Maximizando o Número de Usuários em Servidores de Vídeo sob Demanda

Nelson L. S. da Fonseca **Roberto de A. Façanha**

Instituto de Computação
Unicamp
Caixa Postal 6176
13083-970 — Campinas, SP
e-mail: {nfonseca,facanha}@dcc.unicamp.br

Abstract

The deployment of video services in large scale requires the adoption of bandwidth reduction techniques such as Batching and Piggybacking. In this paper, we introduce a new Batching policy which aims at maximizing the number of users in a video-on-demand system.

Resumo

Aplicações de vídeo, como por exemplo Vídeo sob Demanda, requerem grande largura de banda para a transmissão de fluxos de vídeo. Portanto, o oferecimento de serviços de vídeo em larga escala requer o emprego de técnicas como Batching e Piggybacking. Neste artigo, introduz-se uma nova política de Batching, que visa maximizar o número de usuários suportados por um servidor de vídeo.

Palavras-chave: *Batching*, *piggybacking* e vídeo sob demanda.

1 Introdução

Se por um lado a rápida expansão do paradigma multimídia deve-se à capacidade crescente dos computadores pessoais (*desktops*); por outro, o oferecimento de serviços multimídia remotos em larga escala depende do efetivo dimensionamento das futuras redes de comunicação. Uma das áreas mais promissoras para disponibilização de serviços multimídia em redes é a *Loja de Locação de Vídeo Eletrônico* (*Eletronic Video Rental Store*), que oferece o serviço de Vídeo sob Demanda (*Video on Demand* — VoD). Através deste serviço, um usuário pode escolher um título e ter o filme enviado através de uma rede de telecomunicações até sua casa.

Diversas são as formas de implementação de serviços de vídeo sob demanda. Em um extremo há um serviço mais sofisticado, no qual o vídeo é iniciado instantaneamente e o

usuário pode executar operações de VCR. No outro extremo, encontra-se um serviço mais econômico, no qual a exibição do vídeo não é instantânea mas, ao contrário, o usuário deve esperar o início programado da apresentação. Neste caso, o usuário não dispõe de operações VCR [1].

A transmissão de vídeo requer grande demanda de banda passante restingindo, portanto, o oferecimento de serviços de vídeo a uma grande população de usuários. Assim sendo, diversas técnicas vêm sendo propostas a fim de reduzir a demanda de banda passante. Dentre elas destacam-se *Batching* [2, 3, 4] e *Piggybacking* [5, 6, 7]. Estas técnicas levam em consideração a probabilidade de um conjunto de requisições a um vídeo popular chegar ao sistema dentro de um intervalo de tempo relativamente curto, de forma a atendê-las com um único fluxo.

A técnica de *Piggybacking* baseia-se no fato de que alterações da ordem de 5% na taxa de exibição não são perceptíveis aos usuários. Desta forma, as requisições são atendidas imediatamente e o fluxo compartilhado é obtido através da superposição de fluxos dessincronizados, ou seja, altera-se a taxa de exibição dos fluxos de vídeo de tal forma que estes venham a exibir um mesmo quadro de filme em um determinado instante, descartando-se um dos fluxos após a sincronização. Por outro lado, a técnica de *Batching* consiste em reter requisições por um determinado intervalo de tempo com o objetivo de agrupar o máximo número de requisições e servir ao grupo com um único fluxo de vídeo. A este intervalo dá-se o nome de **Janela** ou **Intervalo de *Batching*** e seu dimensionamento representa um compromisso entre a minimização do tempo de espera do usuário e o aumento do número de usuários suportados pelo sistema (efeito de *Batching*).

O objetivo deste artigo é introduzir uma nova política de *Batching*, denominada *Look-Ahead-Optimize-Batch*, que maximiza o número de usuários suportados por um servidor de vídeo sob demanda.

O restante deste artigo está organizado como segue. Na seção 2, parâmetros utilizados na avaliação de um sistema com *Batching* são descritos. Trabalhos relacionados são apresentados na seção 3. A seção 4 introduz a política *Look-Ahead-Optimize-Batch*. Na seção 5, resultados numéricos são apresentados. Finalmente, na seção 6 conclui-se o artigo.

2 Métricas de Avaliação de um Sistema com *Batching*

Num sistema de VoD sujeito a um processo de chegada de requisições $\mathcal{P}$, com tempo médio entre requisições $1/\lambda$, um usuário escolhe o vídeo, k, que deseja assistir com probabilidade p_k, $1 \leq k \leq V$, em que V é o número de programas armazenados no servidor. A requisição deste usuário é inserida na fila correspondente, onde permanece até que um fluxo seja alocado ou o usuário decida abandonar o sistema sem atendimento. Define-se o *intervalo de espera de um usuário* como uma variável aleatória, T, que descreve o tempo máximo que este tolera esperar na fila antes de abandonar o sistema. O número máximo de fluxos que o sistema suporta é N_{max}, de modo que cada um destes fluxos pode apresentar qualquer um dos V vídeos armazenados. Considera-se um vídeo composto de L quadros.

