Políticas de Gerenciamento de Web Caches

Rodrigo Machado Oliveira

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

Políticas de Gerenciamento de Web Caches

Rodrigo Machado Oliveira

Dezembro de 1999

Banca examinadora:

- Prof. Dr. Nelson Luís Saldanha da Fonseca (Orientador)
  Instituto de Computação – UNICAMP
- Prof. Dr. Luis Carlos Trevelin
  Departamento de Ciência da Computação – UFSCar
- Prof. Dr. Ricardo de Oliveira Anido
  Instituto de Computação – UNICAMP
- Prof. Dr. Celio C. Guimarães
  Instituto de Computação – UNICAMP

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Oliveira, Rodrigo Machado

OL4p Políticas de gerenciamento de Web caches / Rodrigo Machado Oliveira -- Campinas, [S.P. :s.n.], 1999.

Orientador : Nelson Luís Saldanha da Fonseca

Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. World Wide Web (Sistema de recuperação da informação). 2. Cliente/servidor (Computação). 3. Redes de computação. 4. Memoria cache. I. Fonseca, Nelson Luís Saldanha. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Instituto de Computação
Universidade Estadual de Campinas

Políticas de Gerenciamento de Web Caches

Este exemplar corresponde à redação final da dissertação de mestrado devidamente corrigida e defendida por Rodrigo Machado Oliveira, e aprovada pela banca examinadora.

Campinas, 22 de dezembro de 1999

Orientador: Nelson Luís Saldanha da Fonseca

Dissertação de mestrado apresentada ao Instituto de Computação da Universidade Estadual de Campinas – Unicamp, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 22 de dezembro de 1999, pela Banca Examinadora composta pelos Professores Doutores:

Prof. Dr. Luís Carlos Trevelin
DCC-UFSCar

Prof. Dr. Ricardo de Oliveira Anido
IC-UNICAMP

Prof. Dr. Nelson Luís Saldanha da Fonseca
IC-UNICAMP

Agradecimentos

Aos meus pais, Roberval e Rosana, e às minhas irmãs, Camila e Tatiana, que sempre me incentivaram em tudo que fiz. À minha esposa, Luciane, que tem me ajudado desde que entrou na minha vida, e nunca deixou de me apoiar e de estar ao meu lado. Foi graças ao seu amor e compreensão que consegui cumprir mais esta etapa da minha vida.

Ao meu orientador, Dr. Nelson Fonseca, por ter me ajudado tanto no andamento da tese e por ter sido paciente com os erros cometidos. Sua experiência foi fundamental para que eu pudesse concluir as idéias que compõem esse trabalho, e para que os textos tomassem forma de tese.

Finalmente, gostaria de agradecer aos amigos e colegas, principalmente àqueles que me acompanham desde a época da graduação. Eles foram responsáveis por diversas risadas e me ajudaram, em diversos momentos, a amenizar a carga do mestrado.

À Fundação de Amparo à Pesquisa do Estado de São Paulo – FAPESP e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, pelo apoio financeiro.

Ao Centro Nacional de Processamento de Alto Desempenho em São Paulo (CENAPAD – SP) pelo apoio tecnológico.

Resumo

Atualmente, a Web tornou-se um dos principais gargalos no desempenho da Internet. Pesquisas recentes mostram que o parâmetro de Qualidade de Serviço (QoS) mais importante para os usuários da Web é o tempo de resposta na recuperação de objetos. Uma das alternativas para reduzir a latência na recuperação de documentos é a replicação de documentos populares em repositórios fisicamente próximos aos usuários, o que é denominado *Web caching*. As políticas de gerenciamento de Web caches têm grande influência na Qualidade de Serviço. As políticas de expulsão definem quais documentos devem ser retirados da cache, a fim de liberar espaço para um novo documento a ser armazenado. As políticas de admissão, por sua vez, procuram determinar quais documentos podem ser armazenados na cache. Esta dissertação investiga o tempo de recuperação de documentos como chave em políticas de expulsão e de controle de admissão. São propostas diversas políticas com o objetivo de diminuir o tempo de resposta na recuperação de objetos da Web. Além disso, é feita uma investigação sobre os tempos entre misses na cache, na tentativa de caracterizar seu tipo de distribuição.

Abstract

Recent surveys indicate that Web users consider the retrieval time the most important Quality of Service parameter. Web caches have been massively adopted in the Internet in order to reduce the retrieval time of documents as well as to alleviate the ever increasing Internet traffic due to Web traffic. Web cache management policies have great impact on the perceived Quality of Service. Removal policies define which documents should be removed from the cache, to make room for an incoming document. Admission control policies try to determine which documents can be stored in the cache. This dissertation investigates retrieval time of web documents as key for removal and admission control policies. Several policies are proposed. Moreover, the distribution of intermiss time is investigated.

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

A World Wide Web, ou Web como também é chamada, é uma estrutura arquitetônica que permite o acesso a documentos vinculados espalhados por milhares de máquinas na Internet. A Web é baseada no modelo cliente-servidor, onde um usuário (cliente) faz pedidos de documentos para o servidor através de um programa que busca, interpreta e mostra estes documentos, chamado de browser [Tanen97].

A grande explosão de popularidade da Web tem levado a um aumento considerável no tráfego da Internet [AgWoYu97]. Como resultado, a Web se tornou um dos principais gargalos no desempenho da rede. Quando são feitos pedidos para servidores em *links* lentos, existe geralmente uma demora considerável na recuperação de objetos distribuídos. Além disto, a alta taxa de transferência de documentos pela rede leva a um aumento de tráfego, que acaba reduzindo a largura de banda disponível e também introduzindo atrasos perceptíveis pelo usuário.

Existem diversos métodos para reduzir o uso de banda passante e, consequentemente, a latência na recuperação de documentos percebida pelos usuários. Uma das alternativas é a replicação de documentos populares em repositórios fisicamente próximos aos usuários. Tal solução é designada Web caching, e visa diminuir a utilização de links, bem como o tempo de acesso a estes documentos. No entanto, quando projetadas inadequadamente, Web caches podem degradar a Qualidade de Serviço perceptível ao usuário, já que é gasto tempo de processamento para gerenciamento das caches.

Uma Web cache é por definição um repositório finito de documentos de tamanho variável. A idéia é que esses documentos possam ser usados quando o usuário repetir um acesso ao documento, diminuindo assim o tempo de recuperação, já que estão mais próximos ao usuário. Outro uso é o compartilhamento de caches por um grupo de usuários, de forma que o acesso de um usuário possa eliminar a necessidade de acesso remoto à fonte do documento para atender os acessos feitos por outros usuários da mesma cache.

Políticas de gerenciamento de Web caches determinam se um documento irá entrar ou não na cache, quais documentos devem sair da cache para liberar espaço para um novo documento, etc. Essas políticas têm grande influência na Qualidade de Serviço. Políticas de expulsão definem quais documentos devem ser retirados da cache, a fim de liberar espaço para outro documento mais preeminente em um determinado momento. Estudos indicam que a informação utilizada como critério de expulsão tem um grande impacto no desempenho do sistema [WiAbSt96]. Todavia, a grande maioria dos trabalhos realizados até então investiga o impacto em métricas única e exclusivamente relacionadas ao desempenho do sistema, tais como hit ratio. Poucas são as pesquisas que objetivam explicitamente a melhoria de parâmetros de Qualidade de Serviço, tais como tempo médio de resposta por transação [WooAbr97] [BolHos96].

O 10th WWW User Survey [UserSur98] conduzido pelo Graphic, Visualization & Usability Center (Georgia Institute of Technology) e endossado pelo WWW Consortium confirma que o tempo de resposta por transação é o parâmetro de QoS mais importante para os usuários, correspondente a opinião de 61,4% do total dos entrevistados, e de 67,3% dos entrevistados que residem fora dos EUA e da Europa. Define-se como tempo de resposta por transação a diferença temporal entre os instantes de solicitação do documento e o instante em que o mesmo se torna disponível. O tempo de resposta de um pedido depende da existência ou não de uma cópia do documento na cache. Caso exista uma cópia do documento na cache, o tempo de resposta é mínimo. Do contrário, deve-se trazer uma cópia do documento de um servidor remoto para a cache, o que torna o tempo de resposta tipicamente muito mais elevado. Em cenários com um alto miss ratio, o tempo de resposta é tipicamente determinado pelo tempo de recuperação de um documento. Altos tempos de resposta podem levar a um alto índice de evasão de usuários. Além disto, grandes variações do tempo de resposta podem causar irritabilidade no usuário devido a falta de previsão de comportamento do sistema.

Nessa dissertação, investiga-se o tempo de recuperação de um documento como chave em políticas de expulsão de Web caches. Apesar de estudos anteriores abordarem este tópico, acredita-se que os resultados existentes não são conclusivos, como apontam os próprios autores de [BolHos96]. Diversas são as contribuições do presente trabalho. Dentre elas, pode-se mencionar:

- Um estudo sobre o uso do tempo de recuperação em políticas de expulsão em Web caches, baseado em um espaço amostral significativamente maior que os espaços amostrais usados em estudos anteriores [WooAbr97] [BolHos96];
- Avaliação da adoção de políticas de expulsão de documentos antigos mesmo que a chave de expulsão não determine a retirada do documento da cache em um determinado momento;
- Investigação sobre o uso de controle de admissão de documentos na cache.
- Estudo do tempo entre misses consecutivos na cache, na tentativa de caracterizar o tipo de distribuição desses tempos e possivelmente usar os resultados em algoritmos de *prefetching*

Esta dissertação está organizada da seguinte forma. No capítulo 2, fornece-se uma revisão bibliográfica dos tópicos relacionados com a dissertação. O capítulo 3 discorre sobre os experimentos com as políticas de gerenciamento de caches na Web, descrevendo as políticas estudadas e resultados obtidos. No capítulo 4 estudos caracterizam os tempos entre misses na cache. Finalmente, no capítulo 5 apresenta-se conclusões da dissertação.

Capítulo 2

Revisão de literatura

Neste capítulo serão descritos alguns trabalhos existentes na literatura para que se possa situar os problemas que foram investigados nesta dissertação. Inicialmente serão mostrados projetos de caches individuais e de caches cooperativas. A seguir, será discutido o comportamento do tráfego gerado por requisições a servidores da Web e estudos sobre os tipos de distribuições encontrados no tráfego da Web.

2.1. Caches individuais

Caching provou ser uma técnica bastante útil na redução da latência experimentada pelo usuário final na Web [WesCla98]. O conceito fundamental é o armazenamento de cópias de documentos populares em máquinas mais próximas ao usuário. As caches podem estar localizadas em vários pontos da rede, como [AgWoYu97]:

- Cliente: são caches implementadas na maioria dos browsers, podendo armazenar apenas os acessos da sessão corrente (cache não-persistente) ou reter os documentos entre invocações do browser (cache persistente)

• Proxy: são servidores especializados que agem como agentes com o objetivo de encontrar uma cópia em cache de um documento

Uma das primeiras versões de *caching* de objetos na Web foi o armazenamento de cópias dos documentos no cliente, em disco ou na memória [BaMoRo97]. A desvantagem de aplicar esta abordagem isoladamente torna-se óbvia à medida que os documentos da Web se tornam cada vez mais diversificados. Uma abordagem mais interessante para a natureza da Web é o uso de *caches* compartilhados por vários usuários e/ou usuários com mesmos interesses. O uso de servidores proxy para *caching* tem como vantagens a redução do número de pedidos para os servidores populares e a redução do volume de tráfego e da latência percebida pelos usuários.

Diversos estudos que tentaram caracterizar o tráfego da Web mostraram que *caching* de documentos Web pode melhorar o desempenho dos servidores. Tais estudos mostram a existência de um grande número de referências para um número pequeno de documentos e mostram que os tamanhos destes documentos são geralmente bem pequenos.

As políticas tradicionais para *caching*, usadas em sistemas de arquivos, não têm bom desempenho quando usadas para objetos Web, pelas seguintes razões [Markat96]: a granulosidade da *cache* é diferente, pois as políticas tradicionais colocam blocos de arquivos na *cache*, enquanto a *cache* na Web deve guardar todo o documento; e os documentos Web são acessados em um modo somente-leitura, diferente dos sistemas de arquivos que tem que lidar com tráfego somente-leitura e de leitura-escrita, o que introduz uma complexidade adicional para esses sistemas.

Diversos estudos mostraram características específicas na implementação de *caching* para documentos Web, como:

• Documentos Web não têm necessariamente o mesmo tamanho, que é um dos principais motivos da inviabilidade do uso dos mecanismos clássicos de *caching* em memória ou sistemas de arquivos distribuídos;

• O *caching* de objetos multimídia agrava ainda mais o problema anterior, pois neste caso temos ainda objetos com tamanhos grandes e a necessidade de manter uma certa relação entre os blocos da *cache*, para que estes objetos possam ser entregues de maneira contínua [DanSit96]. Além disto, quando é necessária a troca de objetos na *cache*, o tamanho dos objetos multimídia pode ocasionar uma demora muito grande;

• O problema de *caching* em disco [AggaYu96] ocorre quando um novo objeto é inserido na *cache* e há a necessidade de remoção de mais de um objeto, sendo que estes objetos têm que estar posicionados contiguamente no disco, para permitir que o novo objeto ocupe essa posição contígua. Existe, então, a dificuldade na escolha de um grupo contíguo de blocos para serem removidos da *cache*. Existe também o problema da fragmentação gerada pelos pedaços de discos vazios que são muito pequenos para serem utilizados por outro objeto;

• Um dos grandes desafios em se construir um sistema de *caching* é que não se sabe de antemão quais páginas serão interessantes para o usuário, ou onde essas páginas estão localizadas, ou quando elas serão solicitadas [ZhFlJa97];

• A mudança no conteúdo das páginas Web [BaMoRo97] pode fazer com que os documentos armazenados na *cache* fiquem desatualizados. É desejável que as políticas de *caching* levem em conta a coerência dos objetos armazenados, que pode ser garantida, por exemplo, com a introdução de um tempo limite para que os objetos Web fiquem na *cache* ou com a confirmação, com o servidor do documento, da sua validade;

As *caches* de objetos Web, de uma maneira geral, implementam dois tipos de políticas: as políticas de admissão e as de substituição. A política (ou controle) de admissão define se um objeto deve ser inserido ou não na *cache*, pois nem sempre essa inserção é favorável. Por outro lado, a política de substituição ou remoção define os objetos que serão retirados da *cache* para ceder lugar ao objeto que está sendo inserido. Tanto as políticas de admissão quanto de substituição baseiam-se em dois componentes principais: na *predição* do tráfego futuro e na estimação do *valor* do documento [RoSoTa97].

Se dois objetos são acessados com a mesma freqüência, a taxa de acerto (*hit ratio*) é maximizada quando as políticas de admissão e substituição favorecem objetos menores, pois neste caso mais objetos poderão ser armazenados. Por outro lado, se os objetos maiores forem armazenados na *cache*, a probabilidade de se ter uma economia do tempo de transferência aumenta. Políticas de admissão devem levar em conta diversos fatores, tais como a freqüência de acesso, o tamanho dos objetos e a economia nos tempos de transferência. A seguir serão descritas algumas políticas de admissão/substituição existentes na literatura.

2.1.1. PSS (Pyramidal Selection Scheme)

A política PSS (Pyramidal Selection Scheme), proposta em [AgWoYu97], tem como idéia principal uma classificação piramidal dos objetos na *cache*, baseado nos seus tamanhos. Os objetos em cada grupo *i* desta classificação têm seus tamanhos limitados entre 2^{i-1} e $2^{i}-1$, e são ordenados seguindo a política LRU. Em outras palavras, um objeto com tamanho *t* será colocado no grupo $\log_2 t+1$. Sempre que um objeto tiver que ser excluído da *cache*, os objetos menos recentemente usados de cada grupo são comparados levando-se em conta seu tamanho (S) e sua freqüência dinâmica (ΔT), e os objetos que tiverem o maior valor de S x ΔT são escolhidos. Os objetos são retirados, um por vez, até que haja espaço para o novo objeto está sendo inserido na *cache*.

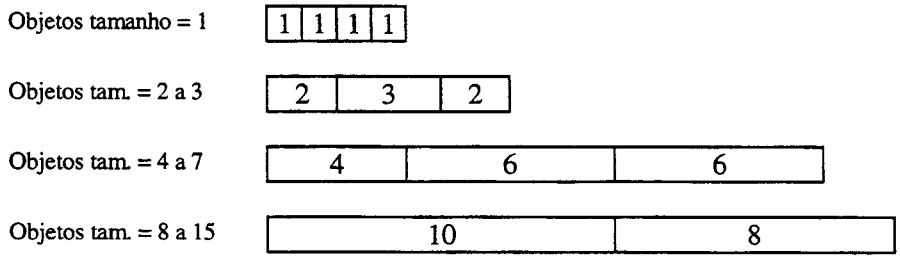

Figura 2.1: Exemplo da política PSS, que separa os objetos da cache em grupos, de acordo com seu tamanho. No exemplo, vemos 4 grupos, cada grupo i *tendo os tamanhos dos objetos que possui limitados entre* 2^{i-1} *e* $2^{i}-1$.

Esta política implementa também um controle de admissão, que usa uma cache auxiliar, mantida de acordo com a política LRU, guardando os documentos mais populares. Só é permitida a entrada na cache de documentos que estejam na cache auxiliar, ou seja, que sejam populares o suficiente.

Aggarwal et al [AgWoYu97] compararam a política PSS com uma variação da política LRU chamada de LRU-MIN [AbStAb95+], e a política SIZE, que retira da cache os documentos com maior tamanho. Foram usadas simulações baseadas em traces e baseadas em eventos. Os parâmetros de comparação foram a taxa de acertos na cache (*hit ratio*) e a economia nos tempos de transferência (*cost savings*). Os experimentos mostraram que a política PSS teve um bom desempenho nas simulações orientadas a eventos e a traces, onde foram usados traces coletados em proxies e em servidores. Na maioria dos casos, PSS superou as outras políticas para os parâmetros de *hit ratio* e *cost savings*.

2.1.2. Algoritmo de substituição baseado no tempo de recuperação dos documentos

Bolot e Hoschka [BolHos96] propõem um algoritmo de substituição em *cache* que objetiva minimizar o tempo de recuperação de documentos, explicitamente levando em conta o tamanho dos documentos e a carga da rede. Este algoritmo tenta minimizar a probabilidade de expulsão de objetos com alta latência de recuperação. Assim sendo, a política armazena, para cada objeto, os seguintes parâmetros: t_i, tempo em que o documento foi referenciado pela última vez; S_i, tamanho do documento; rtt_i, tempo levado para recuperar o documento; e ttl_i, tempo de validade do documento (*time-to-live*).

O algoritmo proposto em [BolHos96] usa a seguinte função para medir o valor de cada documento:

$$W(t_i, S_i, rtt_i, ttl_i) = (w_1 rtt_i + w_2 S_i) / ttl_i + (w_3 + w_4) / t_i$$

W_1, w_2, w_3 e w_4 são constantes, cujos valores devem ser determinados dependendo do objetivo do algoritmo de *caching*. Com essas variáveis de estado, é possível modelar a maioria dos algoritmos existentes. Por exemplo, a função de peso $W(t_i, S_i, rtt_i, ttl_i) = 1 / t_i$ modela a política LRU, pois os documentos têm o peso inversamente proporcional ao tempo de último acesso ao documento.

Resultados de simulações *trace-driven*, usando a taxa de erros (*miss ratio*) na *cache* como medida de desempenho, mostraram que o algoritmo proposto obteve melhor desempenho que o algoritmo LRU. Nesse estudo, foi evidenciada a vantagem de se usar um algoritmo de substituição baseado no tempo de recuperação dos documentos.

Com base no trabalho de Bolot e Hoschka, Wooster e Abrams [WooAbr97] propuseram duas políticas que usavam o tempo de recuperação de um documento. O valor do tempo de recuperação é uma estimativa com base no algoritmo de estimação do RTT (*round-trip time*) do protocolo TCP. Foram feitas comparações com as políticas LRU, LFU e SIZE, usando os parâmetros hit ratio (taxa de acertos na cache), weighted hit ratio (taxa dos bytes pedidos que foi atendida pela cache) e time (tempo médio que um usuário leva para carregar um documento).

A primeira política proposta (LAT) considera apenas o tempo de recuperação dos documentos, e teve desempenho pior que LRU, LFU e SIZE. A segunda política (HYB) é uma política híbrida que usa o tempo de recuperação do documento, a largura de banda da conexão para o servidor de origem do documento, a freqüência de acesso do documento e o seu tamanho. Os autores chegaram à conclusão de que HYB é mais robusto, e na maioria das vezes teve o melhor desempenho considerando diversas medidas de desempenho. Todas as medidas usadas nas políticas foram estimadas com base nos servidores de origem dos documentos, e não na URL completa dos mesmos.

2.1.3. Políticas de remoção vistas como procedimentos de ordenação

Williams *et al* [WiAbSt96] propuseram uma política de remoção em duas fases: na primeira delas, a política ordena os documentos na *cache* de acordo com uma ou mais chaves (primária, secundária, etc.); na segunda fase, ela remove zero ou mais documentos do início da lista ordenada, até que um critério seja satisfeito, como a liberação de um determinado espaço na *cache*.

Foram usadas as seguintes chaves no estudo: SIZE, que representa o tamanho de cada documento na *cache*; $\lfloor \log_2(\text{SIZE}) \rfloor$, que é o piso do logaritmo (na base 2) de SIZE; ETIME, que é a hora que o documento entrou na *cache*; ATIME, que é a hora do último acesso do documento; DAY (ATIME), que é dia do último acesso ao documento; e NREF que é o número de referências ao documento. Com base nestas chaves, podem ser modeladas diversas políticas já conhecidas, como a política LRU, que usa a chave ATIME, a política FIFO, que usa a chave ETIME, ou a política LFU, que usa NREF como chave primária.

Para comparar as políticas, foram usados dois parâmetros, já detalhados anteriormente: hit ratio e weighted hit ratio. Resultados provenientes de experimentos de simulação mostraram que a chave primária que tem melhor desempenho é a chave SIZE. Por outro lado, a chave secundária que produz os menores *miss ratios* é a chave randômica (documento escolhidos aleatoriamente). O resultado de que as políticas baseadas no tamanho dos documentos têm melhor desempenho das que não usam essa medida foi confirmado por outros diversos trabalhos [WooAbr97, WiAbSt96, BolHos96].

2.1.4. Prefetching

A idéia por trás de prefetching é aproveitar o tempo ocioso que normalmente existe entre duas requisições de um usuário a documentos da Web. Este tempo pode ser aproveitado para tentar recuperar os arquivos que provavelmente serão pedidos em um futuro próximo, de forma que o tempo médio de resposta ao usuário possa ser diminuído [JiaKle98].

Apesar de *prefetching* poder ser eficaz na redução da latência, esta técnica pode acarretar um aumento do uso de banda passante. Essa desvantagem inerente é resultado do fato de que parte dos dados que serão buscados antecipadamente (*prefetched*) nunca serão pedidos pelo usuário. Para obter uma maior eficiência, um algoritmo de *prefetching* deve tentar prever os pedidos futuros com o maior grau de certeza possível [WesCla98].

O que se busca com o uso de prefetching é adicionar alguma inteligência às aplicações de rede, de modo que sempre que o usuário pedir um conjunto de informações (um arquivo, uma mensagem eletrônica, etc.), o sistema possa estimar quais informações adicionais serão necessárias nos próximos acessos do usuário, de acordo com algum critério [JiaKle98]. No caso das páginas Web temos uma informação valiosa, que são os links de outras páginas e arquivos referenciados pela página que está sendo visitada.

Os autores de [JacCao98] fazem um estudo da técnica de *pre-pushing*, na qual as decisões de prover informações ainda não solicitadas vem dos caching proxies, e não dos clientes. *Pre-pushing* aproveita o tempo de espera entre os pedidos dos clientes, e usa algoritmos que tentam prever quais serão os próximos pedidos. Estes possíveis pedidos são enviados aos usuários, e armazenados em um *buffer* no cliente. São feitas considerações quanto à quantidade de banda passante usada nessa técnica, já que esta considera pedidos de documentos feitos através de modems. Neste estudo, é avaliada a redução de latência percebida pelo usuário.

