Cristina Duarte Murta

Modelo de Particionamento de Espaço para Caches da World Wide Web

Tese apresentada ao Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Universidade Federal de Minas Gerais
Belo Horizonte, agosto de 1999

Resumo

A WWW apresenta duas características que desafiam a avaliação de desempenho e as propostas de solução para seus problemas: larga escala e grande variabilidade. Esta tese trata de sistemas de cache cujos objetos apresentam variabilidade extrema nos seus tamanhos. Os caches da WWW são um exemplo. Esta tese apresenta um estudo sobre a influência da variabilidade dos tamanhos dos objetos da WWW no desempenho dos seus sistemas de cache e uma solução para o problema gerado por essa variabilidade, o modelo de organização do espaço denominado PART.

A variabilidade nos tamanhos dos documentos tem duas conseqüências importantes. A primeira é um compromisso de desempenho entre as métricas HR (*hit ratio*) e BHR (*byte hit ratio*), devido ao compartilhamento de espaço para arquivos de tamanhos distintos. O armazenamento preferencial de arquivos grandes favorece BHR enquanto melhor HR é obtido armazenando-se os arquivos pequenos. Sistemas de cache existentes da WWW apresentam bons resultados isoladamente em cada uma destas métricas. No entanto, a obtenção de bons resultados em ambas é de reconhecida importância. A outra conseqüência é o impacto causado pelas substituições que não levam em consideração os tamanhos dos arquivos envolvidos.

Esta tese propõe que o gerenciamento do espaço destes caches seja feito em dois níveis: a organização do espaço e a política de reposição. Para a organização do espaço é proposto o modelo PART. O espaço do cache é dividido em partições que armazenam classes de arquivos definidas pelo tamanho. O PART impõe restrições de tamanho para as substituições no cache minimizando os efeitos da variabilidade. A classificação por tamanhos se adequa bem à implementação de políticas específicas para a otimização de cada métrica. Os parâmetros do modelo permitem ajustes para adequação do modelo à carga. A combinação dos vários benefícios assegura o melhor desempenho conjunto em HR e BHR. Estes resultados foram obtidos por simulação e comprovados através do conceito de mapas de desempenho, também introduzido nesta tese. Os mapas de desempenho são construídos a partir de um teorema original que estabelece a relação entre HR e BHR e auxiliam o entendimento do comportamento das estratégias para gerência de espaço nos caches.

Abstract

Caching systems are considered an essential part of the WWW to deal with its remarkable growth. The WWW shows two characteristics that are challenges for performance evaluation and for the design of solutions for its problems: high variability and large scale. This thesis deals with caching systems with objects that show extreme variability in their sizes. WWW caching systems are an example. This thesis presents a study about the influence of the variability of the Web objects' size in the performance of its caching systems and a solution for the problem due to the variability, a model for cache space organization called PART.

The variability of the document sizes has two important consequences. Firstly, there is the performance tradeoff between the metrics HR (*hit ratio*) and BHR (*byte hit ratio*), due to the space sharing of files of very different sizes. While the storage of large files favors BHR, good HR is achieved keeping preferentially the small files in the cache. Existing WWW caching systems show good results for each of these metrics isolately. However, it is recognized the importance of having good results in both of them. The second consequence of the variability is the impact due to the replacement of files that do not take into consideration the sizes of the files.

This thesis proposes two-level space cache management: the space organization and the definition of the replacement policy. To organize the cache space the PART model is proposed. The cache space is divided into partitions that hold classes of files defined according to the file sizes. PART imposes restrictions for the replacements based on the objects' size, minimizing the effects of the variability. Replacement policies specifically designed to optimize each metric can be used in the different partitions. The parameters of the model allow fine tuning according to changings in the workload. The combination of all benefits provides the best combined performance in HR and BHR. The results were obtained by simulation and validated through the concept of performance maps, which is also introduced in this thesis. Performance maps are built from an original theorem which establishes the relationship between HR and BHR. Performance maps help to reveal the strategies of the approaches for space cache management in a combination of both metrics.

Agradecimentos

Pesquisar é pisar em terreno desconhecido, inclui erro e risco, não há como fazer previsão. Como determinar um tempo para cumprir um objetivo de pesquisa? No entanto, ao fazer pesquisa, em algum momento o pesquisador deverá se render à realidade e conviver com as limitações de tempo e recurso, reconhecendo esses limites que o próprio homem impõe ao desenvolvimento da ciência. Mesmo quando há mais experimentos a fazer, mais referências a pesquisar e mais casos a verificar, em algum momento é preciso escrever algo e se contentar com aquilo.

Isso também é verdade para o processo de reconhecer aqueles que tornaram o trabalho possível. Há sempre alguém esquecido e alguma forma mais eloqüente ou bonita de agradecer mas em algum momento temos de escrever mesmo sabendo dos riscos imputados pelos limites de nossa memória e habilidade literária. Dentro destas limitações, aqui vão meus agradecimentos.

A meu orientador, Virgilio Almeida, minha gratidão por seu incentivo constante, pela orientação e pela oportunidade que me proporcionou de fazer na *Boston University* um estágio em pesquisa. Ao professor Mark Crovella, por me receber tão bem na *Boston University* e pela orientação. Ao professor Nivio Ziviani, meu orientador de mestrado, pelo constante incentivo e apoio. Ao professor Ivan Moura Campos, também pelo apoio de longa data. Trabalhar com estes pesquisadores foi um privilégio para mim. Tenho por eles grande admiração.

Agradeço aos vários anônimos profissionais que fizeram os registros das cargas de acesso utilizadas neste trabalho, sem os quais o mesmo poderia não ter sido realizado. Sua contribuição foi inestimável. A todos os colegas do Laboratório de Análise e Modelagem de Desempenho de Sistemas de Computação, do Departamento de Ciência da Computação da UFMG, por tornarem agradável e divertido o convívio diário, em meio a tantas discussões técnicas.

Aos meus colegas professores do Departamento de Informática da Universidade Federal do Paraná devo meus sinceros agradecimentos pelo incentivo, em especial aos chefes de Departamento, professores Carlos Alberto de Carvalho, Armando Delgado e Alexandre Direne. Agradeço também a CAPES pelo apoio financeiro.

Aos meus amigos, que ouviram demasiadamente a palavra "tese" especialmente nos últimos meses, meu agradecimento pela compreensão e pelos momentos tão agradáveis que indubitavelmente tornaram o trabalho mais suave e profícuo.

Aos meus filhos Tomás e Belisa, por tornarem a vida tão interessante e alegre, e saberem respeitar esse trabalho mesmo sem compreender o seu significado. À minha mãe Joanita e meus irmãos Fernando, Kátia e Júnior por seu ilimitado e constante amor e encorajamento. Palavras não são suficientes para expressar meu amor e gratidão.

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

A World Wide Web (referenciada neste trabalho como WWW ou Web) tornou-se, em apenas seis anos, o maior e mais acessado sistema de informação do planeta. Este desenvolvimento ocorreu de forma natural, não planejada, e a facilidade de conexão à Web, como cliente ou servidor, é um dos fatores que permitiu o crescimento acelerado. A popularização da WWW gerou sobrecarga na Internet e nos servidores de páginas, acarretando tempos longos de resposta a requisições, inaceitáveis para os usuários. A utilização generalizada de caches em diversos pontos da rede no mundo aliviou o problema das latências nas requisições e mostrou ser uma solução para a escalabilidade da WWW.

Sistemas de cache são considerados fundamentais na arquitetura da WWW atual. O modelo cliente-servidor da WWW suporta caches em seus três componentes: cliente, servidor e rede. A utilização de sistemas de cache na Web traz vários benefícios. Em primeiro lugar, os clientes observam uma redução na latência de suas requisições, que são servidas por caches localizados em pontos intermediários entre o cliente e o servidor, portanto mais próximos do cliente. Em segundo lugar, há uma redução da carga nos servidores de páginas, que passam a servir cópias de suas páginas para os caches, atendendo a um número menor de usuários individuais. Com isso há um aumento da disponibilidade dos documentos copiados pelos caches. Se o servidor fica indisponível, o documento pode ser servido pelo cache, ainda que sua versão não seja a mais recente. Por último, mas talvez mais importante, caches na rede e no cliente contribuem para a economia na utilização da rede. O tráfego redundante, gerado por múltiplas transferências do mesmo documento, é eliminado do trecho entre o servidor e o cache, e a rede fica dedicada a transferências de documentos novos ou modificados. O descongestionamento da rede permite melhor utilização da Internet e contribui para a melhoria da qualidade de serviço.

Há duas motivações para este trabalho. A primeira é a importância da implementação de caches para a escalabilidade da WWW. A WWW está se desenvolvendo rapidamente e não há indicação de que esta tendência irá se reverter em futuro próximo. Pelo contrário, o advento de interfaces mais baratas como a WebTV permitirá um crescimento do número

de usuários. O uso de redes mais rápidas como a Internet 2 possibilitará a popularização de aplicações que utilizam áudio e vídeo. O crescimento do comércio eletrônico diversificará ainda mais a variedade de serviços oferecidos pela WWW. O tráfego na Internet aumentará como conseqüência destas inovações, em "uma escala que é difícil de imaginar", segundo a *International Data Corporation* (International Data Corporation, 1998). A utilização de caches é uma técnica bem conhecida para melhorar a escalabilidade de sistemas cliente-servidor, isto é, aumentar o número máximo de clientes que podem ser servidos, minimizando os pontos de contenção na rede ou no servidor.

Por outro lado, a gerência de espaço em caches na WWW é um problema novo, com características diferentes dos sistemas de cache encontrados em arquitetura de computadores e sistemas operacionais. Esta é a segunda motivação. Inicialmente, os algoritmos para a implementação dos caches na WWW foram buscados nos estudos sobre sistemas de cache em arquitetura de computadores e sistemas operacionais. No entanto, embora haja consolidação da pesquisa sobre algoritmos de troca de blocos e páginas nestes ambientes tradicionais, sistemas de cache na Web são mais complexos e, devido às suas peculiaridades, compõem um tema a ser estudado separadamente.

A WWW apresenta duas características que desafiam a avaliação de desempenho e as propostas de solução para seus problemas: larga escala e grande variabilidade. O componente de larga escala é dado pelo seu tamanho. A WWW é composta por milhões de objetos, milhões de usuários e milhares de servidores. Para seus sistemas de cache, isto significa que há um grande número de objetos candidatos ao armazenamento. Outra característica fundamental é a variabilidade extrema observada nas medidas da carga, por exemplo, nos tamanhos dos documentos que circulam na Web. Há documentos que são várias ordens de grandeza maiores do que outros. Enquanto os caches tradicionais existentes nos sistemas de computação tratam de objetos de tamanho único (ex.: bloco ou linha), os sistemas de cache na Web operam com objetos de tamanhos extremamente variáveis. Como conseqüência, o armazenamento de apenas um documento grande no cache pode significar a retirada de centenas ou milhares de documentos pequenos, alterando completamente seu estado. A variabilidade também afeta as latências dos Web-caches, cujos valores estão em escalas de tempo muito maiores e são mais variáveis do que as latências dos caches tradicionais. Estas duas características são determinantes nas diferenças apontadas entre os caches da WWW e os sistemas tradicionais de cache. Como pode ser observado, para descrever a WWW o superlativo é comum e necessário.

Estas diferenças são reconhecidas na literatura. Por exemplo, algumas políticas de substituição de objetos em caches da WWW levam em consideração o fato de que os objetos têm tamanhos diferentes. No entanto, não há nenhum estudo que faça uma avaliação do efeito da variabilidade destes tamanhos no desempenho dos caches. Outro aspecto a ser observado é a característica estática das políticas, que não oferecem nenhum parâmetro que possa ser ajustado em função das características da carga, com o objetivo de melhorar o desempenho.

A importância da implementação de caches para a escalabilidade da WWW, aliada aos novos parâmetros característicos do ambiente, abrem uma ampla área de pesquisa cujos resultados tem possibilidade de aplicação imediata.

1.2 O Problema

A área de pesquisa deste trabalho é a gerência de espaço em caches da World Wide Web. A gerência de espaço de caches trata de duas questões: a organização do espaço disponível e a política de substituição. Dado que o espaço do cache é finito, não há como armazenar indefinidamente todos os objetos lidos. Em algum momento será necessário retirar um ou mais objetos para armazenar novos objetos que não estão presentes no cache. A política de substituição decide qual ou quais objetos serão retirados. O modelo de organização do espaço define onde um objeto será armazenado, podendo se referir a um espaço único ou a um espaço distribuído.

Modelos conhecidos para gerência de espaço em Web-caches aplicam uma política global para escolha dos objetos a serem retirados do cache. Arquivos de todos os tamanhos são comparados entre si e classificados utilizando os critérios da política. Não há nenhuma relação entre os tamanhos dos objetos substituídos. Este modelo não funciona bem quando há grande variabilidade nos tamanhos e um grande número de objetos para armazenar, como é o caso da WWW.

Esta tese analisa as implicações da variabilidade observada nos tamanhos dos objetos no desempenho dos Web-caches. Dois problemas são analisados e solucionados:

- Qual é o impacto da grande variabilidade nos tamanhos dos objetos armazenados em sistemas de cache de larga escala, como é o caso da WWW, no desempenho destes sistemas?

- Como deve ser a organização do espaço nestes caches para minimizar os efeitos da variabilidade?

Este trabalho trata de caches de objetos estáticos transmitidos através do protocolo HTTP, tais como arquivos HTML, arquivos de texto, imagens, áudio e vídeo. Este trabalho não inclui cache de objetos "contínuos" como os gerados por servidores de vídeo sob demanda, nem proposta de cache de objetos dinâmicos. Este trabalho também não aborda a questão da consistência dos objetos armazenados no cache.

1.3 Contribuições da Tese

As principais contribuições desta tese são:

- Um modelo de organização do espaço de caches da Web que leva em consideração características fundamentais da WWW: larga escala e grande variabilidade. A utilização deste modelo, denominado PART, gera ganhos relevantes de desempenho para seus sistemas de cache.

- Caracterização detalhada de dez cargas reais da WWW, que mostra as dimensões da larga escala e da variabilidade na WWW, e suas implicações para seus sistemas de cache.

- Um teorema que estabelece a relação matemática entre as métricas *hit ratio* e *byte hit ratio*. A partir desta relação é possível construir mapas de desempenho para os caches onde as duas métricas são plotadas no mesmo gráfico. Estes mapas orientam a pesquisa por melhor desempenho em sistemas de cache.

- Discussão sobre o modelo PART apresentando suas características, vantagens, desvantagens e avaliação de desempenho do modelo proposto baseada em simulação. Esta avaliação é validada pelo teorema referido acima.

- Uma estratégia para provar a existência de compromisso entre as métricas *hit ratio* e *byte hit ratio* para uma determinada carga e mostrar o espaço de soluções que o modelo PART dispõe. Essa estratégia analítica é baseada no problema de otimização conhecido como problema da mochila.

- Revisão bibliográfica que mostra o estado da arte em gerência de espaço em caches WWW.

A elaboração desta tese gerou diversas publicações. Os trabalhos começaram com estudos sobre desempenho de servidores WWW (Almeida *et al.*, 1996a), (Almeida *et al.*, 1996c) e (Murta *et al.*, 1996). A caracterização de alguns aspectos da carga de Web é tratada em (Almeida *et al.*, 1998), (Murta e Almeida, 1998) e (Murta e Almeida, 1999). O modelo PART, proposto nesta tese, é tema dos artigos (Murta *et al.*, 1999), (Murta *et al.*, 1998b), (Murta *et al.*, 1998a), que abordam também a questão do impacto da variabilidade dos tamanhos dos objetos em caches da WWW. A variabilidade dos tamanhos dos objetos WWW também foi estudada no contexto dos servidores Web e os resultados foram publicados em (Crovella *et al.*, 1998a), (Harchol-Balter *et al.*, 1998) e (Harchol-Balter *et al.*, 1999). Estes artigos tratam de políticas de assinalamento de tarefas em servidores Web distribuídos. O desempenho de hierarquias de servidores cache foi tratado em (Meira Jr. *et al.*, 1998a) e (Meira Jr. *et al.*, 1998b).

1.4 Organização da Tese

Esta tese está organizada em seis capítulos. O capítulo 2 está dividido em duas partes. A primeira parte conceitua e descreve as distribuições de cauda pesada, essenciais ao entendimento do trabalho. A segunda parte apresenta uma revisão da literatura relacionada ao tema desta tese. São apresentados os modelos de gerência de espaço conhecidos e as políticas de substituição. As métricas de avaliação de desempenho são descritas e discutidas. Esse capítulo mostra também como esta tese é derivada a partir de trabalhos anteriores. O capítulo 3 apresenta uma detalhada caracterização das cargas utilizadas no estudo de desempenho do modelo com ênfase nos aspectos considerados importantes para o modelo proposto. O capítulo 4 apresenta o modelo proposto para a organização do espaço de armazenamento dos caches e discute a relação entre as métricas *hit ratio* e *byte hit ratio* considerando a variabilidade dos tamanhos. O capítulo 5 descreve as simulações feitas

para a análise de desempenho do modelo e discute os resultados. O capítulo 6 apresenta as conclusões do trabalho e sugere direções para trabalhos futuros.

Capítulo 2

Conceitos e Trabalhos Relacionados

Este capítulo apresenta os conceitos básicos necessários ao entendimento desta tese, descreve os trabalhos relacionados e mostra o estado da arte em gerência de espaço de caches da World Wide Web.

2.1 Variabilidade e Distribuições de Cauda Pesada

Grande variabilidade é uma característica que já foi detectada em vários parâmetros da carga Web. Esta seção descreve o conceito estatístico de variabilidade e apresenta exemplos de ocorrência desta característica na WWW.

2.1.1 Conceito Estatístico de Variabilidade

Uma variável randômica X segue uma distribuição de cauda pesada (*heavy-tailed distribution*) se

$$\overline{F}(x) \;=\; P\{X > x\} \;\sim\; x^{-\alpha}, \qquad 0 < \alpha < 2.$$

A distribuição de cauda pesada mais utilizada para modelar as características do ambiente Web é a de Pareto (Jain, 1991), cuja função de densidade de probabilidade é

$$f(x) = \alpha\, k^{\alpha} x^{-\alpha-1}, \quad \alpha,\ k > 0, \quad x \geq k,$$

e a função de distribuição cumulativa é

$$F(x) = P\{X \leq x\} = 1 - (k/x)^{\alpha}.$$

As distribuições de cauda pesada apresentam propriedades que são qualitativamente diferentes das distribuições identificadas mais comumente, tais como a exponencial e a normal. A função de densidade de probabilidade de uma distribuição de cauda pesada apresenta uma cauda que prolonga-se à direita. Isto significa que o valor da moda é menor que o da mediana que, por sua vez, é menor que a média. Distribuições de cauda pesada são

caracterizadas por uma variabilidade extrema. A variância é infinita e, se $\alpha \leq 1$, a média é infinita. Obviamente os momentos só são infinitos se a cauda estende-se infinitamente à direita. A variabilidade cresce muito à medida que α decresce. A figura 2.1 mostra as curvas da função de distribuição cumulativa para as distribuições exponencial e Pareto. A distribuição exponencial é considerada uma distribuição de cauda leve.

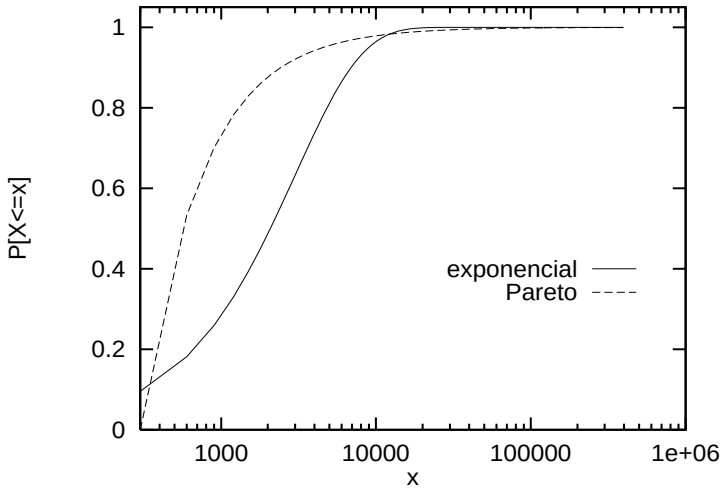

Figura 2.1: Função cumulativa para as distribuições exponencial e de Pareto.

A análise de alguns números pode ajudar no entendimento do significado de uma distribuição com extrema variabilidade. A tabela 2.1 apresenta probabilidades para instâncias das distribuições exponencial e de Pareto com a mesma média (3000), e com parâmetro $\alpha = 1.1$ para a distribuição de Pareto. A distribuição exponencial apresenta pequena variabilidade (cauda leve) e é tradicionalmente utilizada para modelar características de tráfego em modelos de filas (Kleinrock, 1975), (Menascé *et al.*, 1994). Pode ser observado que os valores modelados por uma distribuição exponencial concentram-se mais em torno da média, com pequena probabilidade nos extremos. O mesmo não ocorre na distribuição de cauda pesada: 53% dos valores são pequenos (menores que 600). Além disto, a probabilidade de ocorrer uma observação maior que 50000 é 360000 vezes maior na distribuição de Pareto em relação ao caso exponencial. Há, portanto, uma probabilidade não desprezível de ocorrência de números muito grandes, o que caracteriza a cauda pesada.

As caudas das distribuições mais comuns, por exemplo, Normal, Exponencial, Poisson e Gamma, declinam exponencialmente. No entanto, a cauda das distribuições de cauda pesada declinam com uma função de potência, isto é, muito mais lentamente. Uma forma de visualizar a cauda de uma distribuição é plotar a função $\overline{F}(x)$ num gráfico log-log. A figura 2.2 mostra este gráfico para as distribuições exponencial e Pareto.

$$\log(\overline{F}(x)) \quad \sim \quad -\alpha * log(x)$$

A inclinação $-\alpha$ da reta indica o grau de variabilidade da distribuição.

x	Exponencial	Pareto
300	0.90	1.00
600	0.82	0.47
1500	0.61	0.17
3000	0.37	0.079
6000	0.135	0.037
10000	0.036	0.021
50000	10^{-8}	0.0036
100000	10^{-17}	0.0017

Tabela 2.1: $P\{X > x\}$ para as distribuições exponencial e de Pareto com média 3000. Parâmetros da distribuição de Pareto: $\alpha = 1.1$ e $k = 300$.

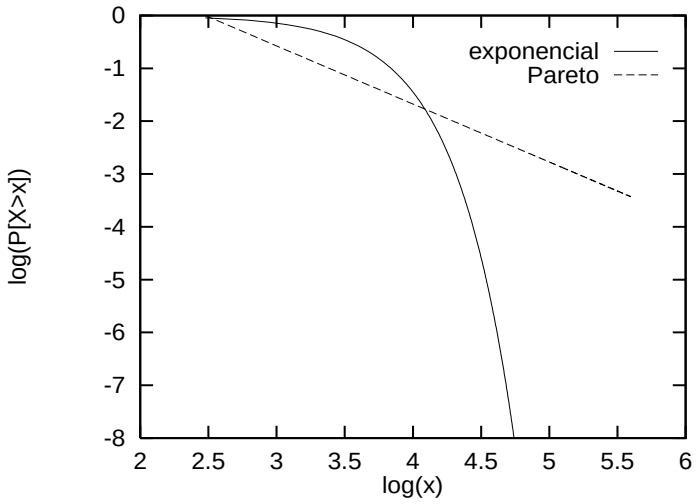

Figura 2.2: Visualização da cauda para as distribuições exponencial e Pareto.

Na prática, momentos infinitos significam uma não convergência dos valores estatísticos. Consequentemente, os sistemas que trabalham com variáveis de cauda pesada podem demorar muito para entrar num regime de estabilidade (Crovella e Lipsky, 1997). O número de pontos necessário para obtenção de uma dada precisão nas medidas é muito sensível ao valor de α (Greiner *et al.*, 1995). Um dos motivos da não observação de distribuições de cauda pesada no passado é o número pequeno de observações colhido para caracterização de dados. Num conjunto pequeno, eventuais observações muito grandes são atribuídas ao acaso e desprezadas.

2.1.2 Variabilidade na WWW

De particular interesse para este trabalho é o fato de que muitas grandezas medidas na Web podem ser modeladas por distribuições de cauda pesada. Isto significa que as propriedades estatísticas da WWW estão caracterizadas pela grande variabilidade. Medições apresen-

tadas em (Crovella e Bestavros, 1996) mostram que o tamanho dos arquivos disponíveis nos servidores, dos arquivos requisitados pelos usuários e dos arquivos transmitidos segue uma distribuição de cauda pesada. O mesmo trabalho mostra que o conjunto dos tempos de duração das transmissões dos arquivos pedidos segue a mesma distribuição como conseqüência do tamanho dos arquivos transmitidos. As taxas de chegadas de requisições a servidores Web também experimentam grande variabilidade (Greiner *et al.*, 1995).

Em (Murta e Almeida, 1998), os autores mostram que a grande variabilidade observada nos tempos de resposta às requisições feitas a caches de *proxy* pode ser explicada pela combinação de vários fatores, incluindo o tamanho da resposta, a variabilidade de *bandwidth* das conexões dos clientes com o cache e o custo inicial (*overhead*) do protocolo TCP para cada conexão. Este custo é amortizado somente para respostas maiores, que requerem mais pacotes para sua transmissão. Os resultados deste trabalho mostram que os tempos de resposta apresentam variabilidade maior do que os tamanhos. Por exemplo, os tempos de resposta podem variar em até quatro ordens de grandeza nas transmissões de mesmo tamanho. Isto comprova que a variabilidade no tamanho explica apenas parcialmente a variabilidade na latência. Este comportamento foi observado tanto para as latências das requisições que são *misses* no cache quanto para os *hits*.

Os tamanhos dos arquivos na Web são modelados por distribuições de cauda pesada. Isto significa que 50% dos arquivos tem tamanho pequeno, menor que a média. Adicionalmente, uma porcentagem pequena dos arquivos tem tamanho muito grande, algumas ordens de grandeza maiores do que a média, e esta pequena porcentagem de arquivos representa a maior parte dos bytes. Para efeito de comparação considere uma distribuição de cauda pesada cujas observações estejam entre 1 e 10^8, valores extremos comuns para o número de bytes dos arquivos na Web. Se esta distribuição representasse, por exemplo, os tempos de duração de ligações telefônicas, em segundos, então teríamos ligações com duração entre 1 segundo e 3.2 anos. Se esta distribuição representasse distância em metros, então teríamos valores entre um metro e cem mil quilômetros, o que equivale a duas vezes e meia a circunferência do Equador. Estas comparações mostram a extrema variabilidade das observações utilizando as grandezas tempo e distância.

Conforme dito anteriormente, a distribuição mais utilizada para modelar a variabilidade da WWW é a de Pareto. No entanto, os momentos desta distribuição não são todos finitos. Para melhor representar um conjunto finito de observações, a distribuição de Pareto limitada (*Bounded Pareto*) (Harchol-Balter *et al.*, 1998) é utilizada. Os valores desta distribuição continuam seguindo uma lei de potência mas têm um limite superior. A distribuição de Pareto limitada é composta por três parâmetros: o expoente α, que caracteriza a lei de potência, k, a menor observação possível, e p, a maior observação possível. A função de probabilidade de massa para esta distribuição, $B(\alpha, k, p)$ é definida por:

$$f(x) = \frac{\alpha k^\alpha}{1 - (\frac{k}{p})^\alpha} x^{-\alpha-1} \quad k \leq x \leq p.$$

Esta distribuição tem todos os seus momentos finitos. Portanto não é uma distribuição de cauda pesada no sentido definido anteriormente. No entanto, apresenta grande varia-

bilidade se $k \ll p$. A distribuição é muito sensível aos valores assumidos por α. Quanto menores os valores de α, maior é a variabilidade. Por exemplo, o segundo momento, $E(X^2)$, cresce exponencialmente (de 10^8 para $\alpha = 2$ para 10^{13} para $\alpha = 0.2$) à medida que α decresce (Harchol-Balter *et al.*, 1998). Valores típicos de α medidos na WWW estão entre 0.8 e 1.3 (Crovella e Bestavros, 1996) (Crovella *et al.*, 1998b).

2.2 Caches na World Wide Web

2.2.1 A World Wide Web

O conceito fundamental da WWW é o de hipertexto, proposto em 1945 por Vannevar Bush (Bush, 1945). Um hipertexto é composto por unidades básicas denominadas documentos. Cada documento pode permitir o acesso a outros documentos implementando um *link* para os documentos referenciados, que é distintamente identificado através de uma âncora. Documentos podem ser compostos por texto, imagens, áudio e vídeo. A facilidade de adicionar informação, ligando-a ao hipertexto global da mesma forma que se escreve uma nota marginal num texto em papel, é um atrativo essencial. As conexões entre os documentos formam o grande sistema hipermídia global que é a WWW. Os termos "hipertexto" e "hipermídia" foram propostos por Ted Nelson em 1965 (Nelson, 1965).

No entanto, apesar da idéia ser conhecida já há algumas décadas, sua implementação é bem recente e deve-se a três tecnologias descritas em (Berners-Lee *et al.*, 1994), a saber: o sistema de endereçamento, o protocolo de comunicação específico para hipertextos e a linguagem de criação de documentos Web. O sistema de endereçamento, denominado URL (*Universal Resource Locators*), permite o acesso a objetos em todos os possíveis endereços de rede que utilizam protocolos que implementam o conceito de objeto. A rede Internet está organizada por uma hierarquia de protocolos. O protocolo da WWW é o HTTP (*HyperText Transfer Protocol*) que permite reconhecer e transferir dados na forma de hipertexto. O protocolo HTTP opera na forma requisição-resposta sobre uma conexão TCP/IP e é utilizado para comunicação entre clientes e servidores. FTP, NNTP, Gopher, WAIS e telnet são exemplos de outros protocolos Internet. A WWW encapsula estas aplicações, eliminando a necessidade de um *browser* para cada tipo de serviço. A linguagem para construção de documentos na forma de hipertexto é denominada HTML (*HyperText Markup Language*).

A WWW é um sistema distribuído cuja comunicação segue o modelo cliente-servidor e a rede de comunicação é a Internet. Este modelo é apresentado na figura 2.3. A Internet, por sua vez, é composta por um conjunto de redes de computadores que se comunicam e cooperam entre si. As redes que compõem a Internet utilizam diversos tipos de tecnologia, protocolos, meios de transmissão e computadores. A WWW é considerada uma aplicação da Internet.

Um cliente Web é um software que permite a requisição de documentos no formato Web, também conhecidos como páginas. Cada página pode ser composta por vários arquivos contendo texto, imagens, animação, áudio, vídeo e código executável. Portanto, um pedido

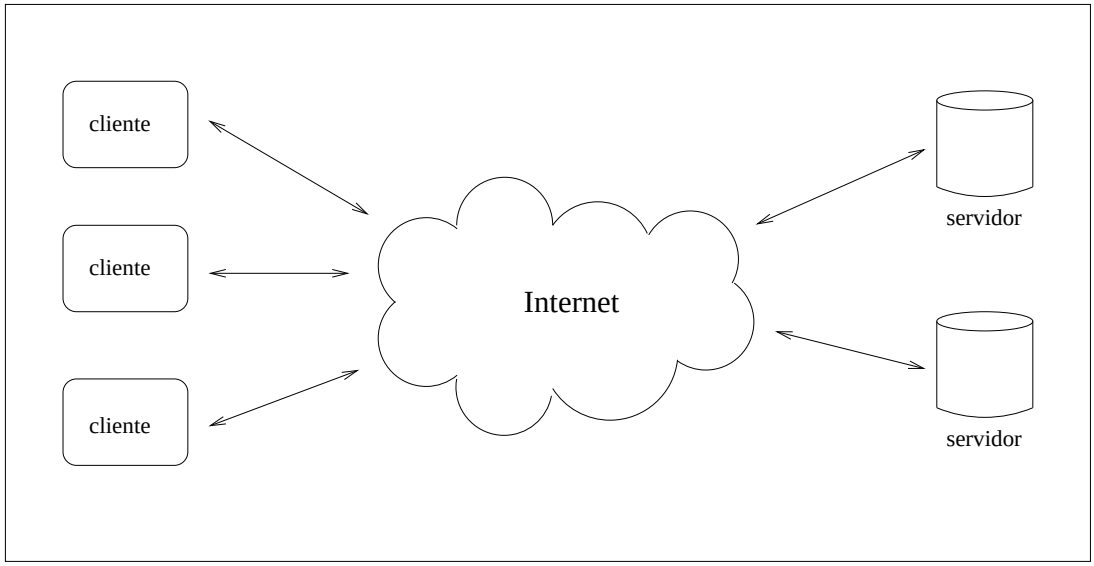

Figura 2.3: A WWW segue o modelo cliente-servidor e a Internet é o meio de comunicação.

do cliente (requisição de uma página) pode gerar vários pedidos ao servidor, um para cada arquivo que compõe a página pedida[1]. Cada arquivo tem um endereço único global, o URL, válido para toda a Internet. As páginas WWW são visualizadas através de um software próprio denominado *browser*. Um usuário da WWW precisa utilizar um *browser* para requisitar e visualizar páginas. O termo "cliente" pode ser estendido para se referir ao computador no qual o *browser* é executado, bem como ao usuário que faz uma requisição.

Um servidor Web é um sistema composto por pelo menos um computador, um software específico e a informação a ser servida. Seguindo o modelo cliente-servidor, o cliente sempre inicia uma interação pedindo ao servidor uma página. O endereço do servidor (*host*) bem como a página pedida são especificados no URL. O software servidor fica em execução contínua, esperando as requisições dos usuários. Ao receber uma requisição o servidor identifica o arquivo requisitado, procura o arquivo no seu espaço de armazenamento e o envia para o *browser* pela mesma conexão. Uma descrição das etapas desta interação pode ser encontrada em (Yeager e McGrath, 1996). O modelo WWW pode ser implementado utilizando uma rede local (LAN) e com acesso restrito a uma comunidade. Este modelo é denominado Intranet.

2.2.2 Funcionamento dos Caches

Os sistemas de cache da WWW têm o mesmo princípio de funcionamento do cache tradicional (Patterson e Hennessy, 1996). Os documentos requisitados são copiados em posições mais próximas do usuário com o intuito de diminuir o tempo do próximo acesso ao documento. Os caches podem ser implementados nos vários componentes do sistema e podem

[1]Neste trabalho os termos "documento", "objeto" e "arquivo" são usados como sinônimos e significam o conteúdo de um URL requisitado, unidade indivisível tratada pelo HTTP.

ser classificados de acordo com a sua localização: cache de cliente, implementado no *browser*, cache de servidor, implementado no próprio servidor, e cache de rede, implementado em pontos estratégicos da rede, entre os demais caches, geralmente em pontos mais próximos dos clientes. O objetivo do cache de rede é atender a um grande conjunto de clientes que utilizam a mesma saída (*backbone*) para a Internet. A figura 2.4 mostra os tipos de cache que podem ser encontrados no caminho de uma requisição, do cliente ao servidor.