Em um sistema com *Batching*, o principal parâmetro de avaliação constitui-se no número de usuários suportados. Os parâmetros seguintes podem ser utilizados na avaliação de sistemas com *Batching*:

- **Probabilidade de Abandono**: É definida como a razão entre o número de requisições que abandona o sistema pelo número total de requisições recebidas em um determinado intervalo de tempo.

- **Retardo Médio de Atendimento**: É definido como o tempo decorrido entre a chegada da requisição e o instante em que o usuário é atendido através da alocação de um fluxo.

- **Injustiça** (*Unfairness*): Seja $P_A(i)$ a probabilidade de abandono para o vídeo i e $\overline{P} = \sum_{i=1}^{V} P_A(i)/V$ a probabilidade média de abandono, define-se a injustiça no sistema como:

$$Unfairness = \sqrt{\frac{\sum_{i=1}^{V}(P_A(i) - \overline{P})^2}{V-1}}. \quad (1)$$

  A Equação 1 informa a variância das probabilidades de abandono no sistema. Objetiva-se minimizar os abandonos tanto nas filas de vídeos populares como dos não populares, tornando assim, o sistema justo. No entanto, os critérios definidos por certas políticas de *Bathing*, com o objetivo de se maximizar o número de usuários suportados, podem impedir a homogeneidade dos valores das probabilidades de abandono, isto é, estes critérios podem beneficiar somente os vídeos populares. Quando isto ocorre, avaliam-se as diferenças entre os índices percentuais de perdas nas filas dos diversos vídeos através de sua variância. Por isso afirma-se que o sistema tende a ser mais justo quanto mais $P_A(i)$ aproxima-se de $\overline{P}$ para todo $1 \leq i \leq V$, ou seja, a probabilidade de abandono de requisições para cada vídeo aproxima-se de média das probabilidades de abandono.

3 Trabalhos Relacionados

Diversas políticas de *Batching* vêm sendo propostas na literatura. Em [4], apresenta-se um estudo considerando as políticas Primeiro a Chegar, Primeiro a ser Atendido (*First Come, First Served* — FCFS) e Maior Tamanho de Fila (*Maximun Queue Length* — MQL). Na política FCFS, todas as requisições são inseridas em uma mesma fila conforme a ordem de chegada e, quando um fluxo torna-se disponível, este é alocado para a primeira requisição da fila, de tal forma que todas as demais requisições pendentes do mesmo vídeo também são atendidas. Deste modo, considera-se somente o fator justiça quando da alocação de um canal.

Por outro lado, no esquema MQL as requisições são inseridas em filas distintas e o vídeo com maior fila é selecionado para alocação. Esta constitui-se numa estratégia agressiva de alocação de fluxos que privilegia os vídeos populares, podendo causar um aumento substancial na probabilidade de abandono dos vídeos não populares. Como conseqüência, tem-se um aumento no nível de injustiça.

Em [3], Philip Yu e H. Sachnai introduziram esquemas de escalonamento de fluxos de vídeo que levam em consideração o tempo de abandono dos usuários. Define-se um *intervalo mínimo de espera*, após o qual pode-se alocar um fluxo segundo os critérios de maior tamanho de fila ou maior número estimado de abandonos. Outras políticas de *Batching* que podem ser mencionadas são Espera Forçada, Controle de Taxa Puro e Desviado [8], dentre outras.

4 A Política *Look-Ahead-Optimize-Batch*

Nesta seção introduz-se uma nova política de *Batching*, *Look-Ahead-Optimize-Batch* — LAO-*Batch*, que utiliza o conhecimento sobre o comportamento de abandono dos usuários para maximizar o efeito de *Batching*. Esta política assume uma janela de *Batching* dependente do usuário e não mais uma janela de *Batching* como um parâmetro global do sistema. Em outras palavras, cada usuário tem associado a ele um tempo (aleatório) máximo de espera na fila após o qual ele abandona o sistema.

A alocação de fluxos ocorre quando o tempo de abandono de um usuário (janela de *Batching*) esgota, isto é, os canais podem ser alocados quando um usuário está prestes a abandonar o sistema. Deste modo, as requisições são inseridas nas filas (por filme) conforme a ordem cronológica, ou seja, a primeira requisição de uma fila é aquela cuja janela de *Batching* esgota mais cedo.

