Arquiteturas para uma Nova Geração de Servidores VoD

Autores
João Caram, Berthier Ribeiro-Neto, Sérgio Campos

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais - 31270-010 Belo Horizonte MG Brasil
{daniela, borges, mribeiro, caram, berthier, scampos}@dcc.ufmg.br

Resumo

Serviços de vídeo sob demanda (VoD) tornaram-se factíveis devido à avanços em tecnologias de redes. O conceito do serviço, entretanto, mudou. Usuários não apenas assistem um vídeo do início ao fim, mas fazem uso de interatividade. Tradicionalmente, a arquitetura é baseada na associação de uma disposição por faixas e um modo de operação do servidor baseado em ciclos. A maioria dos trabalhos na literatura não faz distinção entre as técnicas de disposição dos blocos nos discos e o modo de operação do servidor. Neste artigo, assumimos que eles são independentes. Através de experimentos, estudamos como a disposição e o modo de operação podem ser combinados e o impacto no desempenho do sistema. O modo de operação baseado em ciclos (o mais popular) apresentou os piores resultados. O uso de uma disposição aleatória associada a um modo de operação sem ciclos se mostrou a arquitetura mais promissora.

Abstract

Video on demand (VoD) services have become feasible due to advances in network technologies. The service concept, however, has changed. Users not only watch a video from start to finish, but make use of interactivity. Traditionally, its arquitecture is based on an association between a striping layout and a server operation mode using cycles. The majority of works in the literature does not make distinction between the layout and the server operation mode. In this paper we assume that they are independent. Through experiments, we study how the layout and the operation mode can be combined and the impact of such combinations in the system performance. The operation mode based on cycles (the most popular) presented the worst performance. The use of a random layout associated with an operation mode without cycles resulted in the most promising architecture.

1 Introdução

O objetivo de um sistema de Vídeo sob Demanda (do inglês *Video on Demand* - VoD) é atender clientes interessados em assistir vídeos através de uma rede (de preferência, de alta velocidade). O cliente que deseje assistir um vídeo, armazenado em formato comprimido nos discos do servidor, deve ser capaz de requisitá-lo a qualquer momento. O tempo que o cliente deve esperar desde o momento em que faz uma requisição por um vídeo até o momento em que o vídeo começa a ser exibido é denominado latência.

Um sistema VoD é composto pelos seguintes módulos: um ou mais servidores de vídeo, um sistema de armazenamento que geralmente é representado por um arranjo de discos, clientes e uma rede, servindo como canal de comunicação. O cliente pode ser um PC, uma TV conectada a um dispositivo *set-top box*, ou mesmo um *laptop* conectado a um *link* sem fio. A Figura 1 ilustra um sistema VoD com um único servidor.

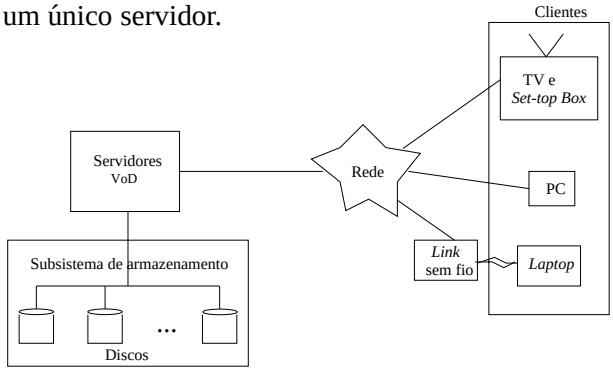


Figura 1: Visão global de um sistema VoD

Serviços VoD são bastante atrativos pois proveêm muito mais flexibilidade a seus clientes do que serviços de TV a cabo ou a tradicional locadora de vídeos ou DVD. Além disso, graças à recentes avanços em tecnologias de redes e comunicações, a disponibilização deste tipo de serviço em larga escala tornou-se factível. Até poucos anos atrás, muitos consideravam que serviços VoD jamais seriam práticos do ponto de vista comercial. A razão deste ceticismo se deve à largura de banda demandada para se transmitir um vídeo (aproximadamente 1,5 Mbps para MPEG1 e 4 Mbps para MPEG2), associada ao fato de que não era comum pessoas terem acesso a serviços de banda larga. Hoje em dia, entretanto, essa situação tem se alterado e, como resultado, existe um grande número de empresas e provedores de serviços investindo dinheiro e gerando lucros com serviços multimídia. Além disso, redes ATM, que são especialmente apropriadas para este tipo de serviço, têm sido implantadas em diversos pontos do mundo, inclusive nas principais cidades do Brasil [Law00, Far97].