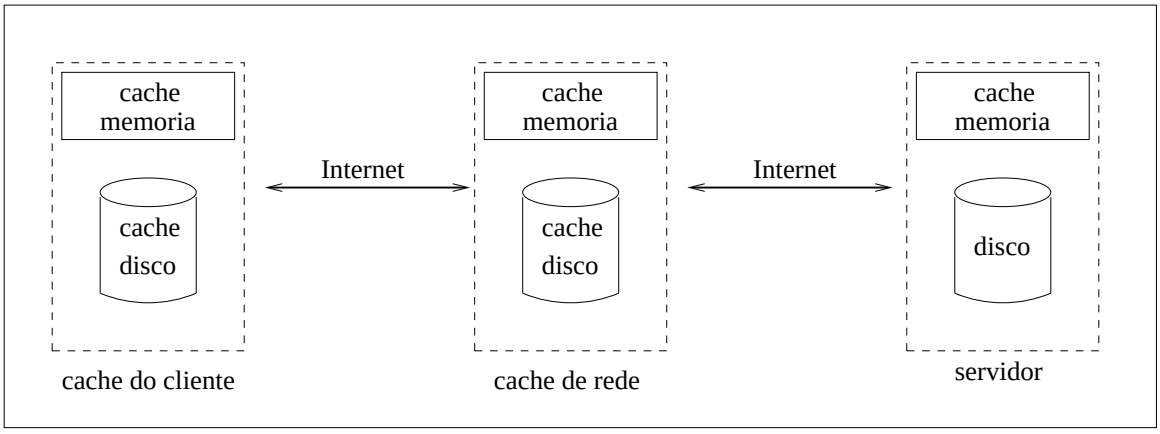

Figura 2.4: Tipos de cache encontrados no caminho de uma requisição.

Os sistemas de cache de cliente e de rede operam como cliente e servidor. O cache recebe requisições de páginas endereçadas a um servidor e verifica se uma cópia da página pedida está presente em seu espaço de armazenamento. Caso afirmativo, o cache envia a página respondendo o pedido do cliente (como faz um servidor). No entanto, se a página não está presente, o cache atua como cliente, enviando um pedido da página ao servidor original. Ao ser atendido, o cache responderá ao cliente servindo a página requisitada e guardará uma cópia da mesma.

Existem dois tipos de cache de rede: os caches explícitos e os transparentes. A utilização de caches explícitos requer que o *browser* do cliente seja configurado para redirecionar todas as requisições HTTP para o cache, ignorando as localizações dadas pelos URLs. Caches explícitos podem ser implementados em qualquer lugar da rede. Localizações típicas são em servidores *proxy* localizados na conexão de uma empresa com a Internet e em *links* de provedores de acesso com a Internet. O cache transparente é utilizado sem nenhuma alteração no cliente. Neste tipo de cache, o roteador é configurado para espiar e redirecionar para o cache todos os pacotes destinados ao porto 80 (tráfego HTTP). A localização dos caches transparentes deve ser cuidadosamente escolhida. Em geral são implementados na conexão de várias subredes com um *backbone* da Internet.

Os três caches mostrados na figura 2.4 experimentam cargas de trabalho diferentes. O cache de cliente mantém cópias do conteúdo dos URLs recentemente exibidos pelo *browser*. Sua vantagem é a de praticamente eliminar o tempo de busca do documento. No entanto, serve apenas um ou poucos usuários e o conteúdo frequentemente fica desatualizado antes de ser reutilizado. O cache de servidor, implementado em memória principal, contém

documentos armazenados no próprio servidor e que foram requisitados pelo menos uma vez. Neste caso, o conjunto de documentos que pode ser armazenado é limitado pelo conjunto de documentos que o servidor dispõe, e o conjunto de usuários deste cache pode se estender ao conjunto de usuários da WWW. O cache de rede pode armazenar documentos residentes em qualquer servidor Web (o mesmo acontece com o cache de cliente) e pode receber requisições de um número grande de usuários, dependendo de sua localização.

A utilização de caches é limitada aos documentos estáticos, por exemplo, páginas Web (arquivos com terminação .html, .htm), páginas de texto (.txt, .ps, .pdf), arquivos de imagem (.gif, .jpeg, .tiff) e arquivos de áudio (.au, .wav) e vídeo (.mpeg, .mp2). Um documento estático é um documento atualizado apenas ocasionalmente. Sua URL determina completamente o que será exibido. Documentos dinâmicos são documentos atualizados a cada requisição, por exemplo, uma consulta a um banco de dados ou uma visualização de imagem em tempo real. Na prática são arquivos identificados pelas terminações .pl, .cgi, .count e também por URLs contendo os strings "?" ou "cgi-bin" na identificação. Estes documentos geram páginas em função de parâmetros próprios definidos em cada requisição. Por isso as páginas não são armazenadas nos caches. Em passado recente acreditava-se que a maioria dos documentos seria dinâmica, o que minimizaria a efetividade dos caches. No entanto, ainda apenas uma fração pequena das requisições corresponde a documentos dinâmicos (Manley e Seltzer, 1997), (Abdulla, 1998), (NLANR, 1999).

Um problema gerado pela utilização de caches é a consistência dos documentos copiados no cache em relação ao documento original. O ideal seria que cada documento original servido fosse acompanhado por sua data de expiração. Assim os caches poderiam avaliar a validade do documento e fazer nova requisição ao servidor caso a cópia esteja inválida no momento de ser servida. No entanto, a maioria dos servidores não fornece esta informação e o cache atribui a seu critério um tempo de vida (*time to live*, TTL) para cada cópia recebida. Se o objeto é requisitado antes do TTL expirar, a cópia do cache é enviada. Caso contrário uma nova cópia é recuperada do servidor. O TTL é fixado considerando o compromisso entre a atualização do documento exigida pelo cache e a carga gerada pelas mensagens de verificação da validade.

O uso de sistemas de cache na Web traz vários benefícios:

- Economia no consumo de *bandwidth* devido à diminuição do tráfego na rede: um documento é enviado uma vez do servidor para o cache e então requisições sucessivas para este documento são servidas a partir do cache. Um documento encontrado no cache significa que a requisição e a resposta vão percorrer o caminho entre o cliente e o cache, isto é, vão trafegar em pequenos trechos dos *backbones* da Internet ou podem mesmo não utilizá-los. Assim, estes caches eliminam o tráfego redundante na rede. A diminuição do tráfego contribui para diminuir a ocorrência de congestionamentos na rede.

- Diminuição da latência: quando ocorre um *hit*, os usuários podem observar uma redução no tempo decorrido entre o pedido da página e a exibição da resposta. Esta redução ocorre porque o documento está no cache e, portanto, está em um caminho

intermediário entre o cliente e o servidor, resultando em uma resposta mais rápida. Mesmo em caso de *miss*, o usuário pode ser beneficiado pela utilização generalizada de caches porque seu pedido percorrerá uma rede menos congestionada e poderá encontrar um servidor menos sobrecarregado, o que contribuirá para que a resposta seja mais rápida. Isto mostra que os benefícios tem efeitos interligados.

- Redução na carga dos servidores: cada *hit* significa uma requisição que deixa de ser enviada a um servidor. Isto beneficia o desempenho dos próprios servidores com a redução da carga, especialmente com a redução nos picos de demanda. Com a utilização de caches de rede, as requisições do usuário podem ser respondidas localmente.

- Aumento da disponibilidade do documento: se o servidor fonte do documento está indisponível ou se há algum problema na rede que impede a conexão entre cliente e servidor, o cliente pode acessar a cópia no cache mesmo se esta não está atualizada. Em geral podemos assumir que uma cópia desatualizada é melhor do que nenhuma cópia.

Três desvantagens são apontadas na utilização de caches: o cliente pode receber um documento desatualizado; o servidor fonte recebe menos *hits* e fica impossibilitado de ter uma estimativa real do número de acessos às suas páginas; o cache de rede pode se tornar um ponto de estrangulamento, com consequente degradação no desempenho.

2.2.3 Medidas de Desempenho

Apesar da semelhança funcional, os sistemas de cache tradicional e de cache na Web apresentam diferenças importantes. Nos esquemas tradicionais de cache os dados são movidos em blocos do mesmo tamanho. Portanto, o número de bytes encontrados no cache em relação ao número de bytes requisitados é exatamente a fração de *hits*. O mesmo não ocorre nos caches da Web porque os documentos são transmitidos e armazenados na sua forma integral; não há o conceito de bloco. Como conseqüência, precisamos trabalhar com duas métricas para medir o desempenho dos sistemas de cache na Web: a fração das requisições atendidas pelo cache, (*Hit Ratio*, HR) e a fração dos bytes requisitados que é atendida pelo cache, (*Byte Hit Ratio*, BHR). Formalmente,

$$HR = \frac{\text{número de requisições atendidas pelo cache}}{\text{número de requisições feitas ao cache}}$$

e

$$BHR = \frac{\text{número de bytes atendidos pelo cache}}{\text{número de bytes requisitados ao cache}}.$$

A otimização de cada métrica, separadamente, atende à primeira vista a interesses diferentes. Usuários Web querem ter respostas rápidas às suas requisições, por isso são atraídos por uma elevada fração de *hits*. Administradores de servidores Web, provedores de acesso e empresas mantenedoras dos *backbones* da Internet querem economia na utilização

da rede, portanto estão interessados em aumentar a taxa de acertos por byte, reduzindo o volume de tráfego na rede. No entanto, a otimização das duas métricas é interessante para ambos pois a redução do número de requisições aos servidores (maior HR) também reduz o volume do tráfego além de diminuir a carga no servidor destino da conexão. Por outro lado, o descongestionamento da rede (maior BHR) influencia fortemente o tempo de resposta, uma vez que as requisições não presentes nos caches poderão ser respondidas mais rapidamente do que quando há sobrecarga na rede. Portanto, o aumento de ambas as métricas, HR e BHR, ajuda a conter o tráfego na rede, evitando gasto desnecessário de recursos (*bandwidth*), além de otimizar o tempo de resposta.

Alguns fatores contribuem para o aumento de ambos, HR e BHR, incluindo o tamanho do cache, a habilidade da política de substituição em manter no cache os objetos que serão requisitados ou mesmo uma carga particular. No entanto, a maximização simultânea das duas frações, HR e BHR, só é possível em caches infinitos, isto é, caches com espaço suficiente para armazenar o conteúdo de todos os URLs pedidos. Quando o espaço é limitado, revela-se um compromisso entre as duas métricas decorrente da associação entre a característica de grande variabilidade nos tamanhos dos documentos e a freqüência de acesso aos documentos, também conhecida como popularidade. Alguns estudos (Almeida *et al.*, 1996b), (Cunha *et al.*, 1995), (Rizzo e Vicisano, 1998) mostram que a popularidade é inversamente correlacionada com o tamanho. Em outras palavras, os documentos menores são os mais requisitados. Para otimizar HR, por exemplo, basta armazenar os arquivos menores, que são os mais populares. Além disto, o armazenamento preferencial de arquivos pequenos permite manter um número maior de arquivos no cache. Os arquivos pequenos são responsáveis por um número grande de *hits* mas constituem uma fração pequena dos bytes servidos. Por isso, a otimização do HR implica num aumento do número de *misses* por byte pedido. Por outro lado, os arquivos maiores são responsáveis por um número pequeno de *hits* que constituem uma grande fração dos bytes requisitados. O aumento do BHR pode levar a uma diminuição do HR porque implica no armazenamento de documentos maiores, que podem ocupar o lugar de vários documentos pequenos, porém mais populares.

A tabela 2.2 mostra dados que revelam a regra 80/20, nome dado em referência à seguinte proporção encontrada nas requisições WWW: 80% das requisições corresponde a aproximadamente 20% dos bytes requisitados e vice-versa, isto é, 80% dos bytes requisitados corresponde a aproximadamente 20% das requisições. Esta tabela foi construída da seguinte forma. Os arquivos transmitidos em resposta às requisições são ordenados por ordem crescente de tamanho. Esta seqüência ordenada é dividida em três grupos: o primeiro grupo é composto por 80% das requisições, que corresponde a primeira parte da seqüência, a parte que contém as menores respostas. O segundo grupo é composto pelos 10% seguintes, que contêm respostas de tamanho médio. O terceiro grupo é composto pelos 10% restantes, que contêm as maiores respostas. Verificou-se que o primeiro grupo, que contém 80% das requisições, responde por aproximadamente 20% dos bytes transmitidos. A parcela de 80% dos bytes transmitidos está distribída nos dois grupos restantes, compostos por apenas 20% das requisições. As cargas referidas nesta tabela são descritas no capítulo 3.

Um princípio importante em projetos de sistemas de computação é otimizar o caso

carga	80%	10%	10%
BL	21	11	68
BR	20	17	63
POP98	22	15	63
U	20	9	71

Tabela 2.2: Regra 80/20: 80% das requisições corresponde a aproximadamente 20% dos bytes requisitados (coluna 2) e apenas 10% dos arquivos são responsáveis por mais de 60% dos bytes requisitados.

mais comum (Patterson e Hennessy, 1996). No entanto, o cenário apresentado mostra dois casos comuns, número de *hits* e número de *hits* por byte, cuja otimização é contraditória, isto é, a utilização de técnicas para otimizar uma métrica não garante o melhoramento do sistema em relação à outra métrica e, pelo contrário, pode-se observar uma deterioração (Arlitt e Williamson, 1996).

Tempo de resposta é também uma métrica importante do desempenho dos caches. O tempo de resposta é dado por

$$\text{Tempo de resposta} = \text{tempo de hit} + (1 - \text{HR}) * \text{tempo de miss}$$

e sofre influência da variabilidade nos tamanhos das requisições. No caso do cache tradicional, a influência do tamanho no tempo de *miss*[2], isto é, o tempo para buscar um bloco ou documento no próximo nível, é constante (já que os blocos têm o mesmo tamanho) e o tempo de acesso, medido em ciclos de clock, pode ser conhecido com precisão[3]. No ambiente Web, onde as escalas de tempo são muito maiores, o tempo de *miss* é função do tamanho da requisição, do *bandwidth* entre o cliente e o servidor e das condições de carga na rede e no servidor no momento da busca. Como já foi mencionado, o tamanho do documento requisitado e os tempos de transmissão dos documentos seguem uma distribuição de cauda pesada.

Do ponto de vista do usuário, a melhor medida de desempenho é tempo de resposta. HR e BHR são medidas indiretas de desempenho. No entanto, o tempo de resposta é completamente dependente do ambiente de execução, da localização relativa entre cliente e servidor, da carga na rede e no servidor no momento da requisição. Como conseqüência, valores de tempos de resposta medidos representam o desempenho apenas do ambiente do experimento, com poucas possibilidades de generalização.

2.2.4 Algoritmos para Gerência de Espaço

O sucesso dos sistemas de cache está no fato de que a distribuição dos acessos aos arquivos é de tal maneira tendenciosa que a maioria das requisições é feita para um subconjunto pequeno do conjunto de documentos. Portanto, o cache deve armazenar este pequeno conjunto, conhecido como *working set* (Denning e Schwartz, 1972) ou localidade (Madison

[2]Denominado *miss penalty* no contexto de cache de memória

[3]Por exemplo, ver tabela 5.37 em (Patterson e Hennessy, 1996)

e Batson, 1976), e ignorar os demais documentos. Diz-se que a carga de um cache tem "boa localidade de referência" se ela referencia pequenos subconjuntos do seu conjunto de documentos distintos por longos períodos de tempo. Esta característica dos acessos é responsável pela viabilidade dos caches em sistemas de computadores.

A gerência de espaço de caches trata de duas questões: a organização do espaço disponível e a política de substituição (ou reposição), que define quais arquivos devem ser retirados para que um seja armazenado. Atualmente são conhecidas duas abordagens para gerenciamento de Web-caches. A primeira consiste em ter grandes quantidades de disco, de forma que praticamente nenhum documento precise ser retirado (Inktomi, 1998), (Infolibria, 1998), (Cao, 1997). Pesquisadores desta corrente afirmam que pode-se ter espaço em disco suficiente para armazenar toda a WWW. Como lembrou Peter Denning em (Denning, 1997), e a história se repete com a WWW, de tempos em tempos várias pessoas argumentam que a tecnologia de memória permitirá ter toda a memória desejada no nível inferior (disco local, no caso da WWW) para armazenar tudo que está no nível superior (disco remoto acessado via Internet). No entanto, as previsões nunca se confirmaram e há pouca razão para acreditar que elas se confirmem dessa vez.

No caso da WWW em particular, a idéia de armazenar em caches de rede todos os documentos acessados não é razoável, mesmo que se disponha de espaço suficiente para armazenar toda a WWW, uma vez que o número de arquivos acessados cresce com o tempo e com o aumento do número de clientes. Além disto, a fração de arquivos úteis para o cache, isto é, que serão acessados mais de uma vez, é pequena em relação ao total de documentos acessados. Então uma grande parte do espaço do cache seria dedicada ao armazenamento de arquivos inválidos (removidos ou atualizados na fonte), ou arquivos que não seriam requisitados novamente no futuro. Portanto, para evitar desperdício de recursos, é necessário adotar uma política de remoção de arquivos. Esta é a segunda abordagem, que é a adotada neste trabalho, encontrar o melhor desempenho com o menor custo possível.

O objetivo da política de reposição é eliminar do cache objetos que não serão requisitados no futuro ou que serão requisitados no futuro mais distante, mantendo no cache os objetos que serão requisitados em futuro próximo. A política de substituição mais conhecida é a LRU (*Least Recently Used*). Esta política privilegia as referências mais recentes, protegendo-as da substituição. Assim, ela explora o princípio da localidade temporal que diz que documentos recentemente acessados tem maior probabilidade de serem requisitados em futuro próximo. A localidade temporal foi observada em seqüências de acesso a código e dados de programas. Neste ambiente de cache tradicional a reposição de blocos fica a cargo do algoritmo LRU por ser baseado no princípio da localidade (Patterson e Hennessy, 1996), (Tanenbaum, 1992).

Os algoritmos de reposição de documentos nos caches da Web procuram levar em consideração os parâmetros da carga que afetam o desempenho. Os parâmetros mais comumente reconhecidos são o tamanho dos documentos, a freqüência de acesso, a recentidade dos acessos aos documentos e o tempo para buscar um documento em caso de *miss*. A identificação destes parâmetros indica que a gerência de espaço nos caches da Web é um problema mais

complexo do que a gerência de espaço em caches de memória.

Associada à política de substituição podemos ter uma política de controle de admissão para definir se um documento requisitado não presente no cache vai ser armazenado. Na maioria dos casos, os caches armazenam todos os documentos que buscam. A restrição mais comum é em relação ao tamanho do documento. Um limite superior é estabelecido e documentos cujos tamanhos estão acima deste limite não são armazenados (Arlitt e Williamson, 1996), (Markatos, 1996). Uma política de controle de admissão mais elaborada é adotada no algoritmo PSS (Aggarwal *et al.*, 1996). Neste caso, um objeto acessado pela primeira vez só vai para o cache se houver espaço. Caso contrário, a informação sobre o objeto é armazenada num cache auxiliar que contém apenas os dados dos documentos (URL, número de acessos, etc.) mas não contém o próprio documento. Quando o documento é acessado pela segunda vez torna-se candidato a ir para o cache. A freqüência dinâmica do documento, definida como o número de acessos ao cache entre dois acessos consecutivos ao mesmo documento, é comparada com as freqüências dinâmicas dos documentos candidatos a sair do cache. É feita a escolha que preserva a maior soma das freqüências dinâmicas no cache.

As políticas de reposição de caches na WWW podem ser classificadas de acordo com os parâmetros citados: baseadas em recentidade, baseadas em freqüência, baseadas em tamanho, baseadas em tempo e as híbridas. Os algoritmos mais conhecidos para cache na Web incluem o LRU, que retira os documentos menos recentemente utilizados, LFU (*Least Frequently Used*), que retira os documentos utilizados com menor freqüência, e SIZE (Williams *et al.*, 1996), que retira o maior documento.

Cada um destes três considera apenas um dos parâmetros citados acima e, portanto, falha em certas situações. LRU e LFU, por exemplo, podem perder vários documentos ante à chegada de documentos grandes no cache embora ambos sejam robustos a alterações na correlação entre popularidade e tamanho. LRU permite que o cache seja inundado por documentos referenciados apenas uma vez (a maioria dos documentos), retirando documentos com maior freqüência de acesso. LFU ignora a recentidade dos acessos e pode, portanto, retirar documentos recém-acessados que serão muito requisitados no futuro. Por outro lado, LFU preserva documentos mais referenciados, o que é uma vantagem, porém pode preservar eternamente documentos muito referenciados no passado e que não serão mais referenciados no futuro. Por exemplo, documentos relativos a descobertas científicas, sucessos ou escândalos de interesse mundial, que experimentam picos de acesso num curto período de tempo, poderiam acumular um número grande de acessos impedindo sua retirada do cache mesmo quando a recentidade dos acessos não justifique mais seu armazenamento. SIZE tem a vantagem de reter os documentos menores, responsáveis pelo maior número de *hits*, mas é mais sensível à variações na relação entre tamanho e popularidade, além de reter documentos pequenos que podem não ser mais referenciados. A preservação por tempo indefinido de documentos no cache pelas políticas de parâmetro único (exceto LRU) causa um efeito equivalente à diminuição do tamanho do cache, fenômeno denominado "poluição do cache".

Variações de LRU foram propostas. LRU-MIN (Williams *et al.*, 1996) verifica o tamanho do documento a ser armazenado e procura no cache documentos de tamanho maior ou

igual. Caso existam documentos que atendem a este requisito, o algoritmo LRU é aplicado a este subconjunto. Caso contrário, todos os documentos cujos tamanhos sejam maiores ou iguais à metade do tamanho do documento a ser armazenado são incluídos no subconjunto. A operação é repetida até que o espaço necessário seja liberado. SLRU (*Segmented LRU*) (Karedla *et al.*, 1994) divide o cache em dois segmentos e implementa duas listas LRU, uma protegida, para documentos com mais de um acesso, e outra para documentos com apenas um acesso. Portanto considera freqüência e recentidade. LRU-k (Robinson e Devarakonda, 1990) retém informação sobre as últimas k referências de um objeto durante um tempo determinado mesmo para objetos removidos do cache. Considera freqüência e recentidade. LRU-*threshold* e LRU-*adaptive* (Markatos, 1996) são variações de LRU nas quais há um limite para o tamanho de documento que pode ser armazenado. Na primeira versão este limite é fixo e na segunda é alterado dinamicamente. Na versão dinâmica, a variação correspondente de desempenho indica alterações posteriores. A definição de um limite para o tamanho do documento que pode ser armazenado é uma forma de controle de admissão e pode ser utilizada em qualquer política. O estabelecimento de um limite de tamanho para armazenamento inferior ao tamanho máximo dos documentos requisitados equivale a um corte na cauda da distribuição dos tamanhos cujo efeito é um possível aumento em HR acompanhado de uma diminuição do BHR.

As principais variações de LFU são implementações de mecanismos de contagem de tempo para "envelhecer" as referências. LFU-*aging* e LFU-DA (*Dynamic Aging*) (Arlitt *et al.*, 1999) são alguns exemplos. Alguns trabalhos (Bolot e Hoschka, 1996), (Bolot *et al.*, 1997) defendem o uso do tempo de *miss* como critério para armazenar e retirar documentos do cache. Maiores tempos justificariam a preservação do documento no cache. No entanto, há pelo menos duas objeções a esta consideração. Em primeiro lugar, a retirada de documentos com menor tempo de busca poderia prejudicar os servidores mais rápidos, e favorecer os mais lentos. Seria uma medida tendenciosa e parcial. Em segundo lugar, há muita imprecisão associada ao tempo de busca de um documento. Foi observado para vários servidores que seus tempos de resposta variavam de acordo com o horário do dia (Wagner, 1996). Segundo medições apresentadas em (Cao e Irani, 1997) a latência de um mesmo documento varia consideravelmente, com variância entre 5% e 500% com média 71%. Essa variabilidade no tempo de busca introduz erros e reduz a efetividade dos algoritmos que utilizam esta medida como parâmetro. Por exemplo, (Wooster e Abrams, 1997) propôs um algoritmo que remove documentos com a menor latência. No entanto, o mesmo trabalho mostra que este não é um bom critério. Em sistemas com grande variabilidade, onde picos numa medida impactam negativamente o desempenho, considerar apenas a média para representar os dados pode levar a conclusões errôneas. Portanto, a utilização destes tempos deve ser feita por meio de valores médios acompanhados por medidas de dispersão dos dados. Isto pode dificultar sua utilização.

O mesmo trabalho (Wooster e Abrams, 1997) propõe um algoritmo híbrido HYB que calcula, para cada documento, seu valor para o cache. Este cálculo é feito por uma função que combina os parâmetros tempo de conexão ao servidor do documento, *bandwidth* da conexão, a freqüência e o tamanho do documento. Documentos com menor valor são descartados. O algoritmo apresentou bom desempenho em HR e tempo de resposta frente

a LRU, LFU e SIZE. Suas desvantagens são a necessidade de armazenar mais informações para cada documento e a necessidade de ajustes cuidadosos das constantes utilizadas na função. Na mesma linha está o MIX (Niclausse *et al.*, 1998) que utiliza os parâmetros latência, número de referências, tamanho e tempo decorrido desde a última referência.

Vários outros trabalhos propõem algoritmos híbridos. Em (Rizzo e Vicisano, 1998), os autores propõem o LRV (*Lowest Relative Value*), baseado na recentidade, tamanho e freqüência do documento. Os autores encontram funções matemáticas que são aproximações da distribuição dos tempos entre acessos a um mesmo documento e da probabilidade de mais acessos dado que o documento foi acessado previamente i vezes. Estas funções são baseadas em uma extensa caracterização de duas cargas de caches de rede. O cache acumula, em tempo de execução, estatísticas de cada documento requisitado, atualizadas a cada acesso. Com base nas funções e nos valores das estatísticas, a probabilidade Pr de um documento ser acessado no futuro é calculada. Pr é diferente para cada documento e é dependente do tempo. Documentos com o menor Pr são descartados. O método é dependente da função de aproximação da distribuição dos tempos entre acessos a um mesmo documento que, por sua vez, é baseada nas cargas estudadas. A implementação é cara. A função de aproximação utiliza as funções logarítmica e exponencial e constantes empiricamente escolhidas e é calculada frequentemente para todos os documentos presentes no cache. A principal desvantagem do LRV, além de seu custo proibitivo para implementação, é sua grande parametrização com base em apenas duas cargas, o que deixa dúvidas sobre o bom desempenho deste algoritmo com cargas diferentes.

GreedyDual-Size (Cao e Irani, 1997), abreviado por GD-Size, é também um algoritmo híbrido que considera recentidade dos acessos, tamanho e custo de busca de um documento. Os autores mostram que considerar a latência não leva a bons resultados devido à grande variabilidade da latência de busca de um mesmo documento, o que confirma resultados anteriores. O algoritmo ordena os documentos de acordo com um valor H definido por $H = \text{custo}/\text{tamanho}$. Documentos com o menor valor H são candidatos a sair do cache. A recentidade é considerada da seguinte forma. A cada acesso, o valor de H do documento é calculado e acumulado ao valor residual anterior. Portanto, quanto mais recente o acesso ao documento, maior o seu H e menor sua probabilidade de ser retirado do cache.

A definição de custo do GD-Size depende do objetivo do algoritmo. Várias opções são analisadas porém o melhor desempenho em HR é obtido com custo é igual a 1, GD-Size(1). Uma variação considera o número de pacotes gerados para a transmissão do documento, GD-Size(pacotes), onde $pacotes = 2 + tamanho/536$[4]. GD-Size(pacotes), apresenta melhor desempenho em BHR do que GD-Size(1). Cada opção oferecida é projetada para otimizar apenas uma métrica.

Outra política híbrida, PSS (*Pyramidal Selection Scheme*) (Aggarwal *et al.*, 1996), considera o tamanho do documento e a recentidade do último acesso de cada documento armazenado no cache. PSS implementa controle de admissão utilizando um cache auxiliar que contém apenas informações sobre os documentos. A freqüência dinâmica de um objeto

[4]O cálculo considera 2 pacotes para estabelecer a conexão mais o número de pacotes necessários de acordo com o tamanho do documento, considerando que um segmento TCP transmite 536 bytes de dados

i é definida por $1/\Delta T_{ik}$ onde ΔT_{ik} é o número de acessos ao cache desde a última vez que o documento i foi acessado e k é a iteração atual. O objeto a ser retirado é o que tem o maior $S_i * \Delta T_{ik}$ onde S_i é o tamanho de i. O objetivo da política é maximizar HR.

Para otimizar a implementação do PSS, é feita uma aproximação. Em vez de calcular o valor $S_i * \Delta T_{ik}$ para todos os documentos, estes são classificados em grupos de acordo com o tamanho e os valores são calculados apenas para o documento menos recentemente utilizado de cada grupo. Posteriormente os escolhidos nos grupos são comparados entre si. O algoritmo apresenta bons resultados para casos de correlação positiva e para casos onde não há correlação entre o tamanho e o número de acessos. Para o caso real, negativamente correlacionado, o algoritmo não apresenta bons resultados.

O desempenho de vários algoritmos em relação a HR e BHR foi comparado em (Arlitt *et al.*, 1998). Os autores analisaram políticas baseadas em freqüência (LFU e LFU-*aging*), em recentidade (LRU, SLRU e LRU-k) e em tamanho (SIZE e GD-Size). Os resultados indicaram que as políticas baseadas em tamanho obtêm melhores resultados em HR enquanto as políticas baseadas em freqüência são melhores quanto a BHR.

Em resumo, as políticas híbridas para substituição em caches na WWW ordenam os arquivos por um valor calculado através uma fórmula do tipo (Wessels, 1995): $valor = f^a.r^b.t^c$, onde f, r e t correspondem, por exemplo, aos parâmetros freqüência, recentidade e tamanho, e as constantes a, b e c são os pesos associados. Outros parâmetros, sempre identificados com os arquivos, também podem fazer parte da fórmula. Arquivos com maior *valor* são mantidos no cache. Por exemplo, a deve ser positivo, valorizando arquivos com maior freqüência e b deve ser negativo para privilegiar arquivos mais recentemente acessados. A constante c é em geral negativa, discriminando arquivos grandes. A fórmula é única para classificar todos os arquivos e obter a ordem na qual os mesmos serão removidos do cache.

A questão de como tratar o tamanho diverso dos objetos em sistemas de cache não é nova. Esta questão é discutida em (Denning e Slutz, 1978) no contexto de cache de memória e em (Copeland *et al.*, 1988) no contexto de caches de sistemas de banco de dados. Neste último trabalho a solução é a mesma descrita no parágrafo anterior, porém os autores só consideram o tamanho e a freqüência. Não há nenhuma referência à variabilidade dos tamanhos dos objetos requisitados. Na primeira referência há um exemplo no qual os tamanhos dos segmentos de memória variavam de dezenas a centenas de bytes com média de 75 bytes. Os objetos não apresentavam a variabilidade nos tamanhos que os objetos da WWW apresentam.

Em relação à organização do espaço dos caches, o modelo de cache particionado de acordo com o tipo de documento, como áudio e não áudio por exemplo, foi proposto anteriormente em (Arlitt e Williamson, 1996). Acreditamos que este particionamento foi uma tentativa de tratar, de forma indireta, a diferença nos tamanhos dos arquivos uma vez que arquivos de áudio são em média muito maiores do que os demais tipos de arquivos. Esta proposta foi analisada em (Williams *et al.*, 1996) em apenas um experimento com uma carga de trabalho e uma métrica, BHR. Os experimentos não são suficientes para gerar uma conclusão segura.

O particionamento por domínio é utilizado em hierarquias de caches de rede (Wessels

e Claffy, 1997). Este particionamento tem como objetivos distribuir a carga entre as máquinas de um mesmo nível da hierarquia e diminuir os custos de comunicação no sistema. A eficiência de diversas configurações de hierarquias de caches foi tratada em (Meira Jr. *et al.*, 1998b) e (Meira Jr. *et al.*, 1998a).

Caches separados têm sido usados em arquitetura de computadores para aumentar o *bandwidth* entre a hierarquia de memória e a CPU. A maioria dos processadores modernos tem caches separados para instruções e dados (Patterson e Hennessy, 1996), que exploram características peculiares ao conjunto servido.

2.3 Estado da Arte em Gerência de Caches

Uma análise das políticas descritas mostra que a pesquisa sobre cache na Web ainda é incipiente. Os artigos publicados sobre o assunto são recentes. No entanto, algumas conclusões podem ser tiradas. Em relação às políticas de reposição, observamos que:

- Os parâmetros da carga mais utilizados nas políticas propostas foram tamanho do documento, recentidade e freqüência dos acessos. Outros parâmetros como tempo de busca, tempo de conexão e *bandwidth* da rede entre o cache e o servidor foram considerados.

- Os algoritmos que levam em conta mais de um parâmetro apresentam, em geral, melhor desempenho no conjunto das métricas (HR e BHR) do que os algoritmos que levam em conta apenas um parâmetro.

- Algoritmos que levam em conta apenas um dos parâmetros, exceto o LRU, provocam poluição no cache, o que equivale a uma diminuição do espaço útil do cache.

- Os autores que consideraram a latência de busca como parâmetro concordam que este não é um parâmetro adequado. É preferível usar parâmetros que dependam da carga em vez de depender do sistema. Portanto, tempos de transmissão, resposta ou *download* não são escolhas apropriadas pois são muito dependentes do sistema em que são executados.

- Embora a maioria dos autores reconheça a importância de obter um bom desempenho em relação às métricas HR e BHR, não há uma proposta para otimizar ou encontrar um compromisso entre as métricas. Este fato é ressaltado na literatura (Cao e Irani, 1997), (Arlitt *et al.*, 1998).

As limitações dos modelos atuais de gerência de espaço são discutidas a seguir. Estes são limites do conhecimento atual na área de pesquisa desta tese e, ao mesmo tempo, oportunidades para desenvolvimento de pesquisa.

1. Embora considerem o tamanho do documento como parâmetro relevante à solução do problema, os autores das políticas de substituição propostas não consideraram nem analisaram as implicações da grande variabilidade do tamanho dos documentos no desempenho da política proposta ou, quando consideraram, adotaram uma política conservativa/defensiva estabelecendo apenas um limite superior para o tamanho dos documentos que podem ser armazenados no cache (Arlitt e Williamson, 1996), (Markatos, 1996). No contexto da substituição de documentos no cache, o armazenamento de um arquivo grande pode significar a remoção de centenas ou milhares de arquivos pequenos para que seja liberado o espaço necessário, com possível perda da localidade de referência.

2. Embora muitos trabalhos reconheçam os benefícios da otimização do desempenho do sistema de cache em relação às duas métricas, HR e BHR, os algoritmos propostos tentam otimizar apenas uma das métricas. Não há na literatura um algoritmo que procure otimizar simultaneamente as duas métricas, embora haja algoritmos com bom desempenho em cada uma, separadamente. Isto significa que, ao escolher um algoritmo, os administradores de caches devem escolher antecipadamente qual métrica será priorizada. Uma mudança nesta decisão implica na implementação de outra política.

3. Os algoritmos conhecidos não oferecem nenhuma possibilidade de adaptação (*tuning*) em tempo de execução, isto é, são completamente estáticos. A única opção para ajuste das métricas em qualquer política é variar o tamanho máximo do arquivo aceito. Limites mais estreitos produzem melhores HR porém diminuem BHR. Os algoritmos utilizam conhecimento prévio do sistema e da carga e, a menos da opção citada, não há nenhum parâmetro que possa ser ajustado em tempo de execução e em função da carga numa tentativa de melhorar o desempenho.

4. As políticas de controle de admissão ainda estão pouco exploradas. O fato de que grande parte dos documentos é requisitada apenas uma vez parece indicar o uso de controle de admissão. No entanto, os trabalhos que utilizam políticas de controle de admissão não avaliaram a efetividade destas políticas, isto é, a relação custo/benefício envolvida na utilização destas políticas ainda não foi pesquisada.

Capítulo 3

Caracterização de Cargas de Caches da WWW

3.1 Importância da Caracterização

A caracterização global da WWW em termos do tráfego e das cargas nos seus diversos componentes é fundamental para seu entendimento. No entanto, o tamanho e a natureza descentralizada da WWW tornam difíceis as medições de âmbito global. Mesmo métricas simples como número de usuários, número de páginas disponíveis e número de servidores são de difícil avaliação. Apesar desta dificuldade, vários esforços têm sido feitos neste sentido e um sumário dos resultados pode ser encontrado em (Pitkow, 1998). Neste sentido, a caracterização apresentada neste capítulo contribui para este esforço global.