2.2. Caches cooperativas

Um grupo de caches Web pode se beneficiar do compartilhamento de outras caches do mesmo modo que um grupo de clientes Web pode se beneficiar do compartilhamento de uma cache [WesCla98]. O uso de uma estrutura hierárquica é particularmente promissor para a escalabilidade e gerência de Web caching. A estrutura comum em caches hierárquicas é a de caches filhas que direcionam pedidos para caches pais, quando estas não conseguem atendê-los (Figura 2.2). Algumas soluções usam o conceito de caches *siblings*, que são caches vizinhas que podem cooperar para que um pedido a um objeto Web seja atendido mais rapidamente.

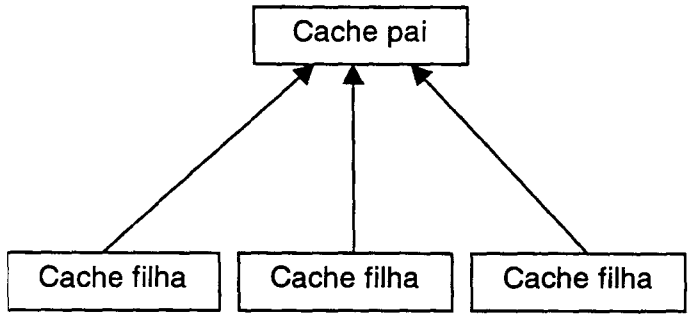

Figura 2.2: Modelo simples de hierarquia de caches. A cache filha repassa o pedido para a cache pai, caso ocorra um miss *[WesCla98].*

2.2.1. Servidores proxy com compartilhamento de conteúdo

Em caches multiníveis, requisições que não são satisfeitas por um servidor proxy são enviadas para outros servidores em níveis mais altos da hierarquia da cache [JefDas97]. *Caching* multinível não leva em consideração a topologia da Internet, pois só é eficiente quando toda a informação flui a partir da mesma localização e este servidor proxy pode compartilhar o conteúdo da sua cache. Ainda, apresenta baixa escalabilidade, pois cada nível que se segue na hierarquia precisa de uma *cache* maior que a do nível anterior, o que não é bom economicamente. Jeffery e Das propõem reformular o modelo hierárquico e fazer com que os servidores *cache* compartilhem recursos de uma forma não-hierárquica. Uma alternativa viável é a seguinte: os servidores proxy, além de serem configurados para permitir pedidos de outros servidores, ainda precisam manter *logs* sobre os outros *proxies* para saber quais colocaram nas suas *caches* os pedidos recentes. Deste modo, quando um pedido chega em um servidor proxy e este não tem a página em sua *cache*, ele pode procurar na lista algum outro proxy que esteja compartilhando recursos com ele, e se achar algum que possua a página, redirecionar o pedido. Neste processo, leva-se em consideração a topologia da rede, pois é escolhido o servidor proxy mais próximo que tenha o recurso requisitado.

2.2.2. *Caching* adaptativo na Web

As páginas mais populares na Web criam os chamados "*hot-spots*", ou seja, sites populares que produzem um tráfego intenso. Nos *hot-spots*, os mesmos dados são transmitidos sobre os mesmos *links* da rede várias e várias vezes, para milhares de usuários diferentes. Entretanto, estudos mostraram que esses *hot-spots* movem-se rapidamente, fazendo com que alguns desses gargalos apareçam e sumam antes que a rede ou os servidores Web possam ser reconfigurados.

Uma possível solução é a transmissão *multicast*, para os diversos usuários interessados nas mesmas informações. A transmissão deve ser feita de modo que os dados sejam recuperados apenas uma vez e transmitidos por um árvore *multicast* a todos os interessados, sem que passe mais de uma vez por um *link*. Entretanto, como os pedidos dos usuários são assíncronos, ou seja, são feitos em momentos diferentes, o processo de *multicast* deve ser feito através de *caching*.

Segundo Zhang *et al* [ZhFlJa97], um sistema de *caching* adaptativo deveria, idealmente, fazer com que uma página popular automaticamente "andasse" pela árvore de distribuição em resposta à intensidade dos pedidos, e quanto maior fosse a demanda por esta página, mais vezes ela seria copiada para alguma *cache* mais próxima do usuário final. Além disso, os pedidos de páginas descobririam automaticamente qual a cópia em *cache* mais próxima ao usuário. A política proposta em [ZhFlJa97] divide os servidores Web em grupos locais *multicast*, onde um servidor *cache* pode se juntar a mais de um grupo para servir de comunicação entre eles. Isso torna a política escalonável, já que diversos grupos podem ser criados à medida que mais servidores Web começam a usar a política.

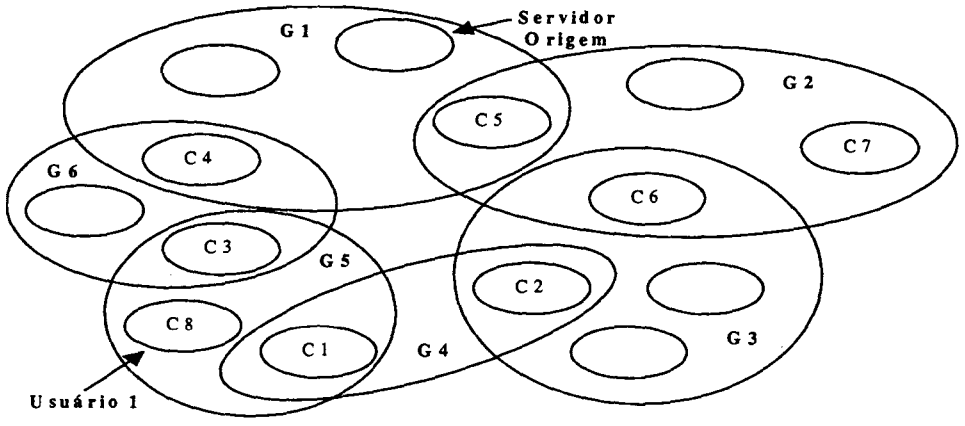

Figura 2.3: Exemplo de como funciona a cache adaptativa. O usuário 1 pede uma nova página, e o pedido vai para o proxy mais próximo (C8). Se a página estiver na cache de C8, ela é retornada; caso contrário, ele faz um multicast do pedido para seu grupo, e um proxy que pertença a outro grupo na direção do servidor de origem repassa o pedido (no exemplo, C3 pode faz multicast para o grupo G6). O pedido vai sendo enviado para grupos que estejam na direção do servidor de origem, até que seja atendido por ele, ou por alguma cache intermediária que tenha a página pedida.

Quando é feito um pedido de um documento para um servidor, ele faz um *multicast* do pedido para o seu grupo. Se ninguém do grupo tiver o documento nas suas *cache*s, ou seja, se ninguém mandou uma mensagem de resposta, cada *cache* do grupo verifica se algum dos seus outros grupos está na direção do servidor. Quando é encontrado este grupo, o servidor que serve de interface entre os grupos recebe o pedido. Com isso, o pedido caminha por entre os grupos, na direção do servidor de origem do documento pedido. Quando a página pedida é encontrada, ela é retornada ou pelo caminho que seguiu, por *multicast*, ou apenas através dos servidores que serviram de conexão entre os grupos e que fizeram os *multicasts* do pedido.

Quando um documento é retornado, ele é armazenado pelas caches que estão no caminho. Com isso, quanto maior a freqüência de pedidos para um documento, maior a probabilidade de que este documento seja encontrado em alguma cache da árvore de distribuição com origem no servidor, indo para os clientes. À medida que este documento deixa de ser pedido (mudança do *hot-spot*), ele é naturalmente expulso das caches, através das políticas locais de substituição.

Em [MiNgRo98+] foi descrito um framework de roteamento de URLs para auxiliar as caches na tomada de decisões para o direcionamento de pedidos para objetos Web. Para auxiliar uma pesquisa rápida em cada grupo de caches, foi permitido o compartilhamento de conteúdo entre caches vizinhas. Neste esquema, as caches podem fazer pesquisas no grupo local, repassando os pedidos de objetos rapidamente e evitando o atraso decorrente dos pedidos *multicast*.

2.2.3. WebWave - protocolo de *caching* baseado em difusão

Heddaya *et al* [HeMiYa97] propuseram um protocolo chamado WebWave, que tenta o balanceamento de carga entre servidores Web. Um dos problemas de um sistema que emprega *caching* é a localização de uma cópia apropriada para usar. Duas técnicas são normalmente empregadas: protocolo para descoberta das fontes de cópias e a pesquisa em um diretório. Em ambas as técnicas, o pedido do usuário irá sofrer um retardo. Qualquer serviço de diretório distribuído que tenta manter um mapeamento de identificadores dos documentos e servidores *cache* tem um *overhead* significativo para atualização deste mapeamento.

O protocolo WebWave tenta eliminar este *overhead* colocando as cópias em *cache* no caminho que o pedido do cliente segue até o servidor destino. Sob este protocolo, os servidores *cache* vizinhos cooperam com o objetivo de suavizar mudanças repentinas na carga. Os servidores *cache* com carga excessiva tentam, periodicamente, passar parte de seus pedidos para seus pais ou filhos na árvore de roteamento, desde que estes estejam com carga abaixo de um certo limite.

2.2.4. Caches hierárquicas (Harvest Cache)

Em uma cache hierárquica, os servidores resolvem um *cache miss* através de outros caches em níveis mais altos na hierarquia [Neal96]. O Harvest Cache é uma implementação de caches hierárquicos que, além de suportar a relação entre pais e filhos, possui a noção de *siblings*, onde servidores cache no mesmo nível são usados para distribuir a carga entre todos os servidores. Cada servidor na hierarquia decide independentemente se vai buscar uma referência no *site* original ou nos servidores pais

ou *siblings*, através de um protocolo de resolução. Por exemplo, se o documento é dinâmico, o pedido vai para o servidor de origem; se a URL do documento casar com o nome de domínio de alguma outra *cache sibling*, o pedido vai para essa cache; caso contrário, o pedido é feito à cache pai.

Entretanto, uma possível desvantagem das caches hierárquicas é o aumento da latência de recuperação de um documento. Isto se deve ao fato das caches em nível mais altos na hierarquia dever suprir os *misses* de seus descendentes, sobrecarregando estes servidores, tornando-se assim um gargalo do sistema. Os autores em [ChaSch96+] discutem a implementação do Harvest Cache, e ponderam a afirmação de que os custos da cache hierárquica não valem os benefícios alcançados. Neste trabalho, mostra-se que a hierarquia não aumenta a latência dos acessos.

Alguns problemas foram encontrados no Harvest cache, como uma implementação ruim do protocolo HTTP, a falta de suporte aos pedidos com cabeçalho *If-Modified-Since*, que é usado para verificar se um documento que está na cache ainda é válido, entre outros. Com isso, surgiu um produto chamado de Squid [WesCla98] como continuação do Harvest cache. Squid é um servidor proxy de alto desempenho que faz cache para clientes Web, suportando objetos FTP, gopher, e HTTP [Squid].

Tanto o Harvest quanto o Squid suportam o protocolo ICP (Internet Cache Protocol), cujo objetivo é fornecer um método rápido e eficiente para comunicação entre caches, permitindo o estabelecimento de hierarquias complexas de caches. O protocolo ICP define um tipo de mensagem pequena e simples, usado para comunicação entre Web caches. Os pedidos e respostas em ICP referem-se à existência de URLs (ou objetos) em caches vizinhas [WesCla97]. As caches trocam mensagens ICP e usam as informações coletadas para selecionar a localização mais apropriada da qual o objeto será recuperado.

2.2.5. Método colaborativo para caches hierárquicas em proxies

Em uma hierarquia restrita de proxies, quando ocorre um *miss* em uma cache, o proxy pede o objeto para o proxy imediatamente superior (pai). Neste tipo de hierarquia, normalmente ocorre duplicação do conteúdo das caches, porque os proxies filhos colocam na *cache* cada objeto Web passado pelo seu pai para o cliente. Em soluções como o Harvest cache e Squid, existem as caches vizinhas ou *siblings*, que podem fornecer um objeto quando ocorre um *cache miss*. Entretanto, a decisão de *caching* é local, baseada no conteúdo da cache e/ou em características do objeto. Essa decisão local pode ser baseada em políticas como LRU, ou em variações como LRU-THOLD, que utiliza uma política LRU em conjunto com uma política de admissão baseada no tamanho dos documentos, na qual nenhum documento com tamanho maior que um determinado limite (*threshold*) entra na cache, ou LRU-MIN, que retira da cache os documentos referenciados menos recentemente e que tenham maiores tamanhos.

O método colaborativo proposto em [YuEdMa97] tenta diminuir a duplicação de conteúdo entre proxies, quando apropriado. Este método tenta também ajudar a melhorar as decisões tomadas pelos proxies. A proposta não requer mensagens adicionais ou verificação do conteúdo das caches, levando em conta apenas o *status* das caches de nível superior.

É usado um protocolo conhecido como PICS (Plataform for Internet Content Selection), que especifica um método para envio de meta-informações sobre conteúdos eletrônicos. PICS é uma Recomendação de Protocolo da *Web Consortium* [WebCon]. Neste protocolo, as informações de *status* de um objeto nos proxies de níveis mais altos são referenciadas como CHLs (Caching Hierarchy Label). O valor de CHL é armazenado no cabeçalho do objeto usando o protocolo PICS, e pode ser usado pelo proxy de menor nível para tomar decisões sobre os objetos. De modo geral, o valor de cache de um objeto reduz quando já está na cache de proxies de níveis mais altos. As informações de status sobre caching pode ser usada para direcionar o pedido do objeto para o nível superior mais próximo, que potencialmente tem o objeto, ao invés de pedir o objeto para o nível imediatamente superior.

É usada uma política de admissão que calcula o valor do objeto e só o insere na cache se este for mais valioso que os objetos que vai substituir. Para calcular o valor do objeto, é usada a sua freqüência de acesso, que pode ser estimada como o inverso do momento da sua última referência. Por isso, além das URLs e status da cache do proxy superior na hierarquia, outras informações são necessárias, como o momento do último acesso de cada objeto.

2.3. Caracterização do tráfego WWW

O entendimento da natureza do tráfego de uma rede é crítico para um bom projeto e implementação desta rede e de seus serviços. Muitos estudos têm sido realizados com o objetivo de caracterizar o tráfego da World Wide Web. Esta caracterização é feita através da análise de dados, obtidos através de servidores configurados para armazenar informações sobre os pedidos dos usuários [ArlWil96, Deng96], ou através de *Web browsers* instrumentados para armazenar informações na máquina dos clientes [CroBes96, CuBeCr95]. A escolha entre coletar dados nos clientes ou servidores depende do tipo de estudo que se quer realizar. Os *traces* obtidos nos servidores são mais apropriados para avaliar medidas de desempenho no nível de sistema, como a demanda de largura de banda, enquanto os *traces* obtidos nos clientes são mais apropriados para caracterizar o comportamento dos usuários.

Arlitt e Williamson [ArlWil96, ArlWil97] realizaram estudos a fim de identificar invariantes no tráfego da Web. Nestes estudos foram utilizados seis *traces* diferentes: três de ambientes acadêmicos, dois de instituições de pesquisa e um de um provedor de Internet. Os conjuntos de dados obtidos foram coletados em intervalos de tempo que variavam de uma semana a um ano de atividade. Foram identificadas dez invariantes no tráfego dos servidores Web, ou seja, dez observações que se aplicam a todos os conjuntos de dados utilizados. Estas invariantes representam, potencialmente, características universais para todos os servidores Web na Internet. As invariantes encontradas são resumidas abaixo:

Invariante	Nome	Descrição
1	Taxa de sucesso	A taxa de sucesso para pesquisas nos servidores é de cerca de 88%
2	Tipo dos arquivos	Arquivos HTML e de imagens compreendem de 90% a 100% dos pedidos
3	Tamanho médio de transferência	O tamanho médio de transferência é menor ou igual a 21 kilobytes
4	Pedidos distintos	Entre todos os pedidos ao servidor, menos de 3% são para arquivos distintos
5	Referência única	Aproximadamente um terço do arquivos e dos bytes acessados nos traces são acessados uma única vez
6	Distribuição do tamanho	O tamanho dos arquivos segue a distribuição de Pareto com $0.40 < \alpha < 0.63$
7	Concentração de referências	10% dos arquivos acessados correspondem a 90% dos pedidos ao servidor e 90% dos bytes transferidos
8	Tempos entre referências	O tempo entre referências de arquivos é exponencialmente distribuído e independente
9	Pedidos remotos	Sites remotos compreendem cerca de 70% dos acessos aos servidores Web e cerca de 60% dos bytes transferidos
10	Uso em áreas amplas	Os servidores Web são acessados por milhares de domínios, com 10% dos domínios correspondendo a cerca de 75% do uso

Tabela 2.1: Invariantes sobre o tráfego em servidores da Web, encontrados em [ArlWil96, ArlWil97]

Estudos adicionais [CuBeCr95] confirmam as invariantes 3 e 6 citadas acima, acrescentando ainda que a distribuição dos tamanhos de transferência é mais influenciado pelo tamanho dos documentos do que pela preferência dos usuários. Ainda chegou-se à conclusão de que a popularidade dos documentos WWW segue a lei de Zipf [CuBeCr95].

Crovella e Bestavros [CroBes96] chegaram à conclusão de que o tráfego WWW exibe um comportamento que é consistente com modelos de tráfego *self-similar* (auto-semelhante), já que o tráfego WWW coletado continha rajadas observáveis em quatro escalas de tempo. Um tráfego em rajadas em várias ou todas as escalas de tempo pode ser descrito estatisticamente usando a noção de auto-semelhança.

Deng [Deng96] mostrou que o processo de chegada de pacotes em um servidor Web pode ser modelado por um processo auto-semelhante, com parâmetro de Hurst igual a 0.9. Uma explicação para a auto-semelhança do tráfego da Web é a seguinte [CroBes96]: um processo auto-semelhante pode ser construído pela superposição de vários processos *on/off*, com tempos entre chegadas tendo uma distribuição com caudas longas nas quais o peso das caudas é determinado pelo parâmetro $\alpha<2$. À medida que o número de fontes cresce, a resultante superposição tende a um processo auto-semelhante com $H=(3-\alpha)/2$. Crovella e Bestavros mostraram que os tempos de transmissão de documentos Web e espera entre transmissões possuem também caudas longas, onde a cauda do período *on* se mostrou muito mais pesada que a do período *off*.

Deng mostrou que o processo de Poisson usado para modelar a chegada de sessões TCP iniciadas pelo usuário não pode ser aplicado à chegada de documentos Web em um servidor, pois as transmissões de documentos WWW não são inteiramente iniciadas pelo usuário [Deng96]. Isto porque uma página Web normalmente contém várias imagens *in-line* (além de outros objetos), e quando o usuário pede uma destas páginas, o browser automaticamente gera uma série de pedidos adicionais para ler estas imagens. Por este motivo, o autor propõe uma modelagem usando um processo *on/off*, onde os períodos *on* correspondem aos períodos de transmissão de arquivos Web, e períodos *off* correspondem aos períodos onde a estação não está recebendo dados, chamado período de silêncio. Usando esta modelagem, o autor chega à conclusão de que as distribuições dos períodos *on* e *off* seguem as distribuições de Weibull e Pareto, respectivamente. Concluiu-se também que o tempo entre chegadas de pedidos dentro de um período *on* segue um distribuição de Weibull.

As invariantes encontradas em [ArlWil96, ArlWil97] aplicam-se a servidores Web. Os autores de [AbFoxAb97+] também identificaram um conjunto de invariantes para o tráfego da Web, identificados a partir de traces de proxies. As invariantes encontradas são mostradas na tabela abaixo:

Invariante	Nome	Descrição
1	Tamanho médio dos arquivos (median)	Aproximadamente 2 Kb
2	Tamanho médio dos arquivos (mean)	Menos de 27 Kb
3	Tipos de arquivos (acessos)	90%-98% dos arquivos acessados são dos tipos gráficos, HTML e cgi-map
4	Tipos de arquivos (bytes)	A maioria dos bytes acessados são do tipo gráficos
5	% de acessos a servidores únicos	Menos de 11% dos acessos são para servidores únicos
6	% de servidores referenciados uma única vez	Menos de 5% dos servidores são acessados uma única vez
7	Concentração de referências (servidores)	25% dos servidores são responsáveis por 80%-95% das referencias
8	Concentração de bytes (servidores)	25% dos servidores são responsáveis por 90% dos bytes acessados
9	Taxa de sucesso	88%-99%
10	Auto-semelhança	$0.59 \leq H \leq 0.94$

Tabela 2.2: Invariantes sobre o tráfego em proxies da Web, encontrados em [AbFoxAb97+]

Comparando as Tabelas 2.1 e 2.2, pode-se notar que diversas invariantes para o tráfego de servidores Web também são verdadeiras para o tráfego de proxies na Web. Uma observação interessante é a de que o estudo das invariantes mostrada na Tabela 2.1 foi feito com traces coletados entre outubro de 1994 e agosto de 1995, enquanto os traces usados o estudo que resultou na Tabela 2.2 foram coletados entre janeiro de 1995 e dezembro de 1996. Em outras palavras, o tráfego caracterizado na Tabela 2.2 foi coletado mais recentemente que o tráfego caracterizado na Tabela 2.1. Pode-se notar que, enquanto invariantes relacionadas com a taxa de sucesso permanecem as mesmas entre as duas tabelas, outras invariantes mudam entre as mesmas. Quanto ao tipo dos arquivos, na Tabela 2.1 os arquivos HTML correspondem de 90% a 100% dos pedidos, enquanto na Tabela 2.2, HTML, gráficos e cgi-maps correspondem de 90% a 98%, sendo que a maior parte dos pedidos é para arquivos de gráficos (de 51% a 84%), vindo depois os pedidos para arquivos HTML (10% a 21%). Provavelmente por causa do aumento do número de pedido para gráficos, o tamanho médio de transferência também é diferente entre as duas tabelas: enquanto na Tabela 2.1 o tamanho médio é menor ou igual a 21 Kbytes, na Tabela 2.2 esse número aumenta para 27 Kbytes. Os estudos sobre referência

única e concentração de referências diferem pois na Tabela 2.1 eles foram feitos com base no número de arquivos, enquanto na Tabela 2.2 foram feitos com base no número de servidores.

Os autores em [AbFoxAb97+] concluem que o tráfego Web é auto-semelhante, resultado consistente com [CroBes96]. Foram estudadas 10 cargas, e duas séries temporais para cada carga, que são a taxa de acessos e o número de acessos. A taxa de acessos é definido como o número de acessos que um proxy recebe por unidade de tempo, enquanto a taxa de bytes é o número de bytes que um proxy manda por unidade de tempo. A auto-semelhança encontrada com a taxa de acessos é mais forte que a encontrada com a taxa de bytes.

Em [AlBeCrOl96], os autores propõem modelos para localidade temporal e espacial das referências em seqüências de pedidos que chegam nos servidores Web. Para isso, usam *stack distances* para representar a seqüência de pedidos nos traces, e usam propriedades destas para medir as noções de localidade temporal e espacial.

A localidade temporal indica que a seqüência de pedidos se beneficia de *caching*, enquanto a localidade espacial indica um benefício se for usado *prefetching*. A primeira pode ser medida pela distribuição marginal da *stack distance*, enquanto a segunda pode ser medida pela estrutura de correlação da *stack distance*.

Capítulo 3

Políticas de Gerenciamento de Web Caches

Neste capítulo, serão estudadas políticas de gerenciamento de Web caches [FonOli99]. Essas políticas determinam se um documento irá entrar ou não na cache, e quais documentos devem sair da cache para liberar espaço para um novo documento.

As políticas de uma Web cache têm grande influência na Qualidade de Serviço total, e são as principais responsável pelo desempenho da cache. As políticas de expulsão definem quais documentos devem ser retirados da cache, a fim de liberar espaço para um documento que quer entrar na cache em um determinado momento. Estudos indicam que a informação utilizada como critério de expulsão tem um grande impacto no desempenho do sistema [WiAbSt96].

A maior parte dos trabalhos realizados sobre políticas de Web cache investiga o impacto usando medidas de desempenho relacionadas somente com o desempenho do sistema, como hit ratio. Poucas são as pesquisas que objetivam explicitamente a melhoria

de parâmetros de Qualidade de Serviço, tais como tempo médio de resposta por transação [WooAbr97] [BolHos96].