Tradicionalmente, a arquitetura de servidores VoD é baseada na associação de uma disposição por faixas e de um modo de operação do servidor baseado em ciclos de serviço [BOS96, BMGJ94, SV97]. Na disposição por faixas, os blocos de vídeo são armazenados nos discos do sistema de maneira *round-robin*. Neste esquema, se o *i-ésimo* bloco de um vídeo X for armazenado no *j-ésimo* disco do servidor, então o (*i-ésimo + 1*) bloco do mesmo vídeo será armazenado no *(j-ésimo + 1) mod (número de discos)* disco do servidor. No modo de operação baseado em ciclos, o servidor envia um bloco de vídeo para cada um dos clientes do sistema durante o ciclo. Esta arquitetura é denominada *striping layout*, sem levar em consideração o componente modo de operação com ciclos.

Existem também outras propostas baseadas em uma disposição aleatória dos blocos de vídeo nos discos do sistema [Kor97] ou na associação desta a um modo de operação sem ciclos [SM98, SMN00]. Em um modo de operação sem ciclos, o servidor simplesmente recebe e processa requisições por blocos de vídeo de seus clientes. Esta arquitetura é denominada *random layout*, sem levar em consideração o componente modo de operação sem ciclos.

Em nosso trabalho, consideramos que a estratégia de disposição e o modo de operação do servidor são independentes e comparamos o desempenho de um servidor VoD em quatro arquiteturas: (i) uma disposição por faixas associada a um modo de operação do servidor com ciclos (Faixas com Ciclos), (ii) uma disposição por faixas associada a um modo de operação do servidor sem ciclos (Faixas sem Ciclos), (iii) uma disposição aleatória associada a um modo de operação do servidor com ciclos (Aleatória com Ciclos) e (iv) uma disposição aleatória associada a um modo de operação do servidor sem ciclos (Aleatória sem Ciclos).

Este artigo é organizado como se segue. Na seção 2, nós caracterizamos o servidor VoD utilizado em nossos experimentos. Na seção 3, nós descrevemos as duas principais estratégias de disposição encontradas na literatura: disposição por faixas e aleatória. Na seção 4, nós caracterizamos os dois modos de operação de um servidor VoD: com ciclos e sem ciclos. Na seção 5, apresentamos os resultados de nossos experimentos com um servidor real. Finalmente, na seção 6, apresentamos nossas conclusões e sugerimos trabalhos futuros.

2 Servidor VoD

O servidor é a parte central de um sistema VoD. Ele é um computador contendo CPU, uma quantidade de memória para *buffering* e discos que são usados para armazenar os vídeos.

O servidor utilizado em nossos experimentos é o ALMADEM-VoD [BCJ+99, CNBM99]. Este servidor foi projetado e desenvolvido, no contexto do projeto pesquisa ALMADEM na Universidade Federal de Minas Gerais para ser um servidor multimídia de baixo custo, construído sobre arquitetura PC e utilizando sistema operacional Linux.

2.1 Controle de Admissão

O controle de admissão é a estratégia usada por um sistema VoD para verificar se um novo cliente pode ou não ser atendido, tendo como base os recursos do sistema e o número de clientes que já estão sendo atendidos.

Estudos encontrados na literatura indicam que as estratégias de controle de admissão estatísticas (onde os *deadlines* das requisições de bloco serão garantidos com uma dada probabilidade) apresentam desempenho superior às determinísticas (onde todos os *deadlines* das requisições de bloco são garantidos) [KY00, MNH97, VGGG94a, VGGG94b]. Por esta razão, nosso servidor usa um controle de admissão estatístico.

Nós adotamos a mesma estratégia usada em [San98]: uma abordagem de garantia de atraso estatística. O servidor garante que as requisições serão atendidas, dado um limite de atraso, com uma probabilidade bastante alta. A probabilidade de se exceder este limite de atraso é de 10^{-6}. Em outras palavras, nós assumimos que o nosso sistema apresenta um comportamento satisfatório se apenas uma em cada um milhão de requisições sofrerem um atraso maior que o limite de atraso.