A construção de soluções eficientes para o gerenciamento do espaço de caches requer um conhecimento detalhado das propriedades da carga. Este conhecimento ajuda a dimensionar o cache e a definir uma política de substituição adequada ao padrão de acesso identificado. Entretanto, não são conhecidos modelos analíticos que possam ser utilizados para predizer a fração de *hits*, dados o tamanho de cache, a política de reposição e as características estatísticas da carga (Blaze, 1993). Em um extremo, a fração de *hits* poderá chegar a 100% se os mesmos dados são repetidamente requisitados e mantidos no cache. Por outro lado, a fração de *hits* será 0% se cada requisição for para um objeto não requisitado anteriormente. As frações de *hits* são completamente dependentes da distribuição dos acessos em cada instância da carga. Seus valores para cada política de substituição são obtidos por técnicas de avaliação experimentais, por exemplo, simulação.

Este capítulo identifica propriedades da carga que afetam as métricas HR e BHR. A carga de um cache consiste da seqüência de requisições recebidas durante um período de observação. As cargas de dez caches reais da WWW são analisadas. O objetivo é descobrir características comuns, que são invariantes para o conjunto de cargas e que são relevantes para o desempenho dos caches em termos das métricas em avaliação. A análise focaliza os parâmetros tamanho das requisições e dos arquivos únicos, e o padrão de acesso identificado nas cargas. A variabilidade nos tamanhos e suas implicações são mostradas. O

entendimento do padrão de acesso revela o potencial e as limitações dos caches na WWW.

As propriedades da carga reveladas neste capítulo formam a base de conhecimento na qual se fundamentam as soluções propostas e contribuições deste trabalho. Alguns números apresentados aqui são similares a caracterizações já publicadas na literatura, o que serve para confirmar dados anteriores e, ao mesmo tempo, validar as cargas utilizadas neste trabalho. A contribuição está nas novas formas de compor e interpretar os dados.

Este capítulo está organizado em sete seções. A seção 3.2 descreve as cargas analisadas. A seção 3.3 mostra evidências da grande variabilidade nos tamanhos das requisições e dos documentos únicos. A seção 3.4 apresenta a caracterização do padrão de acesso das cargas dos Web-caches. A seção 3.5 apresenta caracterizações para classes de requisições pré-definidas de acordo com o tamanho e analisa o comportamento das classes em relação à carga não particionada. Na seção 3.6 são discutidas as limitações e as potencialidades de desempenho dos caches da WWW. A seção 3.7 apresenta as conclusões e mostra como o conhecimento obtido nesta análise pode ser aplicado para aperfeiçoar a gerência de espaço nos Web-caches.

3.2 Cargas de Trabalho Analisadas

Com o objetivo de generalizar os resultados deste trabalho foram analisadas cargas de dez caches da WWW. Estas cargas foram obtidas a partir do registro em ordem cronológica (*trace*) das requisições feitas a sistemas de caches reais. O conjunto apresenta grande amplitude em vários aspectos. Os registros foram feitos ao longo de cinco anos (1995 a 1999), em caches de cliente, de servidor e de rede, em países de três continentes, em estágios diferentes de implantação da Internet. As instituições são pioneiras em alguns aspectos. Esoterica (Esoterica, 1998), POP-MG (POP-MG, 1998) e NLANR (NLANR, 1999) implementam os maiores caches de rede em seus respectivos países. VTech (Virginia Tech, 1995) foi uma das primeiras instituições a registrar e analisar tráfego da WWW. A tabela 3.1 apresenta a identificação das cargas, sua origem, domínio atendido e data de registro. As cargas estão apresentadas por ordem cronológica de ano de registro. Segue uma descrição de cada cache e carga.

- BL: cache local para clientes internos ao *Department of Computer Science* do VTech, com os clientes do Departamento acessando qualquer servidor na Web; as requisições são para servidores dentro e fora do domínio. Os registros compreendem 37 dias de acesso. Esta e as demais cargas do VTech, BR e U, foram utilizados no trabalho de (Williams *et al.*, 1996), onde pode ser encontrada descrição detalhada do procedimento de registro dos acessos.

- BR: cache de servidor do VTech servindo clientes na Web; os clientes estão fora do domínio acessando um *site* no domínio. Compreende 38 dias de acesso.

- U: cache de *proxy* em um *firewall* do VTech; todas as requisições são para fora da rede na qual os clientes estavam conectados, dado que não havia servidores nesta rede. Compreende acessos feitos num período de 190 dias.

Carga	Instituição	País	Domínio	Data
BL	VTech	EUA	todos	set a out 1995
BR	VTech	EUA	local	set a out 1995
U	VTech	EUA	todos	abril a out 1995
NASA	Nasa	EUA	local	04-31 ago 1995
POP98	POP-MG	Brasil	todos	24 out a 4 nov 1997
Portugal	Esotérica	Portugal	todos	01-07 mar 1998
POP99-hug	POP-MG	Brasil	.br	12-15 jan 1999
POP99-zez	POP-MG	Brasil	exceto .br	12-14 jan 1999
NLANR-uc	NLANR	EUA	.com	14 jan 1999
NLANR-bo1	NLANR	EUA	.edu, .gov, etc	14 jan 1999

Tabela 3.1: Identificação, origem e data das cargas caracterizadas.

- NASA: carga do cache de um servidor WWW da NASA (NASA, 1995). Compreende todos os acessos feitos num período de 28 dias. A carga foi obtida em (Archive, 1997).

- POP98: carga do primeiro nível de uma hierarquia de servidores *proxy* do POP-MG. Este é um cache de rede para provedores de acesso a Internet, universidades e empresas. O cache é gerenciado pelo software Squid (Squid, 1997), que também gera o *trace*. Compreende 12 dias de acesso.

- POP99: carga do segundo nível de uma hierarquia de servidores *proxy* do POP-MG. Neste caso, o cache estava organizado por domínios: a carga "hug" registrou as requisições para o domínio **.br** enquanto a carga "zez" registrou as requisições para os demais domínios. Compreende 4 dias de acesso.

- NLANR: carga de dois caches do maior sistema de caches de rede dos EUA. A hierarquia do NLANR, também implementada pelo Squid, é composta por uma dezena de caches espalhados pelo país sendo cinco deles interligados por um *backbone* de alta velocidade (vBNS[1]). Os caches também estão divididos por domínios: "bo1", localizado em *Boulder, Colorado*, atende os domínios **.edu, .gov, .org, .mil e .us**; o cache "uc", localizado em *Urbana-Champaign, Illinois*, atende o domínio **.com**. Cada carga compreende vinte e quatro horas de acesso.

- Portugal: carga do cache de rede implementado pelo maior provedor de acesso a Internet de Portugal. Compreende 7 dias de acesso.

Todas as cargas utilizadas neste trabalho são resultado de pré-processamento aplicado aos *traces* originais. Cada linha de um *trace* corresponde a um URL requisitado pelo usuário e contém as seguintes informações: hora da requisição, o URL, o tamanho da

[1] *very high speed Backbone Network Service*, http://www.vbns.net

resposta (pode incluir o cabeçalho do protocolo HTTP), o tempo de envio da resposta em segundos, o método HTTP e o código de retorno, entre outras informações, que variam de acordo com o tipo de cache e o software que o implementa. O objetivo do pré-processamento é separar as requisições que são pedidos de arquivos ao cache e que alteram o estado de seu sistema de armazenamento ou das listas de metadados, de outras requisições como, por exemplo, o pedido de verificação de presença ou de validade de documentos. As cargas correspondem às requisições HTTP com método GET, que foram transmitidas com sucesso (código 200). Requisições para objetos dinâmicos foram excluídas.

3.3 Caracterização dos Tamanhos

A caracterização dos tamanhos é baseada na seqüência de requisições e no conjunto de arquivos distintos. A seqüência de arquivos transmitidos em resposta às requisições dos clientes é denominada "requisições", e o conjunto de arquivos distintos transmitidos é denominado "arquivos". A seqüência de requisições é composta por todos os URLs requisitados pelos usuários. Esta seqüência pode conter várias vezes o mesmo URL. A repetição ocorre porque um usuário requisita o mesmo arquivo mais de uma vez ou porque usuários diferentes requisitam o mesmo arquivo. Muitas destas requisições repetidas resultam em *hits* no cache, o que significa que elas são satisfeitas sem gerar tráfego no trecho da rede entre o cache e o servidor.

O conjunto de arquivos contém exatamente uma entrada para cada arquivo distinto, independente do número de vezes que o arquivo foi requisitado. O tamanho associado a cada URL é o tamanho da resposta, ou seja, do conteúdo do URL, e não o tamanho da requisição propriamente dita.

A seqüência de requisições e respostas influencia as propriedades do tráfego. A caracterização do tráfego gerado pela WWW e o estudo de desempenho da Internet são baseados nesta seqüência. O tráfego WWW domina o tráfego da Internet, sendo responsável, em 1997, por cerca de 65% a 80% dos bytes e 55% a 75% dos pacotes (Thompson *et al.*, 1997). O conjunto de arquivos é o conjunto de documentos que são armazenados no cache. Sua caracterização é fundamental para a análise de desempenho dos Web-caches. As duas seções seguintes apresentam a caracterização da seqüência de requisições e do conjunto de arquivos distintos para todas as cargas.

3.3.1 Tamanhos das Requisições

A tabela 3.2 apresenta a caracterização básica de cada carga, a saber, o número de requisições, o total de bytes transmitidos, o tamanho médio das requisições e a mediana. No total são analisadas 8 746 315 requisições que compreendem mais de 110 gigabytes de dados transmitidos. A primeira observação é que o crescimento da WWW é refletido na tabela. Analisando os dados dos registros mais antigos para os mais recentes, nota-se um número crescente de requisições e bytes transmitidos para espaços de tempo cada vez menores (o período no qual cada carga foi registrada está na tabela 3.1). Por exemplo, o

número diário de requisições registrado no POP-MG no início de 1999 (carga POP99) é três vezes maior do que o registrado no final de 1997 (carga POP98).

Carga	Requis.	Bytes	Média	Mediana	Menor	Maior	$\frac{\text{Maior}}{\text{Mediana}}$
BL	53399	672807109	12600	2654	8	7949704	2995
BR	179600	10070970465	56074	1999	18	10032887	5018
U	173597	2070556791	11927	2268	1	18825928	8300
NASA	1385259	26766987833	19323	4179	28	3421948	819
POP98	2111766	19142569124	9065	3235	17	20983009	6486
Portugal	1193404	10780304335	9033	2779	51	17919100	6448
POP99hug	1121747	11505589326	10257	2929	17	27164215	9274
POP99zez	1120830	17062836909	15223	2905	16	66017537	22725
NLANR-uc	800534	11291886105	14105	3461	51	53702937	15516
NLANR-bo1	606179	9028332340	14894	3683	51	49977774	13569

Tabela 3.2: Caracterização das requisições: número, total de bytes transmitidos e índices de dispersão para o tamanho. A última coluna é a divisão do tamanho do maior documento pela mediana.

A observação dos dados revela evidências de variabilidade. A mediana está entre 2 e 3.7 kbytes. A média está entre 9 e 19 kbytes, com exceção para a carga BR. Esta carga contém requisições repetidas para arquivos de áudio, que são muito grandes, e elevam a média para 56 kbytes. O fato de a mediana ser menor do que a média indica que os dados seguem uma distribuição com cauda à direita. Os maiores documentos são da ordem de 10^7, seis ordens de grandeza maiores do que os menores documentos. Estes índices de dispersão põem em evidência a grande variabilidade existente nos tamanhos das requisições. A última coluna da tabela 3.2 revela que milhares de documentos de tamanho igual à mediana podem ocupar o mesmo lugar de apenas um dos maiores arquivos.

A figura 3.1 mostra a distribuição de probabilidade acumulada dos tamanhos de cada seqüência de requisições. A idéia não é identificar cada curva em particular, mas mostrar que o comportamento das distribuições é bastante similar. Para todas as cargas, as requisições de tamanho pequeno são a maioria. Há também um número pequeno de requisições muito grandes. O eixo x está em escala logarítmica. As cargas NASA e BR se diferenciam das demais cargas. BR apresenta um número maior de requisições pequenas, menores do que 1 kbyte. Ambas apresentam um número maior de requisições grandes do que as outras cargas. Este comportamento é consistente com os valores das médias e medianas apresentados na tabela 3.2.

A distribuição de Pareto é apontada em (Crovella e Bestavros, 1995) e (Arlitt e Williamson, 1996) como a que melhor representa os tamanhos dos objetos da WWW, enquanto (Abdulla, 1998) indica as distribuições lognormal e Weibull para representar os tamanhos. Em (Barford e Crovella, 1998), os autores propõem que a caracterização dos tamanhos dos objetos da WWW seja feita por duas distribuições, uma para representar o corpo da distribuição, que inclui a maioria das requisições, e outra para caracterizar a cauda. O corpo seria caracterizado pela distribuição lognormal, enquanto a cauda seria melhor representada pela distribuição de Pareto.

Para mostrar a variabilidade, estamos interessados na distribuição que representa a

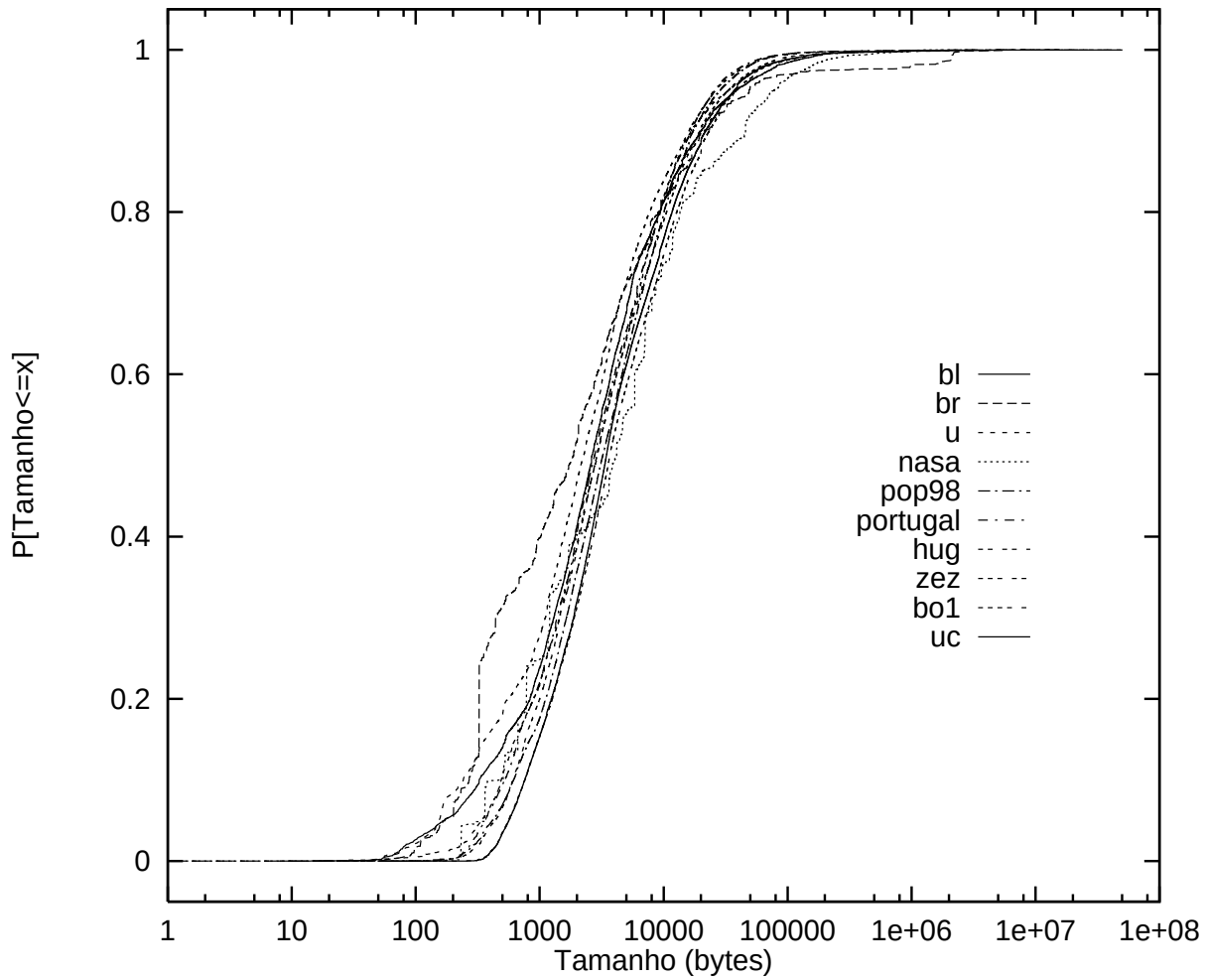

Figura 3.1: Distribuição de probabilidade acumulada empírica dos tamanhos das requisições para todas as cargas.

cauda. A figura 3.2 mostra em um gráfico log-log o complemento da distribuição de probabilidade para as seqüências de requisições e para as funções exponencial e Pareto. Neste gráfico, a inclinação da curva indica o valor do parâmetro α para a distribuição de cauda pesada correspondente. Por exemplo, a reta correspondente à distribuição de Pareto tem $\alpha = 1.3$. O gráfico demonstra que as caudas de todas as cargas têm decaimento similar ao apresentado pela distribuição de Pareto para tamanhos entre 10^4 e 10^6. As cargas NASA, BR e BL apresentam decaimento rápido para valores de tamanho maiores que 10^6 (extremo da cauda) enquanto as cargas restantes continuam seguindo Pareto.

Os métodos *scaling* e CDPlot (Crovella e Taqqu, 1999) foram utilizados para estimar os valores de α para todas as cargas. Os resultados estão na coluna "requisições" da tabela 3.4. Os valores de α, entre 0.85 e 1.5, confirmam as evidências de que as distribuições são de cauda pesada, exibindo, portanto, grande variabilidade.

As diferenças nos valores de α estimados pelos dois métodos devem-se à própria ela-

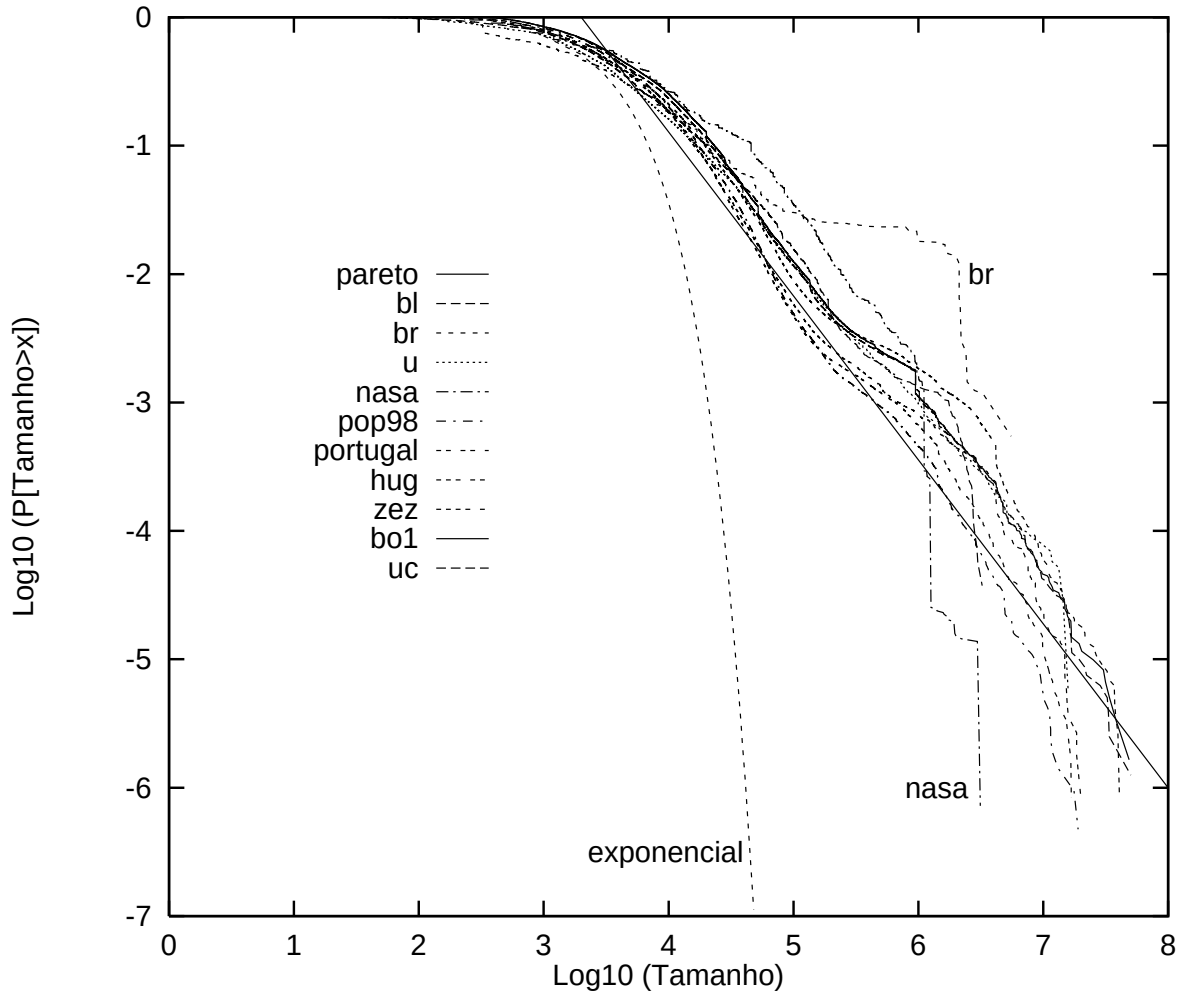

Figura 3.2: Cauda da distribuição de probabilidade empírica dos tamanhos das requisições para todas as cargas e para as funções exponencial e Pareto.

boração dos métodos. O método CDPlot consiste em estimar a inclinação do decaimento da cauda. Para isso é necessário definir um valor inicial a partir do qual o comportamento da lei de potência que define a cauda pesada se inicia. Para arquivos WWW, este valor está em torno de 10^4 (Crovella e Taqqu, 1999). Uma escolha apropriada deste valor é importante porque influencia a estimativa de α. O método *scaling* elimina este problema, mas apresenta algumas tendências que podem mascarar os resultados. Sua estimativa é mais precisa para valores menores de α e para conjuntos grandes de dados (acima de cem mil números). Este método agrega pontos de dados na distribuição e calcula a distribuição complementar do conjunto de dados agregado. Se a distribuição apresenta cauda pesada então as caudas dos conjuntos formados por agregações sucessivas dos dados serão aproximadamente paralelas com inclinação $-\alpha$. Apesar das diferenças nas estimativas dos dois métodos, todos os valores encontrados confirmam que as distribuições apresentam cauda pesada.

Pelos valores das medianas e dos tamanhos das maiores requisições, mostrados na tabela 3.2, podemos observar um crescimento do tamanho dos arquivos ao longo do tempo. As cargas de cache de rede registradas em 1998 e 1999 apresentam medianas maiores do que as medianas das cargas registradas anteriormente. Esta informação pode ser reforçada pela observação de que as cargas de 1999 também registram os maiores arquivos transmitidos. Este crescimento no tamanho das respostas é confirmado em (Abdulla, 1998). Paralelamente, não foi registrado nenhum aumento no tamanho do menor documento.

3.3.2 Tamanhos dos Arquivos Únicos

As estatísticas dos conjuntos de arquivos são apresentadas na tabela 3.3. O número de arquivos e o total de bytes estão representados como uma fração do número de requisições e do número de bytes requisitados, respectivamente, apresentados na tabela 3.2. A cardinalidade do conjunto de arquivos representa uma porcentagem significativa do número de requisições, entre 36% e 67% (com exceção para BR e NASA), o que corresponde a um total de bytes únicos que está entre 46% e 74% dos bytes transmitidos. As cargas BR e NASA apresentam uma fração muito menor de arquivos e bytes distintos, o que é esperado pois seu domínio é restrito apenas aos objetos armazenados no servidor.

Carga	% Arquivos distintos	% Bytes Únicos	Média	Mediana
BL	50.13	63.39	15929	2752
BR	5.21	2.07	22187	1519
U	44.97	61.77	16319	2748
NASA	0.332	0.798	46463	6297
POP98	36.06	46.33	10755	3563
Portugal	46.49	62.26	12052	3389
POP99-hug	40.85	52.41	12743	3284
POP99-zez	60.59	73.61	18339	3512
NLANR-uc	66.59	69.57	14354	3660
NLANR-bo1	66.77	67.67	14801	3777

Tabela 3.3: Caracterização dos conjuntos de arquivos distintos baseada nos dados das seqüências de requisições.

Tanto a média dos arquivos quanto a mediana são maiores neste conjunto do que as respectivas medidas para a seqüência de requisições (exceto BR). Isto significa que os arquivos pequenos são mais requisitados do que os arquivos grandes (maiores que a média). A mediana é menor do que a média também nos conjuntos de arquivos, o que é uma indicação de que uma distribuição de cauda pesada pode modelar os dados.

A figura 3.3 apresenta a distribuição de probabilidade acumulada dos tamanhos para os conjuntos de arquivos. Neste gráfico, as curvas tem um perfil ainda mais similar entre si do que as curvas baseadas nas seqüências de requisições (figura 3.1). As curvas que se distinguem são a da carga BR (mais à esquerda) e a da carga NASA (mais à direita).

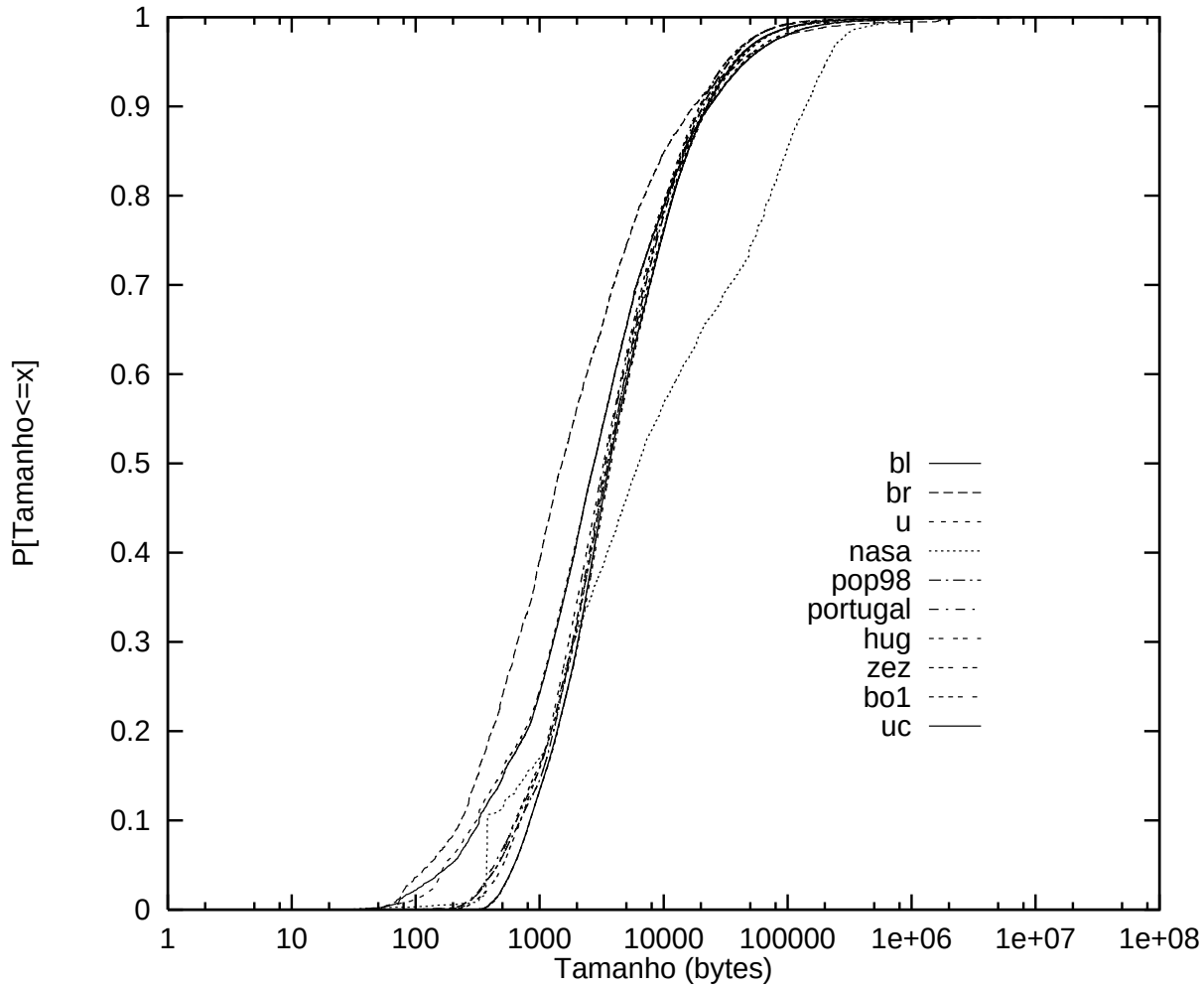

Figura 3.3: Distribuição de probabilidade acumulada empírica dos tamanhos dos arquivos únicos para todas as cargas.

BR contém um número maior de arquivos pequenos, enquanto NASA contém um número menor de arquivos pequenos. A cauda da distribuição de probabilidade é mostrada na figura 3.4. Os valores estimados para α estão na coluna "arquivos" da tabela 3.4. Comparando os dois, temos que o conjunto de arquivos apresenta maior variabilidade (menor α) do que a encontrada para as requisições para a maioria das cargas.

A análise dos tamanhos do conjunto de arquivos e da seqüência de requisições mostra que as cargas apresentam indubitavelmente uma grande variabilidade nos seus tamanhos e que o conjunto de arquivos, que forma a carga de armazenamento do cache, exibe ainda maior variabilidade do que a seqüência de requisições.

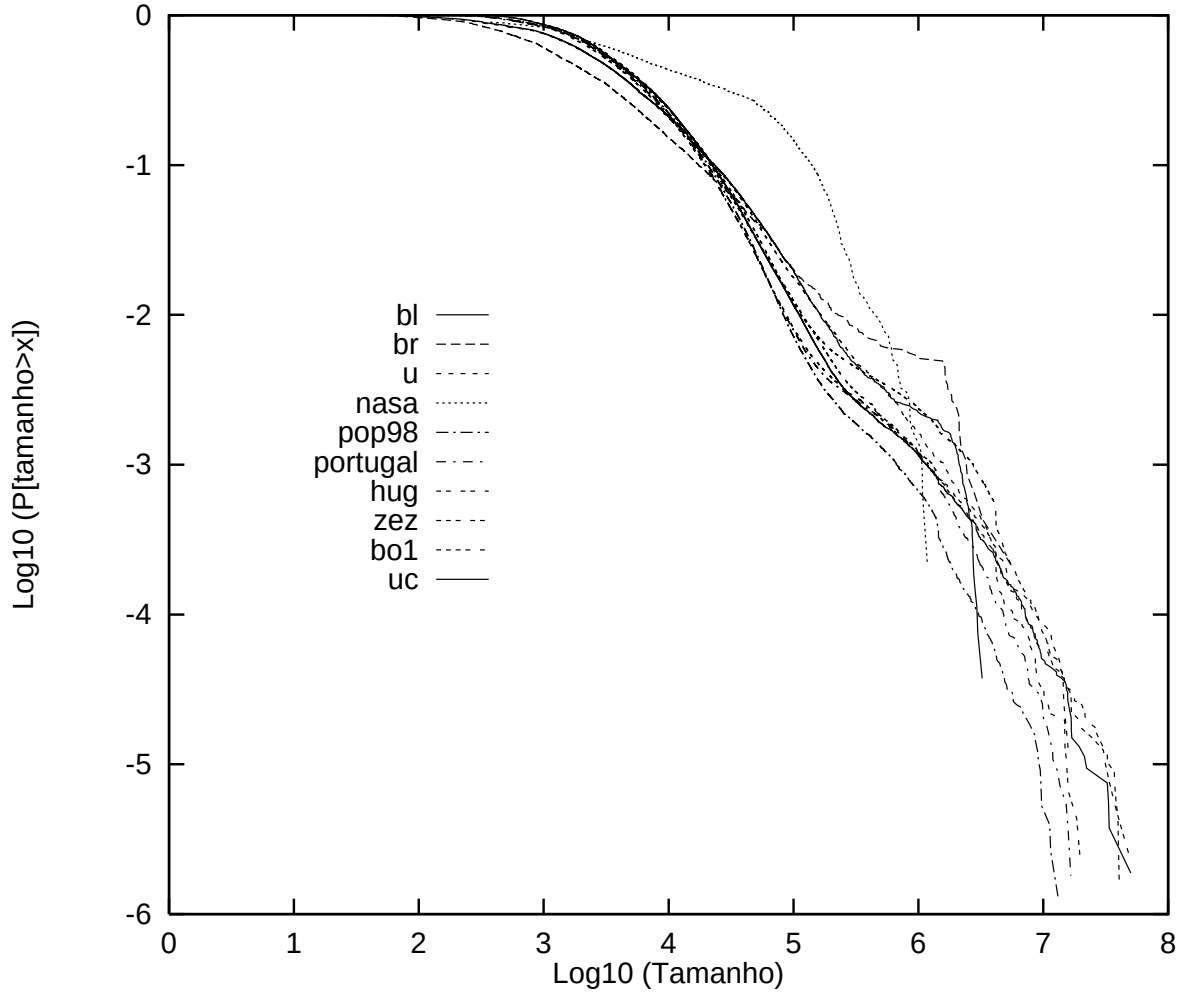

Figura 3.4: Cauda da distribuição de probabilidade empírica dos tamanhos dos arquivos únicos para todas as cargas.

3.4 Caracterização do Padrão de Acesso

Padrão de acesso é um conjunto específico de características de uma seqüência de requisições que define a localidade de uma carga e o limite de desempenho que um sistema de caches pode alcançar quando submetido à referida carga. As características que compõem o padrão de acesso podem ser divididas em dois grupos: as temporais, que dependem da estrutura cronológica (ordem de referência) dos acessos, e as atemporais, que não dependem desta estrutura.

As características atemporais mais comuns são a popularidade dos arquivos e a concentração de referências. A popularidade é dada pela distribuição do número de referências para cada arquivo individual. A concentração de referências pode ser expressa de várias formas. Dentre elas estão a distribuição relativa das requisições, dos arquivos e dos bytes, a fração de arquivos e bytes distintos, e a distribuição dos arquivos por número de acessos,

Carga	requisições		arquivos	
	scaling	CDPlot	*scaling*	CDPlot
BL	1.20	1.10	1.01	1.06
BR	1.50	0.85	0.89	0.80
U	1.03	1.10	1.00	1.10
NASA	0.99	1.16	1.00	1.30
POP98	0.99	1.43	0.99	1.39
Portugal	1.16	1.32	1.06	1.31
POP99-hug	1.02	1.32	0.98	1.28
POP99-zez	0.95	1.24	0.97	1.19
NLANR-uc	1.08	1.23	1.04	1.20
NLANR-bo1	1.07	1.23	1.08	1.22

Tabela 3.4: Estimativas de α para os arquivos e as requisições pelos métodos *scaling* e CDPlot.

por exemplo, a fração das requisições e bytes acessados uma única vez.

Localidade de referência é um princípio observado experimentalmente na execução de programas de computador. Segundo este princípio, a distribuição das referências às páginas de um programa é não uniforme. Diz-se que um programa tem boa localidade quando pequenos conjuntos de seu espaço de endereçamento são referenciados por longos períodos de tempo. A regra básica da localidade temporal é 90% das instruções de um programa referenciam apenas 10% do seu código. A localidade de referência é a base teórica do conceito de hierarquia de memória (Patterson e Hennessy, 1996). Um resumo histórico do desenvolvimento do conceito de localidade e das contribuições de diversos pesquisadores pode ser encontrado em (Denning, 1980).