Pesquisas mostram que o parâmetro de QoS mais importante para os usuários é o tempo de resposta na recuperação de objetos da Web [UserSur98]. Define-se como tempo de resposta a diferença entre os instantes de solicitação do documento e o instante em que o mesmo se torna disponível. Por outro lado, o tempo de recuperação do documento pode ser definido como a diferença entre os instantes do pedido do documento e o término da sua chegada. O tempo de resposta de um pedido depende da existência ou não de uma cópia do documento na cache. Caso exista uma cópia do documento na cache, o tempo de resposta é mínimo. Do contrário, deve-se trazer uma cópia do documento de um servidor remoto para a cache, o que torna o tempo de resposta tipicamente muito mais elevado. Em cenários com um alto miss ratio, o tempo de resposta é tipicamente determinado pelo tempo de recuperação de um documento. Altos tempos de resposta podem levar a um alto índice de evasão de usuários. Além disto, grandes variações do tempo de resposta podem causar irritabilidade no usuário devido a falta de previsão de comportamento do sistema.

Este capítulo objetiva investigar o tempo de recuperação de um documento como chave em políticas de expulsão. Apesar de estudos anteriores abordarem este tópico, acredita-se que os resultados apresentados não são conclusivos, como apontam os próprios autores de [BolHos96]. Ainda, são usados diversos parâmetros para determinar o desempenho de uma política, alguns relacionados com o desempenho do sistema e outros com a qualidade de serviço. Investiga-se também o uso de políticas de controle de admissão de documentos na cache, que procuram determinar quais documentos podem entrar na cache.

Este capítulo está organizado da seguinte forma. Na seção 3.1, descreve-se o experimento realizado. Na seção 3.2, estuda-se o uso do tempo de recuperação como chave de expulsão. Na seção 3.3, algumas políticas híbridas que usam o tempo de recuperação como chave são avaliadas. Na seção 3.4, investiga-se o uso de controle de admissão usando o tempo de recuperação e na seção 3.5 é apresentado um resumo dos resultados mostrados no capítulo.

3.1. O experimento de simulação

A fim de avaliar o impacto da chave tempo de recuperação na QoS perceptível ao usuário, foi desenvolvido um simulador que implementa diversas políticas de expulsão em paralelo. As chaves utilizadas são: "nref", que representa o número de referências ao objeto na cache; "etime" e "atime", que representam a hora de entrada e último acesso do objeto na cache, respectivamente; "size", que representa o tamanho do objeto; e "rtime", que representa o tempo de recuperação do objeto. A Tabela 3.1 mostra as políticas definidas por essas chaves. É importante notar que uma política de expulsão retira o documento com o menor valor de chave. Assim sendo, ao utilizarmos o inverso do tamanho do documento como chave, o maior documento será o documento expulso da cache. Deve-se observar que, neste trabalho, o tempo de recuperação de um documento é estimado pelo valor obtido da última referência ao mesmo.

Política	Chaves usadas	Descrição
LFU	nref / (atime - etime)	política least-frequently-used, baseada na freqüência de uso de um documento
LRU	atime	política least-recently-used, baseada na hora em que um documento foi acessado pela última vez
FIFO	etime	política first-in, first-out, baseada na hora de entrada de um documento na cache
NREF	nref	política baseada no número de referências a um documento
RTIME	rtime	política baseada no tempo de recuperação de um documento
SIZE	1 / size	política baseada no tamanho de um documento

Tabela 3.1: Políticas definidas pelas chaves de expulsão usadas.

As simulações foram baseadas em arquivos de *traces*, contendo uma seqüência de pedidos a objetos Web coletados em proxies. Os dados de entrada do simulador foram coletados em dois ambientes diferentes: um acadêmico (University of California at Berkeley) e outro empresarial (Digital Equipment Corporation). Assim sendo, objetiva-se capturar possíveis diferenças dos resultados devido ao comportamento de usuários distintos. Estes traces estão disponíveis publicamente no Internet Trace Archieve [HTTPTraces].

O trace de Berkeley [BerTraces] corresponde ao serviço Home IP oferecido a estudantes, professores e funcionários e permite conectividade via PPP/SLIP, usando modems de 2.4 kbps, 9.6 kbps, 14.4 kbps ou 28.8 kbps. Todo o tráfego de mais de 600 modems passa por uma rede Ethernet a 10Mbps, onde foi colocado um monitor de tráfego. Todas as chaves mostradas na Tabela 3.1 podem ser direta ou indiretamente extraídas dos valores coletados no trace. Como por exemplo, o tempo de recuperação é dado pela diferença entre o momento da chegada do último byte do documento e o momento da requisição da conexão. O trace da DEC [DigTraces] foi coletado através da instrumentalização de um proxy Squid e representa todos os pedidos externos à sua intranet. Foram utilizados no presente estudo 8.301.739 pedidos, o que corresponde a quase duas ordens de grandeza a mais que o número de pedidos nos traces utilizados por Wooster e Abram [WooAbr97]. Os traces são resumidos na Tabela 3.2.

Trace	Período	Número de acessos
Berkeley 1	01/11/96 - 06/11/96	2.452.516
Berkeley 2	06/11/96 - 10/11/96	1.601.165
Digital 1	29/08/96 – 02/09/96	2.276.231
Digital 2	02/09/96 – 04/09/96	1.971.827

Tabela 3.2: Resumo dos traces usados nas simulações.

Em uma primeira etapa do experimento investigou-se o tamanho da cache para que não houvesse nenhuma reposição de documentos. Em outras palavras, determinou-se o tamanho da cache infinita para cada um dos quatro arquivos de trace. A fim de se obter resultados numéricos passíveis de generalização, os resultados dos experimentos serão descritos em função do tamanho da cache infinita, ou seja, de uma cache capaz de armazenar todas as solicitações. A Tabela 3.3 mostra os tamanhos para as cache infinitas o número máximo de hits obtidos na cache, a maior taxa de acertos possível (HR) e a maior taxa de acertos ponderada pelo tamanho dos objetos possível (WHR).

Trace	MaxHits	HR	WHR	Max.Cache
Berkeley 1	1086834	44.315 %	22.646 %	9.81 Gbytes
Berkeley 2	656328	40.991 %	67.545 %	9.29 Gbytes
Digital 1	1159499	50.939 %	42.236 %	10.63 Gbytes
Digital 2	970057	49.196 %	43.562 %	9.03 Gbytes

Tabela 3.3: Resultados obtidos no experimento de cache infinita.

3.2. O uso do tempo de recuperação como chave de expulsão

Em um primeiro experimento, compara-se o impacto das diferentes chaves de expulsão usadas nas políticas mostradas na Tabela 3.1. Para comparar as políticas, foram usadas as medidas de desempenho definidas na tabela abaixo:

Parâmetro	Descrição
Hit Ratio (HR)	Indica a taxa de acerto na cache, ou seja, a porcentagem de objetos pedidos que puderam ser atendidos pela cache
Weigthed Hit Ratio (WHR)	Indica a porcentagem dos bytes pedidos que puderam ser atendidos pela cache
Time Spent per Miss (TSM)	Indica o tempo gasto para recuperar um objeto, cada vez que este não estava na cache (miss)
Time Spent With Misses (TSWM)	Indica o tempo total gasto na recuperação de objetos que não estavam na cache
Variance (VAR)	Indica o quanto o tempo de recuperação dos objetos variou em relação à média total dos tempos

Tabela 3.4: Parâmetros usados para avaliação das políticas de substituição em cache.

Quando ocorre a necessidade de se disponibilizar espaço na cache para admissão de um novo documento, um ou mais documentos são expulsos de acordo com uma chave específica, de forma que o tamanho total dos documentos expulsos mais o espaço disponível na cache seja pelo menos igual ao tamanho do novo documento.

As Figuras 3.1, 3.2, 3.3, 3.4 e 3.5 mostram o hit ratio, o weighted hit ratio, o tempo médio de recuperação por miss, o tempo total de recuperação gasto em todos os misses e a variância do tempo médio de recuperação em função do tamanho da cache dado como percentagem do tamanho da cache infinita. As figuras que serão exibidas neste capítulo correspondem a todos os traces estudados (Berkeley1, Berkeley2, Digital1 e Digital2) e às medidas de desempenho consideradas mais importantes: taxa de acerto (HR), tempo médio de recuperação dos misses (TSM) e variância do tempo de recuperação dos misses (VAR).

Para as outras medidas de desempenho, taxa de acerto ponderada pelo tamanho (WHR) e tempo total de recuperação dos misses (TSWM), apenas as figuras para o trace de Berkeley2 são exibidas. A fim de se evitar redundância, escolheu-se os gráficos correspondentes ao trace Berkeley2 para se tecer conclusões. Assim sendo, os gráficos dos demais traces são comentados somente quando há divergência de comportamento em relação aos gráficos de Berkeley2.

A Figura 3.1 mostra o hit ratio produzido por diferentes políticas. O hit ratio produzido pela política SIZE é o maior entre todos os hit ratios, enquanto que o hit ratio produzido pela política RTIME é o menor de todos. Este comportamento pode ser facilmente explicado, pois o uso do inverso do tamanho como chave de expulsão retira os maiores documentos da cache. Consequentemente, para um certo tamanho, a cache armazena um maior número de documentos, o que aumenta a probabilidade de um pedido ser um hit. Nota-se, pela Figura 3.1, que enquanto o hit ratio máximo é obtido pela política SIZE para um tamanho de cache de aproximadamente 20% do tamanho da cache infinita, a política RTIME produz o valor máximo de hit ratio para tamanhos de cache maiores que 80% da cache infinita. Nota-se, também para a mesma figura, que os hit ratios produzidos pelas demais políticas convergem para vizinhanças do valor máximo muito mais rapidamente do que os valores produzidos pela política RTIME. Como, por exemplo, os hit ratios produzidos pelas políticas LRU e FIFO para uma cache com tamanho de 30% da cache infinita diferem de aproximadamente 10% do hit ratio máximo. Isso porque a política RTIME mantém na cache os documentos com maiores tempos de recuperação, que normalmente são os documentos com maiores tamanhos. Por este motivo, a cache fica cheia com poucos documentos muito grandes, diminuindo a taxa de acerto na cache, pela pouca quantidade de documentos e pelo fato de que a maior parte das referências são para documentos pequenos. Somente quando o tamanho da cache é quase igual ao tamanho de cache máxima, começa a haver espaço para os documentos pequenos.

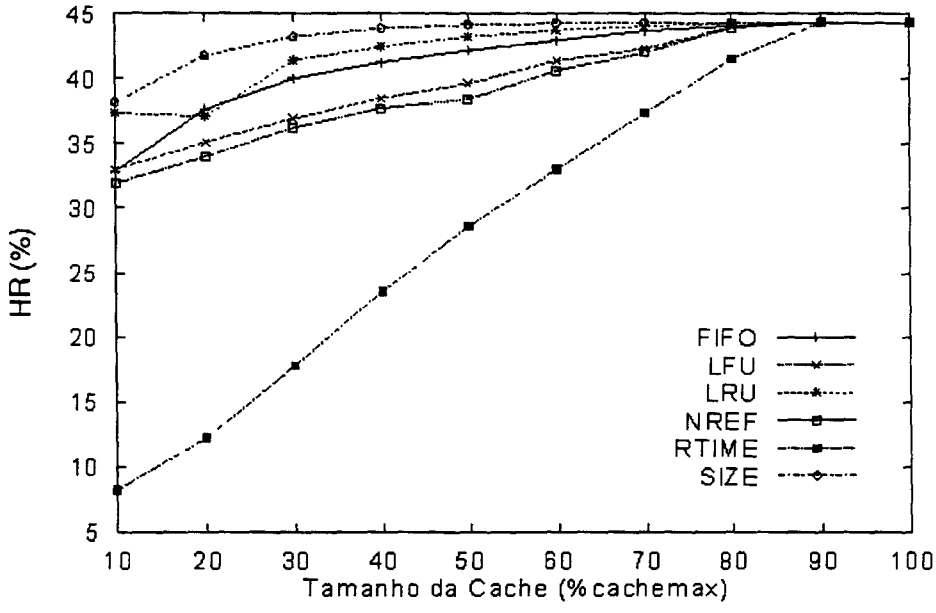

Figura 3.1a: Trace de Berkeley1.

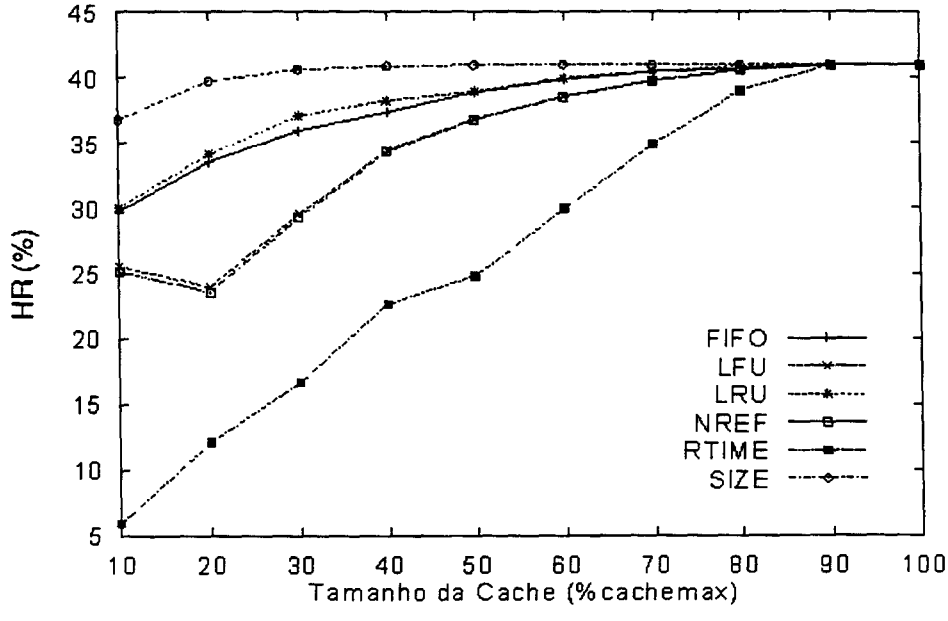

Figura 3.1b: Trace de Berkeley2.

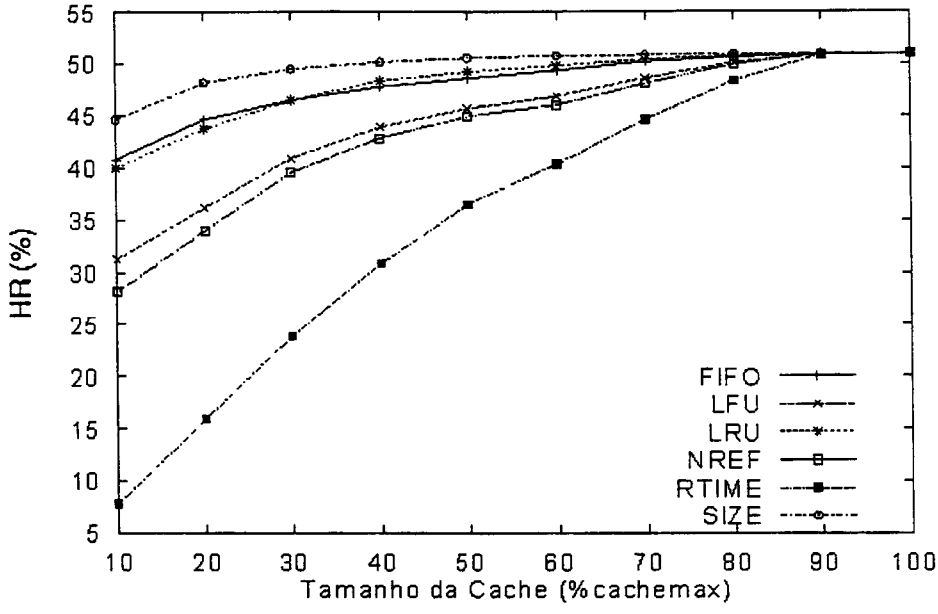

Figura 3.1c: Trace da Digital1.

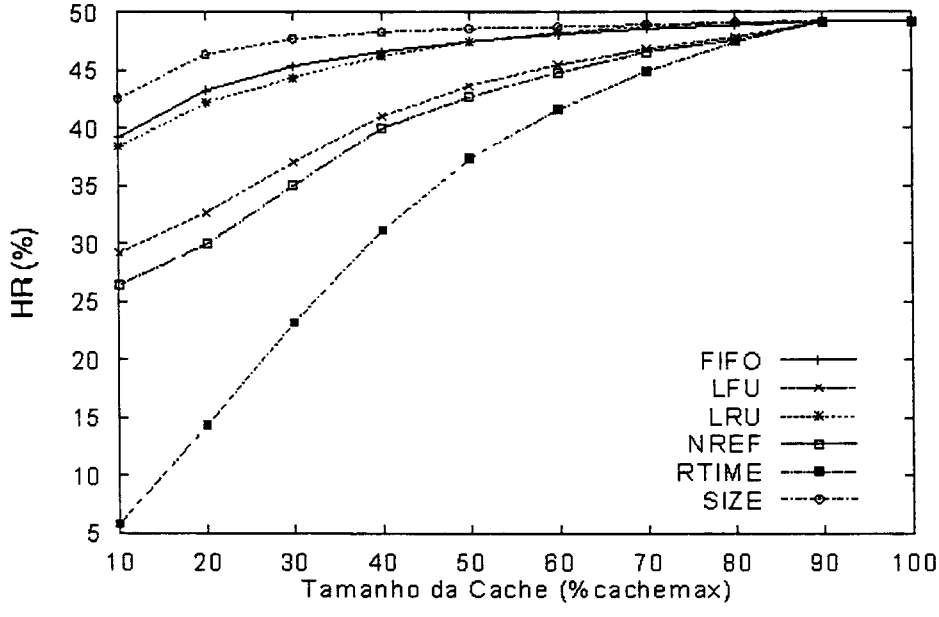

Figura 3.1d: Trace da Digital2.

Figura 3.1: Taxa de acerto na cache (hit ratio) como função do tamanho da cache, para diferentes políticas.

Deve-se notar que os resultados de hit ratios encontrados no presente experimento de simulação fazem com que o tempo de recuperação de um documento tenha um grande impacto no tempo de resposta por transação (pedido de documento). Assim sendo, prover baixos tempos de resposta implica em minimizar o tempo de recuperação de documentos. É notório também que miss ratios da ordem de 10% reportados em [WiAbSt96] não foram obtidos com os traces utilizados no presente trabalho. Acredita-se que esta discrepância seja conseqüência da diversidade da natureza dos traces e que a diferença nos tamanhos dos traces utilizados nos dois estudos não justifique tal discrepância.

Ao se comparar os gráficos da taxa de acerto para o trace de Berkeley2 com os gráficos para os outros traces, podemos concluir que os resultados seguem as mesmas tendências, embora logicamente os resultados para traces do mesmo lugar tenham mais semelhanças entre si, ou seja, os resultados para Berkeley1 e Berkeley2 têm mais semelhanças que os resultados para um trace de Berkeley e outro da Digital. Ainda, um comportamento que pode ser notado ao se analisar os gráficos é uma maior taxa de acerto para os dois traces da Digital, se comparados com os traces de Berkeley, para as políticas LRU, FIFO e SIZE.

Observando o comportamento do weighted hit ratio (Figura 3.2), que traduz o volume de dados transferidos (ou equivalentemente à banda passante utilizada), pode-se notar que a discrepância entre a convergência dos valores produzidos pela política SIZE para o valor máximo de weighted hit ratio e a convergência dos valores gerados pela política RTIME é menor do que a mesma discrepância para o caso de hit ratio. Observa-se, também, que o comportamento dos valores produzidos pela política NREF e pela política LFU são semelhantes ao comportamento dos valores produzidos pela política RTIME e que o comportamento dos valores produzidos pelas políticas LRU e FIFO se assemelham aos valores produzidos pela política SIZE.

Ao se analisar o tempo médio de recuperação gasto por miss (Figura 3.3), observa-se uma tendência ortogonal à tendência encontrada para o hit ratio. Em outras palavras a política RTIME produz os menores tempos médios de recuperação por miss, enquanto que a política SIZE produz os maiores tempos médios de recuperação por miss.

A discrepância entre os resultados das duas políticas é significativa, sendo que o tempo médio produzido pela política RTIME aproxima-se do valor máximo (cache infinita) para tamanhos de cache maiores que 60% da cache infinita, ao passo que o valor máximo para a política SIZE é obtido para tamanhos de cache menores que 10% da cache infinita.

Podemos ver ainda que os resultados para os traces da Digital (Figuras 3.3c e 3.3d) mostram tempos muito menores que os gerados pelos traces de Berkeley (Figuras 3.3a e 3.3b). Isso pode ser explicado pela análise da natureza dos traces. Enquanto o trace da Digital foi coletado a partir de pedidos da intranet da DEC, o trace de Berkeley corresponde à acessos feitos via modems e para um rede que atende ao tráfego de mais de 600 modems. Por estes motivos, podemos concluir que a velocidade de acesso para os pedidos dos traces da Digital são muito maiores que a velocidade experimentada por pedidos existentes nos traces de Berkeley. Essa característica é notada em todos os gráficos relacionados com o tempo de recuperação dos documentos (TSM ,TSWM e VAR), e para todas as políticas estudadas, já que é uma característica da natureza dos tráfegos usados.

Ao se observar as curvas do tempo médio de recuperação por miss, pode-se achar estranho à primeira vista que o tempo médio de recuperação cresça com o aumento do tamanho da cache. Todavia, foi feita uma análise mais profunda dos traces, e notou-se que mais de 90% dos pedidos são para documentos pequenos que têm um baixo tempo de recuperação. Documentos grandes com alto tempo de recuperação são raramente referenciados e muitos deles são solicitados uma única vez, como mostrado na Tabela 3.5. Tal tendência deve estar intimamente ligada ao comportamento do usuário de evitar documentos grandes dada a penalidade de altos tempos de recuperação, como por exemplo documentos com muitas imagens. Assim sendo, com o aumento do tamanho da cache um número maior de documentos com baixo tempo de recuperação e com alto índice de referência passa a não mais contribuir para a computação do tempo médio por miss, o que por conseguinte aumenta a proporção dos documentos grandes com alto tempo de recuperação na computação do tempo médio por miss.

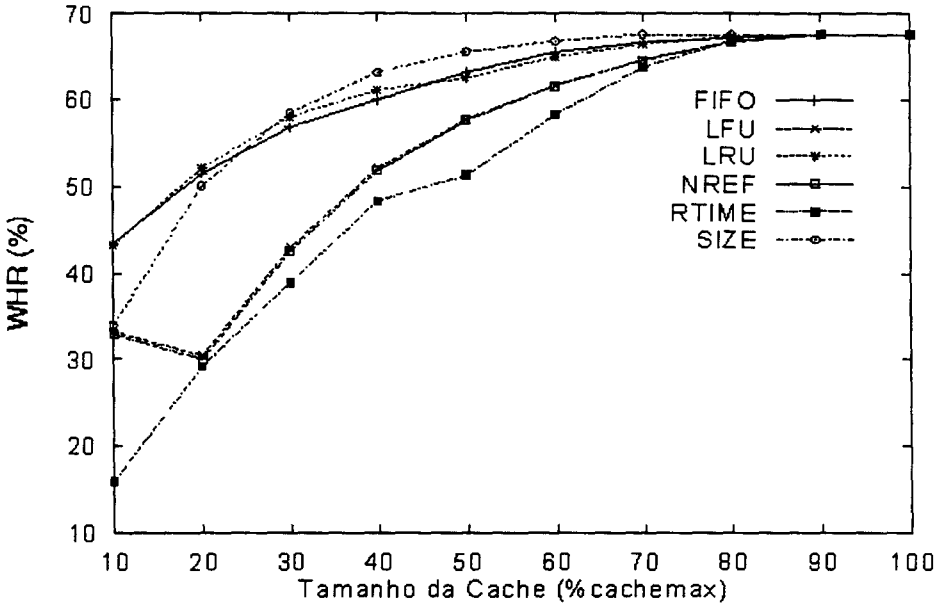

Figura 3.2: Taxa de acerto ponderada pelo tamanho(weighted hit ratio) como função do tamanho da cache, para diferentes políticas. Trace de Berkeley2.