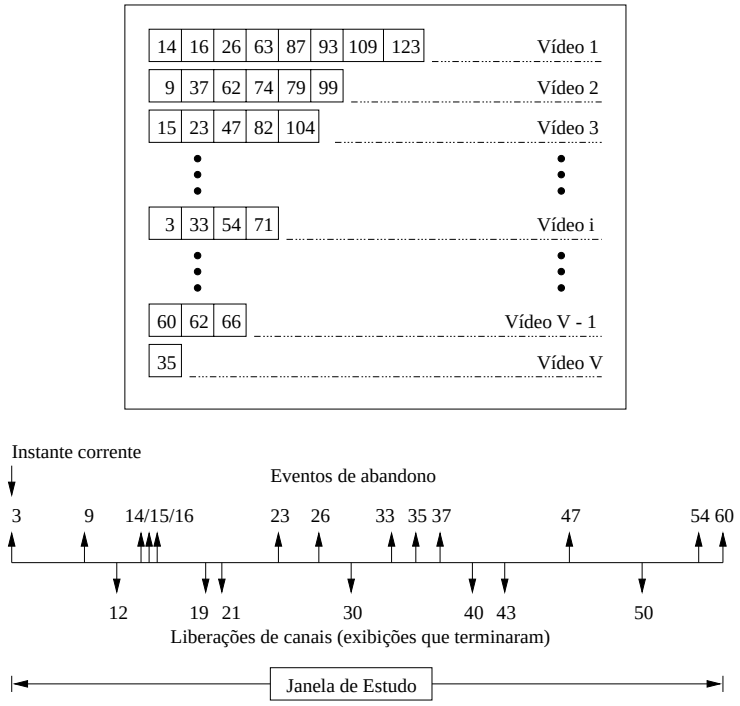

Figura 1: Cenário de filas com requisições pendentes e sua janela de estudo correspondente.

O critério de alocação de fluxos é o mesmo para todos os vídeos (independentemente de sua popularidade). Seja N o número de fluxos disponíveis no sistema no instante de análise, define-se o critério de alocação de fluxos como segue:

i) Se o número de vídeos com requisições pendentes é menor ou igual ao número de canais atualmente disponíveis no sistema, N, aloca-se um fluxo para o vídeo cuja janela de *Batching* esgotou;

ii) Se o número de filas com requisições pendentes é maior que o número de fluxos disponíveis, define-se uma *Janela de Estudo* — JE. Na JE são ordenados os próximos eventos de possíveis alocações e liberações de fluxo. A janela de estudo tem como limite inferior o tempo corrente e como limite superior o instante no qual termina o

maior tempo remanescente de uma requisição na cabeça de uma das filas (*head of the queue*). Com base no cenário de uma JE, formula-se um problema de otimização para a maximização do número de usuários atendidos pelo sistema. Nota-se que este critério beneficia os vídeos populares.

Em outras palavras, a JE consiste num intervalo de tempo cujos limites inferior e superior são definidos, respectivamente, pelos primeiro e último eventos de estouro das janelas de *Batching* das primeiras requisições nas diversas filas. Adicionalmente, consideram-se todos os eventos de potencial alocação e liberação de fluxos que acontecem entre seus extremos, Figura 1.

Levando-se em consideração os possíveis instantes de alocação de canais, bem como de liberação dos mesmos dentro da JE, resolve-se um problema de programação linear inteira para determinar se é vantajoso ou não alocar um canal para o vídeo cujo intervalo de *Batching* esgotou (ou seja, o vídeo que originou a JE em questão). Aloca-se um fluxo para o referido vídeo se e somente se a alocação no presente instante de tempo não acarreta uma perda maior de usuários, por falta de canais disponíveis, em outros instantes de tempo compreendidos entre os limites da janela de estudo. O problema de programação linear inteira pode ser descrito como:

$$\max \sum_{k=1}^{V} \sum_{i=1}^{L_k} (q_k - i + 1) \times x_{ki}$$

Sujeito a:

$$\sum_{i=1}^{L_k} x_{ki} \leq 1, \qquad k = 1 \ldots V \tag{2}$$

$$x_{ki} + \sum_{\substack{\hat{k}=1 \\ \hat{k} \neq k}}^{V} \sum_{j=1}^{L_K} x_{\hat{k}j} \leq A + R_{ki} \qquad \begin{array}{c} t_{\hat{k}j} < t_{ki} \\ k = 1 \ldots V \\ i = 1 \ldots L_k \\ t_{ki} \in \Gamma \end{array} \tag{3}$$

$$x_{ki} \in \{0, 1\}$$

em que : e_{ki} - momento no qual se esgota o intervalo de Batching do i-ésimo usuário da k-ésima fila, isto é, o momento em que o usuário esta prestes a abandonar o sistema.

$$\Gamma - \text{Janela de Estudo}, \Gamma = \left[\begin{array}{c} \min\limits_{k}\{e_{k1}\} \\ \end{array} , \begin{array}{c} \max\limits_{k}\{e_{k1}\} \\ \end{array} \right]$$

em que e_{k1} é o instante no qual o primeiro usuário da k-ésima fila abandonará o sistema.

V - Número total de vídeos armazenados no servidor.

q_k - Número de requisicões pendentes na k-ésima fila, ou seja, tamanho da k-ésima fila, $0 \leq q_k \leq \infty$.