A localidade tem dois componentes, o conjunto de objetos acessados, também conhecido como *working set*, e o intervalo de tempo no qual este conjunto é referenciado. A localidade mostra como as re-referências aos documentos são distribuídas no tempo e é, portanto, uma característica temporal do padrão de acesso.

Esta seção examina o padrão de acesso presente nas cargas apresentando diversas medidas para a concentração de referências e a localidade temporal das cargas da WWW.

3.4.1 Valores Máximos para HR e BHR

A rigor, a concentração de referências vem sendo caracterizada desde o início do capítulo. As frações de arquivos e bytes distintos, apresentadas na tabela 3.3 são uma medida da quantidade de documentos e bytes novos que ocorre em cada seqüência. A partir destes dados podemos calcular os valores máximos de HR e BHR que um sistema de cache pode alcançar quando submetido a estas cargas. HR_{max} e BHR_{max} são, respectivamente, a fração de requisições repetidas e a fração de bytes que compõe as repetições, ou seja, o complemento de 100% das porcentagens da tabela 3.3. Estes valores poderão ser obtidos em caches com tamanho suficiente para armazenar todos os documentos requisitados. Esse

tipo de cache é conhecido na literatura como "cache infinito". Nesta tese dizemos que um cache tem "tamanho de referência" quando seu tamanho é suficiente para armazenar todos os documentos únicos de uma carga em questão. Todas as referências a tamanhos de cache são tomadas a partir do tamanho de referência. Por exemplo, um cache de tamanho dez por cento para uma dada carga tem tamanho em bytes equivalente a dez por cento do total de bytes necessário para armazenar todos os arquivos distintos da referida carga. Portanto, em um cache com tamanho equivalente a cem por cento do tamanho de referência, nenhuma retirada é necessária. Isto elimina a influência das políticas de reposição no desempenho do cache, que fica limitado apenas pela fração de requisições novas feitas pelos clientes ao cache.

A tabela 3.5 apresenta os valores para HR_{max} e BHR_{max}. Estas frações dão uma medida da localidade para o período de tempo que compreende toda a carga. Os valores encontrados para HR, entre 33% e 64%, e BHR, entre 26% e 54%, são comuns para caches de rede (Menascé e Almeida, 1998), (Abrams *et al.*, 1995), (Arlitt *et al.*, 1998). Valores maiores, como os das cargas BR e NASA, são encontrados em caches de servidores (Yeager e McGrath, 1996). Para todas as cargas exceto BR, os valores de HR são maiores do que os de BHR, o que indica que há mais *hits* para objetos pequenos. Os valores reais de HR e BHR são função do tamanho do cache e da política de gerenciamento do espaço.

Carga	HR (%)	BHR (%)
BL	49.87	36.61
BR	94.79	97.93
U	55.03	38.23
NASA	99.67	99.21
POP98	63.94	53.66
Portugal	53.51	46.49
POP99-hug	59.15	40.85
POP99-zez	39.41	26.39
NLANR-uc	33.41	30.43
NLANR-bo1	33.23	32.33

Tabela 3.5: Valores máximos para HR e BHR.

3.4.2 Impacto dos Acessos Únicos

A tabela 3.6 apresenta a distribuição dos arquivos de acordo com o número de acessos e a concentração de requisições feitas aos arquivos mais acessados.

A distribuição dos arquivos por número de acessos apresenta valores bem parecidos para todas as cargas, exceto NASA e BR. A maioria absoluta dos arquivos é requisitada uma única vez. Por exemplo, foi registrada apenas uma requisição para 65% dos arquivos da carga POP99-hug e para 88% dos arquivos das cargas do NLANR. NASA tem 19% dos arquivos com uma única requisição enquanto BR tem 42%.

Considere as seguintes definições:

$$\text{HD} = \{j \mid \text{número de requisições para o arquivo } j \text{ é} \geq 2\}, \quad 1 \leq j \leq m$$

e m é o número de arquivos numa carga.

$$\text{HR} = \{i \mid \text{a requisição } i \text{ é para um objeto } j \in \text{HD}, \} \quad 1 \leq i \leq n$$

e n é o número de requisições de uma carga.

$$\textit{hit-docs} = \frac{|\text{HD}|}{m}$$

$$\textit{hit-bytes} \ = \ \frac{\sum_{j=1}^{m} s_j \ \text{ para todo } j \in \text{HD}}{\sum_{j=1}^{m} s_j \ \text{ para todo } j}, \qquad \text{onde } s_j \text{ é o tamanho do arquivo } j.$$

$$\textit{hit-reqs} = \frac{|\text{HR}|}{n}$$

Em palavras, HD é o conjunto de documentos com mais de um acesso, *hit-docs* é a fração dos documentos únicos com mais de um acesso em relação ao total de documentos únicos e *hit-bytes* é a fração de bytes dos documentos únicos com mais de um acesso em relação ao total de bytes destes documentos. A tabela 3.6 mostra a fração de *hit-docs* (a soma das colunas 2, 3, 4+ da mesma tabela) e de *hit-bytes*, ou seja, a fração dos bytes únicos que pertence aos *hit-docs*, para cada carga. A coluna *hit-reqs* mostra a porcentagem das requisições que são para os *hit-docs*. Por exemplo, para a carga POP98, somente 30% dos arquivos distintos são requisitados mais de uma vez. Estes 30% dos arquivos ocupam 31% dos bytes únicos e respondem por 75% das requisições. Para a carga BR, 98% dos acessos são para 58% dos arquivos que compreendem 84% dos bytes únicos. Finalmente, para as cargas do NLANR, 41% das requisições são para 12% dos arquivos. Isto significa que 59% das requisições do NLANR são para objetos que não foram requisitados anteriormente e não serão requisitados novamente. No entanto, devemos lembrar que as cargas do NLANR contêm apenas registros feitos em apenas vinte e quatro horas de acesso. Cargas com duração mais longa podem apresentar melhores concentrações de referência.

Os *hit-bytes* indicam qual é a fração do tamanho de referência que é realmente útil para o cache, isto é, que produz *hits*. Continuando o exemplo do POP98, para armazenar os documentos que terão pelo menos um *hit* ao longo do período que compreende toda a carga, é necessário um espaço de apenas 31% do tamanho de referência do cache para a carga. Portanto, se todos os documentos são armazenados no cache, a maior parte do espaço será ocupada por arquivos inúteis para o cache.

Isto sugere que uma forma de controle de admissão seria útil para evitar a poluição do espaço de armazenamento dos caches com documentos que são requisitados apenas uma vez. Para isso é necessário distinguir dois tipos de *miss*. O primeiro tipo é o *first time*

	Número de Acessos				H I T S		
Carga	1	2	3	4+	hit-docs	hit-bytes	hit-reqs
BL	0.75	0.12	0.05	0.08	0.25	0.18	0.62
BR	0.42	0.15	0.08	0.35	0.58	0.84	0.98
U	0.76	0.12	0.04	0.08	0.24	0.21	0.66
NASA	0.19	0.08	0.07	0.66	0.81	0.91	0.99
POP98	0.70	0.14	0.05	0.11	0.30	0.31	0.75
Portugal	0.77	0.12	0.04	0.07	0.23	0.38	0.64
POP99-hug	0.65	0.17	0.07	0.11	0.35	0.32	0.73
POP99-zez	0.79	0.13	0.04	0.04	0.21	0.18	0.52
NLANR-uc	0.88	0.08	0.03	0.03	0.12	0.12	0.41
NLANR-bo1	0.88	0.08	0.03	0.03	0.12	0.13	0.41

Tabela 3.6: Distribuição dos arquivos por número de acessos e concentração de referências.

miss, que ocorre a primeira vez que um documento é requisitado. Sua quantidade é função do número de objetos disponíveis e da diversidade de interesses dos usuários da WWW. Um forma de evitar esse tipo de *miss* é tentar antecipar o pedido do cliente buscando um documento que possivelmente será requisitado no futuro e armazenando-o no cache. Esta técnica é conhecida como *prefetching* (Cao, 1996). O segundo tipo de *miss* ocorre quando um documento já esteve no cache e foi retirado. Este tipo é devido à limitação do tamanho do cache e às escolhas das políticas de reposição. Este é o tipo de *miss* que as políticas devem procurar evitar ao máximo.

Dada a grande quantidade de documentos com apenas uma requisição, seria interessante uma política de substituição que discriminasse estes documentos. SLRU (*Segmented LRU*) (Karedla *et al.*, 1994) implementa duas listas LRU, uma protegida, para documentos com mais de um acesso, e a outra para documentos com apenas um acesso. Assim é possível eliminar mais rapidamente os arquivos que não têm o segundo acesso e preservar os que têm mais acessos.

Outra característica importante da carga na WWW é a freqüência de acesso aos documentos individuais, também denominada popularidade. A popularidade é inversamente correlacionada com o tamanho (Almeida *et al.*, 1996b). Em outras palavras, os documentos menores são os mais requisitados. O número de referências a documentos específicos pode ser uma informação importante para o servidor do documento, se o documento é um produto a ser vendido ou traz propaganda associada. Para o cache, o importante é a localidade gerada pela popularidade dos documentos, que está incluída nas estatísticas do padrão de acesso. Cabe às políticas de reposição explorar esta localidade mantendo os documentos populares no cache.

3.4.3 Localidade de Referência Temporal

Localidade temporal refere-se à noção de que o mesmo documento é requisitado sucessivas vezes dentro de curtos intervalos de tempo. Uma medida simples e interessante para localidade é o gráfico de localidade apresentado em (Cunha, 1997). Neste gráfico, cada URL recebe uma identificação distinta, de acordo com a ordem em que aparece na carga. As referências também são numeradas e indicam o tempo decorrido em termos do número de requisições feitas ao cache. O número da referência e o número da URL correspondente são plotados no gráfico. Cada URL aparece apenas a primeira vez em que é referenciada.

A figura 3.5 apresenta este gráfico para todas as cargas. Quanto maior a localidade, mais perto do eixo x estarão as linhas do gráfico, indicando que referências novas ocorrem infrequentemente. Por outro lado, quanto mais freqüentes são as novas referências, mais inclinada é a curva. No pior caso, os dados seguem a diagonal, o que significa que cada requisição é para uma URL distinta. O valor para a localidade é a inclinação, dada por $1 -$ (número de documentos únicos/número de requisições). É interessante observar no gráfico que a localidade se altera lenta e constantemente no intervalo mostrado. Há alterações localizadas na inclinação da curva, porém estas são quase imperceptíveis. Não são observadas fases distintas, como nos caches de memória, que alternam períodos de transição nos quais a localidade é pequena com períodos de localidade elevada.

As cargas com maior localidade são as dos servidores NASA e BR. Dentre as cargas de cache de rede, U, POP98 e POP99-hug são as que apresentam maior localidade, enquanto POP99-zez é a que apresenta menor localidade. Estes resultados são consistentes com os valores máximos de HR e BHR da tabela 3.5.

Várias métricas foram propostas para a quantificar a localidade. Um dos trabalhos pioneiros (Denning e Schwartz, 1972) mostra a relação entre três métricas: distribuição do intervalo entre referências, fração de *miss* e tamanho médio do *working set*. Intervalo de localidade limitada (Madison e Batson, 1976) é uma métrica de localidade que apresenta algumas vantagens sobre o conceito de *working set*. Trabalhos mais recentes estudaram o conceito de localidade aplicado à WWW. A distribuição marginal das distâncias na pilha LRU é apontada como uma medida da localidade temporal para cargas da WWW (Almeida *et al.*, 1996b).

Estes trabalhos têm em comum o fato de que seus próprios autores demonstram que as métricas apresentam resultados compatíveis com a fração de *hits* dada pela política LRU. Por exemplo, a política baseada em *working set* apresenta resultados idênticos à política LRU quando o tamanho da janela utilizado no *working set* é igual ao tamanho da memória utilizado para executar a política LRU. Há também uma equivalência entre a distribuição marginal das distâncias na pilha LRU e a fração de *miss* medida pela política LRU. Isto ocorre porque LRU implementa o corolário da localidade (Patterson e Hennessy, 1996): se os objetos recentemente acessados têm probabilidade maior de serem reacessados então o melhor candidato a ser substituído é o objeto com menor recentidade. Por isso os resultados da aplicação da política LRU são comumente utilizados para expressar a localidade.

A efetividade dos esquemas de cache está na presença da localidade temporal nas seqüências de requisições da WWW e no uso de políticas apropriadas de gerenciamento.

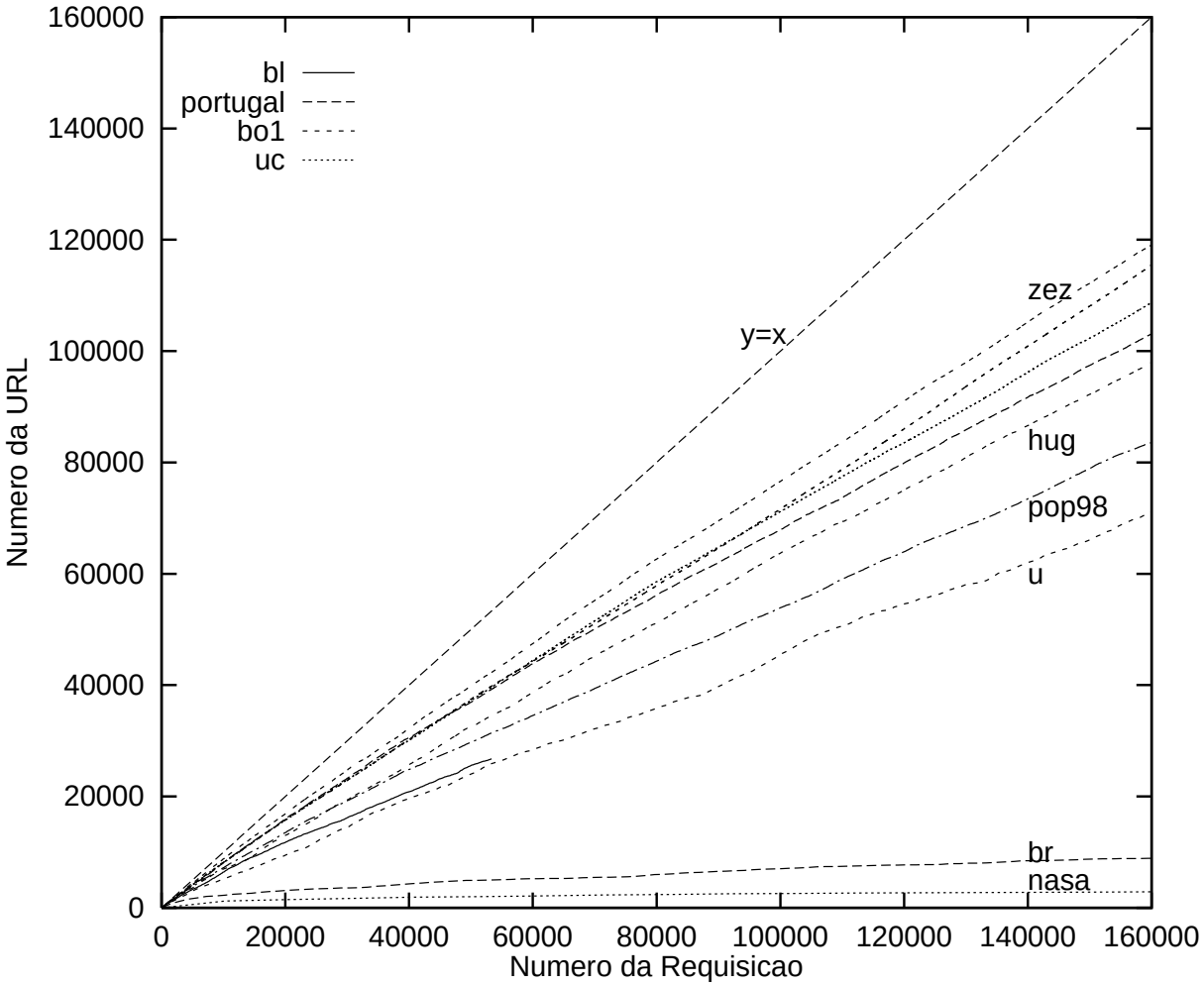

Figura 3.5: Evidência de localidade temporal para todas as cargas.

Como mostrado nesta seção, o grau de localidade observado nos caches da WWW pode ser tão bom quanto o encontrado em caches tradicionais (acima de 90% para as cargas BR e NASA) ou não. Outros trabalhos também mostram que a carga de diversos tipos de cache apresenta boa localidade de referência (Almeida *et al.*, 1996b), (Arlitt e Williamson, 1996), (Cao e Irani, 1997), (Rizzo e Vicisano, 1998) mesmo considerando que, em um cache de rede por exemplo, a seqüência de acessos recebida pelo cache é a composição das seqüências de acesso de dezenas ou milhares de usuários. A localidade existente deve ser explorada. A próxima seção analisa a localidade das seqüências de requisições em função do tamanho e mostra a influência da variabilidade dos tamanhos na localidade.

3.5 Padrão de Acesso em Classes de Requisições

A caracterização do padrão de acesso para a carga completa fornece uma idéia ampla da localidade presente na carga. No entanto, a grande variabilidade dos tamanhos nos impede de avaliar a influência de seqüências de requisições de tamanho similar.

Para estudar a distribuição relativa dos bytes, dos arquivos e das requisições classificamos as requisições de cada carga de acordo com seus tamanhos. As classes foram definidas da seguinte forma:

classe 1	tamanho $< 10^3$ bytes
classe 2	$10^3 \leq$ tamanho $< 10^4$ bytes
classe 3	$10^4 \leq$ tamanho $< 10^5$ bytes
classe 4	$10^5 \leq$ tamanho $< 10^6$ bytes
classe 5	tamanho $\geq 10^6$ bytes

A tabela 3.7 (apresentada no apêndice deste capítulo) mostra a caracterização das seqüências de requisições e dos conjuntos de arquivos por classes de tamanho para todas as cargas. As duas primeiras colunas apresentam a caracterização das requisições, isto é, a fração das requisições e dos bytes em cada classe, enquanto as duas últimas apresentam a fração dos arquivos e bytes únicos em cada classe. Os dados mostram que há uma concentração bastante distinta de referências e bytes nas diversas classes:

- 80% das requisições são para objetos pequenos, das classes 1 e 2;
- 98% das requisições são para objetos das classes 1, 2 e 3;
- entre 75% e 85% dos bytes requisitados são de objetos das classes 3, 4 e 5; estas classes compõem a cauda da distribuição;
- a classe 5, com cerca de apenas 0.2% das requisições, tem de 16% (POP98) a 42% dos bytes requisitados (POP99-zez). A classe 5 de BR tem 2% das requisições e 85% dos bytes requisitados.

A distribuição dos bytes únicos pode ser interpretada como a fração do tamanho de referência do cache para cada carga necessária para armazenar cada classe. Assim, cerca de 0.7% (coluna bytes únicos para as classes 1) do tamanho do cache armazena todos os documentos da classe 1, que corresponde a um número de arquivos entre 13% e 38% do total de arquivos de cada carga. Os arquivos da classe 1 respondem de 15% a 40% de todas as requisições da carga.

A classe 2 contém entre 40% (NASA) e 63% (NLANR e POP98) dos arquivos únicos. Para armazená-los seria necessário um espaço equivalente a valores entre 3% (NASA) e 21% (POP98) do tamanho de referência do cache para estas cargas. Este espaço responderia por um número enorme de requisições, entre 42% e 62%. As duas últimas classes, embora armazenem menos de 2% dos arquivos, ocupam a maior parte do cache, entre 32% (POP98)

e 74% (BR). No entanto, estes arquivos respondem pela maioria de bytes requisitados. A coluna do meio da tabela (média) mostra o tamanho médio dos documentos em cada classe.

Esta diferença significativa na concentração de referências e bytes das classes sugere que a localidade seja diferente nas diversas classes e que a carga completa tenha uma localidade que seja um compromisso das localidades das classes. É esperado que classes de objetos menores tenham melhor localidade do que a carga completa devido à maior popularidade destes objetos.

Para avaliar a localidade em cada classe e compará-la com a localidade da carga completa, escolhemos três cargas de acordo com o gráfico da figura 3.5. As cargas escolhidas representam três níveis de localidade: excelente (BR), boa (U) e regular (POP99-zez). Para cada carga medimos o HR e o BHR dado pelo algoritmo LRU para suas classes e para a carga completa. As medidas estão em função do tamanho do cache, calculado como uma fração do cache infinito para cada experimento (isto é, cada classe). Por exemplo, para a carga que corresponde a classe 1 de U, somamos os bytes de todos os documentos únicos desta classe. Este é o tamanho de referência do cache para esta classe. Os experimentos foram executados para tamanhos de cache de 1, 2, 4, 8, 16, 32 e 64% do tamanho de referência. O mesmo procedimento é repetido para todas as cargas e classes.

Os resultados estão na figura 3.7. Podemos observar que as classes de objetos menores apresentam melhor localidade do que a carga completa, enquanto as classes de objetos maiores apresentam pior localidade. Os resultados das primeiras classes são muito superiores na maioria dos casos. Observando os gráficos de HR, temos que três classes de BR são melhores do que a carga BR completa. O mesmo acontece com as duas primeiras classes de U e com a primeira classe de POP99-zez. Este resultado para HR é bastante positivo porque a maioria das requisições se concentra nas duas primeiras classes. Como estas classes compreendem os documentos menores, elas ocupam um espaço relativamente pequeno. Por isso é possível obter bom desempenho em HR reservando um pequeno espaço do cache para documentos pequenos. Em relação a BHR, a localidade das três primeiras classes é melhor do que a da carga inteira para as três cargas. No entanto, as classes 1 e 2 têm impacto limitado no resultado geral de BHR porque compreendem uma fração pequena dos bytes requisitados. As classes maiores dominam o resultado de BHR.

Estes resultados podem ser atribuídos a dois fatores. A maior popularidade nas classes de arquivos menores contribui para o aumento de HR e BHR nestas classes. A divisão da carga em classes baseadas no tamanho produz novas cargas com uma variabilidade nos tamanhos muito menor do que a variabilidade na carga original. A figura 3.6 mostra a cauda da distribuição de probabilidade empírica dos tamanhos para a carga Portugal e suas classes. Apenas a última classe apresenta um comportamento que mostra evidência de variabilidade. Esta maior homogeneidade nos tamanhos elimina uma "quebra" de localidade que acontece no algoritmo LRU quando há troca de documentos de tamanho muito diferente. Por exemplo, a chegada de um arquivo muito grande pode exigir a retirada de milhares de arquivos pequenos, o que pode alterar significativamente o estado do cache.

Esta análise sugere que a divisão da carga em classes baseadas no tamanho dos documentos pode ser uma boa estratégia para explorar a localidade da carga ao mesmo tempo que minimiza a interferência do tamanho na substituições efetivadas no cache.

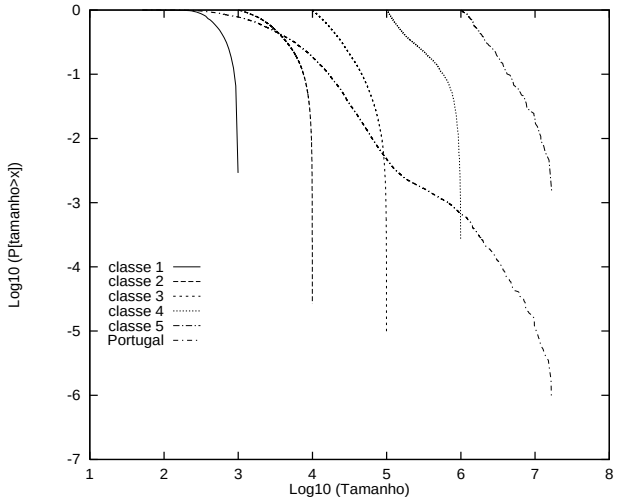

Figura 3.6: Cauda da distribuição de probabilidade dos tamanhos para a carga Portugal e suas classes.

3.6 Potencial e Limitações dos Caches WWW

A presença de caches na WWW mudou o panorama da Internet no mundo. Antes da implementação de caches em vários pontos da rede, os questionamentos sobre a viabilidade da Internet tendo em vista o crescimento do tráfego WWW eram freqüentes (Metcalfe, 1997), (Sedayao, 1994). A implementação generalizada de caches, desencadeada pela campanha *Cache Now!* (Now, 1997), levou a Internet a um novo patamar de operação, ao mesmo tempo que os questionamentos perdiam força.

Quais são os custos e os benefícios da implementacão de caches para o desempenho da WWW? O único custo de desempenho é um provável aumento da latência das requisições que são *miss*. Este aumento pode ser percebido pelo cliente. A presença de caches na WWW não aumenta o volume de tráfego e não é prejudicial a servidores e redes. Portanto, o cache não penaliza a Internet. Por isso todo *hit* é um ganho.

Como quantificar o ganho de desempenho associado a um *hit*? Um *hit* representa um ganho considerável porque beneficia não apenas o cliente que fez a requisição mas todos: usuários individuais, provedores de serviço, provedores de informação e toda a Internet. Os *hits* diminuem a carga na rede e nos servidores. A utilização generalizada de caches leva a melhores tempos médios de resposta para o usuário, e isso se deve não só à presença do arquivo no cache mas também à redução da carga na rede e nos servidores, o que melhora os tempos de *miss*. Os benefícios estão interligados e se diluem na rede dificultando uma avaliação exata. Portanto, quantificar detalhadamente os benefícios de um *hit* ou, generalizando, os benefícios da implantação de caches é uma tarefa muito difícil, se não impossível. Os benefícios são para todos usuários, por isso a dificuldade em quantificá-los.

Em vista dos benefícios gerados por frações mais elevadas de HR e BHR, técnicas que elevam estes valores são de grande interesse. Se as frações de HR e BHR da tabela 3.5 fossem alcançadas, os respectivos sistemas de cache de rede teriam respondido sozinhos a

3 685 138 requisições que poupariam a Internet de um tráfego de cerca de 30 gigabytes de dados, que equivale a mais de 33 horas de transmissão numa linha com capacidade de 2Mbps. Por outro lado, os servidores NASA e BR experimentariam uma queda de 99% no número de requisições servidas e de 98% no número de bytes servidos.

O potencial de contribuição dos caches para a escalabilidade da WWW reside na localidade existente nas cargas e na habilidade do software de gerência em explorar esta localidade. As limitações de desempenho em termos de HR são devidas à característica de larga escala da WWW, isto é, a quantidade e a diversidade de usuários e de informação disponível, bem como à sua própria dinâmica. Documentos novos surgem diariamente e o interesse dos usuários muda constantemente. A diminuição da latência, por sua vez, depende do aumento da taxa de acertos e de melhores tempos de *miss*.

3.7 Conclusão

A variabilidade nos tamanhos dos arquivos e das requisições da WWW, mostrada neste capítulo, tem influência significativa no desempenho dos sistemas de caches destes objetos. O particionamento da carga por intervalos de tamanho gera classes que apresentam características muito distintas como tamanho ocupado e localidade. As análises sugerem que o particionamento do cache é uma estratégia que deve ser pesquisada com o objetivo de explorar características específicas de cada classe e permitir um maior controle sobre o desempenho do cache em relação ao cache não particionado.

3.8 Apêndice

A seguir são apresentadas a tabela 3.7 e a figura 3.7.

BL	req	bytes	média	arquivos	bytes únicos
classe 1	0.2338	0.0083	450	0.2400	0.0067
classe 2	0.5782	0.1641	3575	0.5469	0.1243
classe 3	0.1725	0.3746	27365	0.1933	0.3327
classe 4	0.0142	0.2217	196739	0.0175	0.2278
classe 5	0.0013	0.2361	2206492	0.0023	0.3085

BR	req	bytes	média	arquivos	bytes únicos
classe 1	0.3967	0.0028	392	0.3883	0.0078
classe 2	0.4204	0.0288	3843	0.4577	0.0708
classe 3	0.1527	0.0742	27254	0.1338	0.1795
classe 4	0.0086	0.0496	323146	0.0149	0.1567
classe 5	0.0215	0.8460	2202615	0.0053	0.5853

U	req	bytes	média	arquivos	bytes únicos
classe 1	0.2740	0.0095	416	0.2443	0.0065
classe 2	0.5628	0.1640	3475	0.5473	0.1224
classe 3	0.1514	0.3377	26608	0.1904	0.3211
classe 4	0.0107	0.2084	233347	0.0162	0.2424
classe 5	0.0012	0.3424	3267505	0.0017	0.3076

NASA	req	bytes	média	arquivos	bytes únicos
classe 1	0.2484	0.0069	539	0.1684	0.0017
classe 2	0.4877	0.1099	4355	0.3971	0.0315
classe 3	0.2298	0.3951	33217	0.2943	0.2791
classe 4	0.0319	0.3950	239091	0.1382	0.6281
classe 5	0.0021	0.1270	1190245	0.0020	0.0596

POP98	req	bytes	média	arquivos	bytes únicos
classe 1	0.1764	0.0112	574	0.1440	0.0071
classe 2	0.6195	0.2621	3835	0.6295	0.2074
classe 3	0.1984	0.5038	23021	0.2183	0.4636
classe 4	0.0050	0.1331	240633	0.0073	0.1483
classe 5	0.0006	0.1622	2375845	0.0009	0.1736

Portugal	req	bytes	média	arquivos	bytes únicos
classe 1	0.2196	0.0137	564	0.1572	0.0074
classe 2	0.5931	0.2475	3769	0.6249	0.1969
classe 3	0.1825	0.4506	22306	0.2099	0.4172
classe 4	0.0042	0.1145	246271	0.0068	0.1399
classe 5	0.0007	0.1849	2425073	0.0012	0.2385

POP99-hug	req	bytes	média	arquivos	bytes únicos
classe 1	0.1989	0.0117	605	0.1585	0.0075
classe 2	0.6180	0.2317	3846	0.6295	0.1797
classe 3	0.1772	0.3942	22819	0.2035	0.3743
classe 4	0.0051	0.1186	238610	0.0072	0.1348
classe 5	0.0008	0.2765	3398312	0.0013	0.3037

POP99-zez	req	bytes	média	arquivos	bytes únicos
classe 1	0.1983	0.0072	555	0.1623	0.0051
classe 2	0.5917	0.1404	3613	0.5992	0.1245
classe 3	0.2010	0.3251	24618	0.2264	0.3136
classe 4	0.0071	0.1158	247749	0.0097	0.1297
classe 5	0.0019	0.4246	3385074	0.0024	0.4270

NLANR-bo1	req	bytes	média	arquivos	bytes únicos
part 1	0.1534	0.0069	671	0.1332	0.0061
part 2	0.5925	0.1532	3852	0.6274	0.1623
part 3	0.2414	0.3999	24678	0.2266	0.3646
part 4	0.0115	0.1956	252605	0.0114	0.1698
part 5	0.0012	0.2662	3234596	0.0013	0.2972
NLANR-uc	req	bytes	média	arquivos	bytes únicos
classe 1	0.1544	0.0074	672	0.1335	0.0063
classe 2	0.6115	0.1628	3756	0.6253	0.1622
classe 3	0.2219	0.3850	24478	0.2294	0.3860
classe 4	0.0102	0.1569	216232	0.0105	0.1517
classe 5	0.0020	0.3418	2464358	0.0013	0.2938

Tabela 3.7: Caracterização das seqüências de requisições e dos arquivos por classes de tamanho.

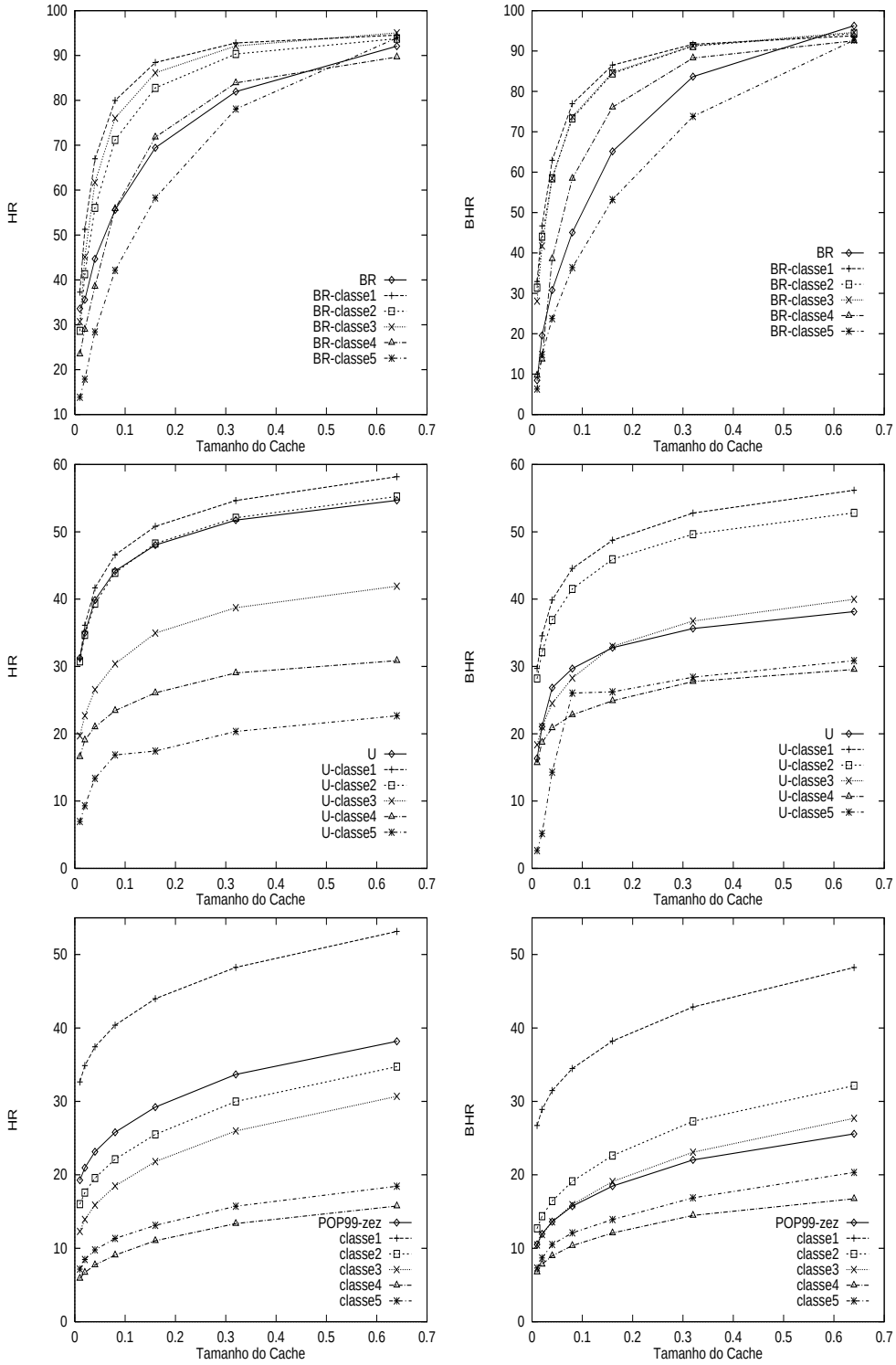

Figura 3.7: Comparação da localidade medida por HR (esquerda) e BHR (direita) entre a carga completa e dividida em classes. Dados para as cargas BR (acima), U (no centro) e POP99-zez (abaixo).

Capítulo 4

Modelo de Particionamento de Espaço para Caches da WWW

4.1 Introdução

Sistemas de cache são amplamente utilizados na WWW atual. Como vimos no capítulo 2, prover bons resultados em HR e BHR são objetivos desejados porém contraditórios em sua implementação, devido ao perfil da carga WWW. Os valores de HR e BHR obtidos pelos caches da Web são resultados das escolhas feitas pelas políticas de reposição, que definem uma única regra para classificar todos os arquivos e decidir quais serão descartados.