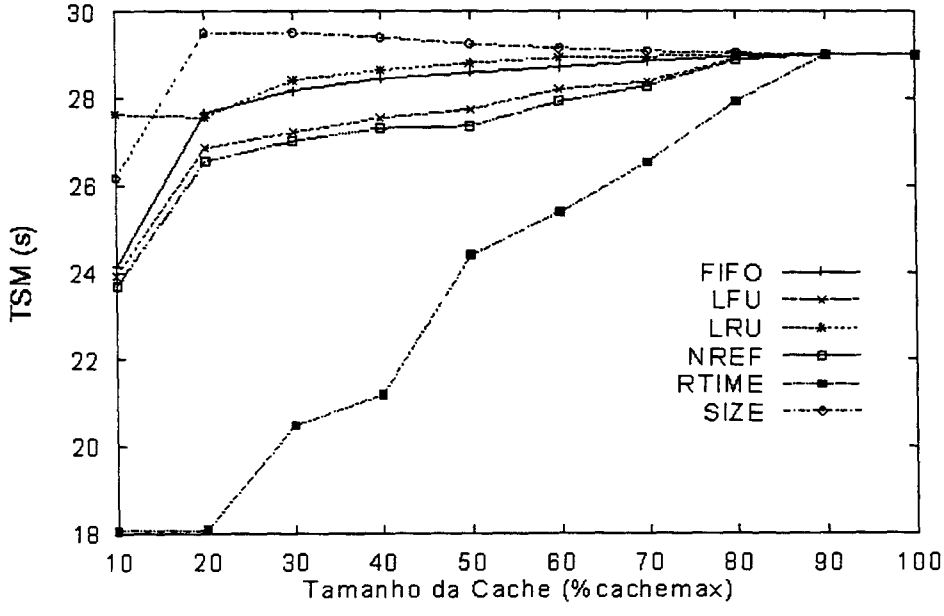

Figura 3.3a: Trace de Berkeley1.

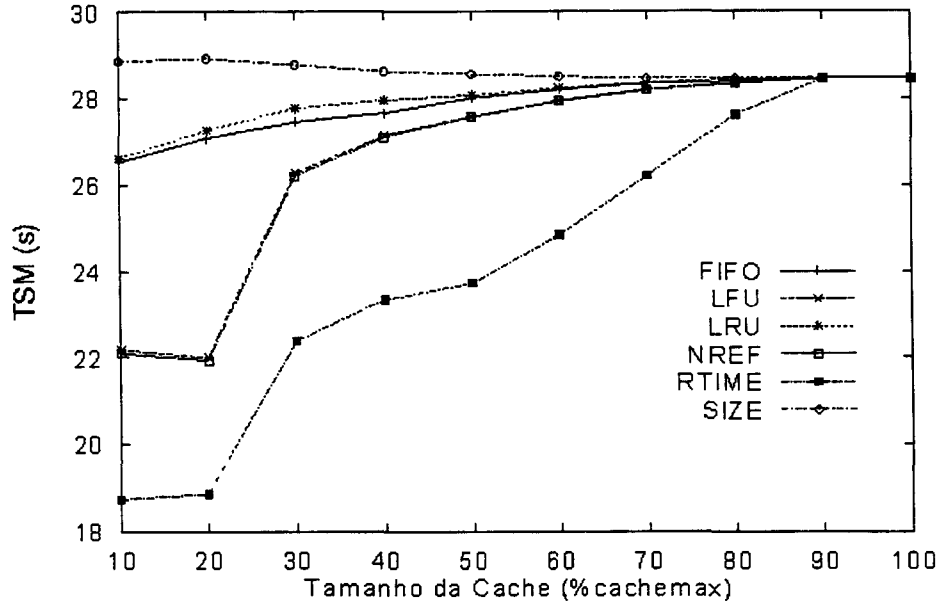

Figura 3.3b: Trace de Berkeley2.

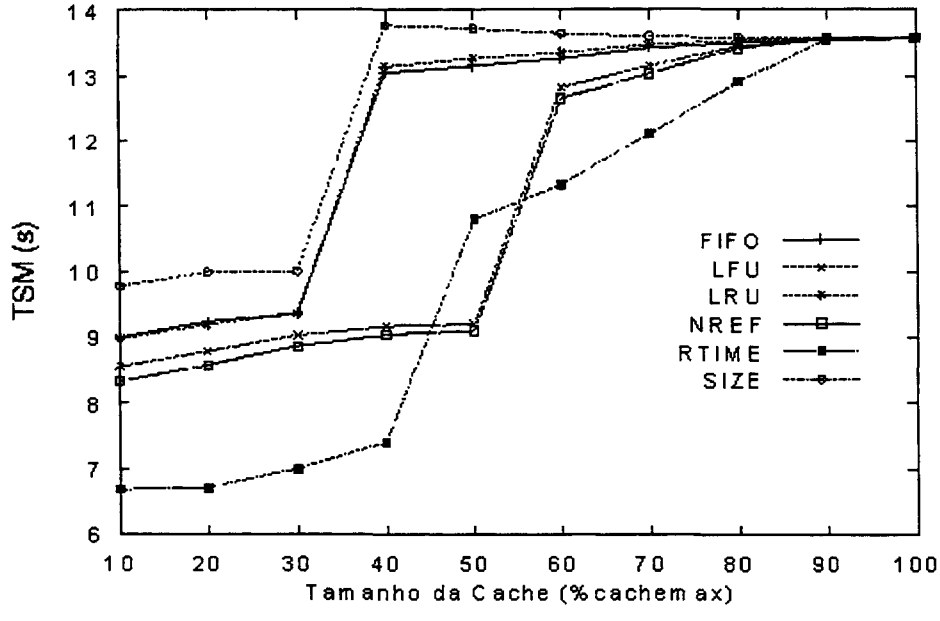

Figura 3.3c: Trace da Digital1.

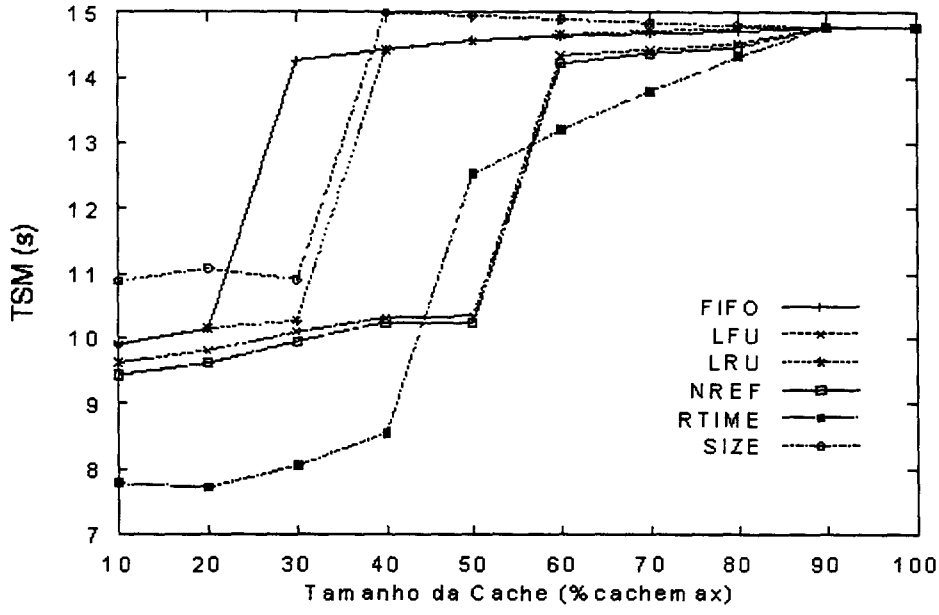

Figura 3.3d: Trace da Digital2.

Figura 3.3: Tempo médio gasto por miss (time spent per miss), como função do tamanho da cache, para diferentes políticas.

Em síntese, quanto menor a cache maior é o número de misses com baixo tempo de recuperação e por conseguinte menor é o tempo médio por miss. Com o aumento da cache o tempo médio de recuperação se aproxima do valor máximo já que existem misses que são única e exclusivamente relativos à primeira referência ao documento. Em outras palavras, com o aumento da cache, o comportamento do miss ratio aproxima-se ao de uma cache infinita. Tal dinâmica explica a razão pela qual o tempo médio de recuperação produzido pela política SIZE gera os maiores valores de tempo médio de recuperação. Esta política faz com que um alto número de documentos com baixos tempos de recuperação tornem-se residentes. Consequentemente, os misses são em sua maioria devido a documentos grandes com alto tempo de recuperação.

Tipo de porcentagem	Berkeley 1	Berkeley 2	Digital 1	Digital 2
Porcentagem do número de documentos	68,43 %	68,68%	73,44%	74,07%
Porcentagem do número de referências	20,13%	22,61%	22,54%	23,36%

Tabela 3.5: Porcentagem de documentos acessados uma única vez, para cada trace.

Ao se observar o tempo total gasto em misses (Figura 3.4), constata-se que a política RTIME produz o maior tempo total gasto em misses devido ao alto miss ratio. Pela mesma razão, a política SIZE produz o menor tempo total gasto em misses. O alto miss ratio para caches pequenas quando comparados a miss rates de caches maiores faz com que o tempo total gasto em misses diminua com o aumento do tamanho da cache.

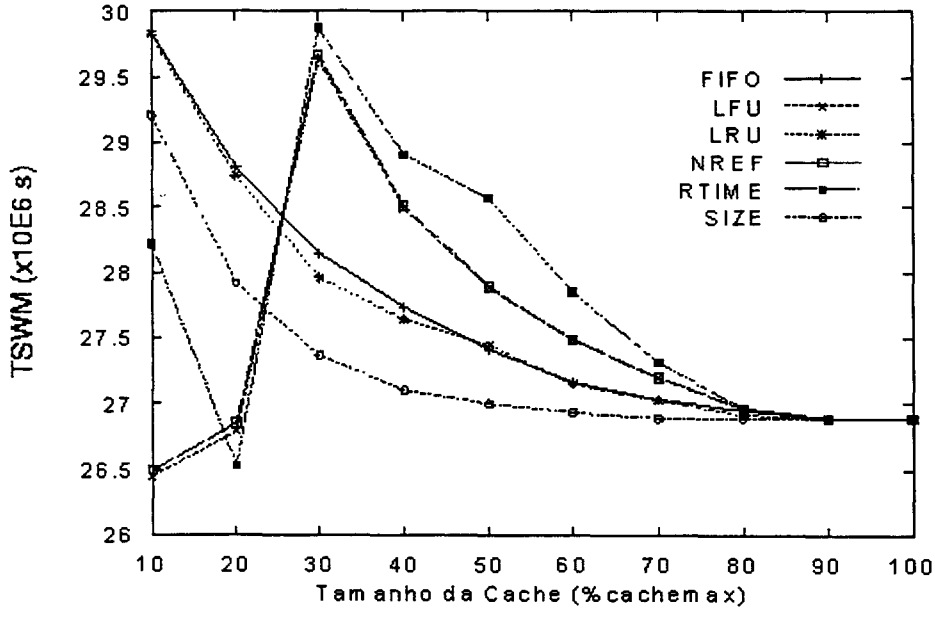

Figura 3.4: Tempo total gasto com misses (time spent with misses), como função do tamanho da cache, para diferentes políticas. Trace de Berkeley2.

Cabe mencionar que o tempo médio de recuperação por miss é apontado como o fator de Qualidade de Serviço mais importante para os usuários. Sabe-se também que quanto menor a variância do tempo de recuperação melhor a satisfação do usuário devido a uma maior capacidade de predição. Os autores desta dissertação desconhecem estudos sistemáticos que apontem que o tempo total de uma sessão de acesso a documentos da Web tenha um impacto significativo na satisfação do usuário.

A variância do tempo de recuperação dos misses (Figura 3.5) segue a mesma tendência do tempo médio de recuperação. Por um lado a política RTIME tende a manter fora da cache um alto número de documentos pequenos com alto índice de referência e baixo tempo de recuperação. Consequentemente, estes documentos contribuem com um maior número de valores próximos à média no cálculo da variância. Por outro lado, a política SIZE mantém residentes na cache os documentos com maior índice de acesso e menor tempo de recuperação. Assim sendo, os misses relativos às primeiras referências a documentos tem um grande impacto no cálculo da variância, o que faz com que o valor produzido por esta política se aproxime rapidamente ao valor obtido quando se tem uma cache infinita.

Nota-se que as demais políticas levam em consideração o número de referências a um documento, bem como o momento da ocorrência destas referências (LFU, LRU, NREF), tendendo a expulsar os documentos menos referenciados, o que coincide com o comportamento da política SIZE. Desta forma, a variância das demais políticas segue padrão semelhante a variância da política SIZE.

Vale a pena ressaltar que os valores da variância para os traces da Digital são muito menores que os valores para os traces de Berkeley porque o cálculo da variância acentua a diferença existente entre os tempos de recuperação para os dois tipos de trace.

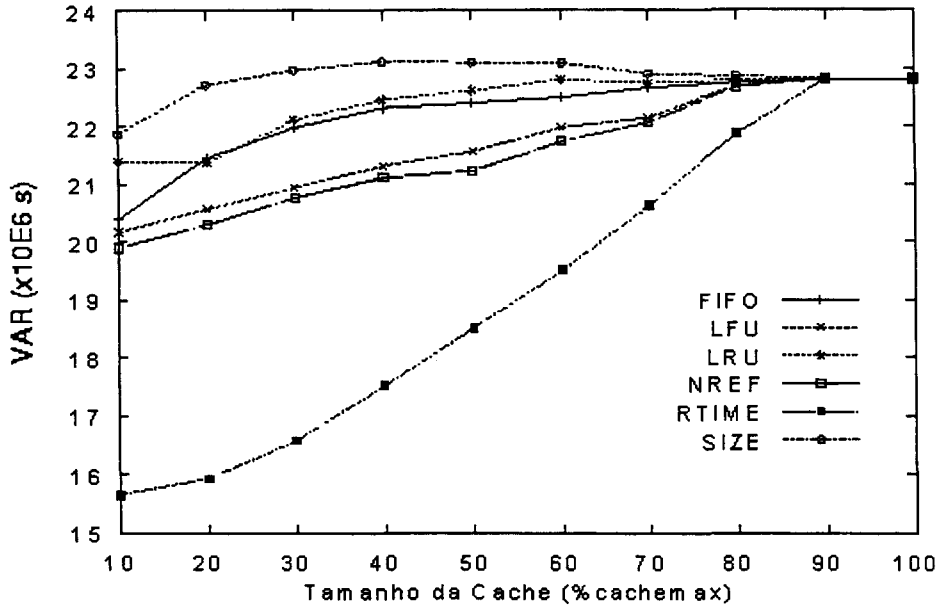

Figura 3.5a: Trace de Berkeley1.

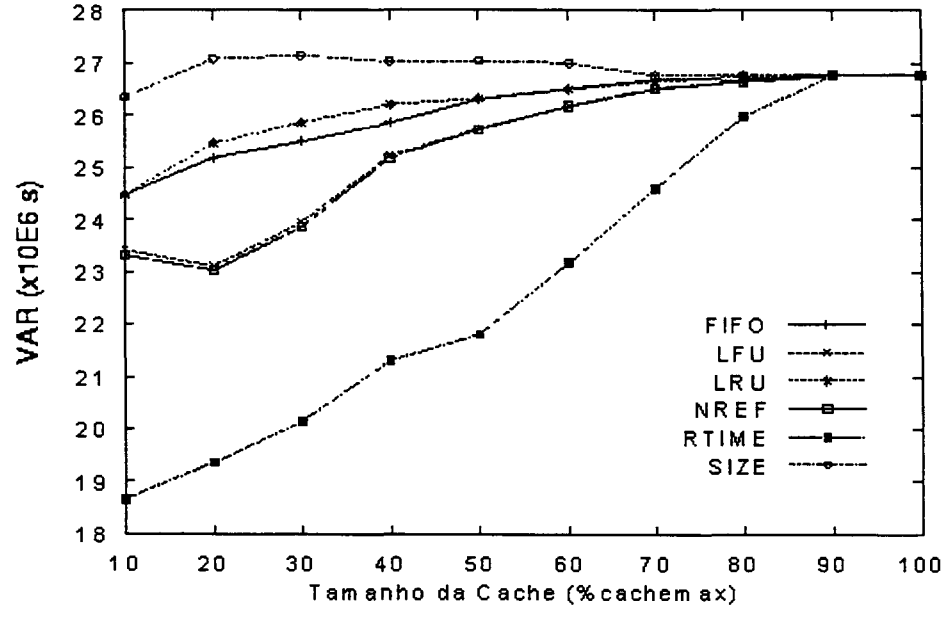

Figura 3.5b: Trace de Berkeley2.

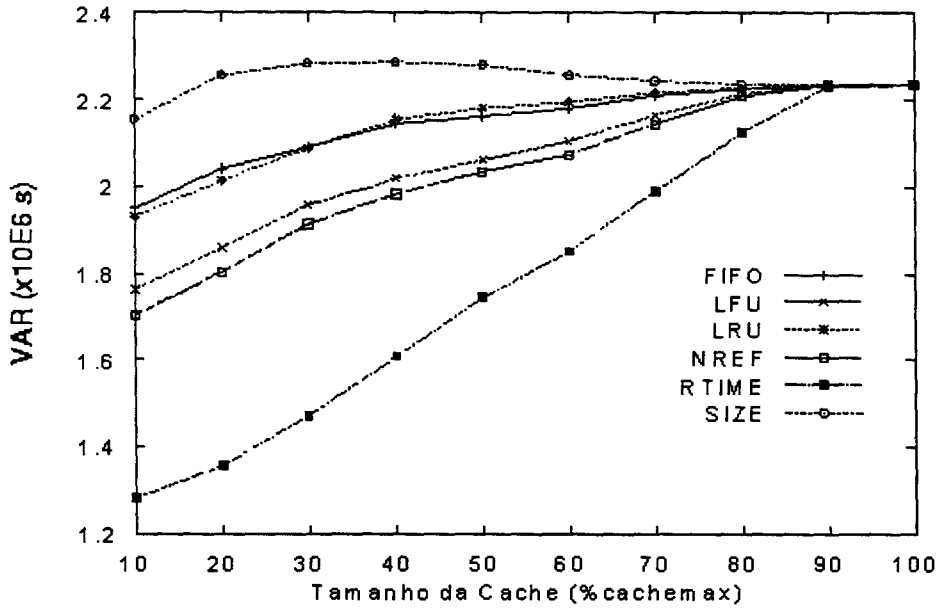

Figura 3.5c: Trace da Digital1.

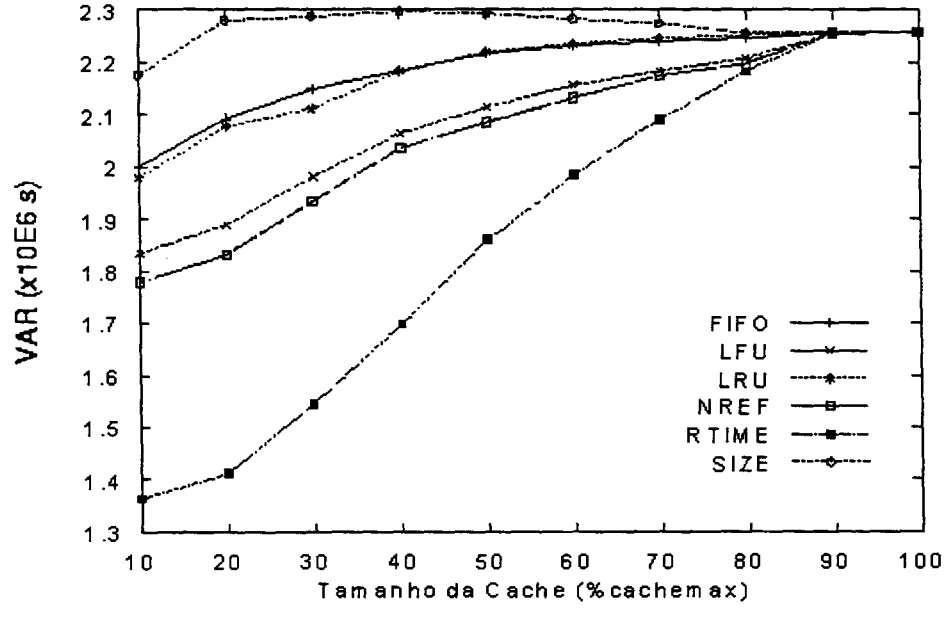

Figura 3.5d: Trace da Digital2.

Figura 3.5: Variância do tempo de recuperação dos misses como função do tamanho da cache, para diferentes políticas.

3.3. Políticas híbridas

Nesta seção, são investigadas políticas que usam o tempo de recuperação de documentos em conjunto com outras chaves de expulsão. O objetivo é tentar achar uma combinação de chaves que leve a uma política com melhor desempenho que as políticas que usam chave única.

Tendo em vista que a análise dos traces utilizados aponta para o fato de que documentos longos têm um baixo índice de referência, e que tais documentos ocupam espaço de cache desnecessariamente quando referenciados uma única vez, definiu-se uma política para tentar expulsar documentos com uma única referência. Para tal, mantém-se uma lista dos documentos ordenada pela hora da última referência (LRU) ao mesmo. Na necessidade de se expulsar um documento, consulta-se primeiramente a ordenação LRU a fim de se verificar se algum documento ultrapassou um certo limite de tempo sem ser referenciado. Caso existam documentos nesta categoria, eles são expulsos. Do contrário, utiliza-se a política RTIME para se determinar qual documento deve ser retirado. Ao se definir um limite para o período de ausência de referências pretende-se identificar documentos que não mais serão referenciados. Essas políticas foram chamadas de THA, já que usam um limite (*threshold*) para o momento do último acesso (*atime*), expressado em número de pedidos.

Nas Figuras 3.6, 3.7, 3.8, 3.9 e 3.10 são exibidos os desempenhos das políticas para diferentes valores do limite dos períodos sem referência. Estas figuras utilizam limites que tentam manter na cache uma certa porção dos documentos referenciados. A Tabela 3.6 mostra as quatro variações desta política usadas nos experimentos, com os valores de limite usados (número de pedidos) e a porcentagem dos documentos que elas englobam em cada trace.

Política	Limite	Berkeley 1	Berkeley 2	Digital 1	Digital 2
THA 1	731.681	65,56%	84,63%	70%	74,14%
THA 2	932.359	75,92%	92,14%	80%	83,81%
THA 3	1.221.024	86,53%	98,17%	90%	92,67%
THA 4	1.515.959	93,35%	99,93%	95%	95,87%

Tabela 3.6: Políticas THAs, com os valores de limite usados e a porcentagem de documentos englobados para cada trace.

Observa-se na Figura 3.6 que à medida que se diminui o limite, mais documentos com pouca referência são retirados da cache, o que aumenta a taxa de acerto. Quanto maior o limite, menos documentos são retirados por causa do tempo que ficaram sem referência e as políticas aproximam-se da política RTIME, já que ela é usada quando nenhum documento excede o limite estabelecido.

Na Figura 3.7, pode-se notar que o comportamento para o parâmetro weighted hit ratio é o mesmo que para o parâmetro hit ratio, mas com uma menor discrepância entre os valores. A explicação para tais comportamentos é a mesma, já que ambos os parâmetros são baseados na taxa de acerto na cache.

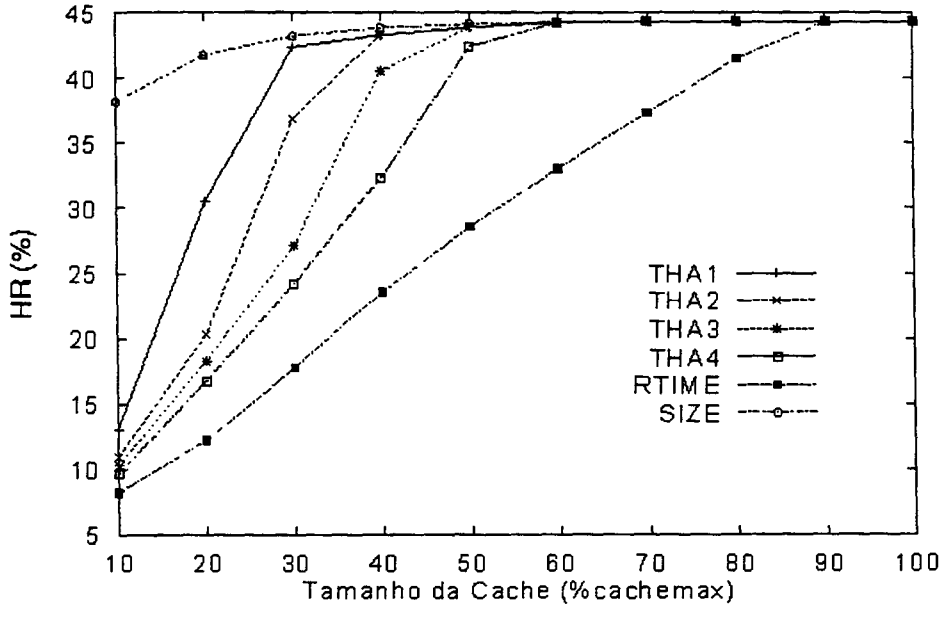

Figura 3.6a: Trace de Berkeley1.