Aqui, o atraso da requisição é o tempo decorrido desde o momento em que a requisição é gerada até o momento em que o bloco de vídeo associado a ela é armazenado no *buffer* do cliente.

Computando o Limite de Atraso

Neste trabalho, computamos o limite de atraso da mesma forma efetuada em [San98]. Assumimos que há um número inteiro *nb* de *buffers* ($nb \geq 2$) no lado do cliente e que cada um desses *buffers* pode armazenar um bloco de vídeo de tamanho *bs*. Em qualquer momento, um dos *buffers* conterá o bloco de vídeo ativo b_i, que está sendo consumido pelo decodificador. Os outros $(nb - 1)$ *buffers* são usados para armazenar os próximos blocos de vídeo a serem consumidos pelo decodificador do cliente. Estes *buffers* podem ter dados de vídeo prontos para consumo ou podem estar esperando dados que ainda serão lidos do sistema. Quando o bloco de vídeo ativo b_i é completamente consumido pelo decodificador, seu *buffer* torna-se disponível para armazenar um novo bloco de dados (b_{i+nb}). O bloco (b_{i+nb}) deve ser armazenado no *buffer* do cliente antes do decodificador começar a consumí-lo, de forma a não causar qualquer interrupção na exibição do vídeo.

Logo, o limite de atraso (*db*), ou seja, o tempo máximo que uma requisição de bloco pode demorar do momento de sua geração até o armazenamento do bloco no *buffer* do cliente é dado por:

$$\begin{aligned} db &= dt * (nb - 1) \\ &= \frac{bs}{dr} * (nb - 1) \end{aligned} \quad (1)$$

Nesta equação, *dt* é o tempo que o decodificador gasta decodificando um bloco de vídeo, que é dado por bs/dr, onde *dr* é a taxa de decodificação em kilobytes por segundo (KB/s).

2.2 Escalonamento de Discos

Nosso servidor ALMADEM-VoD faz uso de dois algoritmos de escalonamento para decidir qual requisição de bloco é a próxima a ser atendida nas filas de seus discos.

Para o modo de operação com ciclos, fazemos uso da política de escalonamento SCAN. Neste algoritmo, as requisições são servidas em ciclos e na ordem das posições de seus cilindros na superfície do disco. A cabeça do disco sempre se move na mesma direção em um dado ciclo, ora em direção aos cilindros mais interiores, ora em direção aos mais exteriores. O objetivo deste algoritmo é minimizar o tempo de *seek* e latência [GKS95]. É o algoritmo de escalonamento mais usado em servidores VoD [SMN00].

Para o modo de operação sem ciclos, nós usamos a política FIFO de escalonamento. Este algoritmo serve as requisições pela ordem de chegada. Apesar de algoritmos baseados em SCAN poderem reduzir a latência de *seek* e proverem tempos de I/O menores, a política FIFO minimiza as variações de atraso de requisições, o que é um importante fator para uma medida de desempenho baseado em limites de atraso estocásticos [San98].

3 Estratégias de Disposição

A disposição é o mapeamento dos blocos de vídeo nos discos do servidor. Há dois esquemas principais de disposição frequentemente citados na literatura e usados no projeto e implementação de um servidor VoD.

3.1 Faixas

A disposição por faixas é discutida em [BOS96, BMGJ94, CLOT96, SV97]. Através deste esquema, os blocos de um vídeo são distribuídos pelos discos do servidor de maneira *round-robin*. Assim, se o *i-ésimo* bloco de um vídeo X for armazenado no *j-ésimo* disco do servidor, então o (*i-ésimo + 1*) bloco do mesmo vídeo será armazenado no *(j-ésimo + 1) mod (número de discos)* disco do servidor. Como resultado, um padrão de acesso sequencial ao vídeo X gerará requisições de bloco que se moverão de um disco para o outro de maneira circular, lembrando o movimento de um carrossel. A Figura 2 mostra um vídeo cujos blocos estão distribuídos pelos discos do servidor de acordo com a disposição por faixas. O disco que possui o primeiro bloco de um determinado vídeo pode ser escolhido aleatoriamente.