L_k - L-ésimo usuário da k-ésima fila tal que:

$$\begin{cases} e_{kL} = \max\limits_{i}\{e_{ki}\}, & e_{kL} \leq \max\limits_{j}\{e_{j1}\}, & e_{kL+1} > \max\limits_{j}\{e_{j1}\}, & q_k > L \\ e_{kL} = \max\limits_{i}\{e_{ki}\}, & e_{kL} \leq \max\limits_{j}\{e_{k1}\}, & & q_k = L \end{cases}$$

t_{ki} - momento de abandono do i-ésimo da k-ésima fila, isto é, o momento em que estoura a sua janela de *Batching* (tolerância à espera para assistir a um filme), $t_{ki} \in \Gamma$.

x_{ki} -variável inteira 0-1 tal que, se o seu valor é 1 aloca-se um canal para a k-ésima fila no instante t_{ki}; caso contrário, não se aloca um canal para a k-ésima fila no instante t_{ki}.

R_{ki} - Número de canais liberados desde o início da Janela de Estudo até o instante $t_{ki}, 0 \leq R_{ki} \leq N_{max}$

N_{max} - Capacidade do Servidor (número total de canais).

A - Número de canais disponíveis no início da Janela de Estudo $(\min\{e_{k1}\}), 0 \leq A \leq C$.

A função objetivo traduz o ganho obtido no sistema com a solução do problema, que leva em consideração todas as requisições inseridas nas filas

A restrição representada pela Equação 2 determina que, para o conjunto de requisições pendentes numa fila, o número total de canais alocados é 1. Isto é, se na solução de um determinado problema a variável $x_{ki} = 1$, a solução obtida indica que as requisições $1, \ldots, i-1$ não são atendidas (abandonando o sistema) e que, no instante t_{ki}, ocorrerá a alocação de um fluxo para o vídeo k suportando as requisições $i, \ldots, q_k$.

O limite superior no número total de fluxos alocáveis é imposto pela restrição da Equação 3. Num determinado instante (t_{ki}), aloca-se um fluxo se o montante utilizado até então é inferior à soma dos fluxos inicialmente disponíveis com os liberados até este instante .

A solução do problema leva em consideração todos os eventos dentro da JE (eventos de possível alocação e de liberação de canais). Mais precisamente, o problema de otimização formulado indica se é atrativo (ou não) para o sistema alocar um fluxo para o vídeo cuja requisição representa o primeiro evento da JE, isto é, a requisição cuja janela de *Batching* esgotou. Em outras palavras, para um vídeo k, que origina a janela de estudo, aloca-se um fluxo se $x_{k1} = 1$, caso contrário, a alocação é retida e a primeira requisição da fila do vídeo k abandona o sistema; de modo que, para cada requisição cujo intervalo de *Batching* esgota, define-se uma nova janela de estudo e formula-se um novo problema.

5 Resultados Numéricos

Para a avaliação da política proposta utiliza-se o seguinte modelo de simulação: o tempo entre chegadas de requisições é exponencialmente distribuído com média $1/\lambda$. A probabilidade de escolha de um vídeo é caracterizada através da distribuição Zipf com parâmetro $\theta = 0{,}271$, valor este derivado da alocação de filmes em lojas de vídeo [4]. Nesta distribuição, a probabilidade de escolha por um vídeo k é dada por $p_k = \frac{c}{k^{(1-\theta)}}$, em que θ é o parâmetro para a distribuição e c é a constante de normalização, dada por:

$$c = \frac{1}{\sum_{k=1}^{V} \frac{1}{k^{(1-\theta)}}}. \tag{4}$$

Assume-se um lucro igual para todos os vídeos ($r_k = 1,\ k = 1, \ldots, V$) e que cada usuário espera durante um intervalo de tempo cuja duração é definida pela distribuição normal (*Gaussiana*) com média μ e variância σ^2, ou seja, $T \sim N(\mu, \sigma^2)$ [3]. Deste modo, considera-se que após T minutos o usuário abandona o sistema sumariamente. Para derivação dos intervalos de confiança com 99% de confidência, utilizou-se o método das replicações independentes. Os programas de simulação foram escritos na linguagem C e para a resolução do problema de programação inteira utilizou-se o pacote CPLEX.

Nos experimentos de simulação, compara-se a política LAO-*Batch* com as políticas *Batch* MBQ e com uma variação da política MQL. Utiliza-se a política MBQ pois esta demonstrou desempenho superior ao das políticas MQL e FCFS [3]. Para a política MQL modificada, assume-se um critério de escalonamento semelhante ao primeiro caso da política LAO-*Batch*, isto é, possíveis alocações somente podem ocorrer quando uma janela de *Batching* esgota. Quando o servidor encontra-se saturado e ocorre uma liberação de canal, este é então alocado para o vídeo de maior fila. Esta modificação objetiva aproximar o desempenho desta política em relação a LAO-*Batch*. Adicionalmente, compara-se a efetividade das políticas de *Batching* em relação a efetividade de *Piggybacking*. Com este objetivo, utiliza-se a política S^2, uma vez que o desempenho desta mostra-se superior ao das demais políticas de *Piggybacking* [7].