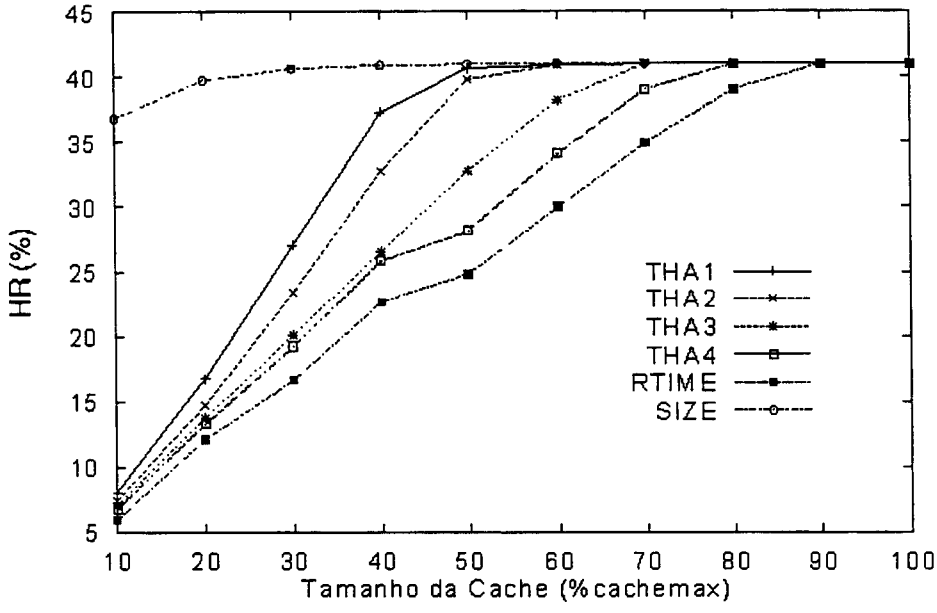

Figura 3.6b: Trace de Berkeley2.

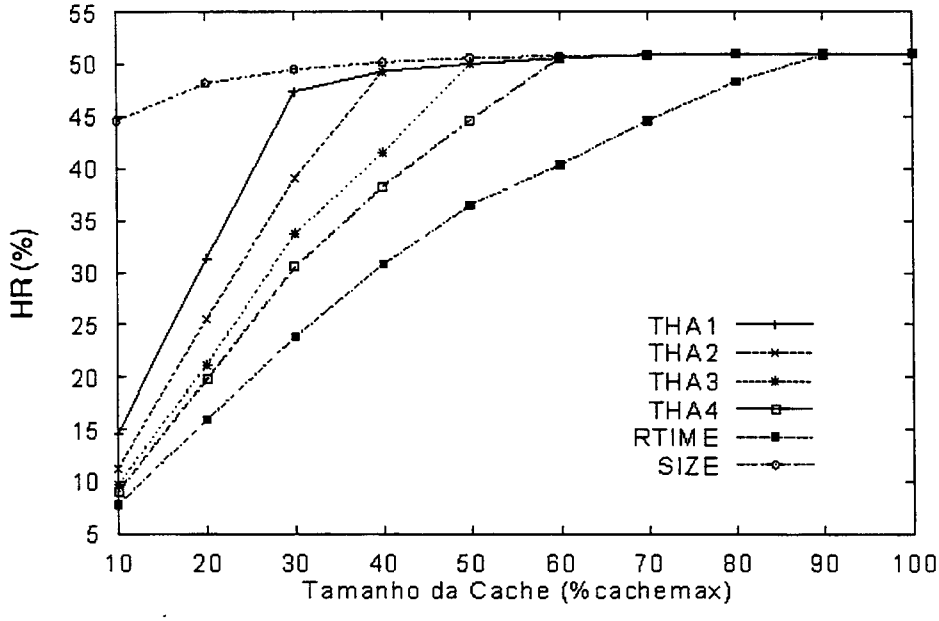

Figura 3.6c: Trace da Digital1.

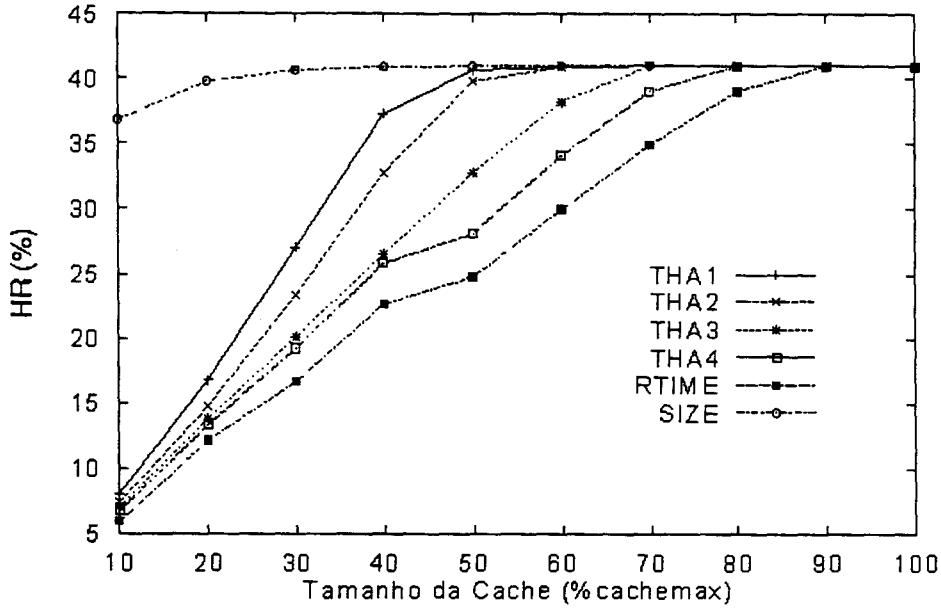

Figura 3.6d: Trace da Digital2.

Figura 3.6: Taxa de acerto na cache (hit ratio) como função do tamanho da cache para diferentes valores limites de período com ausência de referências.

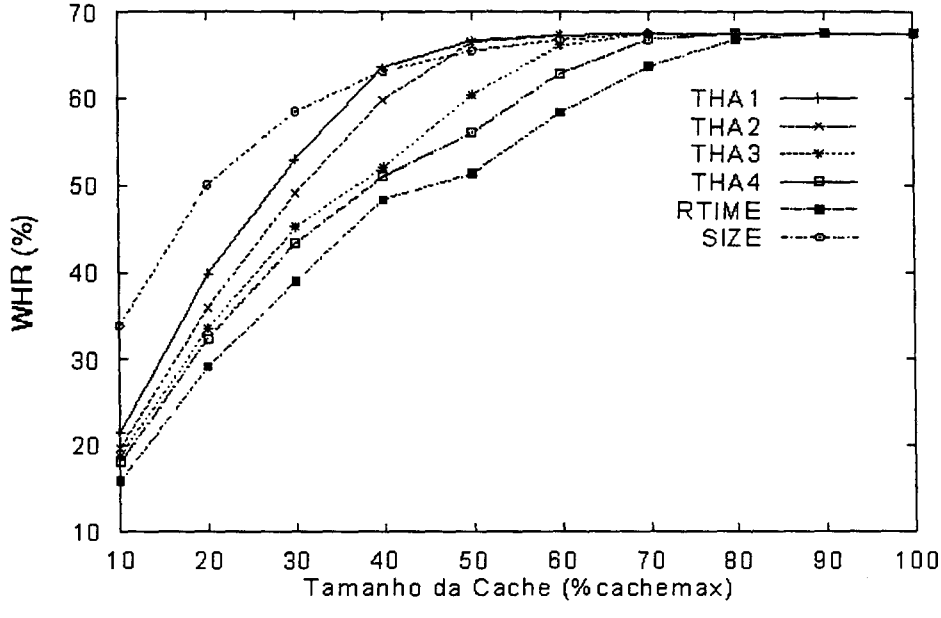

Figura 3.7: Taxa de acerto ponderada pelo tamanho (weighted hit ratio) como função do tamanho da cache para diferentes valores limites de período com ausência de referências. Trace de Berkeley2.

A Figura 3.8 mostra os resultados para a análise do tempo médio por miss, onde pode-se observar que quanto maior o limite das políticas THAs, mais próximos ficam seus desempenhos dos da política RTIME, já que menos documentos são expulsos por causa do tempo que ficaram sem referência. Outra razão para este resultado é o fato de que os documentos expulsos por causa do tempo sem serem referenciados provavelmente terão tamanhos grandes. Por isso limites pequenos, que expulsam mais documentos, irão tirar da cache documentos maiores e, consequentemente, terão maiores tempos nos misses.

Os resultados mostrados na Figura 3.9 para o tempo total gasto com misses são conseqüência do número de misses provocados, e do tempo de recuperação dos documentos nos misses. Quanto maior o limite, mais a chave tempo de recuperação será usada, o que gera muitos misses e aumenta o tempo total gasto com misses, mostrando resultados próximos aos da política RTIME. Por outro lado, quanto menor o limite, mais documentos pouco referenciados são retirados e menos a chave tempo de recuperação é usada, aumentando o número de hits e diminuindo o tempo total gasto com misses.

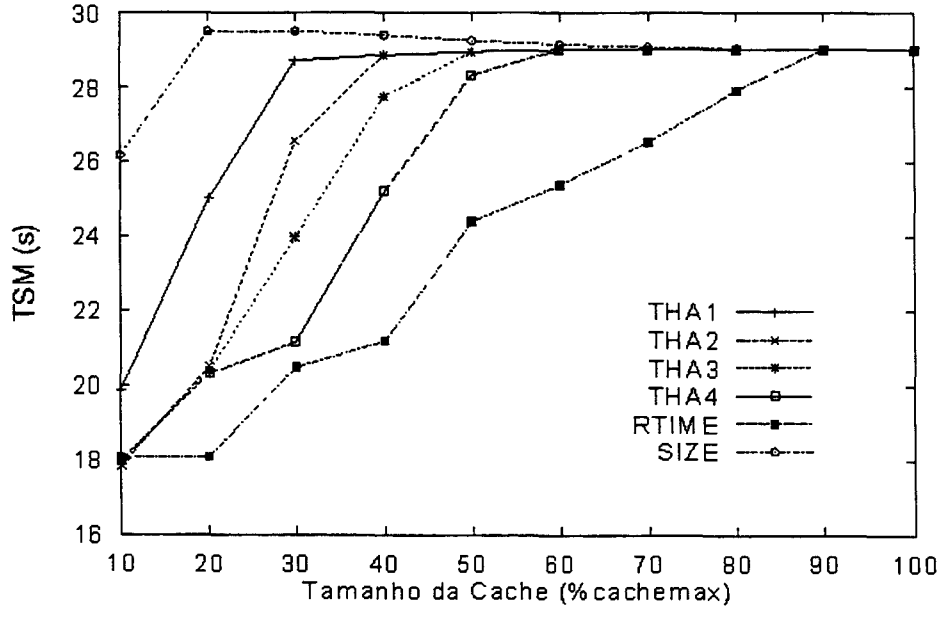

Figura 3.8a: Trace de Berkeley1.

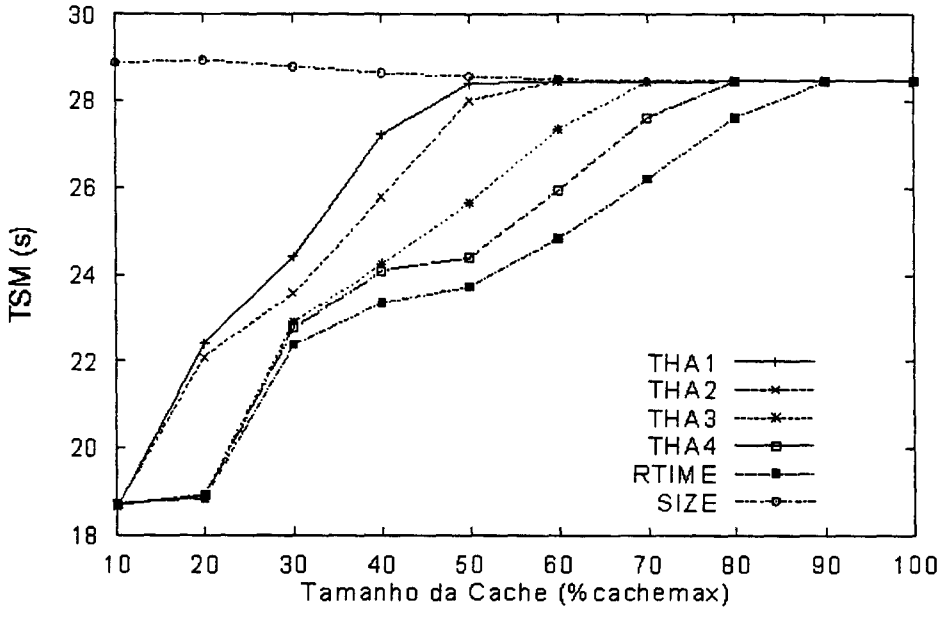

Figura 3.8b: Trace de Berkeley2.

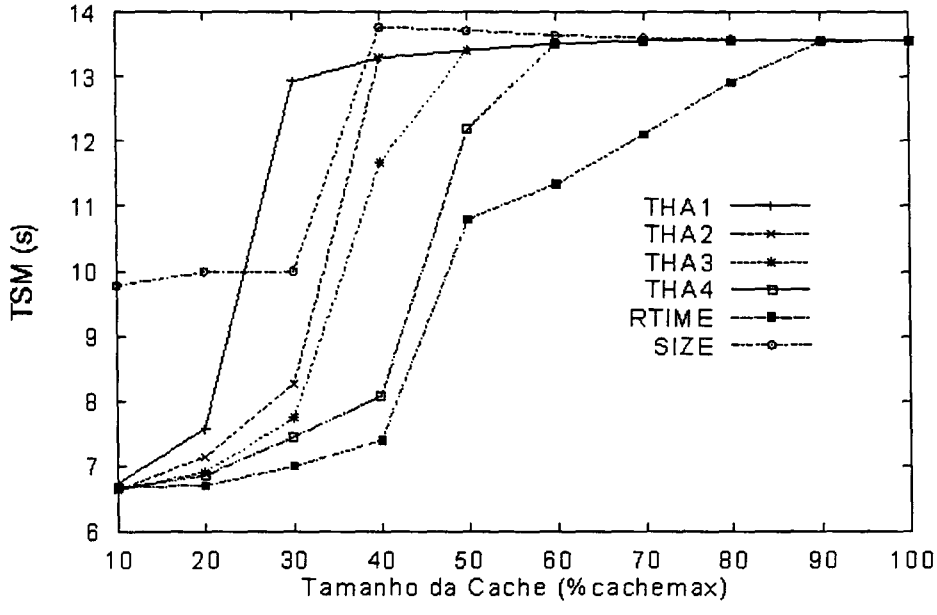

Figura 3.8c: Trace da Digital1.

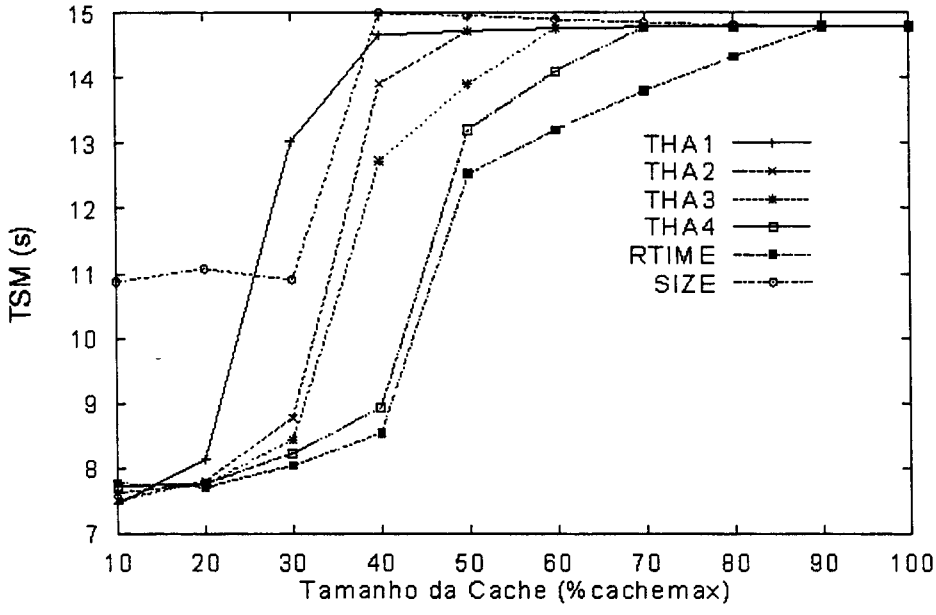

Figura 3.8d: Trace da Digital2.

Figura 3.8: Tempo médio gasto por miss (time spent per miss), como função do tamanho da cache para diferentes valores limites de período com ausência de referências.

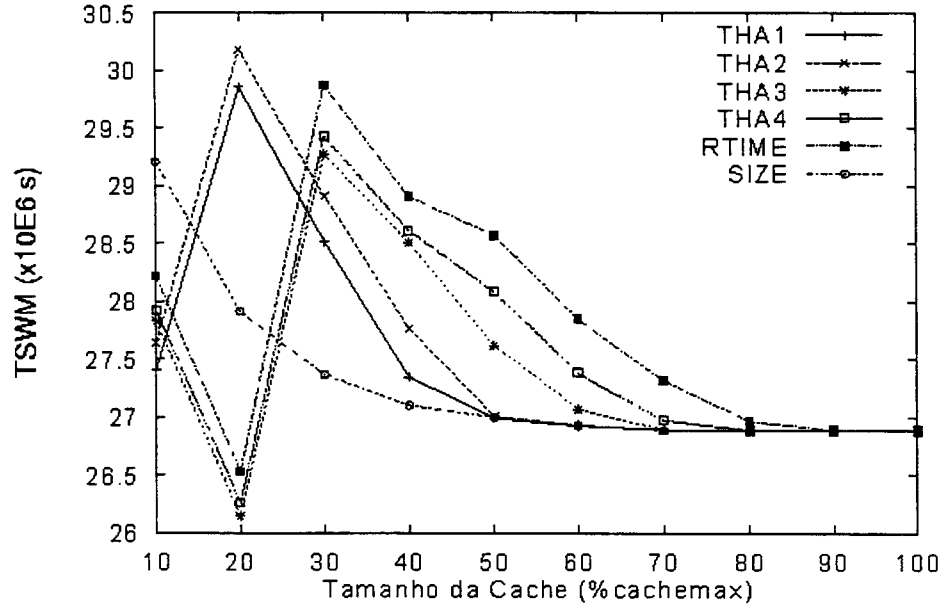

Figura 3.9: Tempo total gasto com misses (time spent with misses), como função do tamanho da cache para diferentes valores limites de período com ausência de referências. Trace de Berkeley2.

Na Figura 3.10, podemos ver que as políticas THAs que usam limites altos, assim como a política RTIME, têm baixa variância do tempo de recuperação dos misses já que mantêm fora da cache documentos pequenos e muito referenciados, e esses documentos mantêm os valores de tempo de recuperação em torno da média. Já as políticas com limites baixos mantêm fora da cache documentos pouco referenciados, que variam muito com relação à média, já que grande parte é referenciada uma única vez.

Foi observado que o desempenho desta política híbrida é bastante dependente do valor escolhido como limite, e que valores mal dimensionados para o limite do período sem referências fazem com que documentos a serem referenciados novamente sejam expulsos desnecessariamente. Também observou-se que um valor muito alto de limite tem pouca validade para uma política híbrida, pois torna a política muito próxima à política RTIME.

Uma alternativa para se tentar obter baixos tempos de respostas é tentar minimizar o miss ratio, bem como o tempo médio de recuperação. Assim sendo, investigou-se políticas híbridas que consideram as políticas SIZE e RTIME, já que estas políticas obtiveram os melhores resultados nos dois parâmetros citados.

Essas novas políticas, em uma primeira etapa, aplicam a chave primária para definir em qual intervalo do valor desta chave o pedido é classificado, dividindo os documentos em grupos, e em um segundo momento aplicam a chave secundária como critério de desempate [AgWoYu97]. Foram definidas as políticas RTS, RTS-Int e SRT. As políticas RTS e RTS-Int definem intervalos para os valores da chave primária tempo de recuperação. Na política RTS, os documentos são agrupados pelo valor de $\log_2$ do tempo de recuperação, enquanto que na política RTS-Int empregam-se valores pré-estabelecidos para os limites dos intervalos dos grupos. Ambas as políticas usam o inverso do tamanho do documento como chave em cada grupo.

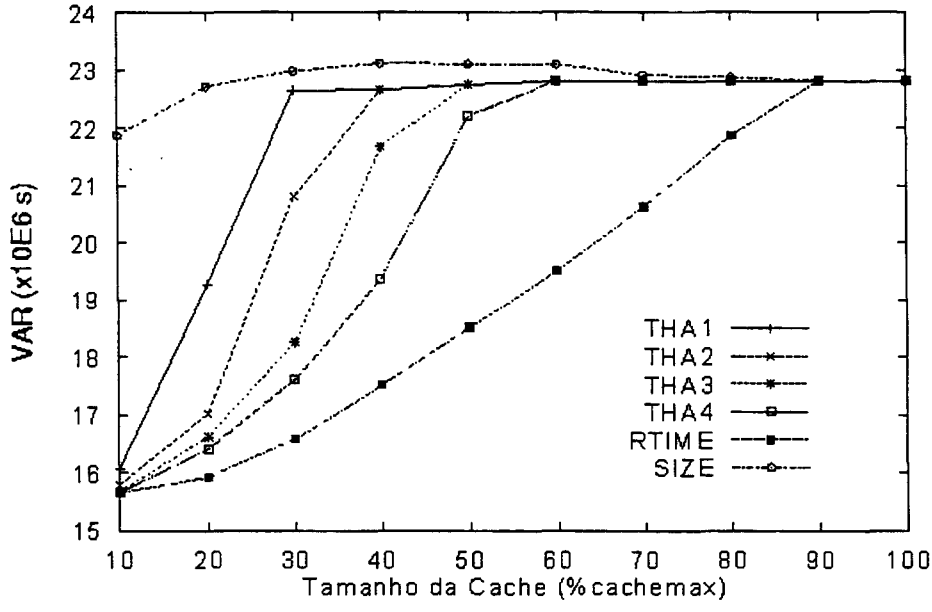

Figura 3.10a: Trace de Berkeley1.