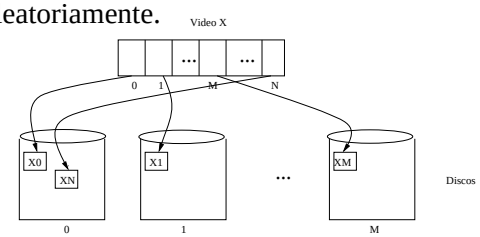


Figura 2: Exemplo da disposição por faixas

A disposição por faixas é considerada boa para padrões de acesso sequenciais, como vídeo e áudio, devido ao fato da carga ser distribuída balanceadamente entre os discos do servidor. No entanto, esta disposição deixa de resultar em um balanceamento de requisições de blocos nos discos quando o padrão de acesso dos clientes é variável (uso funções de videocassete, taxas de decodicação variáveis, mundos virtuais).

3.2 Aleatória

A disposição aleatória é estudada em [Kor97, SMN00, San98, SMW96]. Nesta disposição, os blocos de um vídeo são distribuídos aleatoriamente entre os discos do servidor. Um bloco é armazenado em um disco e em uma posição dentro deste disco escolhidas aleatoriamente.

A disposição aleatória é apropriada para uma variedade de padrões de acesso, como os gerados pelo uso de funções de videocassete, aplicações interativas, mídias codificadas com diferentes técnicas e aplicações heterogêneas. Ela também se adapta bem ao que é chamado na literatura de *incremental growth* [TMDV96]: se discos são adicionados frequentemente ao sistema, a redistribuição dos blocos de vídeo pelos discos do servidor é efetuada de forma bastante simples.

Entretanto, como não se pode assumir nada sobre o padrão de acesso, a disposição aleatória pode causar sobrecarga em alguns discos: é estatisticamente possível que um grande número de requisições de bloco sejam direcionadas simultaneamente para um mesmo disco. Isso é particularmente válido para sistemas de VoD (especialmente quando os blocos de vídeo são acessados sequencialmente) porque vídeos têm diferentes popularidades e, consequentemente, alguns discos podem ser acessados mais frequentemente que outros. A Figura 3 mostra um vídeo cujos blocos estão armazenados no sistema de acordo com a disposição aleatória.

4 Modo de Operação

O modo de operação se refere a qual entidade do sistema é responsável pelo escalonamento das requisições de bloco. As requisições podem ser geradas tanto pelos clientes (sem ciclos) quanto

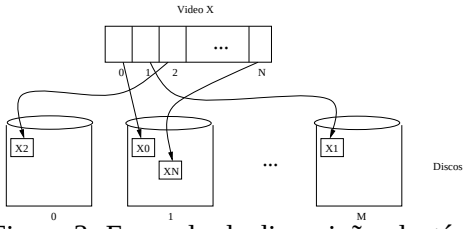


Figura 3: Exemplo da disposição aleatória

pelo servidor (com ciclos).

4.1 Com Ciclos

Neste modo de operação o cliente envia uma requisição por um vídeo ao servidor. Depois disso, o servidor é responsável por controlar quando o próximo bloco de vídeo deve ser lido de um de seus discos e enviado ao cliente. O servidor opera em ciclos de serviço de tamanho fixo e, em cada um destes ciclos, envia um bloco de vídeo para cada cliente sendo atendido no sistema.

O tamanho do ciclo é determinado pela taxa de decodificação dos blocos enviados para o cliente durante aquele ciclo (cada cliente tem que receber um novo bloco antes do término da exibição do bloco corrente). Para um servidor provendo dados a seus clientes a uma taxa aproximadamente constante (tráfego CBR - *Constant Bit Rate*), onde todos os clientes decodificam os dados a uma mesma taxa, o tempo do ciclo é igual ao tempo de decodificação de um bloco.

Apesar de popular na literatura, o uso de um modo de operação com ciclos faz com que os discos operem sincronizadamente, podendo acarretar na ociosidade dos mesmos no final dos ciclos, o que reduz o desempenho do servidor. Por exemplo, suponha um servidor com 2 discos, onde um dos discos atende o número máximo de requisições que pode suportar e o segundo atende apenas metade de sua capacidade. Se um novo cliente entra no sistema no meio de um ciclo, ele terá de esperar até o fim do ciclo para começar a ser atendido, mesmo que o bloco que esteja requisitando esteja armazenado no disco que possui capacidade para atendimento. Além disso, este modo de operação torna mais complexa a implementação do servidor que, além de ler blocos de seus discos, deve também saber o momento de enviá-los a cada cliente e controlar o início e fim de cada ciclo de serviço.