Assume-se que o número de vídeos armazenados no servidor é $V = 100$ e que a duração média de cada vídeo é de 2 horas (120 minutos). Duas taxas de chegada de requisições foram consideradas. Na primeira, ocorrem 50 requisições por minuto (carga alta) e na segunda, somente 10 requisições (carga moderada). A capacidade do servidor assume valores entre 100 e 600 fluxos e o tempo de abandono dos usuários é modelado pela distribuição normal com média $\mu = 10$ minutos e variância $\sigma^2 = 2{,}5$ minutos, $T \sim N(10,\ 2{,}5)$.

No gráfico da Figura 2 mostra-se o desempenho das políticas quanto ao número de usuários suportados. Observa-se que a política LAO-*Batch* apresenta desempenho superior ao das demais políticas consideradas. Esta característica se deve ao fato da política ter a propriedade de "olhar" para momentos futuros enquanto realiza a análise para a alocação de um fluxo, ou seja, a política LAO-*Batch* é capaz de prever se uma alocação em um certo momento vai produzir um número menor (maior) de usuários aceitos pelo sistema dentro da janela de estudo. A política MQL descrita apresenta desempenho inferior ao da política LAO-*Batch*. Em outras palavras, quando o servidor está saturado, podem existir várias filas aptas para alocação. No entanto, quando um fluxo torna-se disponível sua realocação ocorre de forma imediata para tais filas, sem que se espere um momento mais apropriado para a alocação. Quanto à política *Batch* MBQ, o requisito de espera

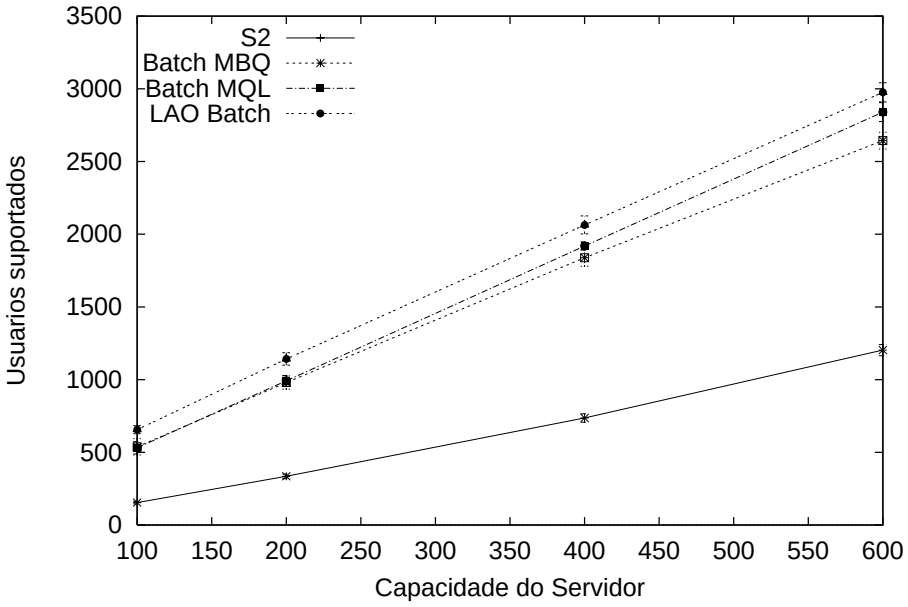

Figura 2: Usuários suportados × capacidade do servidor com taxa de 50 requisições por minuto.

determinado, impõe um tempo mínimo para todas as filas, podendo haver desistências sumárias caso o tempo de abandono de um usuário seja menor que ω. Esta característica explica seu desempenho inferior ao das políticas LAO-*Batch* e MQL modificada. Deve-se notar que a política LAO-*Batch* aceita 20% mais usuários que a política MBQ e 15% a mais que a política MQL modificada para servidores com grande capacidade. Pode-se observar também que todas as políticas de *Batching* apresentam desempenho superior aos da política de *Piggybacking* S^2.

A probabilidade de abandono dos usuários é ilustrada no gráfico da Figura 3. A política LAO-*Batch* possui os índices de probabilidade de abandono mais baixos dentre todas consideradas. Esta constatação é uma conseqüência direta da maximização do efeito de *Batching* no sistema. O comportamento das demais políticas, *Batch* MBQ e MQL modificada são próximos, diferenciando-se cada vez mais com o aumento do número de canais suportados pelo servidor. Com o aumento da capacidade do servidor, reduzem-se as perdas com a política MQL, sendo que esta política busca reduzir a perda de requisições definindo um intervalo de espera cuja duração é igual ao tempo de abandono do usuário. Esta característica não é compartilhada pela política *Batch* MBQ devido à definição de um requisito de espera mínimo para todas as filas que pode exceder o tempo de abandono dos usuários. Nota-se que a probabilidade de abandono dada pela política LAO-*Batch* é 0.10 inferior a probabilidade de abandono da política MQL. Por outro lado, observa-se que a política S^2 apresenta índice de abandonos superior àqueles obtidos pelas políticas de *Batching*. Em outras palavras, em decorrência da maior eficiência de *Batching* em relação a *Piggybacking*, observa-se um número maior de abandonos para a política S^2 que para as políticas de *Batching*.