A caracterização por classes baseadas em tamanho, apresentada no capítulo 3, mostrou enormes diferenças entre as classes em termos de número de requisições, número de bytes requisitados, concentração e localidade de referências. Considerando esta diversidade na carga, dificilmente o estabelecimento de uma única regra é suficiente para (i) alcançar objetivos contraditórios e (ii) tratar a diversidade e a variabilidade, observados na caracterização das cargas, tendo em vista a preservação da localidade. Em resumo, uma regra única não é suficiente para atender a objetivos divergentes. Portanto, um modelo configurável seria vantajoso pois permitiria aos administradores de cache adequarem o sistema às suas necessidades. Esta é a motivação para a investigação de alternativas.

Este capítulo apresenta o modelo PART, que é um modelo desenvolvido para organizar o espaço de sistemas de cache cujos arquivos apresentem grande variabilidade nos seus tamanhos. O modelo PART é definido e seus objetivos são discutidos. O espaço de soluções que o PART dispõe é revelado através de um modelo de otimização. Este modelo quantifica o impacto da variabilidade do tamanho dos arquivos no desempenho de sistemas de cache. Finalmente, o desempenho do modelo é discutido e são indicadas diretrizes para escolha dos seus parâmetros.

Este capítulo apresenta também a relação matemática entre as métricas HR e BHR. Através desta relação podemos fazer gráficos denominados "mapas de desempenho" do cache que mostram em um único gráfico o desempenho em termos de HR e BHR. Os mapas de desempenho podem ser utilizados para comparar modelos de gerência de espaço com

caches particionados e não particionados e para orientar a busca por melhor desempenho nas métricas HR e BHR.

4.2 Definição de Cache Particionado

Esta é uma proposta de organização do espaço disponível no cache considerando a grande variabilidade no tamanho dos objetos candidatos ao armazenamento. A proposta é dividir o espaço do cache em partições. Cada partição é dedicada ao armazenamento de documentos cujos tamanhos pertencem a um intervalo. Cada intervalo define uma classe de documentos. As classes são em número igual ao número de partições e cobrem todos os tamanhos possíveis de documentos sem sobreposição.

Formalmente, seja um cache de tamanho T dividido em n partições de tamanhos T_i, $1 \leq i \leq n$, não necessariamente iguais, $\sum_{i=1}^{i=n} T_i = T$. Seja I o intervalo dos tamanhos possíveis de documentos e t_j, $0 \leq j < n$, n tamanhos (números inteiros) definidos neste intervalo tal que $t_j < t_{j+1}$ para todo $0 \leq j < n-1$. Além disso, $t_0 = 1$. Então cada partição i, $1 \leq i < n$, recebe documentos de tamanho t_d tal que $t_{j-1} \leq t_d < t_j$, e a partição n recebe documentos de tamanho $t_d \geq t_{n-1}$. As classes são delimitadas pelos valores t_j, conforme mostra a figura 4.1.

Substituições de documentos são feitas apenas entre documentos de uma mesma classe. Este modelo não permite nem a retirada de um documento pequeno para a inserção de um documento grande nem a retirada de um documento grande para a inserção de um documento pequeno. Em resumo, não é possível fazer substituição entre documentos de tamanhos extremos.

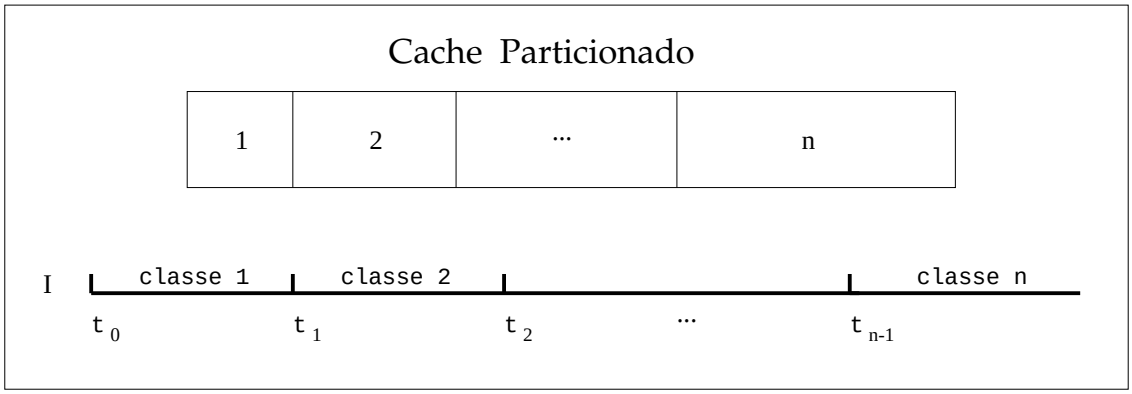

Figura 4.1: Partição do cache e classificação dos documentos

A proposta apresentada coloca a questão da gerência do espaço dos caches em um novo patamar de discussão. Até agora o espaço do cache foi considerado único para todos os documentos, isto é, os documentos concorriam pelo mesmo espaço independente do seu tamanho. A utilização do espaço era definida pela política de reposição. A proposta de dividir o espaço em partições que armazenem classes de documentos de tamanho similar

desdobra a gerência de espaço de caches em dois campos: a organização do espaço e a política de substituição.

4.3 Justificativa para o Modelo PART

4.3.1 Por que particionar por tamanho?

Vamos definir o "estado do cache" como sendo o conjunto de documentos presentes no cache mais a lista ordenada destes documentos. A regra para ordenação é dada pela política de reposição. Por exemplo, a política LRU gera uma lista LRU. A política LFU gera uma lista (ou um *heap*) cuja chave de ordenação é o número de acessos a cada objeto. A lista é atualizada a cada requisição. A atualização consiste na reordenação da lista em caso de *hit* ou na inserção com possibilidade de retirada em caso de *miss*. Nos sistemas tradicionais, o estado do cache é alterado a uma taxa de apenas uma modificação por requisição feita ao cache. Cada inserção envolve no máximo uma retirada. Portanto o estado do cache muda lentamente, é quase estacionário, e a correlação entre o estado do cache no passado imediato e no futuro imediato é alta.

A forma de implementação da reposição nos caches da Web permite a ocorrência de uma alteração significativa no estado do cache, que chamaremos de *cache disruption*, como resultado de uma única inserção. Esta alteração acontece quando a chegada de um documento grande implica na retirada de centenas ou milhares de documentos menores. Diante desta chegada, o estado do cache é consideravelmente alterado. O conjunto de objetos e a lista são diminuídos de centenas ou milhares de documentos em uma única operação cujo objetivo é abrir espaço para o objeto grande. Esta alteração ocorre devido à variabilidade nos tamanhos dos documentos e ao fato de que, nestas políticas, não há nenhuma relação entre o tamanho dos documentos que são armazenados e o tamanho dos documentos que são retirados do cache. Num regime estável, o cache forma seu *working set* e as mudanças devem ser gradativas e lentas, com a menor alteração possível. A retirada em massa de documentos devido a uma única requisição é prejudicial à preservação do estado do cache e à efetividade das políticas.

É importante considerar também a frequência de ocorrência de documentos grandes, que pode gerar um evento de *cache disruption*. Considerando a distribuição de Pareto com $\alpha = 1.3$ e média igual a 13 Kbytes para representar a cauda da distribuição dos tamanhos dos documentos, temos que 0.05% dos arquivos terão tamanho igual ou maior do que 10^6. Se um sistema de cache atende a 200 requisições por segundo então a cada cinco segundos haverá uma requisição para um arquivo grande. Admitindo uma fração de *hits* de 50% para esta faixa de tamanhos, teremos um evento de *cache disruption* a cada dez segundos. Apenas para comparação, numa distribuição exponencial com a mesma média a probabilidade de ocorrer uma observação de valor igual ou maior do que 10^6 é 10^{-33}.

O particionamento por tamanho, proposto pelo PART, elimina a possibilidade de troca entre documentos de tamanhos extremos. Isto evita duas situações indesejadas: o *cache disruption* e a retirada de um documento muito grande (por ser o primeiro na lista de

substituições) diante da chegada de um documento pequeno, liberando espaço maior do que o necessário.

Com o particionamento o estado do cache passa a ser constituído por diversas listas de objetos, uma para cada partição. No cache particionado não há um número grande de objetos envolvidos nas substituições. Não há alteração abrupta no estado do cache. A transição lenta entre os estados do cache é fator importante para a maximização do desempenho da política.

Além disso, para o cache, o tamanho é uma característica intríseca do documento. Se há uma mudança no tamanho de um documento isto significa que o mesmo foi alterado e deve ser novamente requisitado ao servidor. O cache faz esta verificação de consistência e considera o documento desatualizado caso haja alteração de tamanho. O mesmo não acontece com outras características do documento como freqüência, por exemplo.

4.3.2 Objetivos do PART

Nas implementações dos caches da WWW atual, não há nenhuma política que tenha como objetivo obter bom desempenho nas duas métricas, HR e BHR. Também não há, nas políticas conhecidas, variáveis que permitam ajustar o algoritmo para obter melhor desempenho em termos de HR e BHR em função de mudanças na carga ou das necessidades de usuários e administradores. A prática usual é estabelecer um limite superior de tamanho para os arquivos que podem ser armazenados no cache. Esta estratégia gera um aumento no HR e uma diminuição no BHR. Para uma mudança significativa de desempenho a opção é a troca da política de substituição. O modelo PART vem preencher uma lacuna na gerência de espaço dos caches da WWW que é exatamente a possibilidade de configurar o sistema para obter bom desempenho individual em HR ou BHR ou encontrar um ponto de equilíbrio no desempenho de ambas as métricas, sem precisar trocar de política de reposição a cada ajuste.

O PART divide o espaço de soluções (as possíveis escolhas de objetos das políticas de reposição) em grupos com potenciais diferentes para otimizar HR e BHR permitindo a exploração das potencialidades dos grupos. Com a utilização do PART é possível fazer ajustes finos para encontrar o limite de desempenho em termos de uma das métricas ou encontrar uma solução de compromisso que pode ser individual para cada cache, privilegiando o HR ou o BHR. O PART fornece a flexibilidade necessária à experimentação e adequação do cache à sua carga de trabalho, tornando-o com isso mais efetivo.

4.4 Características do Modelo PART

O particionamento do cache apresenta várias características que são discutidas a seguir. O PART reaproxima o problema de cache na WWW do problema de cache tradicional porque elimina a variabilidade extrema dos tamanhos nas substituições, permitindo apenas trocas entre objetos de tamanho similar.

Relação entre tamanho e outras características do arquivo

Como mostrado no capítulo 2, as políticas de reposição dos caches da Web utilizam uma fórmula única que é uma composição de várias características do arquivo como frequência, recentidade e tamanho. Esta fórmula é utilizada para atribuir um valor para cada arquivo. Os arquivos são mantidos ordenados de acordo com esse valor. A lista ordenada indica a ordem de retirada dos arquivos para substituição. No entanto, estas características que compõem a fórmula são difíceis de serem comparadas entre si. Por exemplo, é melhor manter um arquivo de 10 Mbytes com um número razoável de referências ou manter 1024 arquivos de 10 Kbytes? O modelo PART permite que o parâmetro tamanho do arquivo seja tratado de forma ortogonal aos demais parâmetros pertinentes ao desempenho, por exemplo, frequência e recentidade. No contexto do modelo, o tamanho é tratado de forma independente e eliminado desta comparação. As características de frequência e recentidade de um arquivo são comparadas com os mesmos dados de outros arquivos de tamanho similar. Assim, há uma real comparação de localidade sem a influência do tamanho.

Cada partição pode ser tratada como um cache separado

As partições podem ser vistas como caches separados, dedicados a classes específicas de documentos definidas pelo tamanho. Isto permite implementar tratamento diferenciado para as diversas classes de documentos, possibilitando a exploração de características peculiares a um conjunto de documentos de tamanho pouco variável, isto é, da mesma classe. Como exemplo, políticas de substituição diferentes podem ser consideradas para cada classe/partição. Se as partições são implementadas em servidores de um sistema distribuído, cada servidor pode ser ajustado de acordo com as características da carga recebida. Assim, servidores de classes de arquivos pequenos e grandes podem ter seus sistemas de E/S ajustados para otimizar a leitura e escrita de seus arquivos. Por exemplo, sistemas de arquivos que tratam de arquivos grandes obtêm melhor taxa de serviço e melhor eficiência de armazenamento se os blocos de transferência forem grandes (Loukides, 1990). Como regra geral, manter arquivos de tamanho similar torna mais fácil a escolha de opções de configuração apropriadas para o sistema de arquivos. Além disso, se o sistema de cache é distribuído, o particionamento pode ser feito de forma a balancear a carga entre os servidores, conforme será mostrado na seção 4.6.

Diminuição do efeito da variabilidade dos tamanhos

Há uma diminuição substancial da variância dos tamanhos dos arquivos em cada classe, em comparação com a variância do conjunto de arquivos únicos (seção 3.5). Isto elimina o problema da retirada de um número grande de arquivos ante à chegada de um arquivo muito grande e a liberação excessiva de espaço que ocorre quando um arquivo grande é substituído por um pequeno. As consequências da distribuição de cauda pesada ficam limitadas à última partição.

O estado do cache é preservado

A homogeneidade dos tamanhos em cada partição elimina trocas entre objetos de tama-

nhos extremos, preservando ao máximo o estado do cache. Cada partição pode ser vista como um cache com carga de trabalho correspondente à classe atendida. Cada partição terá seu próprio estado do cache e a política de reposição utilizada procurará tirar proveito da localidade da respectiva carga. Assim, a localidade é valorizada uma vez que as substituições vão ocorrer em função da localidade da classe e não em função do tamanho. O particionamento permite que as políticas de reposição explorem a localidade, liberando-as da tarefa de gerência do espaço.

Flexibilidade

A utilização deste modelo permite que o usuário tenha um controle sobre o desempenho, inexistente nos modelos anteriores. O compromisso de desempenho entre HR e BHR pode ser testado e explorado através da variação dos parâmetros do modelo. Melhor desempenho em HR pode ser alcançado aumentando-se o tamanho das partições dedicadas a documentos menores. Melhor desempenho em BHR implica num aumento do tamanho das partições que armazenam os maiores documentos. Adicionalmente, políticas específicas para otimização de cada métrica podem ser utilizadas em diferentes partições. O tamanho das partições pode ser ajustado dinamicamente em função dos resultados de desempenho do cache e em função de mudanças no perfil da carga. Este ajuste pode ser feito durante a operação do sistema.

Fragmentação

A desvantagem do particionamento é a fragmentação, que pode provocar substituições que seriam desnecessárias no cache não particionado. A fragmentação ocorre quando a soma dos espaços vazios é suficiente para armazenar um documento e, no entanto, uma substituição terá de ser feita pois o espaço está fragmentado entre as partições, ou está em uma partição diferente. O efeito da fragmentação e seus custos serão apresentados junto com a descrição dos resultados dos experimentos, no capítulo 5.

4.5 Funcionalidade do PART

Esta seção mostra as hipóteses que o PART assume em relação à carga de trabalho e ao funcionamento do sistema de cache que o implementa. Esta seção também mostra, através de um modelo de otimização, que o conjunto de arquivos que o cache deve manter é muito distinto dependendo da métrica escolhida na otimização, HR ou BHR. Essa distinção representa o espaço de soluções para o PART.

4.5.1 Hipóteses

A utilização do PART pressupõe que (*i*) a carga de trabalho seja caracterizada pela grande variabilidade no tamanho dos objetos, ou seja, que a distribuição dos tamanhos dos objetos seja uma distribuição de cauda pesada; (*ii*) o tamanho dos objetos seja conhecido antes do armazenamento.

Como já demonstrado no capítulo 3, a carga da WWW apresenta uma grande variabilidade no seu tamanho. Além disso, o tamanho dos objetos é inversamente correlacionado com a popularidade. Há mais requisições para objetos pequenos e maior número de requisições repetidas para estes objetos e, por outro lado, há menos requisições para objetos maiores. No entanto, os objetos pequenos representam apenas uma pequena fração dos bytes transmitidos. Deve-se ressaltar que o modelo PART é apropriado para todos os sistemas de cache que apresentem esta característica. A WWW é um exemplo.

Os sistemas de cache de cliente e de rede da WWW são implementados em dois níveis, o cache de memória e o de disco. Tipicamente, o cache de memória é destinado a armazenar a lista de metadados dos objetos presentes no cache de disco, os objetos mais recentemente acessados e os objetos com grande frequência de acesso. Todos os objetos candidatos ao armazenamento são inicialmente copiados no cache de primeiro nível. Posteriormente os arquivos são copiados no cache de disco. No caso do PART, a partição na qual um arquivo deve ser armazenado é identificada pelo tamanho do arquivo. Esta identificação pode ser feita imediatamente após o tamanho do objeto ser identificado. Para objetos pequenos, isto significa que o objeto será primeiramente armazenado no cache de primeiro nível. Para objetos grandes, pertencentes à última partição, a transferência para o disco pode começar tão logo o tamanho do objeto atinja o limite inferior da última classe.

4.5.2 Otimização de HR e BHR: Espaço para Soluções de Compromisso

O PART determina que o recurso espaço em disco deve ser dividido entre as classes em vez de ser compartilhado. Esta estratégia permite que seja aplicada a cada classe uma política de reposição que procure otimizar o desempenho de uma das métricas, HR ou BHR. Com isso é obtido um compromisso de desempenho do sistema de cache em relação a ambas as métricas.

No caso de sistemas de cache onde todos os objetos têm tamanhos iguais o particionamento não se justifica pois o HR é exatamente igual ao BHR. Quando os objetos têm tamanhos diferentes, manter bom desempenho em termos de HR e BHR torna-se uma tarefa difícil pois as métricas são contraditórias. Isto já foi observado na literatura (Arlitt e Williamson, 1996). Quando a popularidade é inversamente correlacionada com o tamanho, como é o caso da WWW, esta dificuldade é enfatizada. A estratégia para aumentar HR, que é manter o maior número de documentos no cache, é contrária à estratégia para aumentar o BHR, que é manter os maiores documentos no cache.

Para provar este comportamento vamos analisar o problema das escolhas dos objetos em um cache sob a ótica de um problema de otimização, o problema da mochila. Vamos também redefinir as métricas HR e BHR (redefinição válida apenas para esta seção). As métricas serão baseadas no número de referências feitas a cada documento. Embora esta seja uma simplificação de um cache real, ela serve aos nossos propósitos. O objetivo é mostrar que as métricas HR e BHR são contraditórias, isto é, as escolhas feitas para otimizar uma delas não implicam necessariamente na otimização da outra.

Suponha um conjunto de n arquivos, cada um com tamanho s_i e número de referências r_i correspondente, $i = 1, \ldots, n$. O problema consiste em determinar quais arquivos devem ser escolhidos para compor o cache quando o objetivo é maximizar independentemente HR e BHR. Este é o problema da mochila 0-1 (Pisinger, 1995), a saber, o problema de escolher um subconjunto de n itens que maximize o ganho sem que o tamanho do conjunto exceda uma capacidade c pré-determinada. A substituição de objetos não é considerada. Este problema pode ser formulado como o seguinte problema de otimização:

$$maximizar\ \mathrm{HR} = maximizar \sum_{i=1}^{n} r_i x_i$$

$$sujeito\ a \sum_{i=1}^{n} s_i x_i \leq c,$$

$$x_i \in \{0,1\}, \quad i = 1, ..., n,$$

ou

$$maximizar\ \mathrm{BHR} = maximizar \sum_{i=1}^{n} r_i s_i x_i$$

$$sujeito\ a \sum_{i=1}^{n} s_i x_i \leq c,$$

$$x_i \in \{0,1\}, \quad i = 1, ..., n,$$

onde x_i é uma variável binária que valerá 1 se o item i for incluído na mochila, e 0, caso contrário.

O resultado da maximização indica qual deve ser a composição de um cache de tamanho c para uma dada carga para que seja obtido o valor máximo em relação a cada métrica definida. Para comparar as soluções ótimas para o cache (mochila) quanto às duas métricas, HR e BHR, utilizamos uma implementação baseada em programação dinâmica (Pisinger, 1997) para o problema da mochila 0-1. As cargas reais já descritas no capítulo anterior foram utilizadas nos testes. O programa foi executado duas vezes para cada tamanho de cache, uma para cada maximização. O resultado de cada execução é o conjunto de documentos que atende à maximização correspondente. O objetivo é verificar a fração de documentos que é comum às duas soluções e o espaço para soluções de compromisso de desempenho entre as otimizações de HR e BHR.

Para analisar os resultados das otimizações vamos definir os seguintes conjuntos: S(HR) é o conjunto de documentos escolhidos na otimização de HR, S(BHR) é o conjunto de documentos escolhidos na otimização de BHR. S(HR) $\cup$ S(BHR) e S(HR) $\cap$ S(BHR) são, respectivamente, a união e a interseção dos conjuntos resposta. O gráfico da figura 4.2 apresenta a cardinalidade desses quatro conjuntos em função do tamanho do cache (fração do tamanho de referência do cache para a carga). Os resultados são para a carga BL. O

gráfico mostra que a otimização de HR gera conjuntos S(HR) com cardinalidade muito superior à dos conjuntos otimizados para BHR. A partir dos dados deste gráfico podemos calcular qual é a fração dos documentos escolhidos pelas duas otimizações que está presente nas duas soluções. Para isso calculamos $|S(HR) \cap S(BHR)|$ / $|S(HR) \cup S(BHR)|$, para cada tamanho. Apenas uma pequena porcentagem dos documentos presentes na união das duas soluções é comum aos dois conjuntos. Esta porcentagem varia entre 4% e 15% para tamanhos de cache entre 1% e 50% do tamanho de referência do cache para a respectiva carga. Isto indica que há um amplo espaço para obtenção de desempenho que seja um compromisso entre a maximização de HR ou BHR.

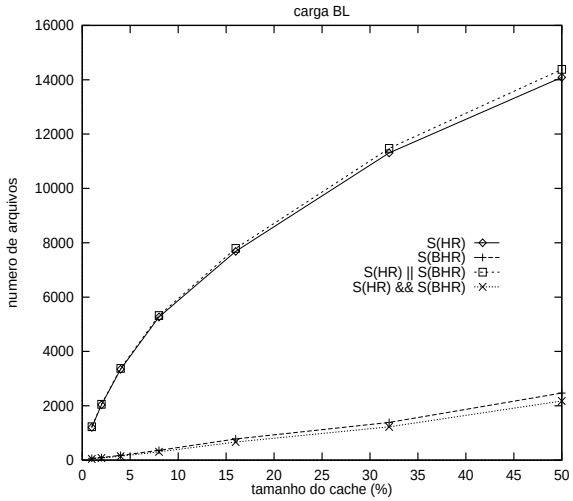

Figura 4.2: Tamanho das soluções para maximização de HR e BHR de acordo com o problema da mochila, em função do tamanho do cache. Os símbolos "||" e "&&" representam, respectivamente, as operações **or** e **and**.

Há uma enorme diferença entre o número de documentos escolhidos na maximização de HR e na maximização de BHR. Lembrando que o espaço do cache para ambos os testes é o mesmo, temos que o pequeno número de documentos dado por $|S(BHR) - (S(HR) \cap S(BHR))|$ está ocupando o mesmo espaço do grande número de documentos que estão incluídos na maximização de HR e não são comuns aos dois conjuntos, isto é, $|S(HR) - (S(HR) \cap S(BHR))|$. Por exemplo, para um cache de tamanho equivalente a 32% do tamanho de referência para a carga, o conjunto S(HR) tem 11310 elementos enquanto o conjunto S(BHR) tem 1389, $S(HR) \cup S(BHR)$ tem 11475 e $S(HR) \cap S(BHR)$ tem 1224. Isto implica que os 10086 documentos de S(HR) e os 165 documentos de S(BHR) que não são comuns a ambos os conjuntos ocupam o mesmo espaço. Este resultado evidencia que (*i*) a variabilidade nos tamanhos dos objetos tem consequências importantes para os sistemas de cache; o exemplo acima mostra que a definição da métrica objetivo é fundamental nas decisões tomadas pelo algoritmo de gerência de espaço; (*ii*) a maximização de HR implica na escolha de um maior número de documentos pequenos, (*iii*) a maximização de BHR implica na escolha de documentos significativamente maiores, e (*iv*) há um grande espaço para a escolha de

uma solução de compromisso entre as duas métricas. Embora este compromisso já fosse conhecido em termos qualitativos, o procedimento adotado aqui mostra como quantificar o espaço de soluções de compromisso para uma carga específica.

Pelo resultado descrito, vimos que há um conjunto pequeno de documentos que está presente nas duas soluções. Quais são estes documentos? Para identificá-los plotamos cada documento do conjunto comum, $S(HR) \cap S(BHR)$, e os documentos que estão exclusivamente em S(HR) e em S(BHR). Cada documento é representado por um ponto cuja abscissa é seu tamanho e a ordenada é o número de referências. A figura 4.3 mostra os documentos destes conjuntos para caches de tamanho 1% para as cargas BL e NLANR-bo1. Estes gráficos mostram a relação entre o tamanho, o número de referências e a pertinência a um ou ambos os conjuntos. Documentos pequenos e pouco referenciados aparecem em S(HR) mas nunca em S(BHR). Documentos pequenos porém muito referenciados aparecem nas duas soluções. Documentos grandes podem também ser divididos em dois conjuntos: os pouco referenciados, que aparecem no conjunto BHR e os muito referenciados que aparecem nos dois conjuntos.

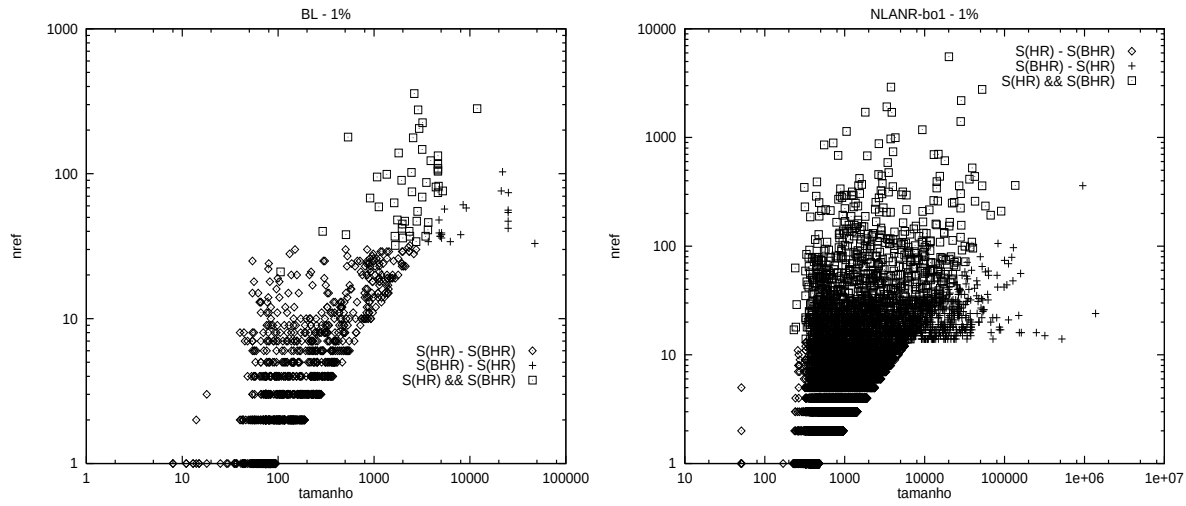

Figura 4.3: Soluções para HR e BHR plotadas em três conjuntos: arquivos exclusivos de cada solução e arquivos comuns. Ambos os eixos estão em escala logarítmica. Cargas BL (esquerda) e NLANR-bo1 (direita).

Esta seção formalizou o compromisso existente entre as métricas HR e BHR e mostrou que há um espaço grande para soluções intermediárias entre as otimizações de HR e BHR.

4.6 Escolha dos Parâmetros do PART

Nessa seção vamos discutir as diretrizes para escolha dos parâmetros do modelo PART, a saber: número de partições, definição das classes e tamanho das partições. A escolha deve ser feita nesta ordem pois a definição das classes depende do número de partições e o

tamanho das partições, por sua vez, depende da definição das classes. A definição final é a da política de reposição e será discutida no capítulo 5.

A escolha dos parâmetros deve ser baseada em duas variáveis: a definição dos objetivos, isto é, se se quer otimizar preferencialmente HR ou BHR, ou se é desejado um compromisso entre estas métricas; e a caracterização da carga.

Dada uma carga e definido o objetivo, é possível indicar diretrizes para escolha dos parâmetros. No entanto, como não são conhecidos modelos para predizer os valores de HR e BHR, a determinação dos valores adequados para os parâmetros do PART só pode ser obtida através de simulação ou experimentação com o sistema real. O capítulo 5 apresenta experimentos baseados em simulação que vão mostrar a sensibilidade do modelo à definição dos objetivos, à variação dos parâmetros e da carga.

4.6.1 Número de Partições

A primeira escolha deve ser a do número de partições. O particionamento tem como objetivo minimizar a variabilidade dos tamanhos dos objetos envolvidos numa substituição. Os maiores ganhos deverão ser obtidos com os primeiros particionamentos. Particionamentos sucessivos vão aumentando os benefícios porém em escalas cada vez menores. A idéia é trabalhar com poucas partições, de duas a cinco. Estas partições podem ser implementadas em um mesmo servidor ou em um ambiente distribuído. Maior variabilidade na carga é indicação para a definição de um número maior de partições.

4.6.2 Divisão da Carga em Classes

A definição das classes determina a fração de documentos em cada classe, isto é, a fração da carga que vai ser atendida pela partição correspondente à classe. A definição das classes pode ser feita com o objetivo de dividir a carga igualmente entre as partições. Em um sistema de cache distribuído a divisão pode ser utilizada para balancear a carga entre computadores de modo que cada um receba carga compatível com seu poder de processamento.

Como exemplo, vamos considerar a situação na qual o cache é implementado por um conjunto de computadores onde cada máquina (*host*) implementa uma partição, isto é, armazena arquivos de uma classe. Assumimos que todos os *hosts* tem o mesmo poder de processamento e que a carga de cada tarefa é dada pelo tamanho da requisição a ser servida. A carga total é a quantidade de E/S feita nos discos. Os *hits* correspondem à leitura e os *misses* à escrita. Portanto, o número total de bytes requisitados (e servidos) é a carga do sistema de E/S. O problema consiste em particionar a carga de modo que cada partição sirva quantidades iguais de bytes. A definição das classes deve ser feita de modo a balancear a carga de E/S nos diversos *hosts*.

O mecanismo para obter o balanceamento da carga nos *hosts* é usar a distribuição dos tamanhos das tarefas para definir os pontos de corte para as classes, isto é, os pontos t_j da figura 4.1. O tamanho das tarefas segue a distribuição limitada de Pareto. A idéia é simples: basta definir as classes de tal forma que a carga de E/S direcionada a cada *host*

seja a mesma. O seguinte teorema mostra como os pontos que definem as classes podem ser obtidos.

Teorema 1 [1]

A função de probabilidade de massa para a distribuição de Pareto limitada (Bounded Pareto), $B(\alpha, k, p)$, é definida por:

$$f(x) = \frac{\alpha k^{\alpha}}{1 - (\frac{k}{p})^{\alpha}} x^{-\alpha-1} \quad k \leq x \leq p.$$

Seja X uma variável aleatória que segue a distribuição de Pareto $B(\alpha, k, p)$. Nessa distribuição α é o grau de variabilidade, $0 < \alpha \leq 2$. k e p são constantes positivas que representam, respectivamente, o menor e o maior valor que pode ser assumido pela distribuição, $k < p$. h é o número de hosts, que é igual ao número de partições. Seja $\mathbf{E}\{X\}$ o primeiro momento da distribuição.

Então, se $\alpha \neq 1$, definindo

$$x_i = \left(\frac{(h-i)}{h} k^{1-\alpha} + \frac{i}{h} p^{1-\alpha} \right)^{\frac{1}{1-\alpha}}, \quad para\ i = 0, \ldots, h,$$

temos $k = x_0 < x_1 < x_2 < \ldots < x_{h-1} < x_h = p$, e

$$\int_{x_0=k}^{x_1} x.f(x) = \int_{x_1}^{x_2} x.f(x) = \int_{x_2}^{x_3} x.f(x) = \ldots = \int_{x_{h-1}}^{x_h=p} x.f(x) = \frac{E\{X\}}{h}$$

Se $\alpha = 1$, a definição correspondente é

$$x_i = k \left(\frac{p}{k} \right)^{\frac{i}{h}}, para\ i = 0, \ldots, h.$$

A prova é apresentada no apêndice deste capítulo. Esse teorema define os pontos de corte dados por x_i, $i = 0, \ldots, h$, sendo $k = x_0 < x_1 < x_2 < \ldots < x_{h-1} < x_h = p$. Cada par adjacente de pontos define uma classe.

Na prática a distribuição dos tamanhos é fácil de ser obtida através da manutenção de um histograma de todos os tamanhos das tarefas executadas durante um período de tempo. Os pontos de corte são calculados uma vez, dada a distribuição, e então é necessário apenas que um escalonador de tarefas mantenha o registro destes pontos. No entanto, um benefício é que se a distribuição dos tamanhos muda significativamente no tempo, os pontos de corte podem ser recalculados e registrados para o sistema se adaptar à mudança na carga.

Com a limitação dos tamanhos dos arquivos servidos em cada *host*, o PART reduz a variância da distribuição dos tamanhos das tarefas em relação ao que cada *host* receberia se a distribuição fosse aleatória. Essa redução na variância dos tamanhos das tarefas gera um melhoramento no desempenho do *host*. Além disso, quando a carga está balanceada, a maioria das tarefas é enviada para os *hosts* que servem os menores arquivos e que são os que apresentam o melhor desempenho. Em (Harchol-Balter *et al.*, 1999) é apresentada uma prova de que, nestas condições, os arquivos menores experimentarão o menor tempo na fila.

[1] Elaborado em conjunto com Mark Crovella e Mor Harchol-Balter (Harchol-Balter *et al.*, 1999).

4.6.3 Tamanho das Partições

O tamanho das partições é estreitamente ligado à definição de classes. Uma vez definidas as classes, o tamanho das partições é escolhido com base na caracterização da carga de cada partição e também com base no objetivo. A caracterização da carga indica qual é a fração de bytes únicos que cada classe ocupa. Maior interesse em HR indica que as partições que armazenam arquivos pequenos devem ter um maior espaço relativo. Por outro lado, maior interesse em BHR indica que maior espaço deve ser dedicado às partições que armazenam arquivos maiores. O tamanho das partições deve ser reconfigurado em função das mudanças no perfil da carga ou em função de mudanças no objetivo.

4.7 Relação Analítica entre HR e BHR

O desempenho conjunto do cache particionado em n partições é dado por:

$$\mathrm{HR} = \sum_{i=1}^{n} H_i . f_i$$

onde H_i é a fração de hits na partição i e f_i é a fração de requisições destinadas à partição i, $1 \leq i \leq n$. Então:

$$H_i = \frac{\text{número de requisições servidas pela partição } i}{\text{número requisições feitas à partição } i}$$

$$f_i = \frac{\text{número de requisições feitas à partição } i}{\text{total de requisições feitas ao cache}}$$

$$H_i . f_i = \frac{\text{número de requisições servidas pela partição } i}{\text{total de requisições feitas ao cache}}$$

$$\mathrm{HR} = \sum_{i=1}^{n} H_i . f_i = \frac{\text{número de requisições servidas pelo cache}}{\text{total de requisições feitas ao cache}}$$

Para aumentar HR deve-se aumentar o número de arquivos servidos pelo cache. Para uma mesma carga e mesmo tamanho de cache, uma forma de fazer isso é aumentar o número de arquivos armazenados no cache, mantendo preferencialmente os arquivos pequenos. A fração de *hits* cresce assintoticamente com o número de arquivos armazenados (Breslau *et al.*, 1999). Utilizando o PART é possível controlar o número de arquivos armazenados no cache e aumentar ou diminuir esse número, o que é feito pela variação do tamanho das partições.