4.2 Sem Ciclos

Neste modo de operação o cliente é responsável por enviar requisições de blocos ao servidor. O servidor não opera em ciclos de serviço, o que permite que os discos operem independentemente (sem pontos de sincronização). O servidor simplesmente recebe e processa a requisição, que é redirecionada para a fila do disco que armazena o bloco requisitado.

5 Experimentos

Nesta seção nós descrevemos os experimentos que realizamos e os resultados obtidos para o desempenho do servidor ALMADEM-VoD operando em cada uma das quatro arquiteturas resumidas na Tabela **??**.

5.1 Descrição dos Experimentos

Em nossos experimentos, o servidor ALMADEM-VoD foi executado em um computador com processador Pentium-Pro à 200 Mhz, com 128 MB de memória RAM e 4 discos rígidos IDE Seagate U8-17221 de 17,2 GB de capacidade cada.

Para comparação entre as configurações, foram efetuados experimentos que mediram o número máximo de clientes que cada arquitetura do servidor podia atender simultaneamente (para cada tamanho de vídeo selecionado) e também a latência média para os clientes atendidos.

Vídeos

Nossos vídeos foram divididos em blocos de 512 KB (kilobytes) e armazenados nos discos do sistema de acordo com uma determinada disposição. Apesar de podermos ter variado o tamanho dos blocos, concentramos nossa atenção em blocos de 512 KB porque, de acordo com [San98], o tamanho de bloco ótimo para vídeos comuns varia entre 500-600 KB.

Nós selecionamos três tipos de vídeos em nossa análise: clipes de notícias (até 4 minutos), seriados (entre 20 e 30 minutos) e filmes completos (entre 1:40 e 2:40 h). Nossa intenção era analisar se o tipo do vídeo teria alguma influência no desempenho do servidor. A Tabela 1 mostra as características destes tipos de vídeos quando armazenados nos discos do servidor ALMADEM-VoD.

Tipo	Número de Blocos	Vídeos por Disco
Filmes	2290 - 3609	11
Seriados	451 - 675	61
Clipes de Notícias	45 - 90	513

Tabela 1: Vídeos

Os vídeos foram codificados utilizando-se da técnica MPEG1 [Jac96]. Logo, a taxa de decodificação dos vídeos foi de aproximadamente 1,5 Mbps ou 192 KB/s.

Número de *Buffers*

Nós consideramos 2 e 4 *buffers* no lado do cliente ($nb = 2$ e $nb = 4$). O primeiro *buffer* armazena o bloco ativo sendo consumido pelo decodificador MPEG no momento e os outros ($nb - 1$) *buffers* armazenam os próximos blocos de vídeo sendo entregues pelo servidor.

Geração dos Clientes

Para cada arquitetura do servidor a ser avaliada, foram geradas cargas com números variados de clientes. Cada cliente gerado requisitava um vídeo e, ao terminar seu atendimento, um novo cliente, requisitando um novo vídeo, era gerado, de modo a manter o número de clientes atendidos constante.

A escolha dos vídeos pelos clientes foi modelada segundo a Lei de Zipf [BYRN99]. Esta lei é utilizada em vários modelos de popularidade, inclusive na modelagem da locação de vídeos em locadoras dos Estados Unidos [Jam99], se mostrando adequada para nossos experimentos.

5.2 Resultados

As tabelas 2 e 3 detalham o número máximo de clientes que o servidor pode atender simultaneamente para as nossas 4 arquiteturas, usando 2 e 4 *buffers* no lado do cliente e variando o tipo do vídeo.