O grau de injustiça é considerado no gráfico da Figura 4. Observa-se que a política LAO-*Batch* apresenta os níveis de injustiça mais elevados, uma vez que privilegia os vídeos populares em detrimento dos não populares. A política MQL modificada privilegia somen-

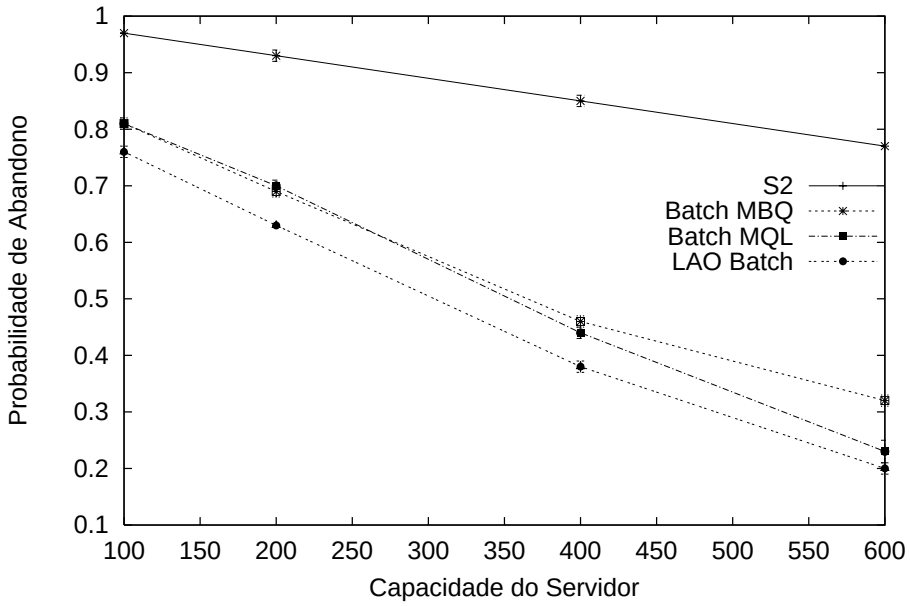

Figura 3: Probabilidade de abandono × capacidade do servidor com taxa de 50 requisições por minuto.

te os vídeos populares quando o servidor encontra-se saturado, ou seja, somente seleciona a maior fila quando o servidor encontra-se saturado e ocorre uma liberação de fluxo. Por isso, seus índices de injustiça encontram-se abaixo daqueles para as políticas LAO-*Batch* e *Batch* MBQ. A política *Batch* MBQ apresenta índices crescentes de injustiça em virtude do requisito de espera mínimo. Quando se considera um servidor de pequeno porte, diminui-se a diferença na probabilidade de abandono entre as diversas filas, uma vez que, mesmo beneficiando os vídeos populares, ainda ocorre um número elevado de desistências em suas filas. No entanto, para servidores de maior porte, aloca-se um número maior de fluxos a vídeos populares, acarretando um número maior de abandonos de requisições pelos vídeos não populares.

Por outro lado, a política de *Piggybacking* (política S^2), proporciona os menores índices de injustiça entre os vídeos. Isto se deve à alocação de fluxos obedecer a seqüência de chegada das requisições, ou seja, emprega-se o esquema FCFS para a alocação de fluxos, não havendo privilégios para qualquer um dos vídeos. Pode-se perceber um nível crescente de injustiça para a política S^2 com o aumento da capacidade do servidor. Isto ocorre porque, à medida em que se aumenta o número de fluxos num servidor, aloca-se uma porcentagem maior destes para os vídeos populares devido à maior freqüência de chegadas de suas requisições, as quais têm maior chance de encontrar fluxos disponíveis para alocação.