Analogamente, o desempenho em BHR é dado por:

$$\text{BHR} = \sum_{i=1}^{n} B_i.b_i = \frac{\text{número de bytes servidos pelo cache}}{\text{total de bytes requisitados ao cache}}$$

onde B_i é a fração de *hits* em bytes na partição i e b_i é a fração de bytes requisitados à partição i, $1 \leq i \leq n$.

A relação entre HR e BHR é fundamental para o entendimento do comportamento de cada modelo de gerência de espaço (composto pelo modelo de particionamento mais a política de reposição). Embora as duas métricas sejam consideradas fundamentais para o desempenho do cache, ainda não há na literatura uma fórmula que estabeleça a relação entre elas. Esta relação é estabelecida pelo teorema enunciado a seguir.

Teorema 2 *Dado que*

$$HR = \frac{\textit{número de requisições servidas pelo cache}}{\textit{número de requisições feitas ao cache}}$$

e

$$BHR = \frac{\textit{número de bytes servidos pelo cache}}{\textit{número de bytes requisitados ao cache}}$$

então

$$\frac{HR}{BHR} = \frac{\textit{tamanho médio das requisições}}{\textit{tamanho médio dos hits}} \tag{4.1}$$

A prova é apresentada no apêndice deste capítulo.

Este teorema mostra que se todas as requisições são do mesmo tamanho então $HR = BHR$. Este é o caso dos caches tradicionais. Além disso, quando não há correlação entre a freqüência de requisições e o tamanho dos objetos requisitados, então $HR/BHR \approx 1$. Se o tamanho médio das requisições é maior do que o tamanho médio dos *hits*, então $HR > BHR$. Isto é, em geral, o que acontece nos Web-caches, onde observa-se que os arquivos mais populares são menores do que a média dos arquivos requisitados. Se os *hits* são preferencialmente para arquivos maiores do que a média das requisições, então $HR < BHR$.

Portanto, para aumentarmos o BHR para uma mesma carga e mesmo tamanho de cache precisamos aumentar o tamanho médio dos *hits*, isto é, preservar no cache arquivos maiores que possam gerar *hits*. O gráfico da figura 4.4 mostra a relação entre HR/BHR e o tamanho médio dos *hits*, dada pela fórmula apresentada no teorema 2, considerando alguns tamanhos médios típicos para as cargas.

Este teorema é válido para todos os tipos de cache, particionados ou não, geridos por qualquer política de reposição. Embora ele seja válido para caches com objetos de tamanho constante, sua aplicação neste caso não é interessante pois HR=BHR sempre.

4.8 Mapas de Desempenho

A partir do teorema 2 podemos gerar gráficos para as métricas HR e BHR para cada carga específica. O único dado necessário da carga é o tamanho médio das requisições.

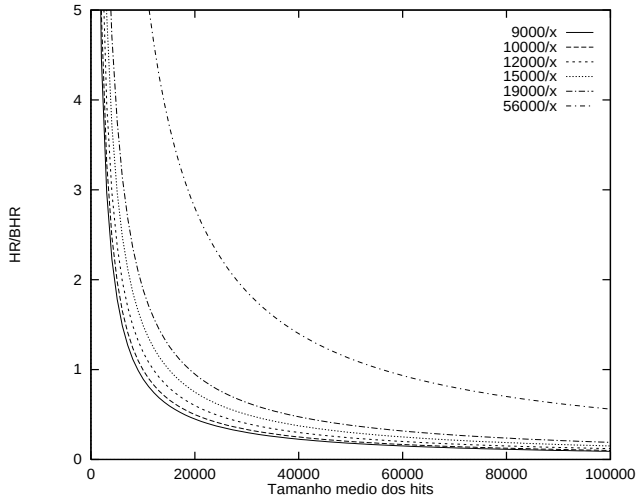

Figura 4.4: Relação entre HR/BHR e o tamanho médio dos *hits* para cargas de tamanhos médios típicos.

Portanto, cargas com médias de tamanho iguais terão gráficos iguais. Para gerar o mapa de desempenho para uma determinada carga basta plotar a seguinte fórmula

$$HR = BHR * \frac{\text{tamanho médio das requisições}}{\text{tamanho médio dos hits}}.$$

No eixo y é plotado HR, e no eixo x, o tamanho médio dos *hits*. O gráfico apresenta curvas que representam diversos valores para BHR.

A figura 4.5 apresenta dois mapas de desempenho, um para a carga BR, cujo tamanho médio das requisições é de 56074 bytes, e o outro para a carga NLANR-uc, cujo tamanho médio das requisições é 14105 bytes. Em cada mapa há curvas rotuladas para BHR = 0.1 e BHR = 1.0. As outras curvas representam os valores intermediários de BHR, com incrementos sucessivos iguais a 0.1.

A principal conclusão obtida a partir do mapa de desempenho é que o desempenho dos sistemas de gerência de espaço dos caches, que incluem as políticas de reposição, é diretamente relacionado com o tamanho médio dos *hits*. Valores altos para HR só são alcançados com médias menores para os *hits*. Médias maiores tornam mais fácil a obtenção de melhores valores para BHR. Pelos mapas vemos que, fixada uma curva para BHR, pequenas variações em torno de valores menores do eixo x geram alterações significativas no HR. Por outro lado, fixando um valor em HR, o BHR só é aumentado significativamente quando há um grande aumento no tamanho médio dos *hits*. Conclui-se que o desempenho de um cache é determinado por suas escolhas de objetos que geram *hits*.

Dada a característica da carga WWW, composta de um número grande de requisições pequenas e populares e um número pequeno de requisições grandes, podemos intuir que é fácil aumentar HR. Como a maioria das requisições é para objetos pequenos, basta manter preferencialmente este tipo de objeto no cache. Ao armazenar os arquivos menores teremos mais arquivos no cache e aumentaremos a probabilidade de *hit*. Por outro lado, aumentar BHR parece ser muito mais difícil. Em primeiro lugar, há menos arquivos grandes e menos

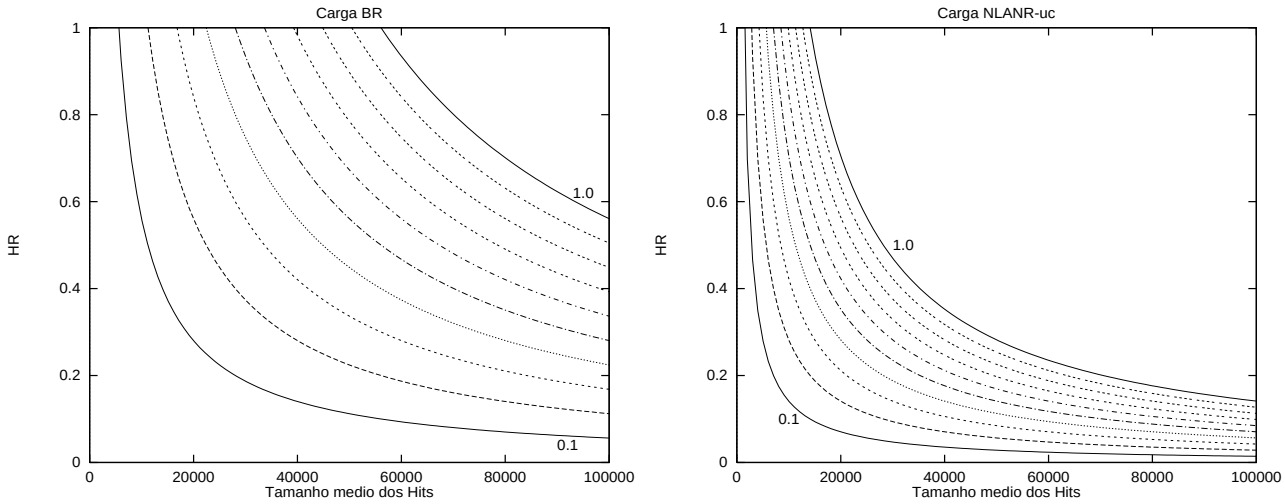

Figura 4.5: Mapas de desempenho para as cargas BR e NLANR-uc.

requisições repetidas para estes arquivos. Portanto, o BHR está muito mais limitado pelas características da carga. É preciso ter *hits* para os arquivos grandes e estes são eventos mais raros do que *hits* para arquivos pequenos. Além disto, armazenar estes arquivos significa excluir muitos arquivos pequenos. A utilização de mapas de desempenho será discutida no capítulo 5.

4.9 Conclusão

Neste capítulo apresentamos o modelo PART e discutimos extensamente suas características. Este modelo tem como pressuposto que os tamanhos dos objetos a serem armazenados apresentem grande variabilidade. O modelo foi projetado com os objetivos de (i) diminuir a diferença de tamanho entre os objetos trocados, minimizando o efeito da variabilidade dos tamanhos na construção do *working set* do cache, preservando seu estado, e (ii) permitir maior controle sobre o desempenho do cache. O particionamento diminui a diferença de tamanho dos arquivos envolvidos numa substituição, o que contribui para preservar o estado do cache construído pela política de reposição, que procura refletir a localidade da carga.

Através do particionamento é possível controlar o desempenho do cache. Provamos que o desempenho em HR e BHR é função do tamanho médio dos arquivos requisitados e do tamanho médio dos *hits*. Enquanto o tamanho médio das requisições é característica inerente à carga, o tamanho médio dos *hits* é resultado das escolhas feitas pelas políticas de reposição. O controle de desempenho é feito através dos parâmetros do PART. O desempenho é limitado pelos valores máximos de HR e BHR possíveis tendo em vista a carga, o tamanho do cache e o compromisso entre as duas métricas.

Neste capítulo apresentamos também um resultado novo que é a relação entre as métricas HR e BHR. Através dessa relação podemos contruir mapas de desempenho que

mostram com clareza as opções dos modelos de gerência de espaço. Estes mapas podem também auxiliar o projeto de novas políticas de reposição.

4.10 Apêndice

4.10.1 Prova do Teorema 1

Prova: Demonstramos que quando

$$x_i = \left(\frac{(h-i)}{h} k^{1-\alpha} + \frac{i}{h} p^{1-\alpha} \right)^{\frac{1}{1-\alpha}}, para\ i = 0, \ldots, h,$$

então o trabalho alocado para o i-ésimo *host* é exatamente $\frac{E\{X\}}{h}$, para todo i, assumindo $\alpha \neq 1$.

$$\int_{x_i-1}^{x_i} x.f(x)dx = \int_{x_i-1}^{x_i} \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha} dx$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha} \int_{x_i-1}^{x_i} x^{-\alpha} dx$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha (1-\alpha)} . \left(x_i^{1-\alpha} - x_{i-1}^{1-\alpha} \right)$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha (1-\alpha)} . \frac{(-k^{1-\alpha} + p^{1-\alpha})}{h}$$

$$= \frac{E\{X\}}{h}$$

Se $\alpha = 1$, então definindo

$$x_i = k(p/k)^{(i/h)}, \quad para\ i = 1, \ldots, h$$

temos

$$\int_{x_i-1}^{x-i} x.f(x)dx = \int_{x_i-1}^{x_i} \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha} dx$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha} \int_{x_i-1}^{x_i} x^{-\alpha} dx$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha (1-\alpha)} . (ln(x_i) - ln(x_{i-1}))$$

$$= \frac{\alpha k^\alpha}{1-(k/p)^\alpha (1-\alpha)} . \left(\frac{1}{h} . (ln(p) - ln(k)) \right)$$

$$= \frac{E\{X\}}{h}$$

4.10.2 Prova do Teorema 2

Prova: Considere uma seqüência de n requisições para um cache. Cada requisição i, $1 \leq i \leq n$, é para um objeto j, $1 \leq j \leq m$. Seja x_i uma variável que assume valores $\{0, 1\}$ que indicam se o objeto j pedido na i-ésima requisição está presente ($x_i = 1$) ou não ($x_i = 0$) no cache. Seja s_i o tamanho do objeto j associado à requisição i. Então podemos definir HR e BHR como:

$$HR = \frac{\sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} 1}$$

e

$$BHR = \frac{\sum_{i=1}^{n} x_i * s_i}{\sum_{i=1}^{n} s_i}.$$

Então:

$$\frac{HR}{BHR} = \frac{\frac{\sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} 1}}{\frac{\sum_{i=1}^{n} x_i * s_i}{\sum_{i=1}^{n} s_i}}$$

$$= \frac{\sum_{i=1}^{n} x_i}{n} * \frac{\sum_{i=1}^{n} s_i}{\sum_{i=1}^{n} x_i * s_i}$$

$$= \frac{\sum_{i=1}^{n} s_i}{n} * \frac{\sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} x_i * s_i}$$

onde $\frac{\sum_{i=1}^{n} s_i}{n}$ é o tamanho médio das requisições e a razão $\frac{\sum_{i=1}^{n} x_i * s_i}{\sum_{i=1}^{n} x_i}$ é o tamanho médio dos *hits*. Portanto,

$$\frac{HR}{BHR} = \frac{\text{tamanho médio das requisições}}{\text{tamanho médio dos hits}}$$

Capítulo 5

Estudo de Desempenho do Cache Particionado

O objetivo deste capítulo é apresentar diversas análises de resultados de experimentos com o modelo de cache particionado. Utilizando simulações, o modelo PART foi analisado e comparado com o modelo não particionado buscando não apenas medir o desempenho em termos das métricas relevantes mas também identificar o efeito e as conseqüências da variabilidade dos tamanhos dos arquivos na operação e no desempenho dos caches. Os resultados obtidos mostram como esta variabilidade interfere no desempenho do modelo não particionado e como o PART consegue minimizar seus efeitos.

As cargas de trabalho utilizadas nos experimentos são as mesmas descritas no capítulo 3. Os resultados são apresentados para duas ou três cargas distintas, dependendo do experimento. Para um mesmo experimento procuramos utilizar cargas com variabilidades diversas em relação aos tamanhos dos objetos, conforme a caracterização apresentada no capítulo 3. Entre experimentos distintos procuramos repetir pelo menos uma carga utilizada anteriormente para que os resultados pudessem ser comparados. Outras cargas foram introduzidas para aumentar a diversidade da carga de trabalho testada.

A primeira seção deste capítulo descreve o simulador implementado e o projeto da simulação. As medidas monitoradas durante a simulação são discutidas e a estrutura básica do simulador é apresentada. Os experimentos estão divididos em dois grupos. O primeiro grupo compara o desempenho do cache particionado com o do cache não particionado. Neste experimento, todos os parâmetros são mantidos constantes: tamanho do cache, política de reposição e cargas. Apenas o particionamento é introduzido para comparação. A análise dos resultados deste grupo de experimentos, apresentada na segunda seção, contribui para o entendimento do impacto da variabilidade dos tamanhos dos objetos no desempenho do cache.

O segundo grupo de experimentos foi projetado com o objetivo de investigar o comportamento do modelo PART quando seus parâmetros são variados. A análise dos resultados deste grupo, descrita na terceira seção, auxilia a definição dos parâmetros do modelo PART.

A quarta seção analisa a utilização dos mapas de desempenho para o acompanhamento simultâneo do desempenho medido em HR e BHR e para o auxílio do projeto de novas

políticas. A última seção apresenta um sumário dos resultados e a conclusão do capítulo.

5.1 Projeto da Simulação

Diferentes estratégias de cache são tradicionalmente comparadas através de simulação por software (Jain, 1991) cujo objetivo é mais avaliar o desempenho relativo das diversas estratégias do que obter valores absolutos para as métricas. Para avaliar o desempenho do modelo PART foi projetada a estrutura básica de um simulador para caches particionados. O cache não particionado foi definido como um cache com uma única partição.

A estrutura básica do simulador do PART é apresentada na figura 5.1. O simulador recebe como entrada a carga e os parâmetros do cache a ser simulado, a saber, o tamanho do cache, a definição das classes, o número e o tamanho das partições, e a política de substituição a ser empregada em cada partição. A partir do tamanho do objeto requisitado e da definição de classes identifica-se a partição na qual o objeto deve ser ou estar armazenado. Cada partição é vista como um cache individual e tem sua própria lista de documentos. Substituições só ocorrem entre objetos da mesma classe, que compartilham a mesma partição.

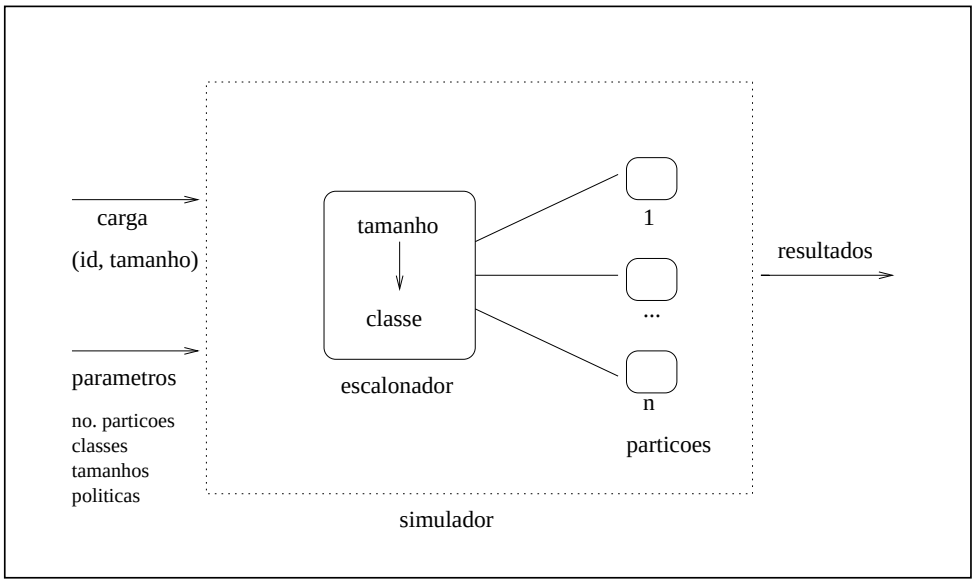

Figura 5.1: Estrutura do simulador.

Um aspecto essencial para o bom desempenho do simulador é o pré-processamento da entrada (carga). A entrada consiste de uma seqüência cronológica de requisições, cada uma composta pelo identificador (URL) e o tamanho da resposta. O pré-processamento consiste em converter esta entrada em um formato mais compacto, no qual cada URL é substituído por um identificador numérico único. Esta etapa permite um ganho substancial no desempenho do simulador.

A tarefa principal do pré-processamento é mapear os URLs em identificadores inteiros únicos no intervalo $[1 \ldots n]$ onde n é o número de arquivos únicos da carga. A partir do mapeamento, a entrada é reconstruída com os identificadores inteiros, e o custo computacional do casamento de padrão, necessário para comparar URLs, é substituído por uma comparação entre inteiros. Este é um ganho importante. Outra conseqüência do mapeamento é a possibilidade de definir a estrutura de dados que contém os metadados de cada objeto armazenado no cache como um arranjo linear de tamanho n. Portanto, o ganho principal é o acesso à informação de cada objeto (URL) em tempo constante durante a simulação.

A figura 5.2 apresenta as principais estruturas de dados utilizadas no simulador. A estrutura *Files* é o arranjo referido anteriormente. A informação armazenada para cada arquivo consiste do seu tamanho, status (se o arquivo está presente no cache ou não), informações específicas necessárias em cada política implementada e apontadores para composição da lista de objetos do cache. A estrutura de dados *Caches* é também um arranjo cujo tamanho é igual ao número de partições definido para o experimento. Cada partição contém apontadores para o início e o fim da lista de arquivos armazenados no seu espaço. As informações associadas a cada partição são o tamanho, o espaço disponível e os contadores para o cálculo das métricas.

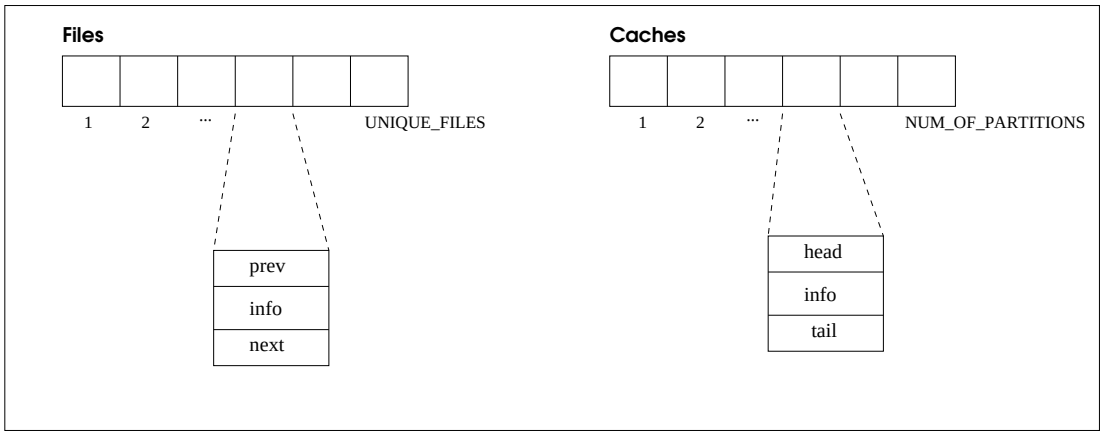

Figura 5.2: Estruturas de dados para os arquivos e os caches do simulador. UNIQUE_FILES e NUMBER_OF_PARTITIONS são, respectivamente, os tamanhos das estruturas FILES e CACHES.

As políticas implementadas para experimentação foram LRU, GD-Size e LFU-DA. LRU foi escolhida por ser a mais utilizada nos caches reais da WWW e também a mais utilizada como base de comparação de desempenho nos experimentos com caches Web. Além de LRU, foram utilizadas outras políticas com desempenho reconhecidamente bom em cada uma das métricas, HR e BHR. Políticas baseadas em tamanho são escolhas apropriadas para otimização de HR. Dentre as duas opções mais comuns, SIZE e GD-Size (Cao e Irani, 1997), escolhemos a última porque implementa um mecanismo de envelhecimento que evita a poluição do cache. Os melhores valores para BHR são obtidos pelas políticas baseadas em freqüência, por exemplo, LFU. A política escolhida foi *Least Frequently Used with*

Dynamic Aging (Arlitt *et al.*, 1999), LFU-DA, também porque implementa um mecanismo de envelhecimento.

A estrutura do código do simulador é apresentada na figura 5.3. As principais operações são inserir um arquivo no cache em caso de *miss* e atualizar a estrutura de dados em caso de *hit*. A cada inserção verifica-se se o espaço disponível é suficiente para o armazenamento do arquivo a ser inserido. Se necessário, alguns arquivos são removidos até que haja espaço suficiente. A atualização requer a retirada do arquivo da lista da sua respectiva partição e a reinserção do mesmo em sua nova posição. A ordem de inserção na lista depende da política. Por exemplo, para LRU a inserção ocorre no início da lista. Para as demais políticas, a lista é mantida ordenada de acordo com um valor H calculado para cada arquivo. Para a política GD-Size, H é definido por $H = 1/\text{tamanho} + L$, onde L é o fator de envelhecimento. Para a política LFU-DA, o valor H de cada arquivo é calculado por $H =$ número de referências $+ L$. A retirada não depende da política, é sempre feita no extremo da lista onde estão os arquivos LRU ou os arquivos com menor valor H. O valor de L é atualizado a cada objeto i retirado com o valor de H deste objeto, isto é, $L = H_i$.

```
for each request (name, size)
{
    // hit or miss?
    if(Files[name].status == NOT_IN_CACHE)
    {
        // make free space
        while(ptrtopartition->FreeSpace < size)
            RemoveLRU(ptrtopartition,Files);
        // insert node at the top of LRU stack
        InsertLRU(Cache[partition], Files, name);
    }
    // otherwise, a cache hit has occurred
    else
    {
        // must change place in the stack due to LRU policy
        RemoveFile(Cache[partition], Files, name);
        // insert node at the top of LRU stack
        InsertLRU(Cache[partition], Files, name);
    }
}
```

Figura 5.3: *Loop* principal do código do simulador.

O código foi escrito em linguagem C, sem auxílio de linguagens de simulação. A simulação é determinística e baseada em registros de cargas reais (*trace-driven simulation*),

cujas caracterizações estão descritas no capítulo 3. O tempo de simulação depende do número de requisições da entrada e da política de substituição executada. Com os benefícios do pré-processamento da entrada, o tempo de simulação com a política LRU é de poucos minutos mesmo para entradas da ordem de milhões. Este tempo foi obtido em experimentos executados em um computador Sun Ultra Enterprise 3000 com memória principal de 512 MB e sistema operacional SunOS versão 5.5.1.

O aquecimento do cache (*warm-up*) consiste no processo de operá-lo de maneira usual, armazenando os objetos requisitados sem, no entanto, coletar estatísticas. O objetivo é preencher o cache levando-o de um estado inicial transiente a um estado estável de operação. Esse processo evita a contabilização de um número grande de *misses* iniciais, que ocorrem porque o cache está vazio.

A quantidade de carga utilizada na fase transiente, em número de requisições, é função do tamanho do cache e do modelo de particionamento. A diversidade de experimentos gerada pelos diferentes tamanhos de cache e modelos de particionamento implica em utilização de quantidades diferentes de uma mesma carga no período transiente, o que desconfigura a carga original e torna difícil a comparação mesmo entre experimentos iguais com tamanhos diferentes de cache.

Em função disso, consideramos que a inicialização com o cache vazio é a mais justa para comparação de resultados. Assim, todos os experimentos e contadores foram iniciados com o cache vazio e a execução foi feita para carga completa. Esta abordagem foi também utilizada em outras simulações de cache (Karedla *et al.*, 1994), (Williams *et al.*, 1996). Para garantir que a política de reposição é utilizada o suficiente, calculamos o tamanho do cache com base no tamanho máximo necessário para armazenar todos os arquivos requisitados (tamanho de referência). Utilizando frações deste tamanho garantimos que o cache será completamente preenchido e o sistema alcançará uma situação de utilização plena das estratégias em questão.

Esta experimentação tem dois objetivos. O primeiro é avaliar os efeitos da variação dos parâmetros de entrada do modelo PART no desempenho do cache. O segundo é comparar o desempenho e o custo de sistemas de cache cuja organização do espaço segue o modelo PART com sistemas que seguem o modelo convencional não particionado. As características do modelo PART, descritas no capítulo anterior são exploradas. As medidas monitoradas estão em três grupos. O primeiro grupo compreende as medidas de desempenho, HR e BHR. O segundo grupo compreende as medidas que contribuem para o entendimento do efeito do particionamento, que são medidas de número de objetos armazenados no cache, tempo médio de permanência dos arquivos no cache, número de arquivos inseridos e retirados, número médio de retiradas em cada operação de reposição e número de retiradas devido à fragmentação. O terceiro grupo é composto por apenas uma medida, o tamanho médio dos *hits*, utilizada para validação da simulação.

Os resultados da simulação foram validados pelo Teorema 2 apresentado no capítulo 4. Para isso, os valores de tamanho médio dos *hits*, HR e BHR, medidos durante a simulação, foram utilizados para calcular o tamanho médio das requisições de acordo com a fórmula

$$\text{tamanho médio das requisições} = \frac{HR}{BHR} * \text{tamanho médio dos hits}$$

parâmetro	PORTUGAL	POP99-zez	BR
classe 1	tamanho $<$ 1500 bytes	tamanho $<$ 1000 bytes	tamanho $<$ 2000 bytes
classe 2	1500 $\leq$ tamanho $<$ 7000	1000 $\leq$ tamanho $<$ 8000	2000 $\leq$ tamanho $<$ 10000
classe 3	tamanho $\geq$ 7000	tamanho $\geq$ 8000	tamanho $\geq$ 10000
partição 1	4%	5%	4%
partição 2	22%	17%	4%
partição 3	restante	restante	restante

Tabela 5.1: Definição dos parâmetros para o experimento com o cache com três partições.

Este resultado foi comparado com o tamanho médio das cargas apresentadas na tabela 3.2, para todos os experimentos. Os valores comparados diferiram no máximo em 0.5%.

Técnicas de verificação do modelo de simulação (Jain, 1991) como execução de casos simples, registros de eventos e variáveis em detalhe durante a execução, execução passo a passo com auxílio de ferramentas de depuração de código e testes de continuidade e de consistência foram empregadas para verificar a correção da implementação. Os resultados obtidos são reproduzíveis. O desempenho relativo das políticas conhecidas implementadas no simulador é similar ao reportado nos trabalhos citados.

5.2 Comparação entre Cache Particionado e Não Particionado

Nesta seção são apresentados os resultados da comparação entre o modelo de cache particionado e o modelo não particionado. O experimento consiste em simular os modelos particionado e não particionado para vários tamanhos de cache e registrar várias métricas. Os tamanhos de cache simulados foram 1%, 2%, 4%, 8%, 16%, 32%, 50% e 64% do tamanho de referência do cache para cada carga. Foram medidos os seguintes valores: HR e BHR, tamanho médio dos *hits*, número de arquivos presentes no cache ao final da simulação, número médio de arquivos removidos em cada operação de reposição, tempo médio de permanência de um arquivo no cache, número de arquivos inseridos e removidos durante a simulação e número de arquivos removidos devido à fragmentação.

As cargas utilizadas neste teste foram PORTUGAL, POP99-zez e BR. Para o cache particionado foram definidas três partições. Os parâmetros do modelo PART definidos para este experimento estão descritos na tabela 5.1. A seção 5.3 aborda a questão de como definir os parâmetros de entrada do modelo PART. A política de reposição para todos os casos foi LRU.

Os resultados desses experimentos são apresentados e analisados nas subseções seguintes. Os gráficos apresentam rótulos PART e NP que significam, respectivamente, os resultados da simulação do cache particionado e do não particionado. Os gráficos apresentados estão em função do tamanho do cache utilizado no experimento, calculado como uma porcentagem do tamanho do cache necessário para que não haja nenhuma remoção (tamanho de referência).

5.2.1 HR e BHR

Os gráficos da figura 5.4 apresentam os valores de HR e BHR para ambos os caches, para cada carga. As curvas de BHR para os caches particionado e não particionado estão sobrepostas nos gráficos. Enquanto o PART preserva o BHR obtido pela política LRU no cache não particionado, os valores de HR obtidos são sempre melhores, porém o ganho varia com a carga. O incremento médio em HR é de 5.0% para a carga PORTUGAL, 9.6% para POP99-zez e 21.5% para BR. No caso do POP99-zez, os valores obtidos para HR com o cache particionado são maiores do que os obtidos com o cache não particionado com o dobro do tamanho. Esta afirmativa é válida para os caches menores. Para BR, os ganhos são ainda melhores para todos os tamanhos de cache.

Estes resultados apresentam evidências de que os ganhos são maiores para cargas que apresentam maior variabilidade nos tamanhos dos arquivos. Conforme revelado no capítulo 3, das três cargas utilizadas neste experimento, PORTUGAL é a que apresenta menor variabilidade e BR é a que apresenta maior variabilidade. A análise dos valores encontrados para as demais métricas auxilia o entendimento do comportamento do PART e de seus fundamentos.

O ganho em HR é importante porque contribui para melhorar a latência das requisições, servindo objetos a partir de um dispositivo de E/S mais rápido. A Internet pode ser considerada como um nível adicional na hierarquia de E/S. Portando, se o Web cache está na rede, o cache é o disco e os *hits* vão evitar o acesso à Internet. Se o Web cache está num servidor WWW, o cache é implementado em memória e os *hits* vão evitar o acesso a disco. A seção 5.3.1 mostra como o PART pode obter melhor desempenho em HR e BHR utilizando políticas diferentes nas partições.

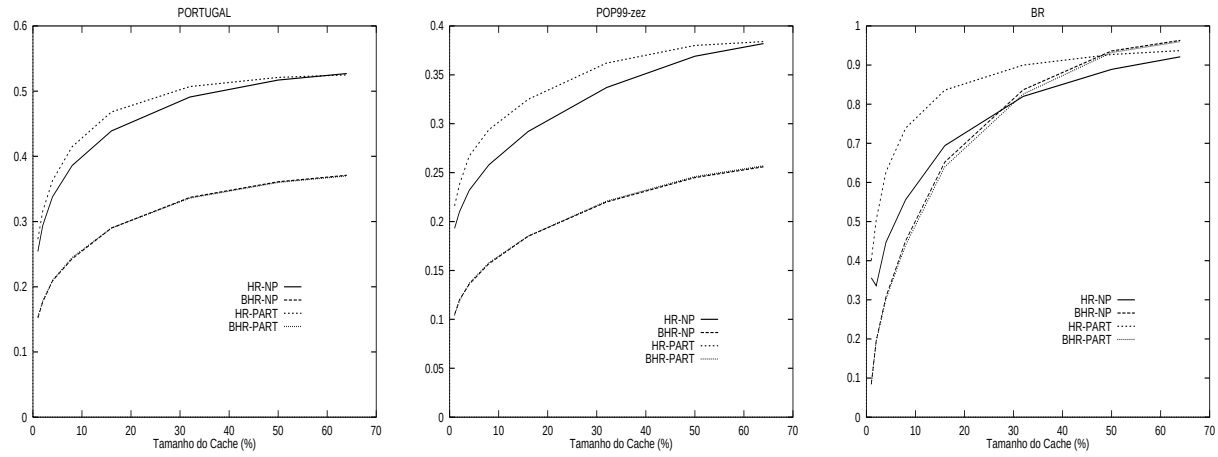

Figura 5.4: HR e BHR para caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

5.2.2 Número de Arquivos no Cache

O número de arquivos presentes no cache ao final da simulação para ambos os modelos é apresentado na figura 5.5. Como esperado, o número de arquivos armazenados cresce com o tamanho do cache. No entanto, este número é sempre maior no modelo PART. Isto ocorre porque o particionamento reserva espaço para os arquivos pequenos e permite maior controle sobre o número de arquivos armazenados. Assim, é possível manter mais arquivos no cache. Nos experimentos realizados o PART armazenou, em média, 56% mais arquivos do que o cache não particionado para a carga PORTUGAL, 97% a mais para a carga POP99-zez e 132% a mais para BR.

O aumento do número de arquivos no cache pode ter como conseqüência o aumento no número de acertos. Em (Breslau *et al.*, 1999), por exemplo, é apresentada uma fórmula para calcular o HR assintótico em função do número de objetos armazenados no cache. A fórmula é HR $\approx \Omega\ ln$(número de objetos no cache), onde Ω é uma constante calculada para cada carga.

O aumento da fração de *hits*, apresentado na seção anterior, é bem menor do que o aumento correspondente no número de arquivos mantidos no cache. Isto pode ser explicado pela grande proporção de arquivos requisitados uma única vez (tabela 3.6). Apenas uma pequena parte dos arquivos armazenados é requisitada mais de uma vez, por isso não há relação diretamente proporcional entre o número de arquivos armazenados e a fração de *hits*.