Vídeo	Faixas c/ Ciclos	Faixas s/ Ciclos	Aleat. s/ Ciclos	Aleat. c/ Ciclos
Filmes	76	80	78	58
Seriados	76	80	78	60
Clipes de notícias	76	80	78	60

Tabela 2: Número máximo de clientes no sistema para as 4 arquiteturas, com 3 tipos de vídeos e 2 *buffers*

Vídeo	Faixas c/ Ciclos	Faixas s/ Ciclos	Aleat. s/ Ciclos	Aleat. c/ Ciclos
Filmes	76	90	90	60
Seriados	76	90	90	60
Clipes de Notícias	76	90	95	60

Tabela 3: Número máximo de clientes no sistema para as 4 arquiteturas, com 3 tipos de vídeo e 4 *buffers*

Nós observamos que, para nossos experimentos, o tamanho do vídeo quase não causou impacto no desempenho do servidor. Para todos os três tamanhos de vídeo selecionados, o servidor conseguiu atender aproximadamente o mesmo número de clientes.

As arquiteturas baseadas em um modo de operação sem ciclos apresentaram o melhor desempenho. A arquitetura Aleatória sem Ciclos chega a apresentar um desempenho 34% melhor do que a Aleatória com Ciclos (2 *buffers* e filmes completos). Já para a disposição por faixas, a arquitetura Faixas sem Ciclos apresenta um desempenho 5% melhor do que a arquitetura Faixas com Ciclos.

Pelos dados exibidos nas tabelas podemos perceber que as arquiteturas com modo de operação baseado em ciclos não são muito sensíveis ao aumento do número de *buffers* no lado do cliente. Tanto com 2 quanto para 4 *buffers* o servidor atendia aproximadamente o mesmo número de clientes. Já para as arquiteturas com modo de operação sem ciclos, o servidor chega a atender 21% mais clientes (Clipes de notícia e arquitetura Aleatória sem Ciclos) quando utiliza 4 *buffers* ao invés de 2.

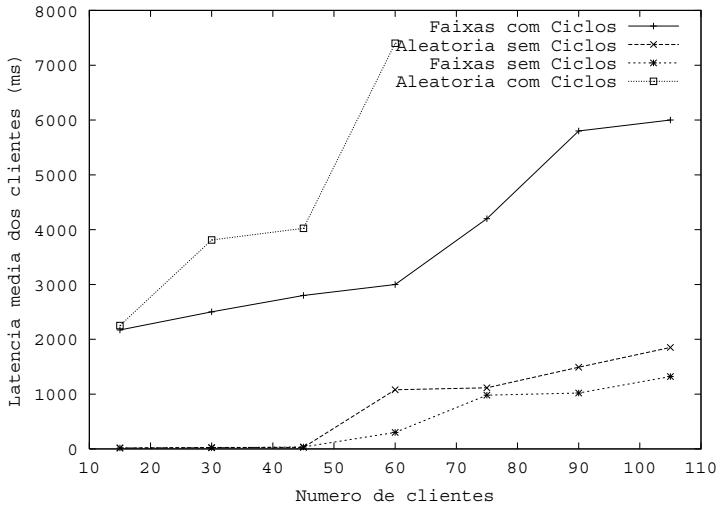
Outro fato observado é que a disposição por faixas (para qualquer modo de operação) apresenta um desempenho superior à disposição aleatória (até 31% para arquiteturas com modo de operação baseado em ciclos, 2 *buffers* e filmes completos). Entretanto, é importante ressaltar que para um modo de operação sem ciclos essa vantagem de desempenho é extremamente pequena (cerca de 2%) e que para 4 *buffers* as arquiteturas que utilizam a disposição aleatória tiveram desempenho igual ou superior (clipes de notícias) as que utilizam a disposição por faixas.

A Figura 4 ilustra como a latência média dos clientes variou com o aumento do número de clientes no sistema para nossas quatro arquiteturas, 2 *buffers* no lado do cliente e com o servidor armazenando clipes de notícias.

Observando as curvas, notamos que, mais uma vez, as arquiteturas que utilizam um modo de operação baseado em ciclos apresentam o pior desempenho, ou seja, resultam em latências iniciais muito mais altas para um cliente que acabou de entrar no sistema.

Com relação à disposição, concluimos que uma disposição por faixas leva à latências menores que a disposição aleatória. Entretanto, para um modo de operação sem ciclos, esta diferença é extremamente pequena.