Nas Figuras 5–6 analisam-se os mesmos experimentos das Figuras 3–4 considerando-se uma taxa de chegadas de 10 requisições por minuto. Na Figura 5, mostram-se as curvas de probabilidade de abandono. Observa-se que a política LAO-*Batch* apresenta os níveis de abandono mais baixos, chegando a assumir valor zero quando se considera um servidor com maior capacidade. A política *Batch* MBQ reduz o nível de perdas de requisições com o aumento da capacidade do servidor. No entanto, mesmo com servidores de maior porte, a determinação do requisito mínimo de espera impede uma redução maior no número

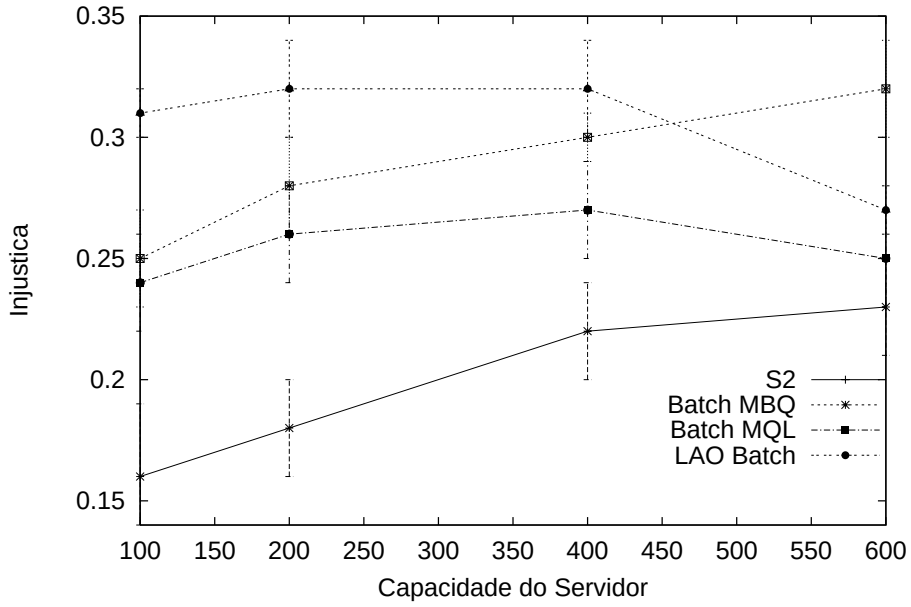

Figura 4: Injustiça × capacidade do servidor com taxa de 50 requisições por minuto.

de abandonos. Percebe-se que para servidores com capacidade de 600 fluxos, o nível de perdas da política *Batch* MBQ mantém-se alto, inclusive maior que a taxa de perdas da política S^2 (que para as demais capacidades de servidor consideradas apresenta índices de abandono mais elevados).

Na Figura 6, mostra-se o nível de injustiça no sistema. Como conseqüência direta da queda da probabilidade de abandono para todas as filas, a política LAO-*Batch* transforma-se na mais justa de todas as políticas. De modo geral, a política S^2 novamente mostra-se menos injusta que as políticas de *Batching*, com exceção das políticas LAO-*Batch* e MQL modificada, quando se considera um servidor com capacidade de 600 fluxos. A política *Batch* MBQ é a que mais favorece aos vídeos populares, mais uma vez devido ao intervalo de espera definido, que sob baixa taxa de chegadas, conduz a um grande número de abandonos, em termos percentuais, entre os vídeos não populares.

6 Conclusões

O serviço Vídeo sob Demanda requer grande largura de banda para a transmissão de vídeo, sendo necessário o emprego de técnicas, como *Batching* e *Piggybacking*, para reduzir esta demanda. Neste artigo, introduz-se uma nova política de *Batching*, LAO-*Batch*, que visa maximizar o número de usuários suportados por um servidor de vídeo. A política LAO-*Batch* utiliza informações sobre o potencial abandono dos usuários e formula um problema de programação inteira.

Compara-se a política proposta com outros esquemas de *Batching* e com a política de *Piggybacking* S^2. Verifica-se que a política LAO-*Batch* permite o atendimento a uma população de usuários maior, uma vez que os vídeos populares são priorizados em situações de carga intensa. Observa-se também que sob cargas moderadas, a política LAO-*Batch* assume postura mais flexível, permitindo o atendimento de usuários que solicitam a exibição

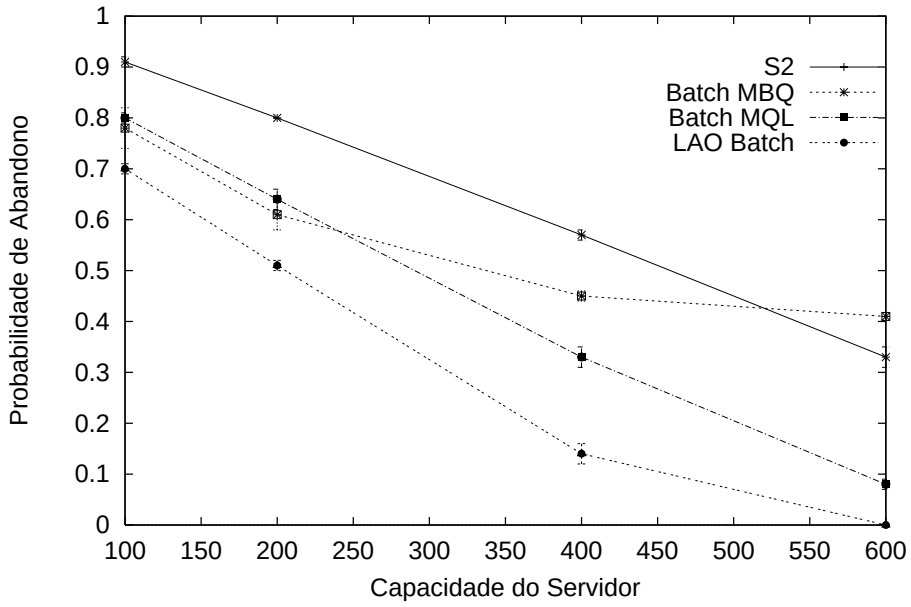

Figura 5: Probabilidade de abandono × capacidade do servidor com taxa de 10 requisições por minuto.

de vídeos não populares.