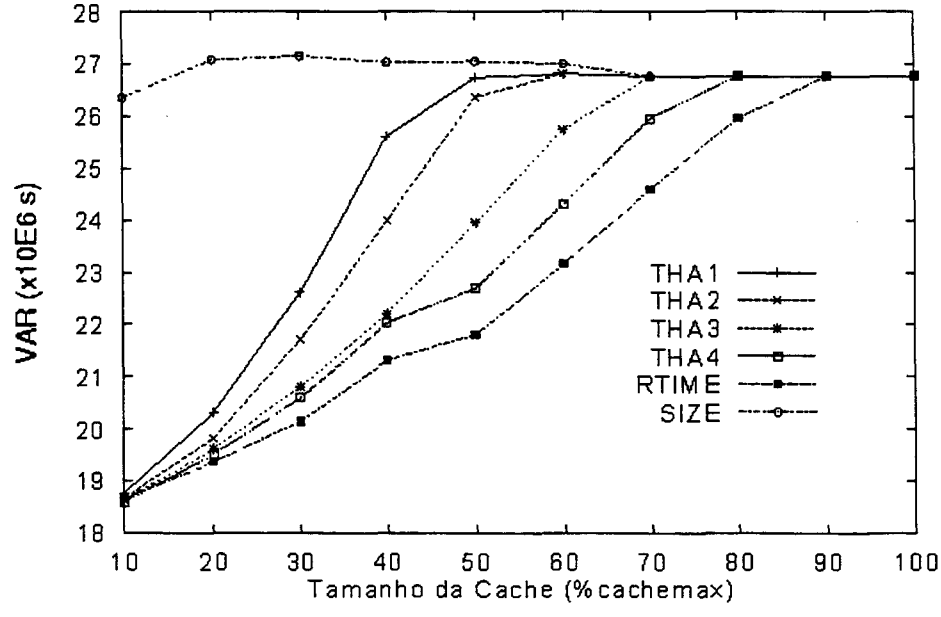

Figura 3.10b: Trace de Berkeley2.

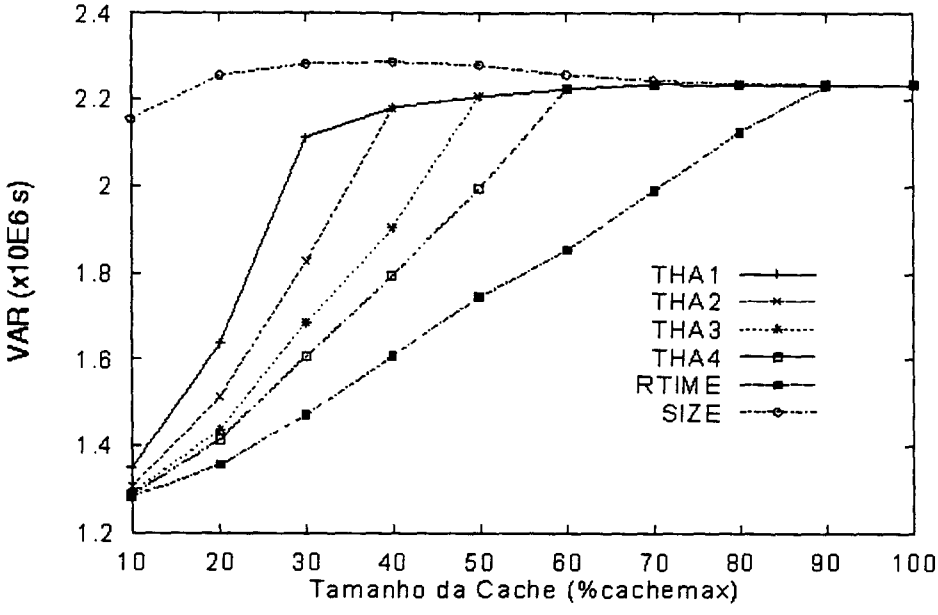

Figura 3.10c: Trace da Digital1.

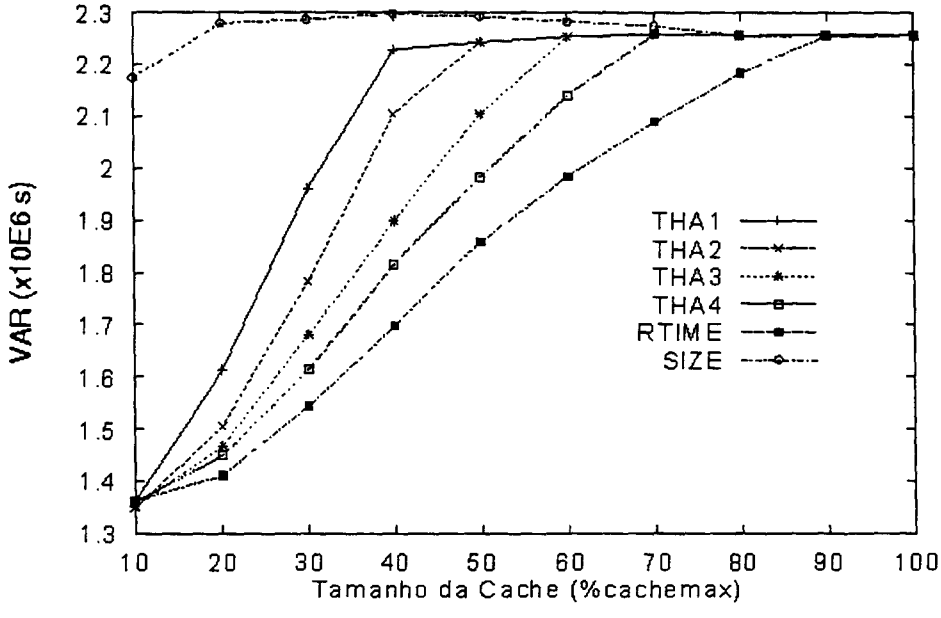

Figura 3.10d: Trace da Digital2.

Figura 3.10: Variância do tempo de recuperação dos misses como função do tamanho da cache para diferentes valores limites de período com ausência de referências.

Já a política SRT utiliza o $\log_2$ do tamanho do documento como chave primária e o tempo de recuperação como chave secundária. Essa divisão em grupos feita pelas políticas RTS e SRT assemelha-se ao particionamento de caches feita por Virgilio et al [MuAlMe98], onde a cache é dividida em partes que armazenam classes de documentos com base em seus tamanhos, e uma política é aplicada a cada parte. A grande diferença entre os estudos reside no fato de termos usado uma única cache particionada logicamente, enquanto em [MuAlMe98] as caches estavam particionadas fisicamente.

Na Figura 3.11 são mostrados os hit ratios para RTS, RTS-Int e SRT, enquanto na Figura 3.12 mostram-se os resultados para o parâmetro weighted hit ratio. Para a política RTS-Int utiliza-se os intervalos em segundos [0, 1], [1, 2], [2, 3], [3, 5] [5, 9], [9, 16] e [16, 30]. A política SRT produz praticamente os mesmos resultados que o uso isolado do inverso do tamanho do documento. Isto se deve ao fato de que como o tamanho do documento contribui para o tempo de recuperação de um documento, o agrupamento de documentos com tamanhos parecidos pode levar ao agrupamento de documentos com tempo de resposta também parecidos o que elimina as vantagens do uso do tempo de recuperação como chave secundária.

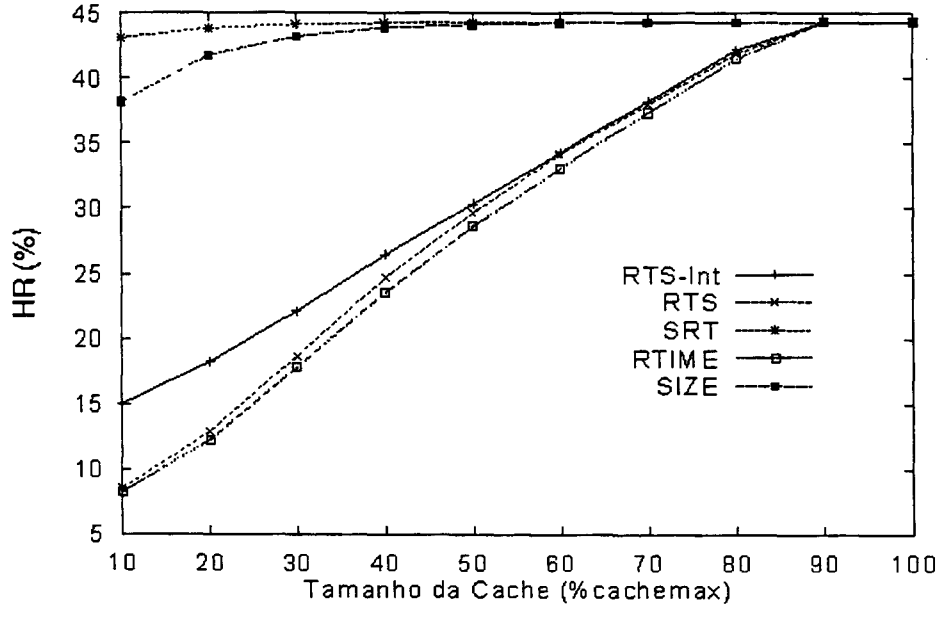

Figura 3.11a: Trace de Berkeley1.

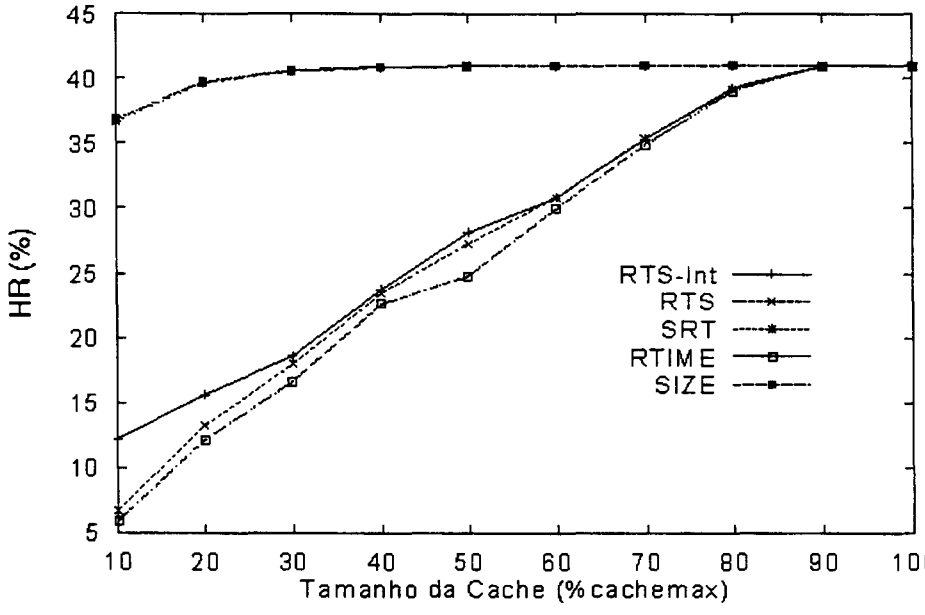

Figura 3.11b: Trace de Berkeley2.

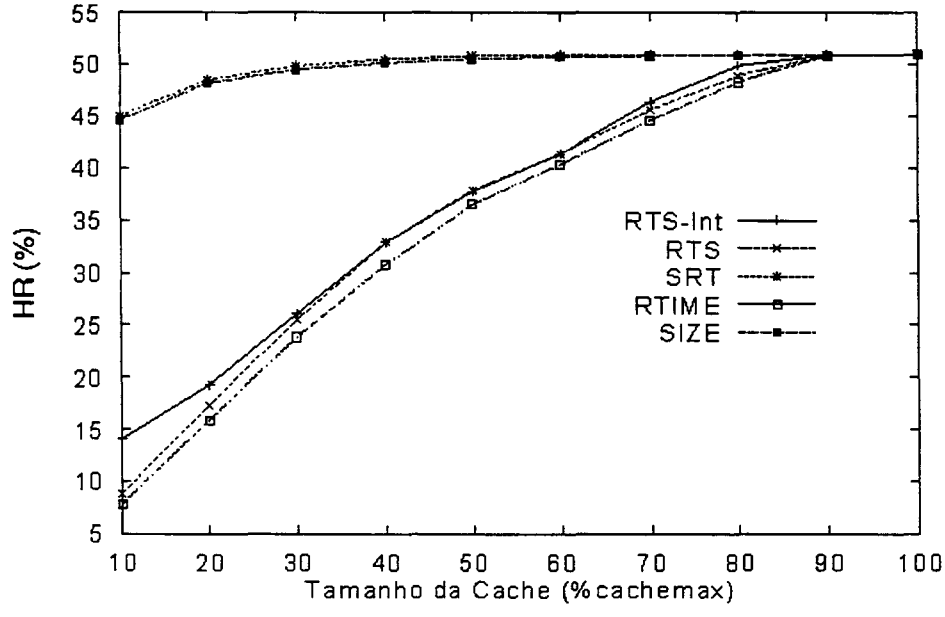

Figura 3.11c: Trace da Digital1.

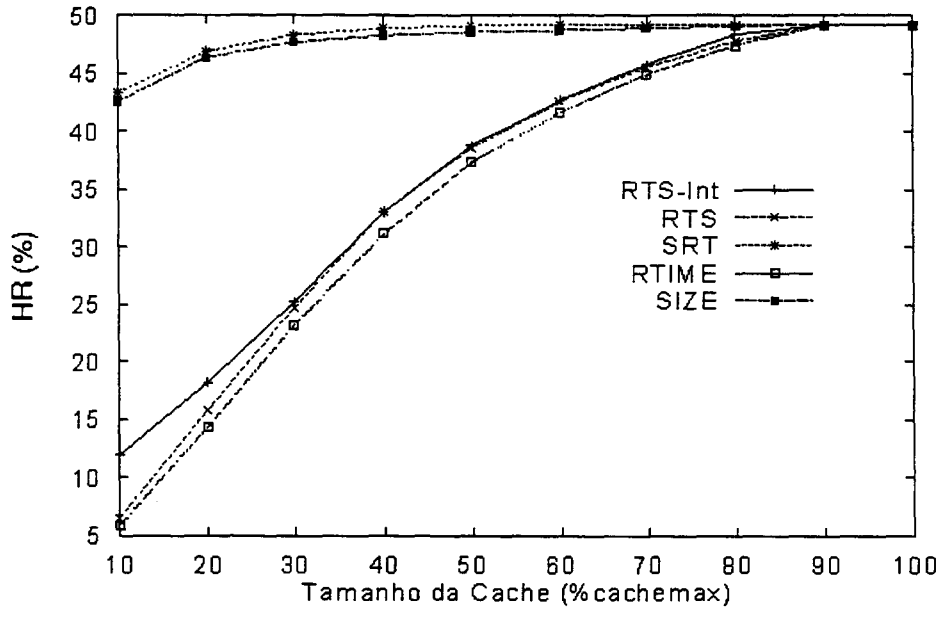

Figura 3.11d: Trace da Digital2.

Figura 3.11: Taxa de acerto na cache (hit ratio) como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

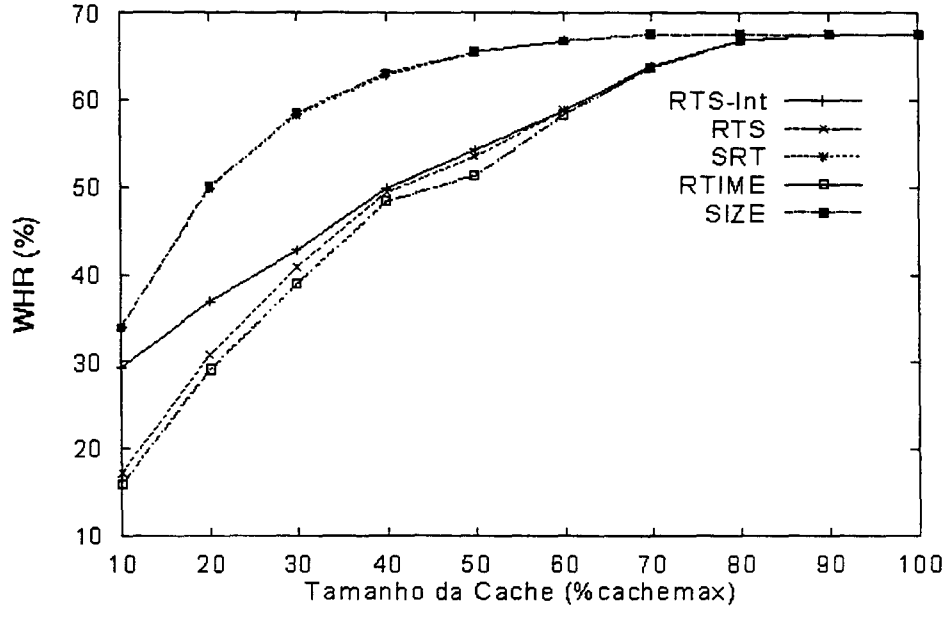

Figura 3.12: Taxa de acerto ponderada pelo tamanho (weighted hit ratio) como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves. Trace de Berkeley2.

A explicação para a semelhança entre os resultados de SRT e da política SIZE é válida também para a semelhança entre os resultados gerados pela política RTS e pela política RTIME. Os resultados de RTS-Int diferenciam-se dos resultados da política RTIME para caches menores de 20% da cache infinita, mostrado nos gráficos de Berkeley2, o que indica a vantagem de se aplicar um critério com granulosidade fina para este tamanho de cache. Entretanto, observa-se que o propósito de se diminuir o miss ratio e o tempo de recuperação não é alcançado.

A Figura 3.13 mostra o tempo médio de recuperação para as políticas RTS, RTS-Int e SRT. Ao analisar a figura, podemos reafirmar que o desempenho das políticas RTS e SRT ficou muito próximo ao desempenho das políticas RTIME e SIZE, respectivamente. A política RTS-Int teve um desempenho intermediário com relação às políticas anteriores, embora ainda tenha comportamento muito parecido com a política RTS.

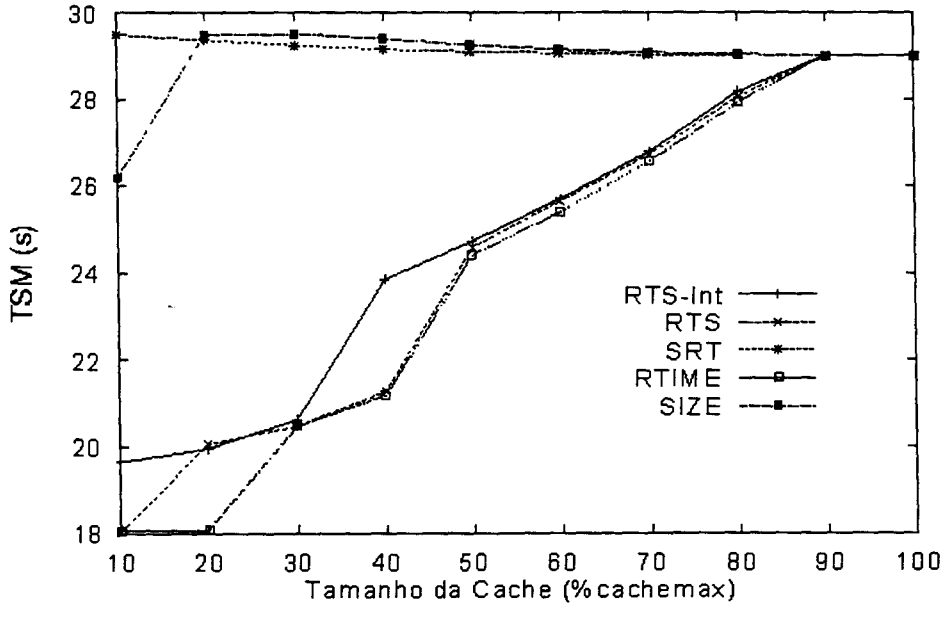

Figura 3.13a: Trace de Berkeley1.

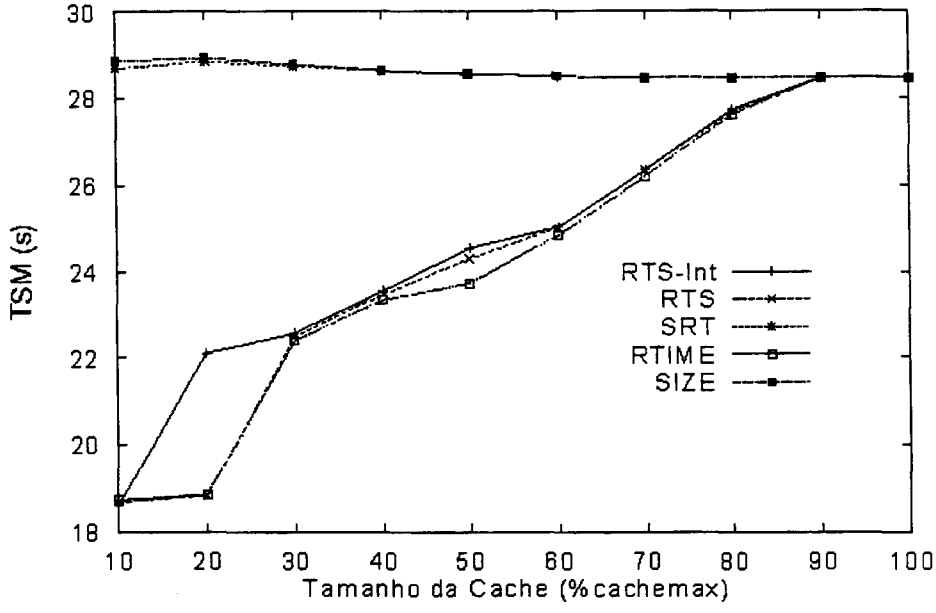

Figura 3.13b: Trace de Berkeley2.

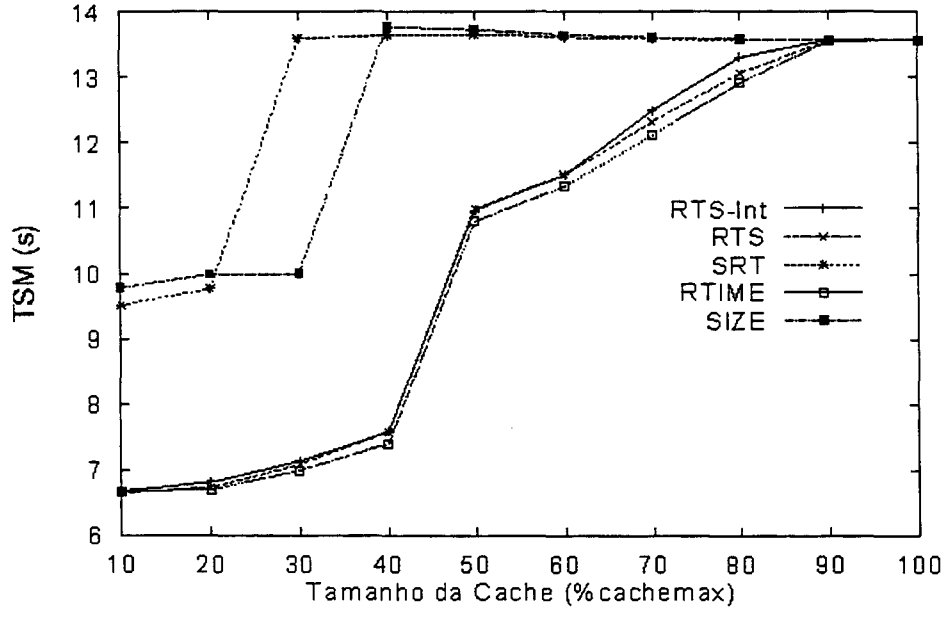

Figura 3.13c: Trace da Digital1.

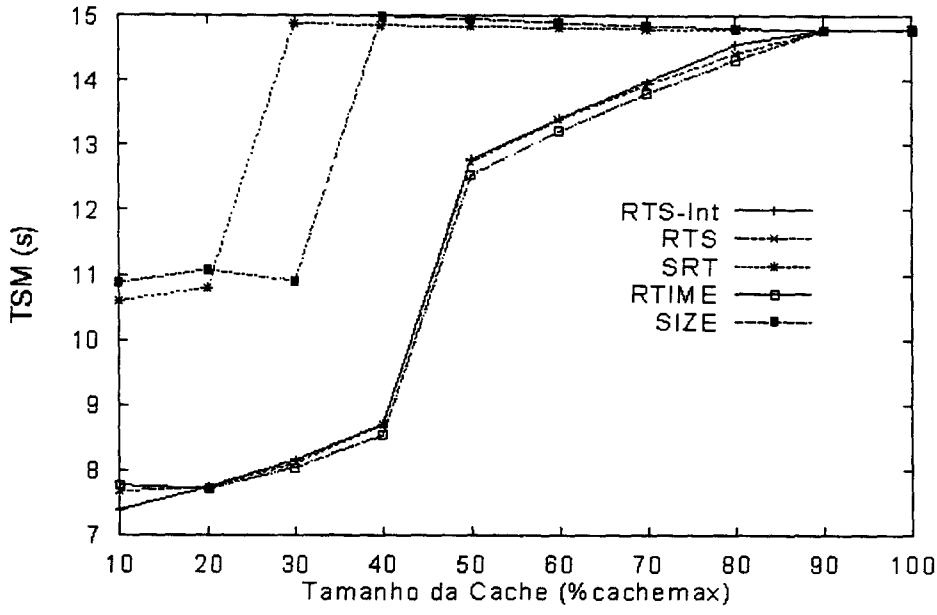

Figura 3.13d: Trace da Digital2.