Pelos resultados obtidos, vemos que o uso de um modo de operação baseado em ciclos de serviço, em qualquer das combinações analisadas, apresenta o pior desempenho. Como já men-



(a) (b)

Figura 4: Latência média dos clientes x Número de clientes - Clipes de notícias - 2 *buffers*

cionamos anteriormente, neste modo de operação os discos do servidor operam de forma sincronizada. A cada ciclo, o servidor direciona um certo número de requisições aos seus discos, e mesmo que um disco receba um número pequeno de requisições, após atendê-las, ele deve esperar ociosamente pelo fim do ciclo corrente para que possa atender novas requisições no próximo ciclo. Se uma nova requisição é gerada por um novo cliente no meio de um ciclo, o disco que armazena o bloco requisitado só irá atender o novo cliente no próximo ciclo, mesmo que esteja atendendo um número pequeno de clientes ou esteja ocioso. Isso resulta em um aumento significativo da latência média dos clientes. O uso de ciclos de tamanho fixo faz com que o servidor atenda um número simultâneo de clientes bem menor que sua capacidade. Mesmo que haja tempo sobrando em determinado ciclo, uma requisição de bloco só é atendida se o tempo que o servidor for gastar para ler dos discos o bloco requisitado couber inteiramente no tempo que resta do ciclo. E isso, pode resultar em uma sub-utilização do servidor.

Com relação à disposição, a disposição por faixas resultou, em nosso estudo, em latências menores que a disposição aleatória. Isso pode ser explicado pelo fato de que o padrão de acesso neste caso foi estritamente sequencial, ou seja, os clientes assistiam os vídeos do início ao fim, sem interrupção. E para padrões de acesso sequenciais, a disposição por faixas provê uma distribuição mais balanceda dos blocos de vídeo nos discos do sistema que a disposição aleatória.

6 Conclusões e Trabalhos Futuros

Tradicionalmente, a arquitetura de servidores VoD é baseada na associação de uma disposição por faixas e de um modo de operação do servidor baseado em ciclos de serviço Esta arquitetura é denominada *striping layout*, sem levar em consideração o componente modo de operação com ciclos. Existem também outras propostas baseadas em uma disposição aleatória dos blocos de vídeo associada a um modo de operação sem ciclos. Esta arquitetura é denominada *random layout*,

sem levar em consideração o componente modo de operação sem ciclos.

Nos trabalhos encontrados na literatura, não se faz uma distinção entre a estratégia de disposição e o modo de operação como componentes distintos que constituem a arquitetura de um servidor. Em nosso trabalho, consideramos que a estratégia de disposição e o modo de operação do servidor são independentes e comparamos o desempenho de um servidor VoD em quatro arquiteturas: (i) uma disposição por faixas associada a um modo de operação do servidor com ciclos (Faixas com Ciclos), (ii) uma disposição por faixas associada a um modo de operação do servidor sem ciclos (Faixas sem Ciclos), (iii) uma disposição aleatória associada a um modo de operação do servidor com ciclos (Aleatória com Ciclos) e (iv) uma disposição aleatória associada a um modo de operação do servidor sem ciclos (Aleatória sem Ciclos).

Nossos resultados indicam que um modo de operação baseado em ciclos de serviço não é a escolha mais apropriada para qualquer disposição escolhida, apresentando sempre o pior desempenho. Este resultado desafia o conhecimento tradicional na área, pois a maioria dos trabalhos envolvendo servidores VoD hoje encontrados na literatura utiliza este modo de operação. As arquiteturas com modo de operação sem ciclos apresentaram melhor desempenho tanto em termos do número máximo de clientes que o servidor podia atender quanto em termos da latência média experimentada pelos clientes.

Com relação à escolha da disposição, apesar da disposição por faixas ter apresentado um desempenho um pouco melhor que a disposição aleatória (diferença pequena para o modo de operação sem ciclos), nós acreditamos que a disposição aleatória apresenta mais vantagens que a por faixas: a disposição aleatória oferece muito mais flexibilidade e menos complexidade ao sistema quando levamos em consideração questões como tolerância a falhas dos discos, inclusão de discos ao sistema, utilização de discos e aplicações heterogêneas, interatividade dos clientes e das aplicações (mundos virtuais) e mudanças no padrão de acesso. Logo, acreditamos que a arquitetura Aleatória sem Ciclos (uma disposição aleatória combinada com um modo de operação do servidor sem ciclos) é a mais promissora para a uma nova geração de servidores VoD. Ademais, concluimos que o tamanho dos vídeos exerce pouca ou quase nenhuma influência no desempenho do servidor.