Agradecimentos

Este trabalho foi parcialmente financiado pelo CNPq/Protem-III ALMADEM, CNPq e FAPESP, processo Nº 96/09739-1, PRONEX. Agradecemos também as sugestões do prof Luiz Fernando Gomes Soares.

Referências

[1] Jean M. McManus and Keith W. Ross. Video on Demand over ATM: Constant-Rate Transmission and Transport. *IEEE Journal on Selected Areas in Communications*, 14(6):1087–1098, Agosto 1996.

[2] Phikip S. Yu, Joel L. Wolf, and Hadas Shachnai. Design and Analysis of a Look-Ahead Scheduling Scheme to Support Pause-Resume for Video-on-Demand Applications. *Multimedia Systems*, 3(4):137–149, Setembro 1995.

[3] Hadas Shachnai and Philip S. Yu. The Role of Wait Tolerance in Effective Batching: A Paradigm for Multimedia Scheduling Schemes. Technical Report RC 20038 (88607), IBM Research Division, T. J. Watson Research Center, Abril 1995.

[4] Asit Dan Dinkar Sitaram and Perwez Shahabuddin. Dynamic Batching Policies for an on-demand Video Server. *Multimedia Systems*, 4:112–121, 1996.

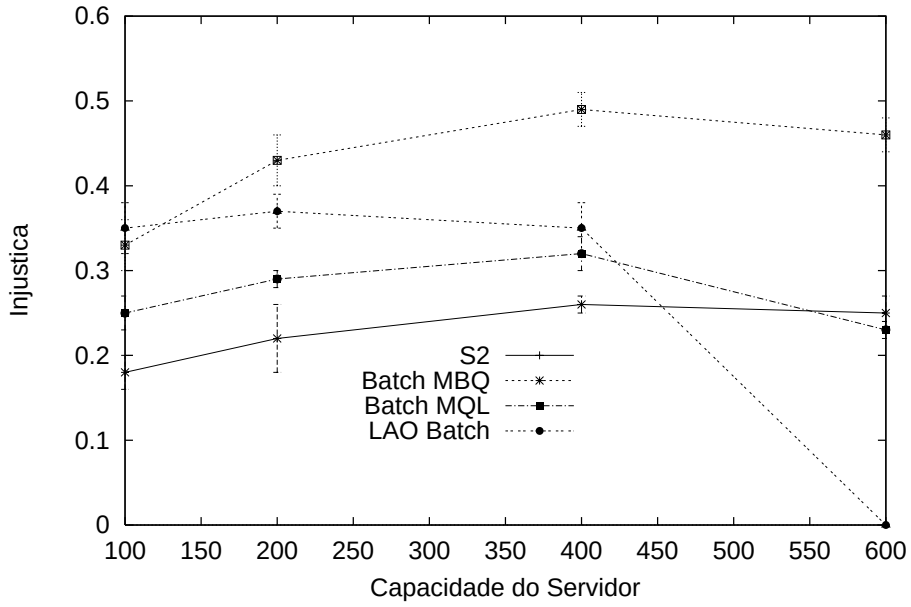

Figura 6: Injustiça × capacidade do servidor com taxa de 10 requisições por minuto.

[5] Leana Golubchik, John C. S. Lui, and Richard Muntz. Adaptive Piggybacking: A Novel Technique for Data Sharing in Video-on-Demand Storage Servers. *Multimedia Systems*, 4(3):140–155, 1996.

[6] Charu C. Aggarwal, Joel Wolf, and Philip S. Yu. On Optimal Piggybacking Merging Policies for Video-on-Demand Systems. In *Proceedings of the ACM Sigmetrics*, volume 24, pages 200–209, New York, Maio 1996. ACM Press.

[7] Roberto de A. Façanha and Nelson L. S. da Fonseca. A Política de Piggybacking S^2. In *Anais do IV Simpósio Brasileiro de Sistemas Multimídia e Hipermídia*, pages 125–136, Rio de Janeiro, RJ, Maio 1998.

[8] Kevin C. Almeroth, Asit Dan, Dinkar Sitaram, and William H. Tetzlaff. Long Term Channel Allocation Strategies for Video Applications. Research Report RC 20249 (89566), IBM Research Division, Outubro 1995.