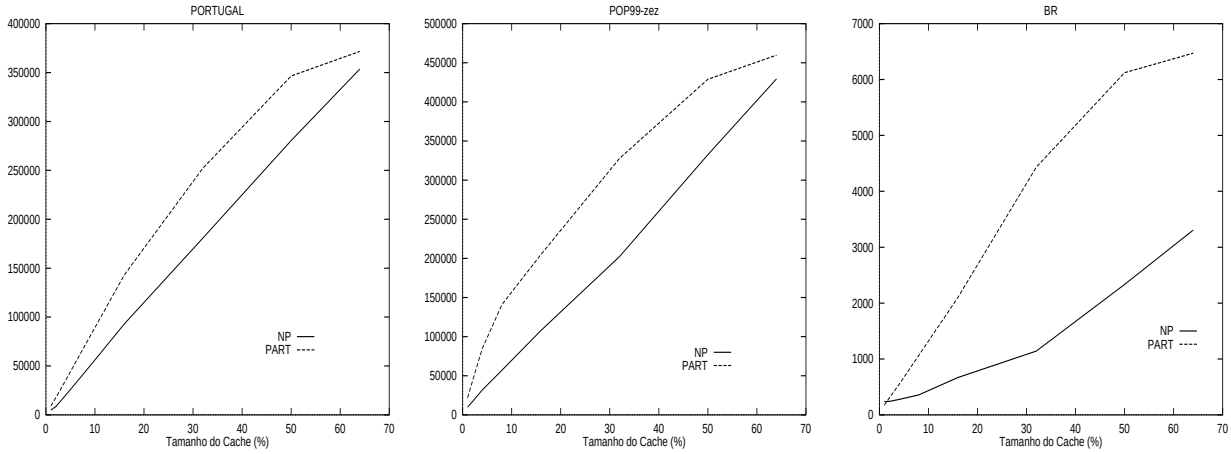

Figura 5.5: Número de arquivos armazenados no cache ao final da simulação para caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

5.2.3 Tamanho Médio dos Hits

O tamanho médio dos *hits* é menor no modelo particionado. Devido à reserva de espaço para arquivos pequenos, um número maior destes arquivos é armazenado no cache, au-

mentando a probabilidade de *hit* nestes documentos. Como consequência, a média dos tamanhos dos *hits* diminui. A figura 5.6 apresenta os resultados desta medida nos experimentos realizados. Conforme dito anteriormente, os valores de tamanho médio dos *hits* obtidos durante a simulação foram utilizados para validar os resultados.

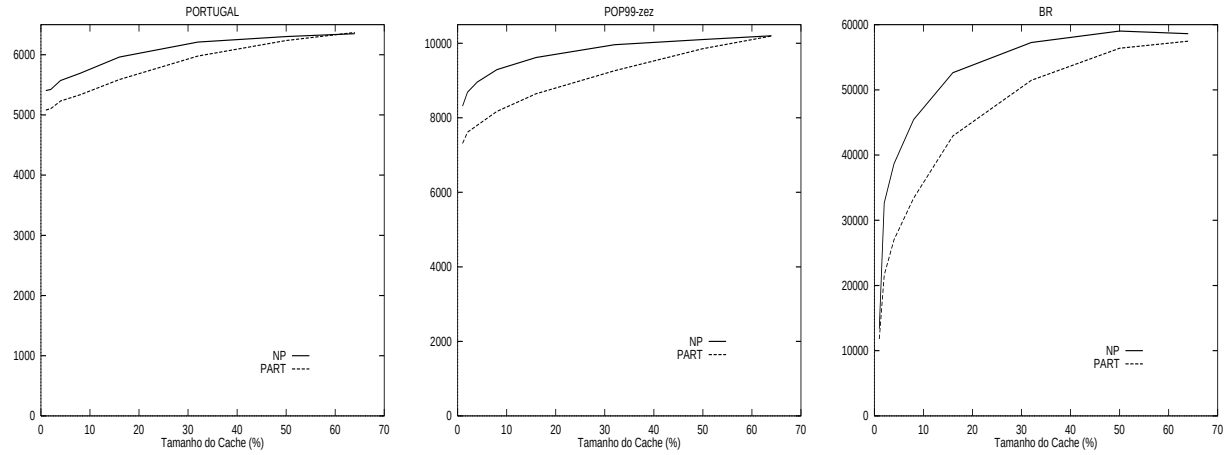

Figura 5.6: Tamanho médio dos *hits* para caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

5.2.4 Tempo Médio de Permanência no Cache

O tempo médio, medido em número de requisições, de permanência de um arquivo no cache foi também avaliado. Este valor é conhecido como LRU-*age* quando a política é LRU. A figura 5.7 mostra os resultados. O PART preserva os arquivos no cache por um período de tempo maior, cujos números são 32% a mais para a carga PORTUGAL, 82% para POP99-zez e 233% para BR, todos em relação ao cache não particionado. Preservar os arquivos no cache por um período de tempo mais longo aumenta a probabilidade de acerto.

A medida do LRU-*age* é interessante para ajustar o tamanho relativo entre as partições e foi utilizada na fase de definição dos parâmetros. Valores muito discrepantes para as diversas partições indicam que seu tamanho relativo não está adequado à carga. Por exemplo, se o LRU-*age* da primeira partição é muito alto comparado com a mesma medida da segunda partição, isto indica que o tamanho da primeira partição está grande e pode ser diminuido em favor da segunda partição.

5.2.5 Inserções e Remoções de Arquivos

Um dos argumentos a favor do PART é que, devido à variabilidade nos tamanhos dos arquivos, cada substituição individual no cache não particionado poderia envolver um número grande de remoções de arquivos, alterando significativamente o estado do cache. Para fazer esta avaliação foi medido o número de arquivos removidos a cada reposição, isto é, a cada

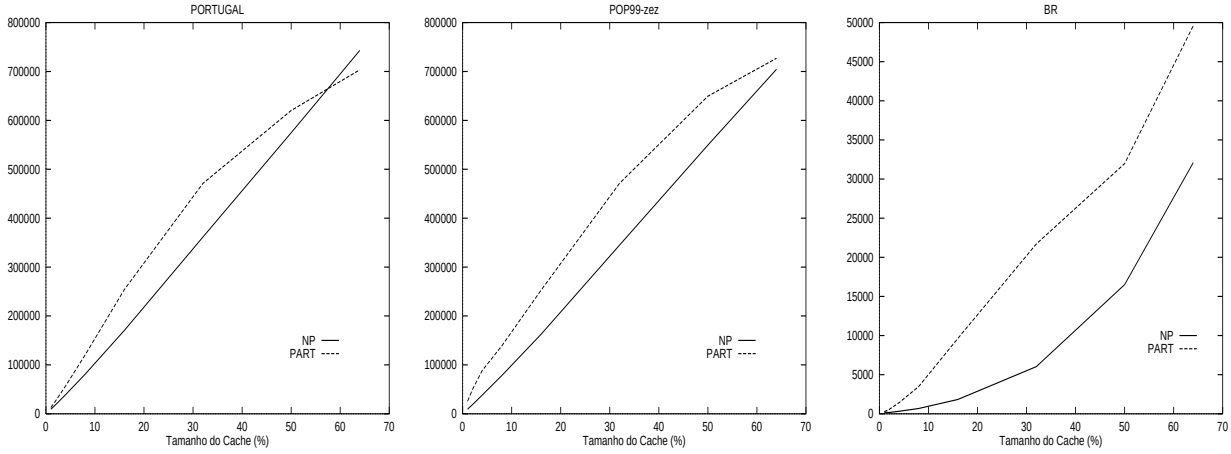

Figura 5.7: Tempo médio de permanência, em número de requisições, dos arquivos em caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

inserção na qual foi necessário liberar espaço no cache. A figura 5.8 compara os resultados dos valores médios dessa medida, calculados sobre todas as reposições feitas durante a simulação, para cada modelo de cache. O modelo PART apresenta um comportamento bastante estável nesta métrica, com uma média aproximadamente constante e inferior a dois arquivos removidos a cada operação de reposição para todas as cargas.

No caso do cache não particionado, as cargas PORTUGAL e POP99-zez apresentam valores em torno de quatro e seis, respectivamente, com discreto crescimento à medida que o tamanho do cache aumenta. O crescimento pode ser explicado pelo fato de que nos caches maiores é necessário um número menor de remoções de arquivos. No entanto, as remoções que ainda são necessárias são exatamente as causadas pela inserção de arquivos grandes envolvendo, portanto, muitas retiradas. Como as remoções com poucas retiradas diminuem, a média cresce. No caso da carga BR, os valores são instáveis para caches pequenos e isso se deve à grande variabilidade nos tamanhos dos arquivos desta carga. Para caches com tamanhos acima de 32%, os valores caem consideravelmente mas ainda se mantêm acima do PART. A média da carga BR para todos os tamanhos de cache é de 17 arquivos removidos em cada operação de reposição.

Esses resultados comprovam o argumento utilizado. Um número maior de arquivos é envolvido numa substituição no cache não particionado porque não há relação entre o tamanho dos arquivos inseridos e retirados. Na inserção de um arquivo grande, muitos arquivos pequenos podem ser retirados. Por outro lado, se um arquivo grande é o LRU (ou o próximo a ser retirado), sua retirada pode ser causada pela inserção de um arquivo pequeno. Essa operação libera um espaço maior do que o necessário. Assim, as próximas inserções encontrarão espaço suficiente no cache e não implicarão em retiradas. O PART retira menos arquivos a cada substituição. Porém, como o espaço liberado por vez é pequeno devido à maior uniformidade nos tamanhos dos arquivos envolvidos nas trocas, o número de operações para abrir espaço é maior. Em resumo, o PART retira poucos

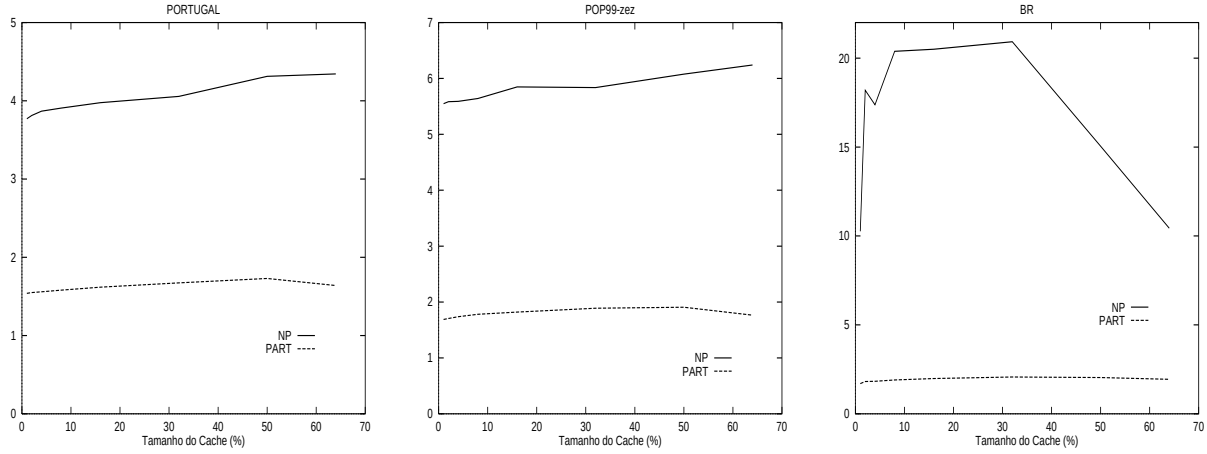

Figura 5.8: Número médio de arquivos removidos em cada operação de reposição em caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

arquivos de cada vez mas faz isso mais frequentemente enquanto o cache não particionado retira muitos arquivos por vez e faz remoções menos frequentes.

O próximo passo é conhecer o número total de arquivos removidos. A figura 5.9 apresenta o número de arquivos removidos e inseridos no cache durante a simulação. O PART invariavelmente insere e remove menos arquivos. O cache não particionado remove 13% mais arquivos do que o PART para a carga PORTUGAL, 20% mais para a carga POP99-zez e 84% mais para a carga BR. As remoções executadas pelo PART são em menor número e mais regulares, melhor distribuídas no período do experimento. Isto explica o valor maior para LRU-*age* obtido pelo PART. Ao liberar apenas o espaço necessário à próxima inserção, evita-se retiradas desnecessárias, o que favorece a preservação dos arquivos no cache. O número de inserções é igual ao número de *misses*. Portanto, o PART insere menos porque acerta mais e isto é devido ao fato de manter mais arquivos no cache, e mantê-los por mais tempo, preservando o estado do cache.

Esta variabilidade apresentada pelo modelo não particionado não é benéfica para o desempenho no ambiente real pois o sistema vai experimentar picos de requisições de retirada em certos instantes, intercalados por períodos nos quais nenhuma retirada é feita. Uma melhor distribuição das retiradas, como a obtida pelo PART, contribui para a obtenção de uma melhor taxa de serviço.

O número de remoções por operação de reposição nas partições individuais do cache particionado foi avaliado. O número médio de arquivos removidos é maior na última partição em todos os experimentos e isto é devido à maior variabilidade nos tamanhos dos arquivos da última classe. Para PORTUGAL, a média é de 1.3 para as duas primeiras partições e 2.6 para a última. Para POP99-zez, temos 1.2 para a primeira partição, 1.4 para a segunda e 3.7 para a última. Finalmente, para BR temos, 1.6 para a primeira partição, 1.3 para a segunda partição e 5.2 para a terceira partição.

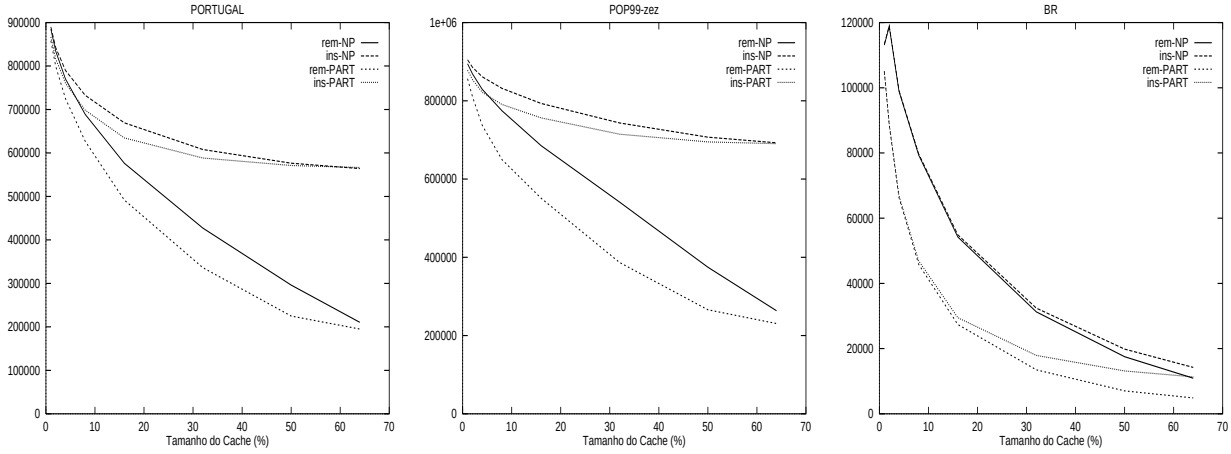

Figura 5.9: Número de remoções e inserções de arquivos em caches particionados (PART) e não particionados (NP) com política LRU. Cargas PORTUGAL (esquerda), POP99-zez (centro) e BR (direita).

5.2.6 Fragmentação

O particionamento do cache provoca retiradas de arquivos devido à fragmentação do espaço entre as partições. Isto ocorre quando um arquivo deve ser armazenado em uma partição na qual não há espaço suficiente, sendo necessária a remoção de outros arquivos nessa partição. No entanto, a soma do espaço livre em todas as partições seria suficiente para armazenar o arquivo. Isso não ocorre no cache não particionado.

Como vimos na seção anterior, o número de arquivos removidos no cache particionado, que inclui as remoções devido à fragmentação, é menor do que no modelo não particionado. Portanto, mesmo incluindo o custo da fragmentação, o cache particionado apresenta um número inferior de remoções de arquivos.

A fragmentação dos arquivos no disco, que causa impacto no tempo de acesso, não foi modelada na simulação. Assumimos que os arquivos são armazenados de forma contígua.

5.2.7 Conclusão

Nesta seção foi discutido e comparado o desempenho dos modelos de cache particionado e não particionado. O PART se mostrou melhor em todas as métricas avaliadas. Não foram observados resultados contraditórios para um mesmo experimento e cargas diferentes. A substituição de arquivos no cache sem considerar o tamanho produz duas situações que são prejudiciais ao desempenho do cache: retirada desnecessária de arquivos e liberação desnecessária de espaço. Os resultados evidenciam que o modelo PART minimiza esses problemas e apresenta um comportamento mais regular, preservando mais arquivos no cache e por mais tempo. Os melhores resultados em todas as métricas foram obtidos em caches de tamanhos intermediários, entre 2% e 50% do tamanho de referência. A escassez extrema do recurso gerenciado, o espaço em disco, representada pelo tamanho de 1%, e a plena disponibilidade do recurso, representada pelo tamanho de 64%, são situações

extremas nas quais a mudança de modelo não configura alteração significativa no resultado.

5.3 Efeito da Variação dos Parâmetros de Entrada do Modelo PART

No experimento descrito na seção anterior os parâmetros do modelo PART foram mantidos constantes e a comparação foi feita com o cache não particionado. Nessa seção são analisadas configurações diferentes para o particionamento. Em cada experimento apenas um parâmetro é variado. O objetivo desse conjunto de experimentos é avaliar a sensibilidade do modelo PART frente à mudança nos seus parâmetros.

5.3.1 Políticas Diferentes nas Partições

Um dos parâmetros de entrada do modelo PART é a política de reposição utilizada em cada partição. Neste experimento foram utilizadas políticas diferentes nas partições, com o objetivo de obter melhores valores para HR e BHR. Foram definidas três partições. Na primeira partição, que contribui mais para HR, foi implementada a política GD-Size. A política LFU-DA foi implementada na terceira partição para favorecer BHR. As duas políticas foram implementadas na segunda partição, em experimentos diferentes numa fase de testes. LFU-DA produziu os melhores resultados e foi a escolhida. Para comparação foram executados experimentos com o cache não particionado e as políticas LRU, LFU-DA e GD-Size.

As figuras 5.10 e 5.11 apresentam os resultados para as cargas BR e U, respectivamente. A legenda PART indica o cache particionado com as diferentes políticas e as demais legendas indicam os resultados para o cache não particionado com a respectiva política. GD-Size apresenta os melhores resultados para HR e os piores para BHR. LFU-DA tem o melhor desempenho em BHR mas é bastante inferior em HR. LRU apresenta desempenho próximo porém inferior ao de LFU-DA nas duas métricas. O modelo PART obtém bons resultados em ambas as métricas, com ganhos tanto em HR quanto em BHR. O particionamento oferece as condições necessárias para que o cache possa se beneficiar de políticas específicas para cada métrica. No entanto, o desempenho do PART em ambas as métricas parece estar limitado pelo desempenho da melhor política para cada métrica.

Nos resultados para a carga BR, o PART com política mista obtém ganho médio (sobre todos os tamanhos) de 25% em HR e 7% em BHR, quando comparado com o cache não particionado e a política LRU. No entanto a maior parte do ganho em HR é devida ao particionamento. A seção 5.2.1 mostrou que o cache particionado com LRU nas três partições obtém ganho de 21.5% em relação ao cache não particionado com LRU. A utilização de políticas diferentes parece ser mais efetiva para BHR, quando comparada com LRU. Enquanto o desempenho do cache com a política LRU nessa métrica é igual para os dois modelos de particionamento, o ganho com a utilização de diferentes políticas é de 7%.

Para a carga U, o ganho médio do cache particionado com políticas variadas em relação ao não particionado com política LRU é de 12% para HR e 6% para BHR.

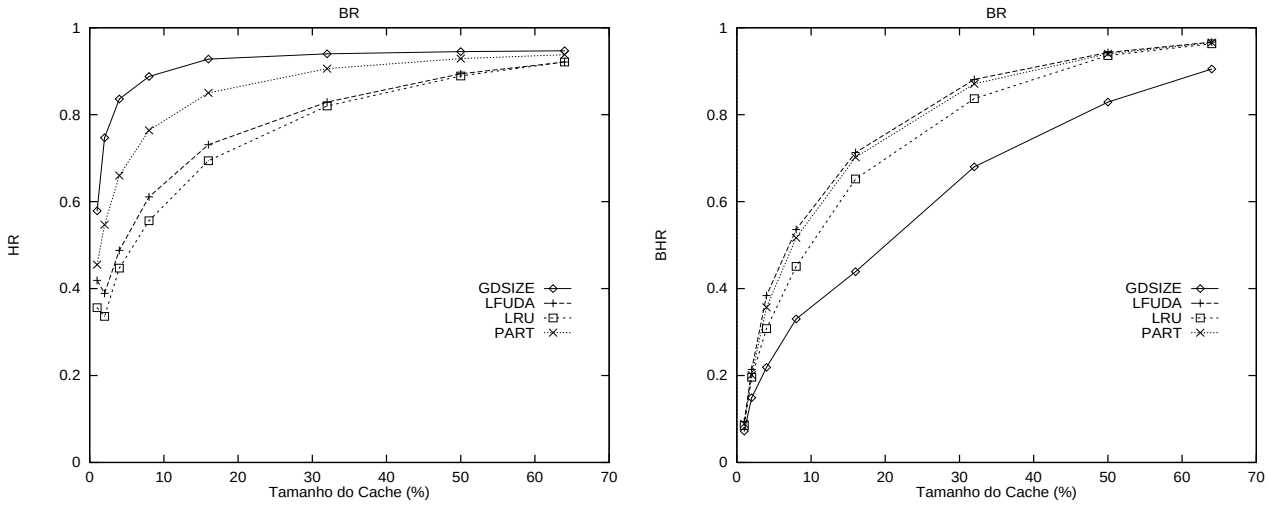

Figura 5.10: HR e BHR para diversas políticas aplicadas ao cache sem particionamento e para PART com três partições com GD-Size na primeira partição e LFU-DA nas demais.

5.3.2 Variando o Número de Partições

Esta seção apresenta os resultados de um experimento com variação do número de partições. Os resultados, apresentados na figura 5.12, são para HR e para as cargas U e POP98. Os valores para BHR não são apresentados pois as curvas são sobrepostas, indicando valores praticamente idênticos. A fração de *hits* cresce para o mesmo tamanho de cache e um número maior de partições. No entanto, os ganhos são maiores nos primeiros particionamentos e para os tamanhos intermediários de cache.

A mesma definição de classes e tamanhos de partição foi utilizada neste experimento para as duas cargas, com o objetivo de mostrar que o particionamento gera ganhos mesmo quando a definição de parâmetros não é ajustada para a carga. Em outras palavras, uma definição mais adequada pode gerar maiores ganhos.

5.3.3 Variando o Tamanho das Partições

Nesta seção vamos avaliar a sensibilidade do modelo em relação à variação do parâmetro tamanho das partições. Neste experimento são definidas duas classes fixas e duas partições cujos tamanhos são variados. O tamanho do cache é constante em cada experimento. Inicialmente a primeira partição ocupa um espaço pequeno do cache deixando o restante para a segunda partição. A seguir, o tamanho da primeira partição é aumentado sucessivamente com consequente diminuição do tamanho da segunda partição. Os valores de HR, BHR e tamanho médio dos *hits* são monitorados em cada alteração. A variação das métricas HR e BHR é obtida pela variação do tamanho das partições.

O experimento foi repetido para vários tamanhos de cache, a saber, 1%, 2%, 4%, 8%, 16%, 32%, 50% e 64% do tamanho de referência. O tamanho da primeira partição foi calculado de acordo com a caracterização dos bytes únicos da primeira classe. Assim, para

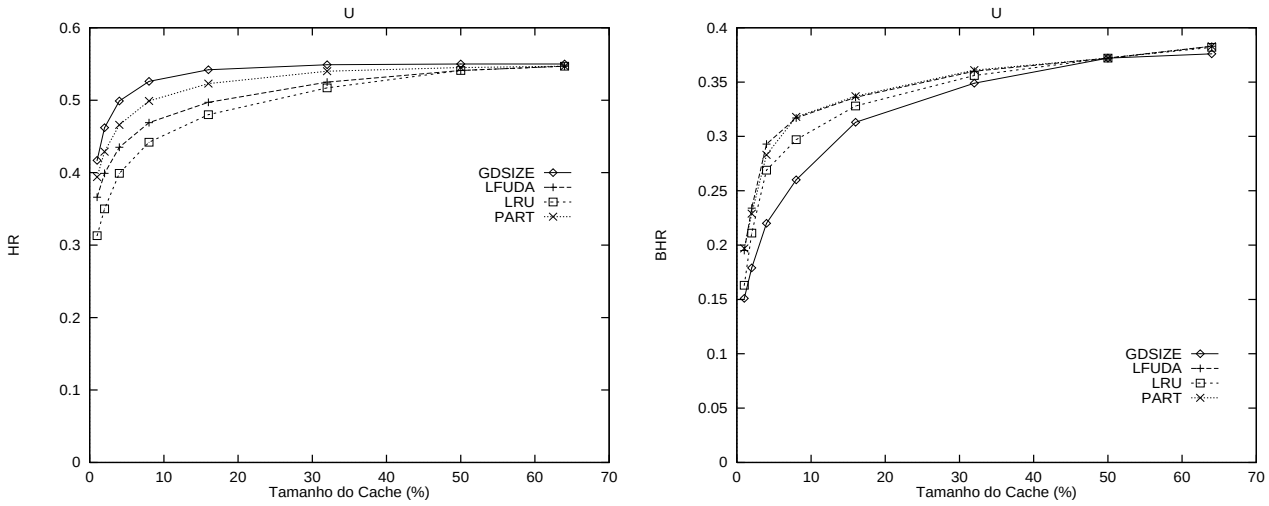

Figura 5.11: HR e BHR para diversas políticas aplicadas ao cache sem particionamento e para PART com três partições com GD-Size na primeira partição e LFU-DA nas demais.

todos os tamanhos de cache, a primeira partição teve tamanhos correspondentes a 0.1%, 0.5%, 1%, 2%, 4%, 8%, 16%, 32%, 64% e 100% dos bytes únicos da primeira classe. O espaço restante do cache foi alocado para a segunda partição. O objetivo é aumentar gradativamente o espaço dedicado à primeira partição sem, no entanto, desperdiçar espaço alocando mais do que o máximo necessário.

As cargas utilizadas neste experimento foram BR e POP98. Para a carga BR foram definidas a classe 1, com arquivos menores do que 2000 bytes e a classe 2, com arquivos de tamanho igual ou maior que este valor. A primeira classe é responsável por 56% dos documentos únicos e 52% das requisições. Em relação aos bytes, ela compreende 29% dos bytes únicos e apenas 6% dos bytes requisitados. A classe 1 definida para a carga POP98 é composta pelos arquivos de tamanho menor do que 3000 bytes. A classe 2 agrupa os demais arquivos. Assim, a classe 1 engloba 45% dos arquivos únicos, 48% das requisições, 7% dos bytes requisitados e 6% do bytes únicos.

A figura 5.13 mostra os mapas de desempenho para as cargas BR e POP98, incluindo os resultados do experimento. O mapa é caracterizado por ter HR no eixo y, o tamanho dos *hits* no eixo x e curvas que indicam os valores de BHR, começando de 0.1 para a curva mais próxima ao eixo x, e aumentando com passo 0.1. No gráfico da carga BR temos dez curvas para BHR. No gráfico da carga POP98, as curvas de BHR vão até 0.6 pois a própria carga não alcança valores maiores. As curvas marcadas com pontos e legendadas são os resultados do experimento para valores diferentes do tamanho do cache. Numa mesma curva cada ponto corresponde a um tamanho relativo das partições, conforme descrito anteriormente. O primeiro ponto é o que tem o maior tamanho médio de *hits* e corresponde a 0.1% dos bytes únicos da primeira classe.

Observamos que ao variar o tamanho das partições, de um tamanho pequeno a um tamanho grande para a primeira partição, o HR varia perceptivelmente enquanto o BHR

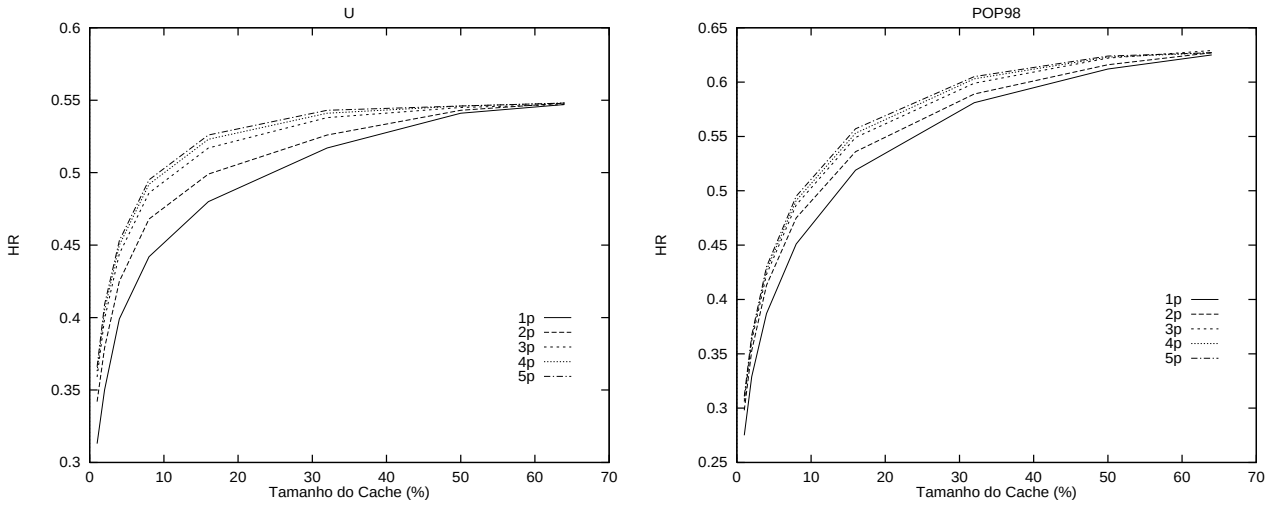

Figura 5.12: Valores para HR para as cargas U (esquerda) e POP98 (direita) com número de partições variando entre um e cinco.

varia pouco. À medida que o tamanho da primeira partição aumenta, maior espaço é reservado para os arquivos pequenos, mais arquivos pequenos são armazenados, com consequente diminuição do tamanho médio dos *hits* e aumento do HR. No entanto, quando uma fração grande do cache é reservada para a primeira partição, o BHR pode diminuir. Veja, por exemplo, as curvas de 1% a 8% para as duas cargas. Este efeito é minimizado nos caches maiores. A melhor proporção entre os tamanhos das partições é a que dedica entre 8% e 64% dos bytes únicos da primeira classe para a partição correspondente, sendo o menor valor, 8%, mais adequado para caches pequenos, e os maiores valores indicados para caches maiores.

Uma avaliação precisa das variações em HR e BHR obtidas neste experimento pode ser feita com o auxílio da figura 5.14. Esta figura apresenta os valores máximo e mínimo obtidos para HR e BHR, dentre todos os resultados gerados pelas diferentes proporções nos tamanhos das partições, para cada tamanho do cache. A variação de HR é maior do que a de BHR. Portanto, para um mesmo tamanho de cache é possível aumentar o significativamente o HR aumentando o tamanho das partições dedicadas a documentos pequenos. O crescimento significativo de BHR só é alcançado com o aumento do tamanho do cache. Este experimento dá uma idéia da flexibilidade do PART para gerar variações em HR.

5.3.4 Ajuste Dinâmico dos Tamanhos das Partições

A variação do tamanho da partição permite alteração substancial do HR sem que o BHR seja alterado significativamente. Esta propriedade permite que o ajuste dos tamanhos das partições seja feito em tempo de execução. Por exemplo, para um cache com duas partições, o tamanho da primeira partição é inicialmente definido como sendo uma fração pequena do cache. O tamanho é então progressivamente aumentado em função da resposta de HR e

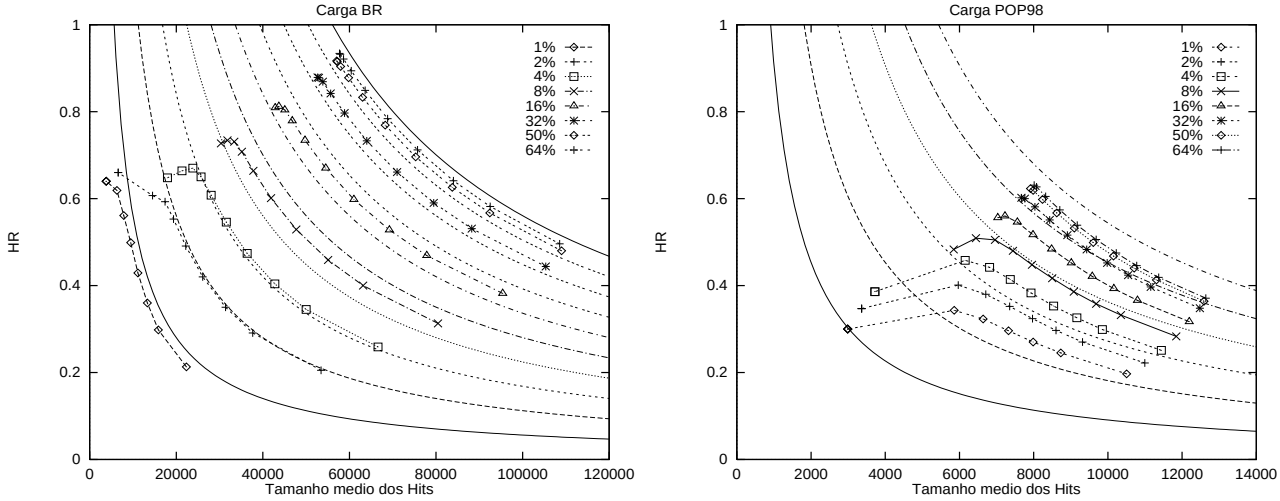

Figura 5.13: HR e BHR em função do tamanho do cache e do tamanho relativo das partições para as cargas BR e POP98. A legenda indica o tamanho do cache em porcentagem do tamanho de referência para a carga.

BHR aos aumentos. Enquanto HR for crescente e BHR for aproximadamente constante o tamanho da partição é aumentado. Em um cache com três ou mais partições o ajuste deve ser feito inicialmente na primeira partição e posteriormente na segunda partição porque a contribuição maior para HR é provida primeira partição.

5.3.5 Estimativa dos Parâmetros do Modelo PART

Os parâmetros do PART a serem definidos para a utilização do modelo são o número de partições, a definição de classes, o tamanho das partições e a política de reposição para cada partição, que devem ser definidos nessa ordem, conforme discutido no capítulo 4.

Nas subseções anteriores o comportamento do modelo PART foi analisado para cada variação individual de um dos seus parâmetros. Em função dos resultados apresentados é possível inferir algumas regras para a definição dos parâmetros. No entanto, deve ser ressaltado que não é conhecido nenhum procedimento para determinação do valor ótimo de cada parâmetro.

Número de Partições

Os benefícios do particionamento advêm da quebra da variabilidade dos tamanhos dos objetos substituídos. Nos experimentos realizados os maiores ganhos foram obtidos nos dois primeiros particionamentos. Particionamentos adicionais aumentam os benefícios porém em escalas progressivamente menores. Deve ser observado que os tamanhos dos arquivos nas cargas utilizadas nos experimentos variam de poucos bytes a dezenas de megabytes. Cargas com variações maiores podem se beneficiar de maior número de partições.

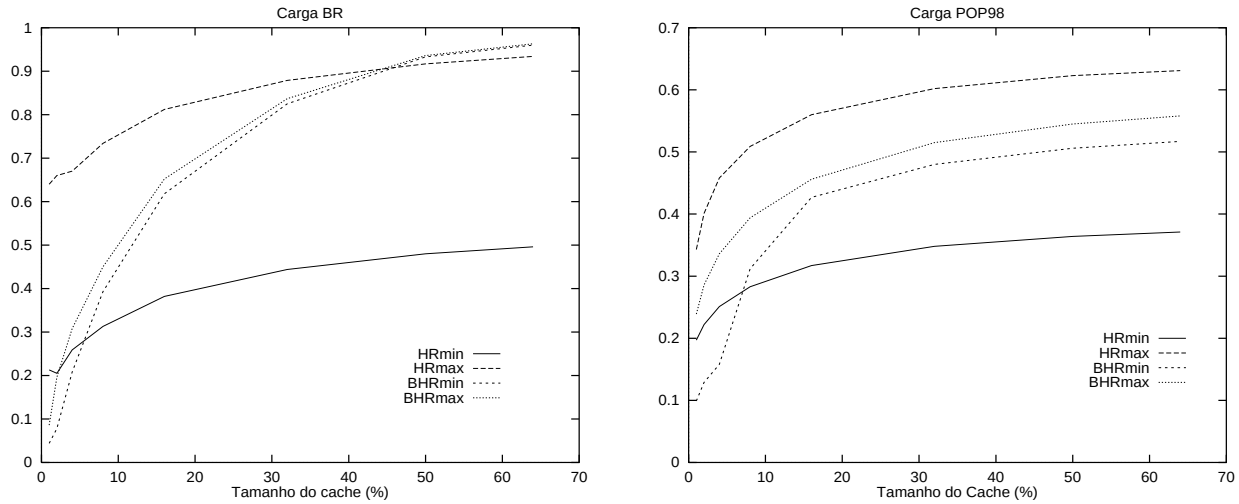

Figura 5.14: Limites encontrados para HR e BHR em função do tamanho do cache para as cargas BR (esquerda) e POP98 (direita).