Figura 3.13: Tempo médio gasto por miss (time spent per miss), como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

Esperava-se que a política RTS-Int obtivesse um desempenho que fosse considerado razoável em todos os parâmetros avaliados. Entretanto, constatou-se que essa política é muito sensível ao tamanho dos seus intervalos e, pelo seu funcionamento e natureza do tráfego, tende a se parecer muito com a política RTIME.

A Figura 3.14 mostra o resultado dessas políticas para o parâmetro tempo total gasto com misses, para o trace de Berkeley2, e a Figura 3.15 mostra os resultados da variância destas políticas. Podemos notar que o mesmo comportamento para os parâmetros anteriores ainda pode ser observado.

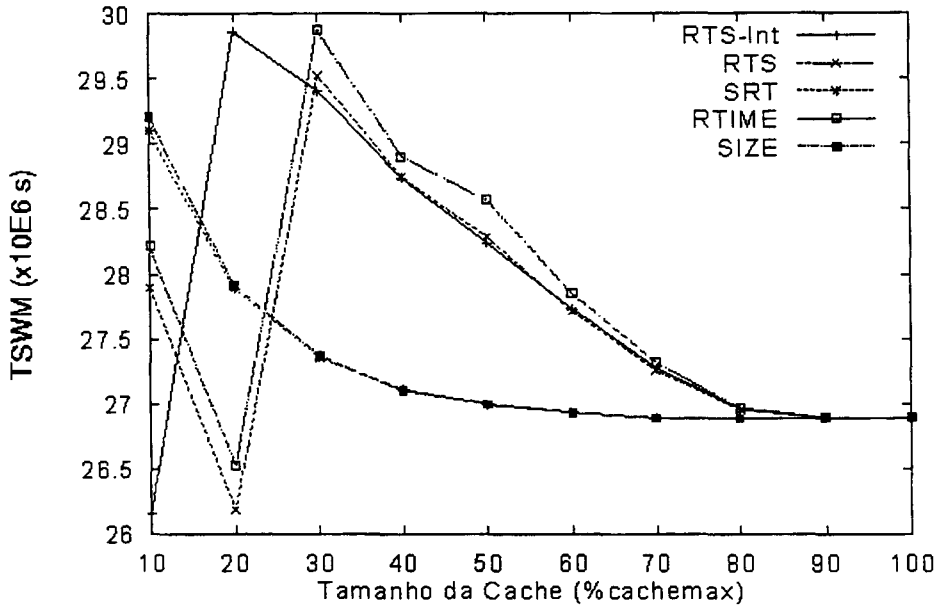

Figura 3.14: Tempo total gasto com misses (time spent with misses), como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves. Trace de Berkeley2.

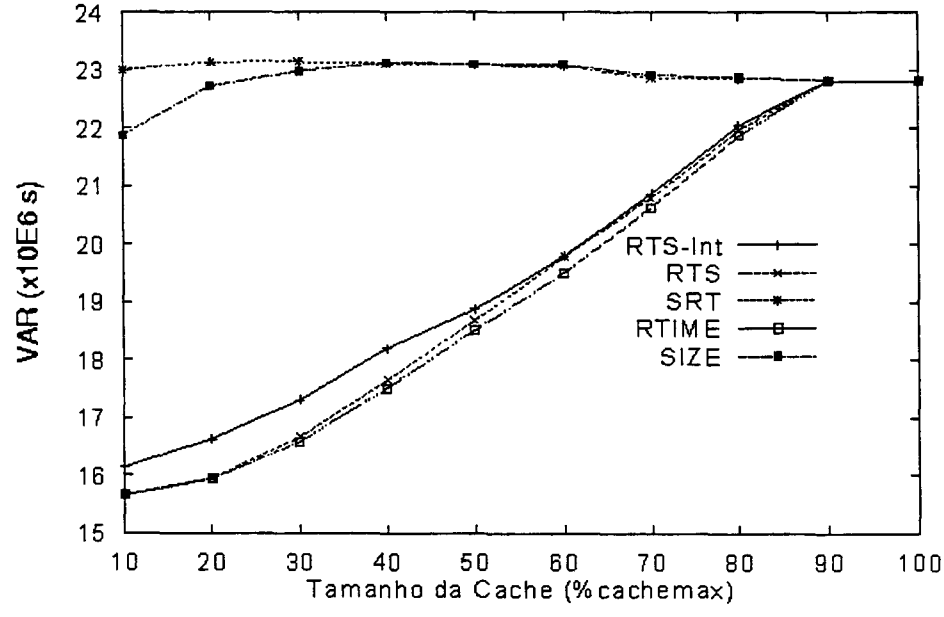

Figura 3.15a: Trace de Berkeley1.

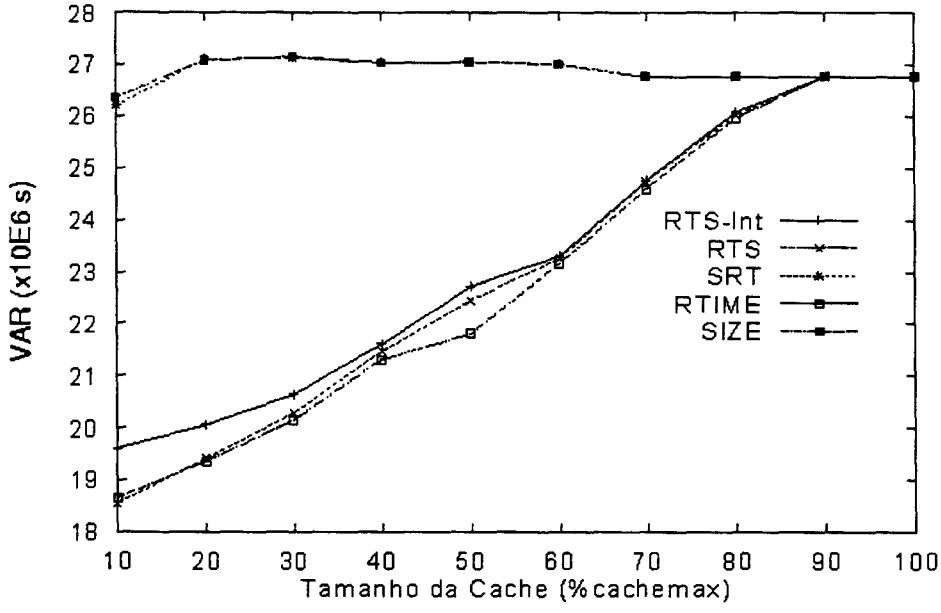

Figura 3.15b: Trace de Berkeley2.

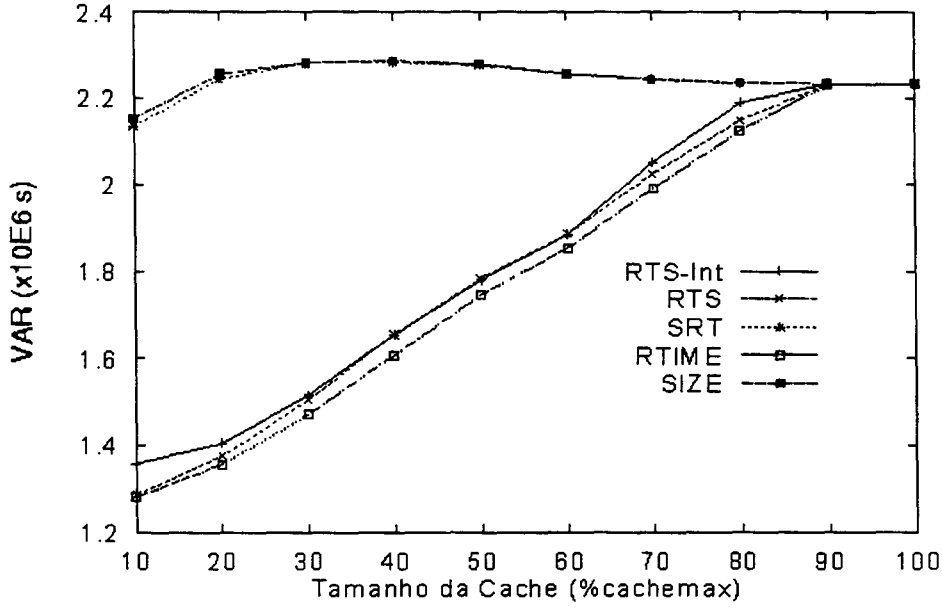

Figura 3.15c: Trace da Digital1.

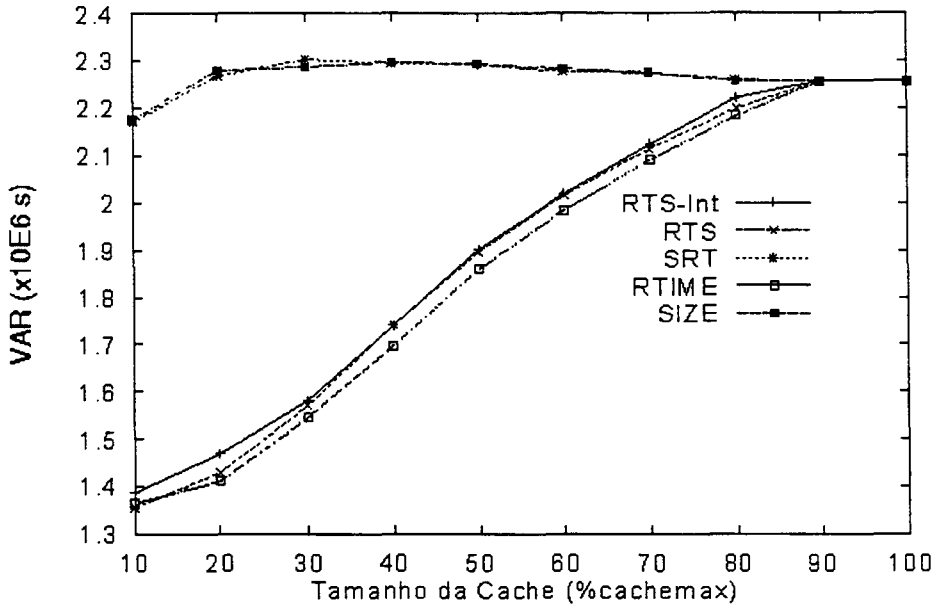

Figura 3.15d: Trace da Digital2.

Figura 3.15: Variância do tempo de recuperação dos misses como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

3.4. O uso de controle de admissão em Web caches

Um mecanismo de Controle de Admissão estabelece critérios para admissão de documentos na cache. A idéia por trás deste mecanismo é evitar que documentos com baixo valor em termos de desempenho (ou QoS) possam substituir um ou mais documentos de maiores valores. Políticas mais elaboradas de controle devem comparar o valor do novo documento com todos os documentos em cache, bem como predizer com um certo grau de precisão futuras requisições para os documentos na cache. Na presente investigação, pretende-se avaliar a eficácia de um mecanismo simples de controle de admissão.

Sabe-se, a partir dos resultados da seção 3.2, que a chave tempo de recuperação minimiza o tempo médio de recuperação de um documento, ou seja, que é possível diminuir o tempo médio de recuperação mantendo-se documentos com alto tempo de recuperação na cache. Assim sendo, propõe-se o uso de um valor limite para o qual documentos com tempo de recuperação menores que este limite não sejam armazenados na cache. Um mecanismo de controle de admissão avalia, portanto, o compromisso entre gerar misses desnecessários e armazenar somente documentos com um certo valor.

As políticas analisadas são resumidas na Tabela 3.7, e funcionam da seguinte forma: só entram na cache os documentos que tiverem tempo de recuperação maior que o limite estabelecido. Desta forma, tenta-se deixar na cache apenas documentos que são valiosos na diminuição da latência percebida pelo usuário. Os limites apresentados na tabela são expressados em segundos. Na Figura 3.16, podemos ver o hit ratio para valores de corte iguais a 6s, 30s e 90s. Podemos notar que o uso de controle de admissão aumenta muito o miss ratio, uma vez que impede a entrada de documentos na cache que podem ser referenciados no futuro. O mesmo comportamento pode ser observado na Figura 3.17, para o weighted hit ratio.

Política	Descrição	Limite	Digital 1	Digital 2
RTIMETHR	RTIME com limite para controle de admissão	30s	22,26%	18,86%
RTIMETHR2	RTIME com limite para controle de admissão	6s	12,90%	10,84%
RTIMETHR3	RTIME com limite para controle de admissão	90s	34,25%	27,53%
RTHA1THR	THA1 com limite para controle de admissão	30s	22,26%	18,86%
RTHA2THR	THA2 com limite para controle de admissão	30s	22,26%	18,86%

Tabela 3.7: Políticas com controle de admissão, os limites usados e a porcentagem de documentos que é impedida de entrar na cache para os respectivos limites.

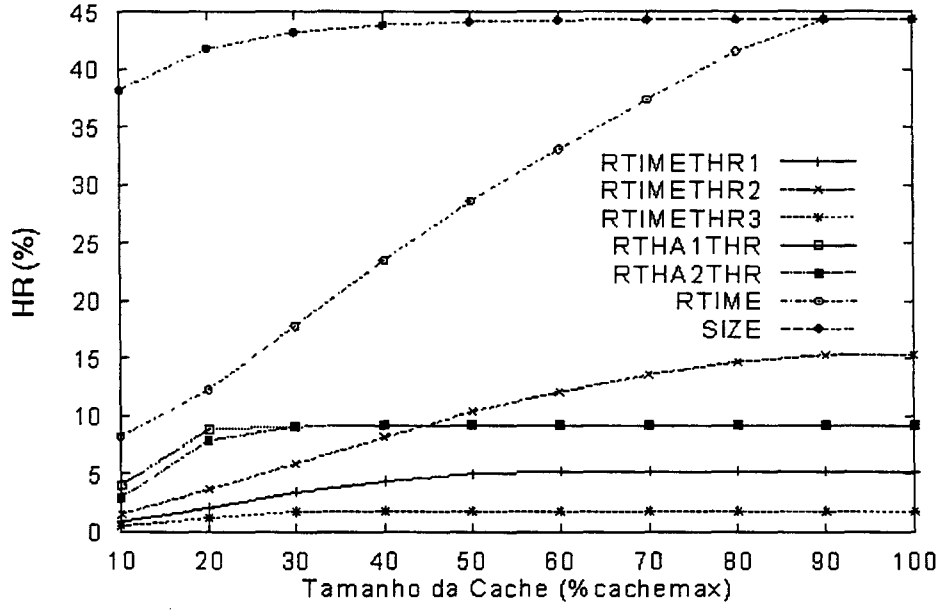

Figura 3.16a: Trace de Berkeley1.

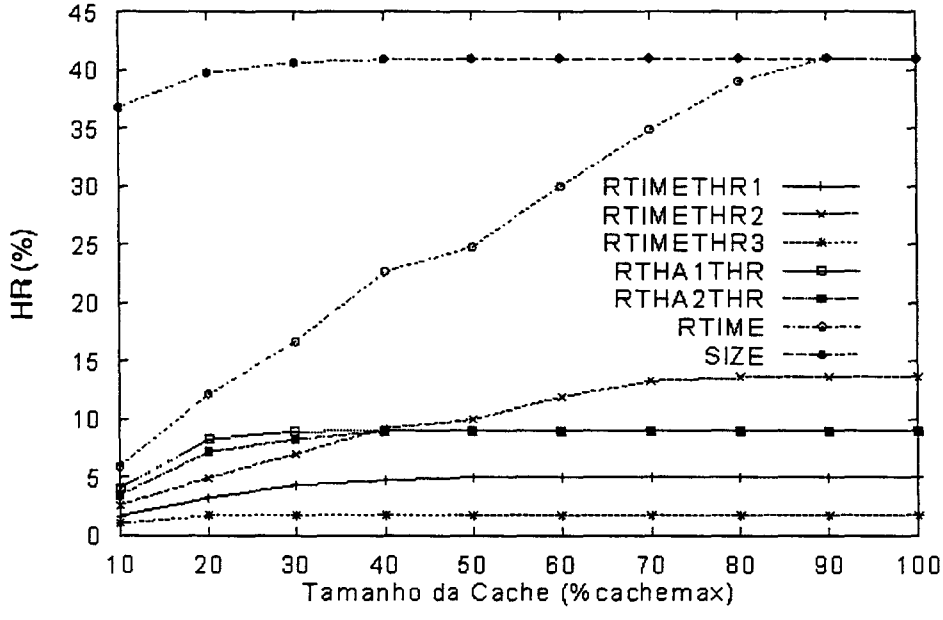

Figura 3.16b: Trace de Berkeley2.

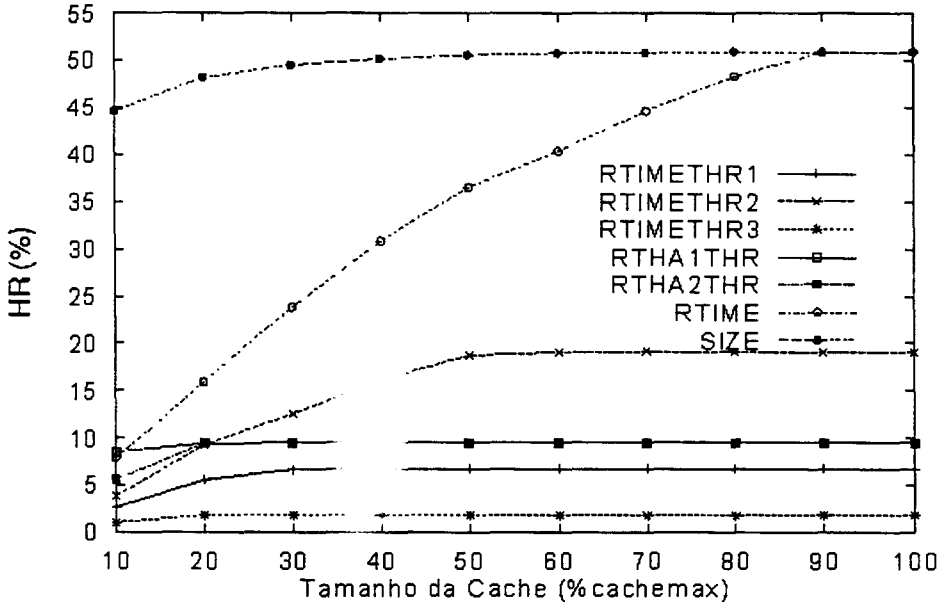

Figura 3.16c: Trace da Digital1.

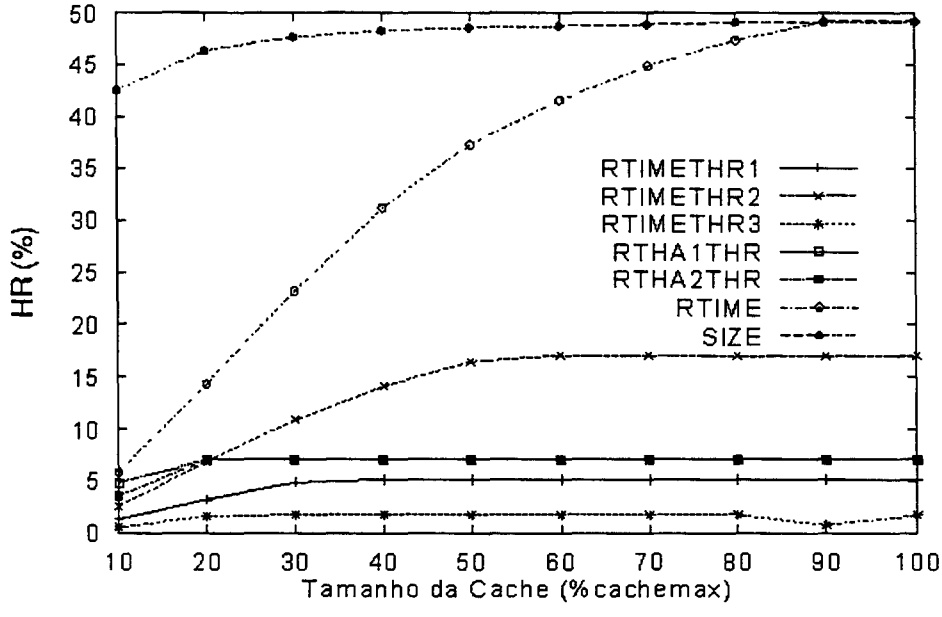

Figura 3.16d: Trace da Digital2.

Figura 3.16: Taxa de acerto na cache (hit ratio) como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

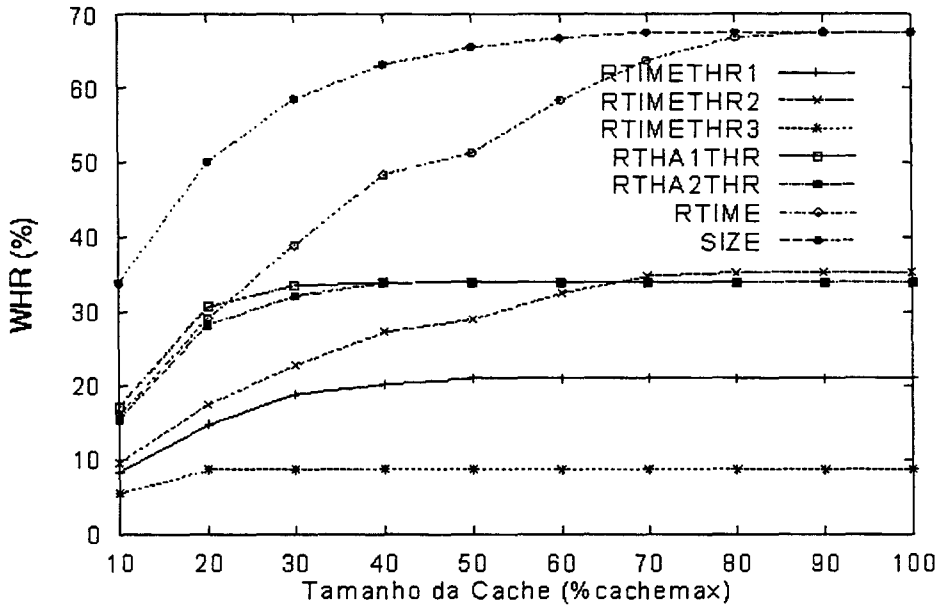

Figura 3.17: Taxa de acerto ponderada pelo tamanho(weighted hit ratio) como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves. Trace de Berkeley2.

Na Figura 3.18 plota-se o valor médio do tempo de recuperação para as políticas da Tabela 7. Nota-se que é possível obter tempos de recuperação consideravelmente menores com o uso de controle de admissão do que com o uso exclusivo da chave tempo de recuperação. Ao se aumentar o valor de corte obtém-se valores menores para o tempo médio de recuperação. Por outro lado, ao se aumentar o valor de corte, aumentamos o tempo gasto com misses, já que o número de misses passa a ser bem maior. Esta tendência nos resultados do tempo médio de recuperação deve-se à diminuição da proporção dos documentos com tempos de recuperação acima do valor limite no cálculo do tempo médio de recuperação. Neste caso, os documentos permanecem na cache ao invés de serem expulsos para armazenar uma nova solicitação com tempo de recuperação abaixo do valor limite. Verifica-se que com o aumento do tamanho da cache, a vantagem de se ter um mecanismo de controle de admissão aumenta. Na realidade, com o aumento da cache diminui-se o miss ratio e um miss devido a um documento com alto tempo de recuperação previamente expulso tem um peso maior na computação do tempo médio de recuperação.