Como trabalhos futuros pretendemos efetuar esta análise utilizando outras variáveis como aplicações heterogêneas, clientes utilizando funções de videocassete, e vídeos com diferentes tamanhos de bloco e taxas de decodificação.

Referências

[BCJ+99] L. Bertini, S. Campos, G. Jamil, A. Macêdo, B. R. Neto, C. Santos, and D. S. Alvim. Análise de desempenho do servidor de vídeo almadem-vod. *V Simpósio Brasileiro de Sistemas Hipermídia e Multimídia - SBMIDIA'99*, June 1999.

[BMGJ94] S. Berson, R. Muntz, S. Ghandeharizadeh, and X. Ju. Staggered striping in multimedia informatiom systems. *ACM Sigmod*, 1994.

[BOS96] Rajeev Rastogi Banu Ozden and Avi Silberschatz. Disk striping in video server environments. *IEEE International Conference on Multimedia Computing and Systems*, pages 172–180, June 1996.

[BYRN99] R. B. Yates and B. R. Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[CLOT96] T. S. Chua, J. Li, B. C. Ooi, and K. L. Tan. Disk striping strategies for large video on demand servers. *ACM Multimedia 96*, pages 297–306, June 1996.

[CNBM99] Sérgio C., B. R. Neto, Luciano Bertini, and Autran Macêdo. Verification and analysis of multimedia systems. *ACM Multimedia 99*, pages 131–140, November 1999.

[Far97] David J. Farber. Communications technology and its impact by 2010. *Communications of the ACM*, 40(2):135–138, February 1997.

[GKS95] S. Ghandeharizadeh, S. Kim, and C. Shahabi. On configuring a single disk continuous media server. *ACM SIGMETRICS Performance*, June 1995.

[Jac96] K. Jack. *Video Desmystified*. HighText Publications, 1996.

[Jam99] George Leal Jamil. Análise de desempenho do servidor de vídeo almadem - vod por simulação. Master's thesis, Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, 1999.

[Kor97] J. Korst. Random duplicated assigment: An alternative to striping in video servers. *ACM Multimedia 97*, pages 219–26, 1997.

[KY00] Sooyong Kang and Heon Y. Yeom. Statistical admission control for soft real-time vod servers. *ACM Multimedia 2000*, 2000.

[Law00] George Lawton. Video streams into the mainstream. *Computer*, 33(7), July 2000.

[MNH97] D. Makaroff, G. Neufeld, and N. Hutchinson. An evaluation of vbr disk admission algorithms for continuous media file servers. *ACM Multimedia 97*,1997.

[San98] José Renato Gonçalves Santos. *RIO: A Universal Multimedia Storage System Based on Random Data Allocation and Block Replication*. PhD thesis, Department of Computer Science, University of California, Los Angeles, 1998.

[SM98] José Renato Santos and Richard R. Muntz. Performance analysis of the rio multimedia storage system with heterogeneous disk configurations. *ACM Multimedia 98*, pages 303–308, September 1998.

[SMN00] J.R. Santos, R.R. Muntz, B.R. Neto. Comparing random data allocation and data striping in multimedia servers. *ACM SIGMETRICS 2000*, pages 44–55, June 2000.

[SMW96] S.Berson, R. Muntz, and W. Wong. Randomized data allocation for real-time disk i/o. *Compcon 96*, pages 286–90, 1996.

[SV97] P. Shenoy and H. Vin. Efficient striping techniques for multimedia file servers. *7th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 97)*, pages 25–36, May 1997.

[TMDV96] R. Tewari, R. Mukherjee, D. M. Dias, and Harrick M. Vin. Design and performance tradeoffs in clustered video servers. *International Conference on Multimedia Computing and Systems*, pages 144–150, 1996.

[VGGG94a] Harrick M. Vin, Pawan Goyal, Alok Goyal, and Anshuman Goyal. An observation-based admission control algorithm for multimedia servers. *IEEE International Conference on Multimedia Computing and Systems*, May 1994.

[VGGG94b] Harrick M. Vin, Pawan Goyal, Alok Goyal, and Anshuman Goyal. A statistical admission control algorithm for multimedia servers. *ACM Multimedia 94*, 1994.