Definição de Classes

Novamente o objetivo é minimizar a variabilidade dos objetos trocados no cache. Por isso a definição deve procurar manter objetos de tamanho homogêneo nas primeiras partições. Isto é garantido pela própria definição das classes. Adicionalmente, a definição de classes pode ser feita com o objetivo de balancear a carga de E/S entre as diversas partições, com consequente melhoria no desempenho, conforme argumentado na seção 4.6.

Tamanhos das Partições

O tamanho das partições deve ser utilizado para ajustar o desempenho em HR conforme descrito anteriormente. No ambiente real, o controle do tamanho das partições pode ser feito pelo tempo médio de permanência dos arquivos nas partições (LRU-*age*). Se esse tempo é muito grande para alguma partição, em comparação com as demais, seu tamanho pode ser diminuído em favor de outra partição com menor tempo médio de permanência. Esta estratégia permite fazer ajustes em função de mudanças nas características da carga.

Políticas de Reposição

A seção 5.3.1 mostrou que o melhor compromisso entre HR e BHR é obtido pelo cache particionado, com a política que otimiza HR na partição menor, responsável pelo maior número de *hits*, e a política que otimiza BHR nas demais partições. Para obter o melhor desempenho em HR a opção é utilizar o cache não particionado com uma política baseada em tamanho. Para obter o melhor desempenho em BHR, a opção é também utilizar o cache não particionado porém com política baseada em freqüência. No entanto, ambas as soluções não apresentam desempenho satisfatório na métrica que não otimizam. O melhor desempenho global é obtido com o particionamento. A próxima seção discute mais essa questão.

5.4 Utilização dos Mapas de Desempenho

Os mapas de desempenho são construídos para cada carga individual, em função do tamanho médio das requisições conforme descrito no capítulo 4. Nessa seção apresentamos em mapas de desempenho os resultados do experimento descrito na seção 5.3.1 para as cargas BR e U. Os experimentos são execução do cache não particionado para cada uma das políticas GD-Size, LRU e LFU-DA, e execução do PART com as políticas GD-Size na primeira partição e LFU-DA nas demais. As linhas legendadas representam os resultados de cada experimento. Cada ponto marcado representa um tamanho de cache, sendo o ponto mais perto da origem o tamanho de 1% e os demais são 2%, 4%, 8%, 16%, 32%, 50% e 64% do tamanho de referência.

Nos mapas apresentados na figura 5.15, os pontos estão bem próximos ou mesmo sobrepostos para os maiores caches. Estes mapas revelam as estratégias de cada política. GD-Size apresenta crescimento mais significativo em HR para tamanhos menores de cache. Para os caches maiores, quando os valores de HR estão perto do máximo possível, o BHR ganha aumentos mais expressivos. As políticas não baseadas em tamanho, LFU-DA e LRU, fazem o caminho inverso. Apresentam maior ganho em BHR nos primeiros aumentos de tamanho e posteriormente apresentam crescimento expressivo em HR.

O modelo PART apresenta um comportamento intermediário, com crescimentos mais homogêneos em ambas as métricas. Em termos matemáticos, podemos definir a taxa média de variação de cada curva como a inclinação da reta traçada entre os pontos inicial e final da respectiva curva. Para o PART a taxa média é aproximadamente igual à inclinação da curva em cada ponto (derivada constante diferente de zero). O mesmo não ocorre para as demais curvas.

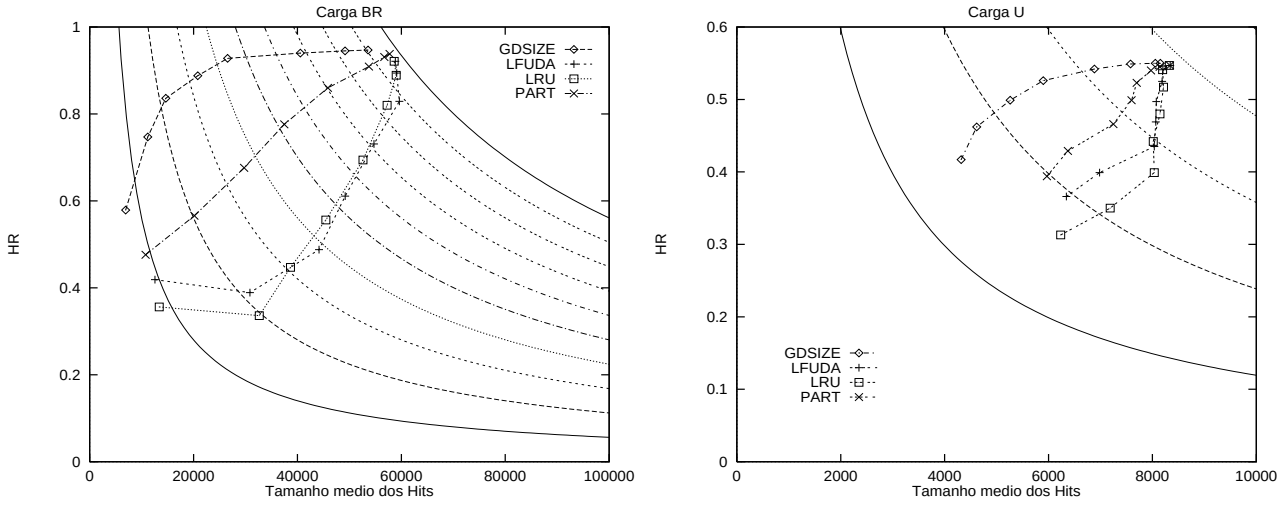

Figura 5.15: Mapa de desempenho para as cargas BR (esquerda) e U (direita).

5.5 Sumário dos Resultados e Conclusão

Neste capítulo foram descritos os estudos sobre o desempenho e o comportamento dos modelos de cache particionado e não particionado, e foram analisados os resultados destes estudos, baseados em simulação. Os resultados provêem respostas para as perguntas que esta tese procura responder, colocadas no capítulo 1.

A primeira pergunta é como a variabilidade nos tamanhos dos arquivos afeta o desempenho dos caches. Nos modelos tradicionais (não particionados) de organização de cache não há relação entre o tamanho do arquivo que é retirado e o tamanho do arquivo que será inserido. Como conseqüência podem ocorrer dois eventos: a retirada de muitos arquivos para a inserção de um arquivo muito grande ou a retirada de um arquivo muito grande para a inserção de um pequeno. Ambas as ocorrências são prejudiciais ao cache. A primeira implica na retirada de até milhares de documentos, alterando o estado do cache e prejudicando a ocorrência de possíveis *hits* no futuro para os documentos retirados. Na segunda situação o cache fica com um espaço vazio que pode ser enorme, o que tem o efeito instantâneo de uma diminuição no tamanho do cache seguida de uma série de inserções sem retirada porque há espaço livre, o que caracteriza um estado transiente. Essas ocorrências são observadas no instante da chegada de uma requisição mas seus efeitos são notados no desempenho do cache. Como observado nos experimentos, esses efeitos são mais graves em cargas com maior variabilidade.

Nesse capítulo foi mostrado que o cache não particionado, com espaço indivisível e regra única para classificar os documentos, remove mais arquivos do cache e faz mais inserções, porque gera mais *misses*. Além disso esse cache também mantém em média menos arquivos e por menos tempo. Outra medida a ser considerada é o número de arquivos retirados em cada substituição, que oscila muito no cache não particionado, com picos de retiradas intercalados por inserções sem nenhuma substituição. Essa característica irregular também pode prejudicar o desempenho do cache. Pode-se concluir que o tamanho domina as decisões sobrepondo-se às outras características que preservam o *working set* do cache, como recentidade e freqüência. Isso ocorre mesmo quando as políticas utilizadas procuram preservar por pressuposto a localidade, como LRU.

O outro problema gerado pela variabilidade é o compromisso entre as métricas HR e BHR. No modelo não particionado o espaço do cache é igualmente compartilhado entre arquivos de todos os tamanhos. Utilizando regras baseadas em tamanho, as políticas de reposição podem otimizar HR e utilizando regras baseadas em freqüência e recentidade é possível otimizar BHR. No entanto, a utilização de uma regra única para alcançar objetivos contraditórios é inviável.

O modelo PART apresenta soluções para os problemas gerados pela variabilidade. O PART impõe restrições baseadas em tamanho para as substituições no cache obtendo com isso maior regularidade e melhor desempenho. Como foi mostrado em comparação com o cache tradicional, o PART manteve mais arquivos no cache, fez menos inserções e retiradas e reteve os arquivos por um tempo maior. Como conseqüência obteve melhores valores para HR no experimento em que ambos os modelos utilizaram a política LRU. Adicionalmente, a classificação por tamanhos é extremamente adequada para a implementação de políticas

específicas para otimização de cada métrica. Com isso é possível atingir o objetivo de obter bom desempenho simultaneamente em HR e BHR.

A utilização dos mapas de desempenho foi essencial para provar que a estratégia do PART é procurar obter bom desempenho nas duas métricas e para mostrar as estratégias de outros modelos de gerência de cache. Neste sentido a utilização dos mapas de desempenho parece bastante promissora.

A desvantagem do modelo é a parametrização empírica. O modelo proposto foi avaliado com parâmetros definidos a partir de poucos experimentos. A pergunta natural é como definir parâmetros ótimos, isto é, que produzem o melhor desempenho para uma dada carga. Esta é uma questão para os trabalhos futuros. No entanto, essa interrogação deixa espaço para pensarmos que se os parâmetros definidos nos experimentos não são os parâmetros ótimos, então resultados ainda melhores do que os apresentados nesse capítulo podem ser obtidos pelo PART.

Capítulo 6

Conclusões e Direções para Trabalhos Futuros

A área de pesquisa desta tese é a gerência de espaço nos caches da WWW, que trata de dois aspectos, a organização do espaço do cache e a política de reposição. Esta tese apresentou um estudo original sobre como a característica de variabilidade nos tamanhos dos objetos da WWW influencia o desempenho dos seus sistemas de cache, e uma solução para o problema gerado por essa variabilidade, o modelo de organização do espaço denominado PART. Este capítulo conclui a tese com um sumário do trabalho feito, uma reflexão sobre as contribuições e relevância desta tese e sugestões para trabalhos futuros.

6.1 Sumário dos Resultados e Contribuições

Os resultados que esta tese apresenta podem ser divididos em dois grupos: os resultados que contribuem para o entendimento do efeito da variabilidade dos tamanhos dos arquivos no desempenho dos sistemas de cache e a solução proposta para minimizar este efeito.

Essencial para ambos os grupos é a caracterização da carga apresentada no capítulo 3. Essa caracterização foi necessária para demonstrar o pressuposto do trabalho, que existe grande variabilidade nos tamanhos dos arquivos que circulam na WWW. Foi observado que a variabilidade está presente em todas as dez cargas analisadas. Esse capítulo apresentou também uma caracterização das cargas por classes de tamanhos. Através da caracterização e de experimentos foi demonstrado que as classes apresentam características muito distintas. Essa foi a primeira indicação de que o particionamento por tamanho pode gerar ganhos pelo aproveitamento de características específicas de cada classe. Esse capítulo confirmou a existência de uma porcentagem significativa de arquivos e bytes que são requisitados apenas uma vez. A existência desta característica representa um limite importante para o desempenho dos sistemas de cache e reflete o componente de larga escala na WWW, a composição de milhões de usuários e objetos.

Esta tese apresenta importante contribuição para o entendimento do efeito da variabilidade dos tamanhos dos arquivos no desempenho dos sistemas de cache. Essa contribuição

é apresentada de três formas. A primeira é o modelo e a teoria de quantificação do compromisso entre HR e BHR, baseada no problema de otimização da mochila. Esse modelo propõe uma forma de quantificar o conceito que era intuitivo revelando através de gráficos o compromisso entre HR e BHR para cada carga.

A segunda contribuição é a fórmula que expressa o relacionamento entre HR e BHR. Essa fórmula traz duas contribuições. A partir dela é possível construir mapas de desempenho e apresentar os resultados de desempenho em HR e BHR de um modelo de gerência de espaço em um único gráfico, em vez de um gráfico para cada métrica. Isso torna possível uma comparação completa do desempenho, com a possibilidade de uma única visão global do comportamento das estratégias, em vez de uma visão segmentada dada pelos gráficos individuais de HR e BHR. A segunda contribuição da fórmula é mostrar que a relação entre as medidas de HR e BHR depende do tamanho médio dos *hits*, que é determinado pela estratégia de organização e reposição do cache. Esse conhecimento deve orientar a construção de novos modelos de gerência de espaço de caches.

A terceira contribuição da tese é o modelo PART de organização do espaço do cache. Esse modelo tem um objetivo bem definido que é minimizar o efeito da variabilidade. O PART cumpre esse objetivo impondo restrições de tamanho para a substituição de objetos no cache e permitindo a utilização de políticas específicas para incrementar o desempenho em cada métrica, minimizando o compromisso entre as mesmas. O modelo foi descrito e extensamente discutido no capítulo 4 e foi avaliado por simulação conforme descrito no capítulo 5. O PART atende a objetivos de desempenho distintos e obteve melhores resultados no conjunto das métricas do que as demais abordagens experimentadas. Há evidências de que o particionamento pode levar a um melhor desempenho computacional do cache devido à regularidade do comportamento, à possibilidade de balanceamento da carga entre as partições e de adequação dos parâmetros do sistema de arquivos para cada classe/partição.

O estudo do efeito da variabilidade dos tamanhos dos arquivos no desempenho dos sistemas de cache é importante pois apesar da diferença nos tamanhos dos arquivos já ser considerada nas propostas para cache da WWW, o impacto da grande variabilidade nestes tamanhos não havia sido considerado na literatura.

6.2 Relevância

Este trabalho pertence a um contexto maior que é o esforço para diminuir o *performance gap* de aplicações cujos tempos de resposta dependem essencialmente de operações de E/S. A Internet pode ser vista como um novo nível na hierarquia de memória. O disco local opera como cache para um disco remoto acessado via TCP/IP. Enquanto os tempos de acesso ao disco local são da ordem de milissegundos, o acesso à Internet é da ordem de segundos. O entendimento dos efeitos da variabilidade nos sistemas de cache da WWW, uma das contribuições desta tese, auxilia o projeto de sistemas com melhor desempenho, o que contribui para esse esforço.

Outro tópico importante de pesquisa é a escalabilidade da WWW, necessária para

viabilizar seu crescimento contínuo. Sistemas de caches são reconhecidamente um método fundamental nesse objetivo. O modelo PART proposto nesta tese obtém simultaneamente bom desempenho nas métricas HR e BHR, contribuindo ao mesmo tempo para reduzir a carga na rede e nos servidores e melhorar os tempos de resposta. Em resumo, contribui para melhorar a escalabilidade da WWW. Além disso, há evidências de que este modelo também contribui para o melhor desempenho computacional do próprio cache, levando a melhores tempos de resposta.

Há evidências empíricas de que a variabilidade dos tamanhos dos objetos Web tende a aumentar. Entre essas evidências estão o crescimento do tamanho médio dos arquivos, da mediana e do tamanho do maior arquivo no tempo. Há também grande interesse na utilização da WWW para transmissão de arquivos de áudio e vídeo, que são em geral arquivos muito grandes. Os arquivos de vídeo não implicam necessariamente na utilização de sistemas de vídeo sob demanda, que tratam da transmissão de vídeos de longa duração. Documentos grandes, em especial áudio e vídeo, são ao mesmo tempo um desafio e uma oportunidade para sistemas de cache. Arquivos grandes sobrecarregam a rede, demoram muito tempo para serem transmitidos e são geralmente estáticos, por isso são ideais para serem armazenados no cache.

Outro fator a ser considerado é a resposta dos usuários da WWW em relação a melhoramentos no sistema como, por exemplo, a implantação de redes mais rápidas. Os usuários tendem a mudar suas demandas focalizando dispositivos do sistema que podem oferecer melhor desempenho. À medida que os meios de transmissão tornam-se mais rápidos, as demandas por arquivos cada vez maiores poderão crescer. Isso implica num aumento da variabilidade nos tamanhos das transmissões. Neste contexto de aumento de variabilidade, o PART é uma proposta importante a ser considerada.

6.3 Direções para Trabalhos Futuros

Esta tese abre espaço para algumas direções de pesquisa. A determinação dos parâmetros do modelo PART foi feita de forma empírica. Uma direção para o trabalho é um estudo sobre como definir parâmetros ótimos para o PART, isto é, parâmetros que assegurem o melhor desempenho para uma dada carga. Não está claro se há possibilidade de encontrar um algoritmo ótimo para definir os parâmetros ou se esse é um problema NP-completo. A avaliação da aplicação do PART para os diversos tipos de cache (cliente, servidor e rede) também necessita mais estudo. A implementação do modelo PART em um sistema de cache distribuído permitirá a avaliação completa do efeito do particionamento. Além das métricas HR e BHR, outras métricas não focalizadas na simulação poderão ser avaliadas como, por exemplo, o desempenho do sistema de E/S.

Outra área de pesquisa que esta tese revela sem explorar é o controle de admissão nos caches da WWW. O fato de que grande parte dos documentos é requisitada apenas uma vez parece indicar o uso de controle de admissão para evitar que o cache seja inundado por documentos que não serão referenciados novamente.

A caracterização continuada da carga em longos períodos de tempo é necessária para

planejamento de capacidade. A carga da WWW ainda não é completamente entendida e, mais importante, não há uma situação de estabilidade na WWW. O crescimento em termos de usuários, documentos e volume de dados transmitidos é exponencial e a mudança nas características da carga devido a novas aplicações é frequente. Comércio eletrônico é um exemplo. É necessário conhecer a carga para predizer as cargas futuras possibilitando o planejamento de capacidade, de utilização dos recursos e a qualidade dos serviços.

Bibliografia

[Abdulla, 1998] Ghaleb Abdulla. *Analysis and Modeling of World Wide Web Traffic*. Tese de Doutorado, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, May 1998.

[Abrams *et al.*, 1995] Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Stephen Williams e Edward A. Fox. Caching Proxies: Limitations and Potentials. In *Electronic Proceedings of the Fourth International World Wide Web Conference: The Web Revolution*, http://www.w3.org/Conferences/WWW4, 1995.

[Aggarwal *et al.*, 1996] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu e Marina Epelman. On Caching Policies for Web Objects. Technical Report RC 20619, IBM Research Division — Thomas J. Watson Research Center, Yorktown Heights, New York, August 1996.

[Almeida *et al.*, 1996a] Virgílio Almeida, Jussara M. Almeida, Cristina D. Murta, Adriana A. Oliveira e Marco A. S. Mendes. Performance Analysis and Modeling of a WWW Internet Server. In *Proceedings of the Fourth International Conference on Telecommunication Systems*, March 1996.

[Almeida *et al.*, 1996b] Virgílio Almeida, Azer Bestavros, Mark Crovella e Adriana de Oliveira. Characterizing Reference Locality in the WWW. In *Proceedings of PDIS'96: The IEEE Conference on Parallel and Distributed Information Systems*, Miami Beach Florida, December 1996.

[Almeida *et al.*, 1996c] Virgílio A.F. Almeida, Cristina D. Murta e Jussara A. Marques. Performance Analysis of a WWW Server. In *Electronic Proceedings of the CMG'96 — Computer Measurement Group — International Conference on Technology Management and Performance Evaluation of Enterprise-Wide Information Systems*, San Diego, California, 1996.

[Almeida *et al.*, 1998] Virgílio Almeida, Márcio G. Cesário, Rodrigo C. Fonseca, Wagner Meira Jr. e Cristina D. Murta. The Influence of Geographical and Cultural Issues on the Cache Proxy Server Workload. In *Proceedings of the Seventh International World Wide Web Conference - edited by Computer Networks and ISDN Systems*, volume 30, p. 601–603, 1998.

[Archive, 1997] Internet Traffic Archive. Traces available in the Internet Traffic Archive . http://ita.ee.lbl.gov/html/traces.html, 1997.

[Arlitt *et al.*, 1998] Martin F. Arlitt, Rich Friedrich e Tai Jin. Performance Evaluation of Web Proxy Cache Replacement Policies. Technical report, HP Labs, http://fog.hpl.external.hp.com/techreports/index.html, 1998.

[Arlitt *et al.*, 1999] Martin Arlitt, Ludmilla Cherkasova, John Dilley, Rich Friedrich e Tai Jin. Evaluating Content Management Techniques for Web Proxy Caches. In *Second Workshop on Internet Server Performance - WISP'99*, Atlanta, Georgia, 1999. In conjunction with ACM SIGMETRICS 99.

[Arlitt e Williamson, 1996] Martin F. Arlitt e Carey L. Williamson. Web Server Workload Characterization: The Search for Invariants. In *Proceedings of the 1996 ACM Sigmetrics Conference*, p. 126–137, May 1996.

[Barford e Crovella, 1998] Paul Barford e Mark E. Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *Proceedings of Performance '98/ACM SIGMETRICS '98*, p. 151–160, Madison, Wisconsin, November 1998.

[Berners-Lee *et al.*, 1994] Tim Berners-Lee, Robert Cailliau, Ari Luotenen, Henrik Frystyknielsen e Arthur Secret. The World Wide Web. *Communications of the ACM*, 37(8), August 1994.

[Blaze, 1993] Matthew Addison Blaze. *Caching in Large Scale Distributed File Systems.* Tese de Doutorado, Princeton University, January 1993.

[Bolot *et al.*, 1997] Jean-Chrysostome Bolot, S. M. Lamblot e A. Simonian. Design of Efficient Caching Schemes for the World Wide Web. In V. Ramaswami e P. Wirth, editors, *Proceedings of the 15th International Teletraffic Congress ITC–15*, p. 403–412, 1997.

[Bolot e Hoschka, 1996] Jean-Chrysostome Bolot e Philipp Hoschka. Performance Engineering of the World Wide Web: Application to Dimensioning and Cache Design. In *Proceedings of the Fifth International Conference on the WWW*, Paris, France, 1996.

[Breslau *et al.*, 1999] Lee Breslau, Pei Cao, Li Fan, Graham Phillips e Scott Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proceedings of Infocom'99*, New York, USA, March 1999.

[Bush, 1945] Vannevar Bush. As We May Think. *The Atlantic Monthly*, July 1945.

[Cao e Irani, 1997] Pei Cao e Sandy Irani. GreedyDual-Size: A Cost-Aware WWW Proxy Caching Algorithms. In *Proceedings of the Second Web Caching Workshop*, Colorado, 1997.

[Cao, 1996] Pei Cao. *Application-Controlled File Caching and Prefetching.* Tese de Doutorado, Princeton University, 1996.

[Cao, 1997] Pei Cao. Disks Solve Web Scalability Problems. Panel Presentation, ICDCS'97 - The 17th International Conference on Distributed Computer Systems, May 1997. http://www.cs.wisc.edu/ cao/talks/icdcs97-panel/index.htm.

[Copeland *et al.*, 1988] George Copeland, William Alexander, Ellen Boughter e Tom Keller. Data Placement in Bubba. In *ACM SIGMOD*, p. 99–108, 1988.

[Crovella *et al.*, 1998a] Mark Crovella, Mor Harchol-Balter e Cristina D. Murta. Task Assignment in a Distributed System: Improving Performance by Unbalancing Load (poster paper). In *Proceedings of the 1998 ACM Sigmetrics Conference*, Madison, Wisconsin, June 1998.

[Crovella *et al.*, 1998b] Mark E. Crovella, Murad S. Taqqu e Azer Bestravos. Heavy-Tailed Probability Distributions in the World Wide Web. In *A Practical Guide to Heavy Tails*, chapter 1, p. 3–26. Chapman & Hall, New York, 1998.

[Crovella e Bestavros, 1995] Mark Crovella e Azer Bestavros. Explaining World Wide Web Traffic Self-Similarity. Technical Report TR-95-015, Boston University, CS Dept, Boston, MA 02215, August 1995.

[Crovella e Bestavros, 1996] Mark Crovella e Azer Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. In *Proceedings of the 1996 ACM Sigmetrics Conference*, 1996.

[Crovella e Lipsky, 1997] Mark E. Crovella e Lester Lipsky. Long-Lasting Transient Conditions in Simulations with Heavy-Tailed Workloads. In *Proceedings of the 1997 Winter Simulation Conference*, December 1997.

[Crovella e Taqqu, 1999] Mark E. Crovella e Murad S. Taqqu. Estimating the Heavy Tail Index from Scaling Properties. *Methodology and Computing in Applied Probability*, 1999.

[Cunha *et al.*, 1995] Carlos Cunha, Azer Bestavros e Mark Crovella. Characteristics of WWW Client-based Traces. Technical Report TR-95-010, Boston University, CS Dept, Boston, MA 02215, April 1995.

[Cunha, 1997] Carlos Cunha. *Trace Analysis and Its Application to Performance Enhancements of Distributed Information Systems.* Tese de Doutorado, Computer Science Department - Boston University, Boston, Massachusetts, 1997.

[Denning e Schwartz, 1972] Peter J. Denning e Stuart C. Schwartz. Properties of the Working-Set Model. *Communications of the ACM*, 15(3):191–197, March 1972.

[Denning e Slutz, 1978] Peter J. Denning e Donald R. Slutz. Generalized Working Sets for Segmented Reference Strings. *Communications of the ACM*, 21(9):750–759, September 1978.

[Denning, 1980] Peter J. Denning. Working Sets Past and Present. *IEEE Transactions On Software Engineering*, SE-6(1):64–84, January 1980.

[Denning, 1997] Peter J. Denning. *In the Beginning: Recollections of Software Pioneers*, chapter Before Memory was Virtual. IEEE Press, 1997.

[Esoterica, 1998] Esoterica. http://www.esoterica.pt, 1998.

[Greiner *et al.*, 1995] Michael Greiner, Manfred Jobmann e Lester Lipsky. The Importance of Power-tail Distributions for Telecommunication Traffic Models. Technical report, Institut Für Informatik, Technische Universität München, Germany, 1995.

[Harchol-Balter *et al.*, 1998] Mor Harchol-Balter, Mark Crovella e Cristina D. Murta. On Choosing a Task Assignment Policy for a Distributed Server System. In *Proceedings of Performance Tools'98: 10th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, Palma, Spain, September 1998.

[Harchol-Balter *et al.*, 1999] Mor Harchol-Balter, Mark E. Crovella e Cristina Duarte Murta. On Choosing a Task Assignment Policy for a Distributed Server System. *Journal of Parallel and Distributed Computing*, November 1999. accepted for publication.

[Infolibria, 1998] Infolibria. Web Performance Solutions. http://www.infolibria.com, 1998.

[Inktomi, 1998] Inktomi. Traffic Server - The Economics of Large Scale Network Caching. http://www.inktomi.com, 1998.

[International Data Corporation, 1998] International Data Corporation. National Laboratory for Applied Network Research (NLANR). http://www.idc.com, 1998.

[Jain, 1991] Raj Jain. *The Art of Computer Systems Performance Analysis.* John Wiley & Sons, 1991.

[Karedla *et al.*, 1994] Ramakrishna Karedla, J. Spencer Love e Bradley Wherry. Caching Strategies to Improve Disk System Performance. *IEEE Computer*, p. 38–46, March 1994.

[Kleinrock, 1975] L. Kleinrock. *Queueing Systems – Volume 1 – Theory.* Wiley, New York, 1975.

[Loukides, 1990] Mike Loukides. *System Performance Tuning.* O'Reilly & Associates, Inc., 1990.

[Madison e Batson, 1976] A. Wayne Madison e Alan P. Batson. Characteristics of Program Localities. *Communications of the ACM*, May 1976.

[Manley e Seltzer, 1997] Stephen Manley e Margo Seltzer. Web Facts and Fantasy. http://www.eecs.harvard.edu/ vino/web/sits.97.html, 1997.

[Markatos, 1996] Evangelos P. Markatos. Main Memory Caching of Web Documents. In *Proceedings of the Fifth International WWW Conference*, 1996.

[Meira Jr. *et al.*, 1998a] Wagner Meira Jr., Erik L. S. Fonseca, Cristina D. Murta e Virgílio A. F. Almeida. Analyzing Performance of Cache Server Hierarchies. In *Anais do XVIII International Conference of the Chilean Society of Computer Science*, p. 113–121, 1998.

[Meira Jr. *et al.*, 1998b] Wagner Meira Jr., Erik L. S. Fonseca, Cristina D. Murta e Virgílio A. F. Almeida. Performance Analysis of WWW Cache Proxy Hierarchies. *Journal of the Brazilian Computer Society*, 5(2):16–30, November 1998.

[Menascé *et al.*, 1994] Daniel A. Menascé, Virgílio A. F. Almeida e Larry W. Dowdy. *Capacity Planning and Performance Modeling — From Mainframes to Client-Server Systems.* Prentice Hall, 1994.

[Menascé e Almeida, 1998] Daniel A. Menascé e Virgílio A. F. Almeida. *Capacity Planning for Web Performance - Metrics, Models, and Methods.* Prentice Hall, Upper Saddle River, New Jersey, 1998.

[Metcalfe, 1997] Bob Metcalfe. Internet Futures. http://www.camcon.org/bobtalk.htm, 1997.

[Murta *et al.*, 1996] Cristina D. Murta, Jussara A. Marques e Virgílio A. F. Almeida. Análise de Desempenho de Um Servidor WWW. In *Anais do XXIII Seminário Integrado de Software e Hardware — Semish'96*, p. 391–402, 1996.

[Murta *et al.*, 1998a] Cristina D. Murta, Virgílio A. F. Almeida e Wagner Meira Jr. Cache Particionado: Uma Nova Abordagem para Cache na WWW. In *Anais do XXV Seminário Integrado de Software e Hardware — Semish'98*, p. 259–275, 1998.

[Murta *et al.*, 1998b] Cristina Duarte Murta, Virgílio Almeida e Wagner Meira Jr. Analyzing Performance of Partitioned Caches for the World Wide Web. Position paper at the Third International WWW Caching Workshop, June 1998.

[Murta *et al.*, 1999] Cristina D. Murta, Virgílio Almeida e Wagner Meira Jr. Efficient Storage Management in World Wide Web Caches. In P. Key e D. Smith, editors, *Teletraffic Engineering in a Competitive World - Proceedings of the ITC'16 - 16th International Teletraffic Congress*, volume 3 of *Teletraffic Science and Engineering*, p. 1189–1198, Edinburgh, UK, June 1999. Elsevier Science.

[Murta e Almeida, 1998] Cristina D. Murta e Virgílio Almeida. Characterizing Response Time of WWW Caching Proxy Servers. In *Proceedings of the Workshop on Workload Characterization, realizado em conjunto com o IEEE/ACM Micro'31, International Symposium on Microarchitecture*, p. 110–116, Dallas, Texas, November 1998.

[Murta e Almeida, 1999] Cristina Duarte Murta e Virgilio A. F. Almeida. Caches na WWW: Limitações e Potencial. In *11th Symposium on Computer Architecture and High Performance Computing*, September 1999. Accepted for publication in the Proceedings.

[NASA, 1995] NASA. National Aeronautics and Space Administration. http://www.nasa.gov, 1995.

[Nelson, 1965] Ted Nelson. A File Structure for the Complex, the Changing and the Indeterminate. In *Proceedings of he ACM 20th National Conference*, 1965.

[Niclausse *et al.*, 1998] Nicolas Niclausse, Zhen Liu e Philippe Nain. A New and Efficient Caching Policy for the World Wide Web. In *Proceedings of the 1998 Workshop on Internet Server Performance*, June 1998.

[NLANR, 1999] NLANR. http://ircache.nlanr.net, 1999.

[Now, 1997] Campanha Cache Now. http://vancouver-webpages.com/CacheNow/, 1997.

[Patterson e Hennessy, 1996] D. A. Patterson e J. L. Hennessy. *Computer Arquitecture: A Quantitative Approach, 2nd Edition.* Morgan Kaufmann Publishers, California, 1996.

[Pisinger, 1995] David Pisinger. *Algorithms for Knapsack Problems.* Tese de Doutorado, Department of Computer Science - University of Copenhagen, Universitetsparken 1, DK-2100, Copenhagen, Denmark, February 1995.

[Pisinger, 1997] David Pisinger. A Minimal Algorithm for the 0-1 Knapsack Problem. *Operations Research*, 45:758–767, 1997.

[Pitkow, 1998] James E. Pitkow. Summary of WWW Characterizations. Proceedings of the Seventh World Wide Web Conference, 1998.

[POP-MG, 1998] POP-MG. Ponto de Presença da RNP em Minas Gerais. http://www.pop-mg.rnp.br/index.html, 1998.

[Rizzo e Vicisano, 1998] Luigi Rizzo e Lorenzo Vicisano. Replacement Policies for a Proxy Cache. http://www.iet.unipi.it/ luigi/research.html, 1998.

[Robinson e Devarakonda, 1990] J. Robinson e M. Devarakonda. Data Cache Management Using Frequency-Based Repalcement. In *Proceedings of the 1990 ACM SIGMETRICS Conference*, p. 134–142, Boulder, Colorado, May 1990.

[Sedayao, 1994] Jeff Sedayao. "Mosaic Will Kill My Network!" - Studying Network Traffic Patterns of Mosaic Use. In *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/sedayao/mostrafpaper.html, 1994.

[Squid, 1997] Squid. Squid Internet Object Cache. http://squid.nlanr.net/Squid, 1997.

[Tanenbaum, 1992] Andrew S. Tanenbaum. *Modern Operating Systems.* Prentice Hall, Englewood Cliffs, N.J., 1992.

[Thompson *et al.*, 1997] Kevin Thompson, Gregory J. Miller e Rick Wilder. Wide-Area Internet Traffic Patterns and Characteristics. *IEEE Network*, 11(6):10–23, Nov/Dec 1997.

[Virginia Tech, 1995] Virginia Tech. Traces from Virginia Polytechnic Institute. http://www.cs.vtech.edu, 1995.

[Wagner, 1996] Thomas A. Wagner. Towards an Empirical Model of WWW Site Response Times. Technical Report TR 96-19, Department of Computer Science - University of Massachussetts, DCC-UMASS Amherst, March 1996.

[Wessels e Claffy, 1997] Duane Wessels e K. Claffy. ICP and the Squid Object Cache. http://ircache.nlanr.net/Cache/reading.html, 1997.

[Wessels, 1995] Duane Wessels. Intelligent Caching for World Wide Web Objects. In *INET'95 - Internet Society's International Networking Conference*, Honolulu, Hawaii, 27-30 June 1995. Hypermedia Proceedings: http://www.isoc.org/inet95/proceedings/.

[Williams *et al.*, 1996] Stephen Williams, Marc Abrams, C. R. Standridge, Ghaleb Abdulla e Edward A. Fox. Removal Policies in Network Caches for World Wide Web Documents. *ACM Sigcomm*, p. 293–305, August 1996.

[Wooster e Abrams, 1997] Roland P. Wooster e Marc Abrams. Proxy Caching That Estimates Page Load Delays. In *Sixth International World Wide Web Conference*, 1997.

[Yeager e McGrath, 1996] Nancy J. Yeager e Robert E. McGrath. *Web Server Technology.* Morgan Kaufmann Publishers, San Francisco, California, 1996.