protocols. In *IEEE International Conference on Network Protocols*, pages 226–235, Beijing, China.