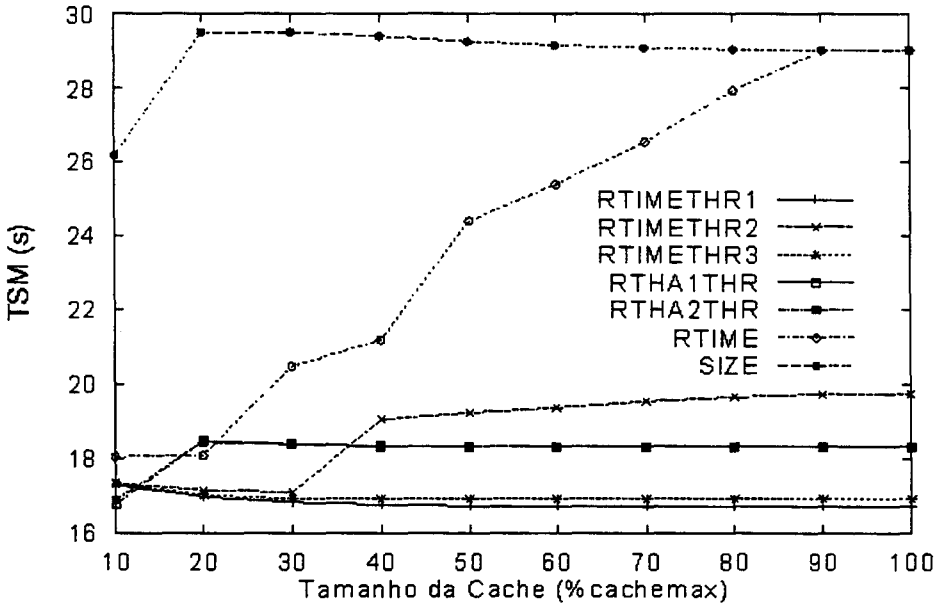

Figura 3.18a: Trace de Berkeley1.

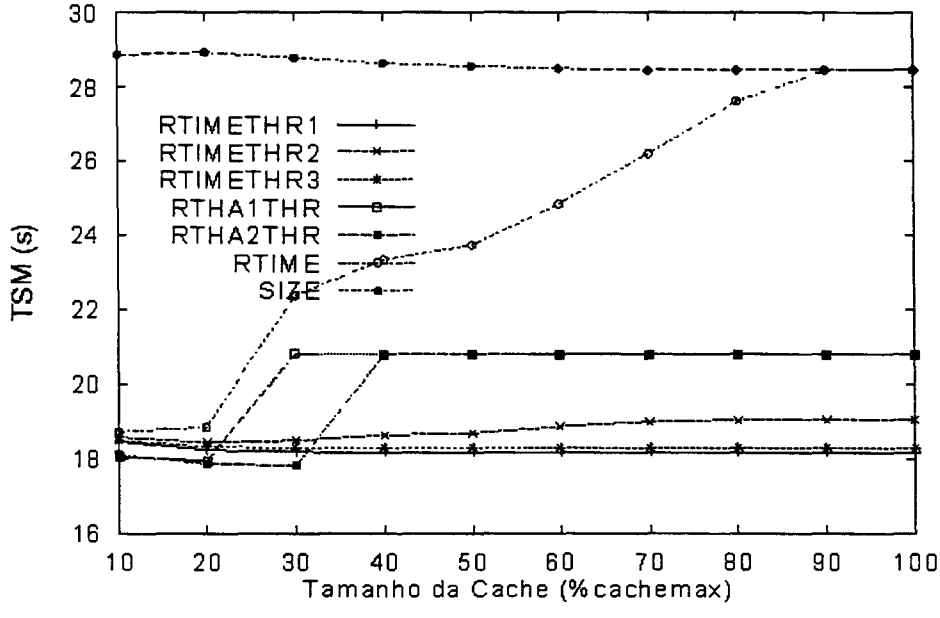

Figura 3.18b: Trace de Berkeley2.

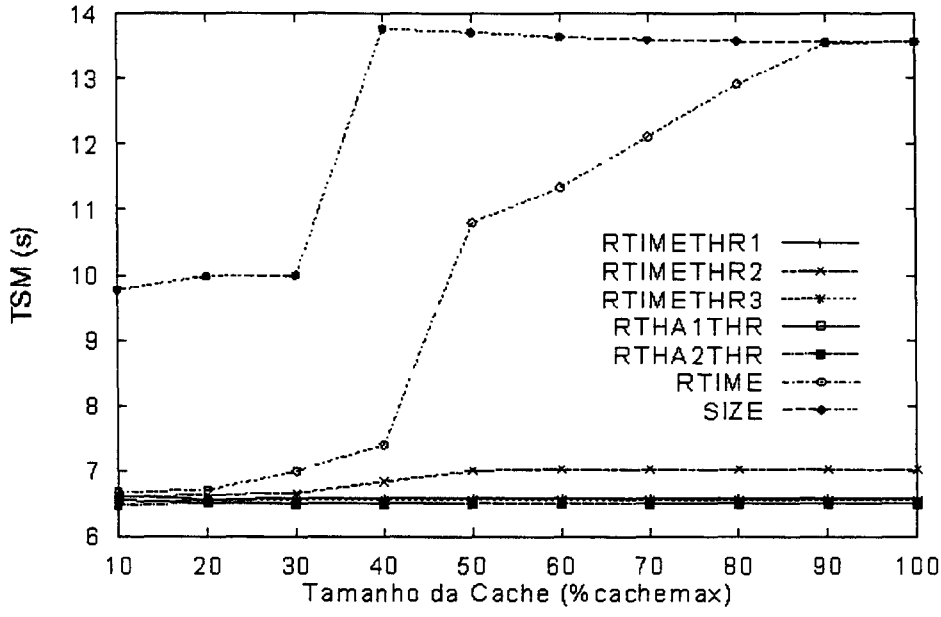

Figura 3.18c: Trace da Digital1.

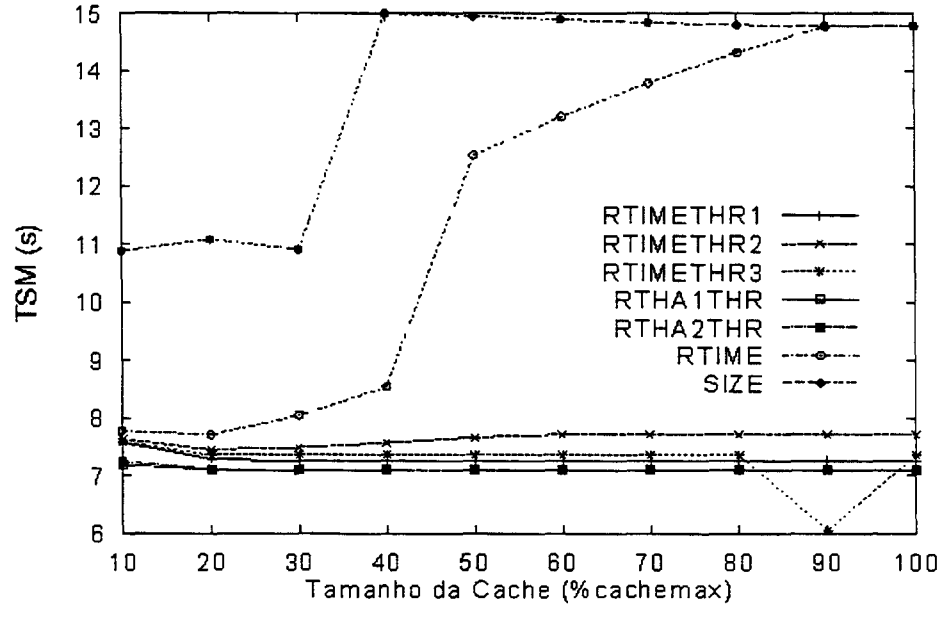

Figura 3.18d: Trace da Digital2.

Figura 3.18: Tempo médio gasto por miss (time spent per miss), como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

As mesmas observações feitas para os parâmetros anteriores são mantidas para o tempo total gasto com misses e variância, mostrados nas Figuras 3.19 e 3.20.

É possível observar que a variância do tempo de recuperação diminui com o aumento do valor de corte, pois mantém-se cada vez mais um número maior de documentos com longos tempos de recuperação na cache, evitando-se, assim, uma elevada contribuição de altos tempos de recuperação no cálculo da variância.

Analisando a política que combina controle de admissão e a política de expulsão de documentos não referenciados por longos períodos, podemos notar que a política de controle de admissão tem um impacto muito maior do que políticas que expulsam documentos antigos.

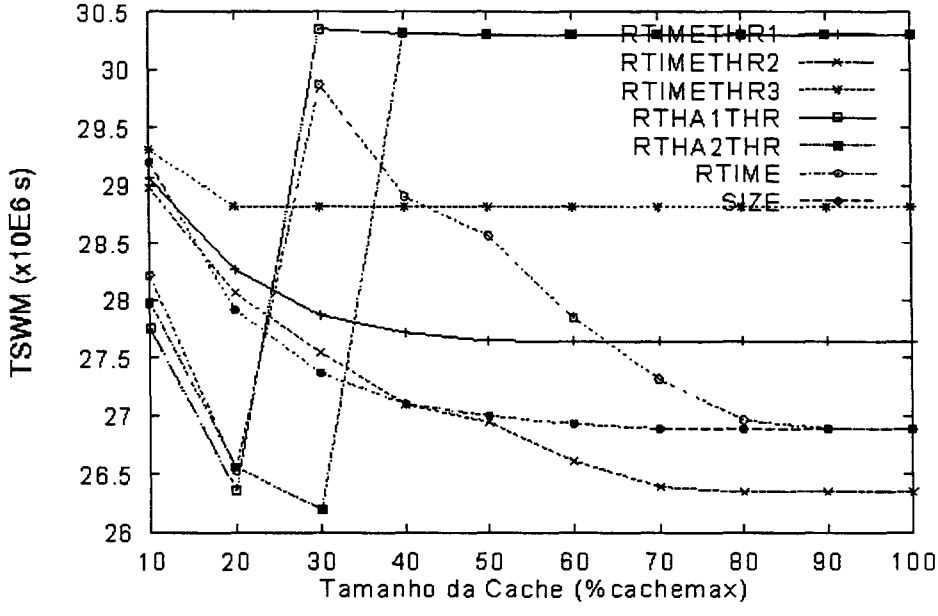

Figura 3.19: Tempo total gasto com misses (time spent with misses), como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves. Trace de Berkeley2.

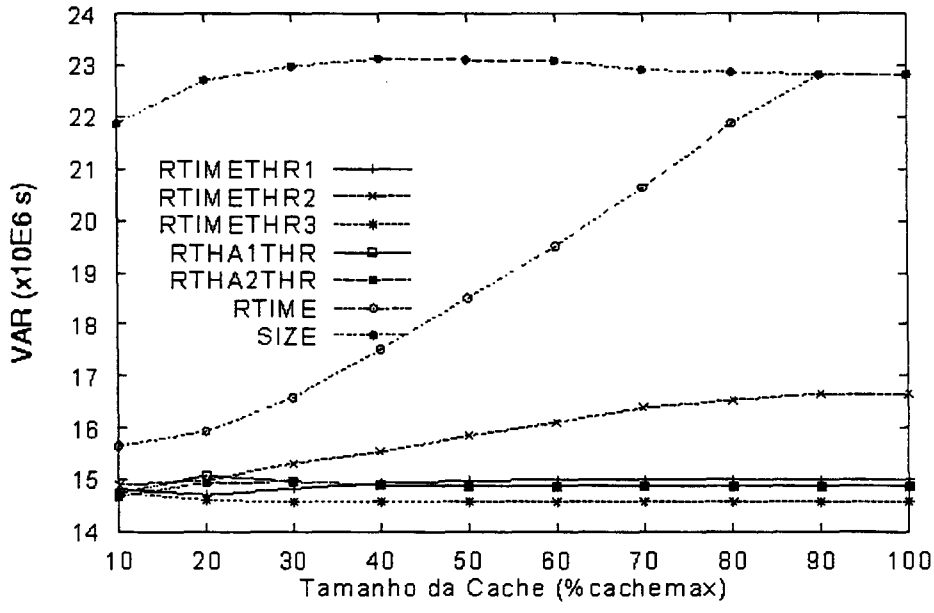

Figura 3.20a: Trace de Berkeley1.

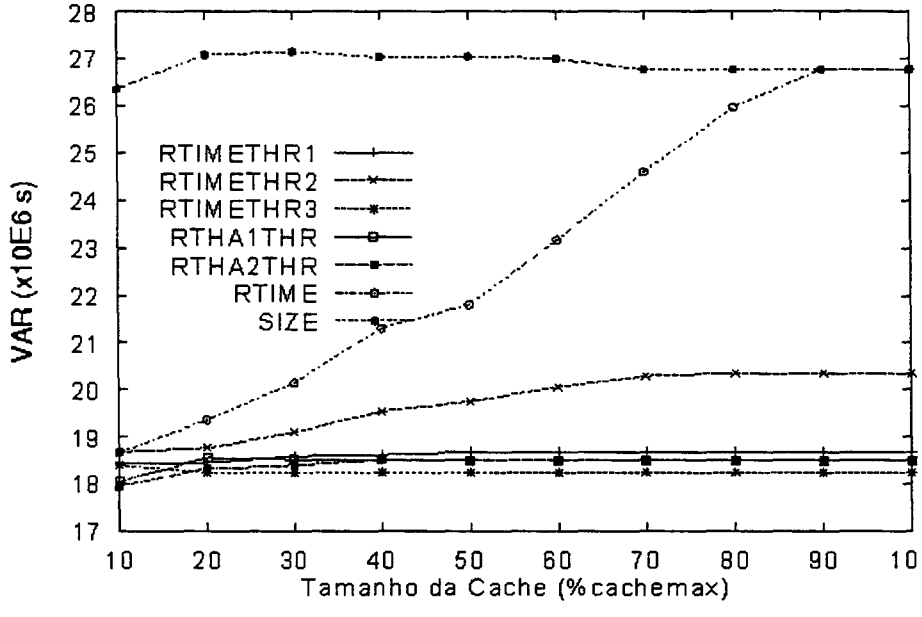

Figura 3.20b: Trace de Berkeley2.

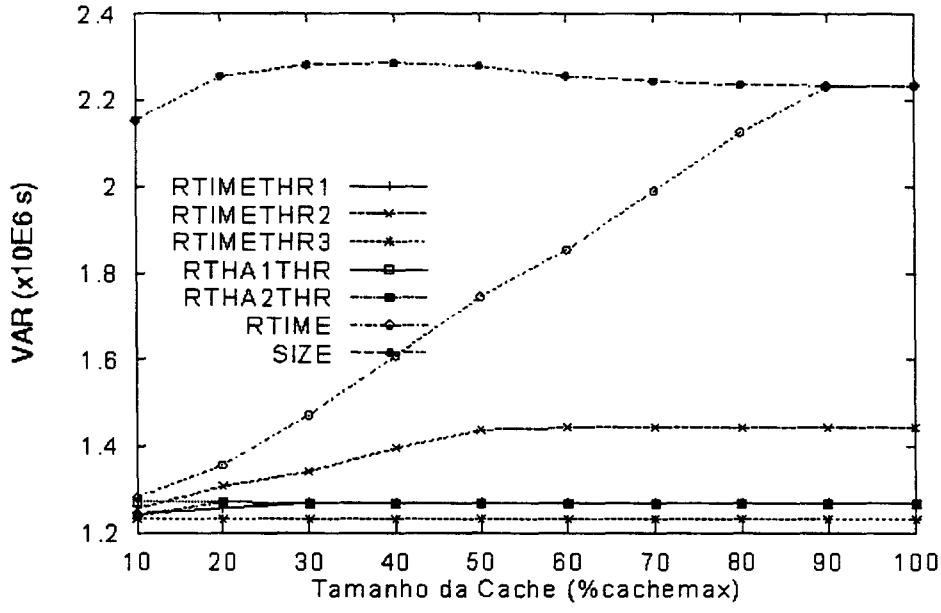

Figura 3.20c: Trace da Digital1.

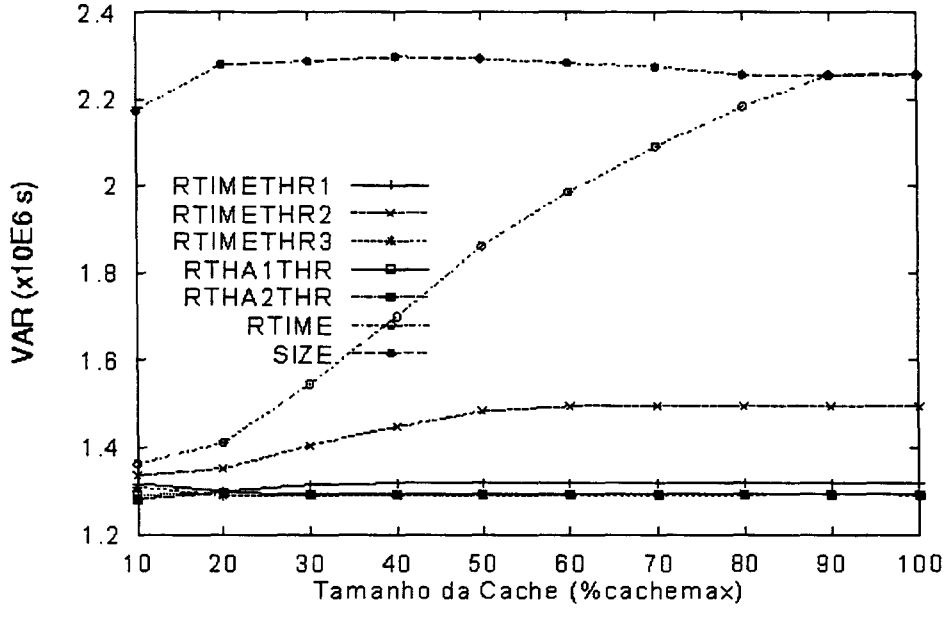

Figura 3.20d: Trace da Digital2.

Figura 3.20: Variância do tempo de recuperação dos misses como função do tamanho da cache para diferentes políticas híbridas, que utilizam o tempo de recuperação e o inverso do tamanho do documento como chaves.

3.5. Resumo dos principais resultados

Políticas de expulsão têm um impacto significativo no desempenho de uma Web cache bem como em parâmetros de Qualidade de Serviço. Este capítulo mostrou a investigação do uso da chave tempo de recuperação em políticas de Web cache. Resultados indicam que a chave tempo de recuperação fornece a menor média e variância do tempo de recuperação em todas as políticas analisadas. Isso se deve ao fato de esta chave tentar deixar na cache documentos com altos tempos de recuperação, o que tende a diminuir o tempo médio de recuperação e manter a variância do tempo dos misses baixa, já que os misses são compostos por documentos muito referenciados e com baixo tempo de recuperação.

Evidenciou-se que é vantajoso expulsar documentos antigos da cache, porém os resultados são altamente sensíveis à escolha do valor limite para a decisão de expulsão. Se este valor de limite for muito grande, o limite não influenciará muito a política, já que poucos documentos alcançarão o limite. Neste caso, a política se aproxima da política RTIME. Entretanto, se esse limite for muito pequeno, podem ser expulsos documentos que serão referenciados novamente, o que diminui o desempenho da política. Um limite pequeno deixa fora da cache documentos grandes (que são pouco referenciados), o que faz com que os resultados da política se aproximem da política SIZE.

Investigou-se o uso de políticas que dividem a cache em grupos, segundo algum critério, e aplicam uma política de substituição a cada grupo. Usamos o tempo de recuperação e o inverso do tamanho do documento para divisão em grupos e como política de substituição. Os resultados mostraram que, dependendo da divisão dos grupos, pode-se obter uma política que tenha um desempenho intermediário para os parâmetros tempo médio de recuperação por miss e taxa de acerto na cache. Isso é um bom resultado, já que chegou-se à conclusão de que esses dois parâmetros são antagônicos, e que quando procuramos otimizar um deles, estamos diminuindo o desempenho no outro.

O uso de políticas de controle de admissão baseadas em valor de corte para o tempo de recuperação mostrou-se bastante eficiente na diminuição da média e da variância do tempo de recuperação. Entretanto, a diminuição da taxa de acerto na cache foi muito grande, o que exige que este controle seja usado com cuidado. O controle de admissão tem a grande vantagem de permitir uma filtragem dos documentos que podem entrar na cache, com o objetivo de otimizar algum parâmetro que, no caso deste estudo, foi o tempo médio de recuperação de um documento.

Capítulo 4

Uma descrição do tempo entre misses

Neste capítulo, investiga-se a caracterização do tempo entre misses em Web caches. Durante os experimentos de simulação apresentados no capítulo 3, foram gerados dados sobre os tempos entre misses consecutivos, para diferentes políticas. Esse tempo entre os misses é igual ao número de pedidos feitos entre dois misses consecutivos. Um valor de tempo entre misses igual a 1 indica que houve um hit entre dois misses na cache.

O valor do tempo entre os misses em uma cache representa um período sem solicitações a documentos remotos e pode ser usado, entre outras coisas, para orientar algoritmos de *prefetching* sobre quanto tempo existe disponível para que documentos possam ser recuperados antecipadamente, antes que o usuário volte a pedir documentos que não se encontram na cache.

No presente capítulo, investiga-se a existência de auto-semelhança nos tempos entre misses coletados a partir das políticas mostradas anteriormente. Em seguida, verifica-se se os tempos entre misses podem ser modelados segundo alguma distribuição conhecida. A modelagem segundo alguma das distribuições estudadas poderia ser usada para tentar prever, na ocorrência de um miss, quanto tempo haveria até que um novo miss ocorresse.

4.1. O padrão de correlação dos tempos entre misses

4.1.1. Processos auto-semelhantes

Os processos estocásticos auto-semelhantes são invariantes em sua estrutura de correlação para diferentes escalas de tempo. Quando são analisados diversos gráficos de um processo auto-semelhante, em diferentes escalas de tempo, podemos ver que eles permanecem estatisticamente inalterados [CroBes96, LelTaq94+]. Esses tipos de processo têm interesse nas mais variadas áreas, como física, economia, geofísica, etc.

Um processo auto-semelhante tem picos observáveis em todas as suas escalas de tempo, exibindo uma dependência de longa duração (LRD – *long range dependence*). Isso significa dizer que valores em qualquer instante estão normalmente correlacionados com valores futuros [CroBes96]. A importância da dependência de longa duração pode ser observada em estudos como [WilTaq97+].

Atualmente, os processo auto-semelhantes têm despertado um interesse especial na modelagem de tráfego de redes, já que diversos estudos mostram que esse tráfego tem um comportamento estatisticamente auto-semelhante [LelTaq94+, WilTaq97+, AbFoxAb97+, CroBes96]. Para se medir o grau de auto-semelhança de um processo usa-se um único parâmetro, chamado de parâmetro de Hurst, ou parâmetro *H*. O parâmetro representa o quão rapidamente decai a função de auto-correlação da série. Para séries auto-semelhantes, o valor de *H* deve variar entre 0.5 e 1. À medida que *H* tende ao valor 1, o grau de auto-semelhança aumenta.

Podem ser usados diversos métodos para testar o grau de auto-semelhança de uma série, ou ainda, para estimar o valor do parâmetro *H*. Abaixo, alguns destes métodos serão brevemente descritos [CroBes96]. Os três primeiros são métodos gráficos, normalmente usados para confirmar resultados, enquanto os dois últimos métodos fornecem uma estimativa do parâmetro *H* com um intervalo de confiança.

- Variance-time plot: Baseia-se na variância, que decai lentamente em séries auto-semelhantes. A variância de $X^{(m)}$ é plotada contra *m* em um gráfico log-log. Uma linha reta com uma inclinação β maior que –1 é um indicativo de auto-semelhança, e o parâmetro *H* é dado por $H = 1 - \beta/2$.

- R/S plot: Usa o fato de que, para um conjunto de dados auto-semelhante, a estatística R/S cresce de acordo com uma lei de potência com expoente *H*, como função do número de pontos incluídos (*n*). Desta forma, a plotagem de R/S contra *n* em um gráfico log-log irá resultar em uma reta cujo valor da inclinação é uma estimativa de *H*.

- Períodograma: Usa a inclinação do espectro de potência das séries à medida que a freqüência se aproxima de zero. Em um gráfico log-log, a inclinação do períodograma será a inclinação de uma linha reta β, onde $\beta - 1 = 1 - 2H$.

- Estimador de Whittle: É necessário que se forneça a forma do processo estocástico da série, que normalmente vai ser fGn (Fractional Gaussian Noise) ou Fractional ARIMA. Esses dois modelos diferem nas suposições sobre as dependências de curta duração nas séries: enquanto fGn assume que não existe nenhum dependência de curta duração, Fractional ARIMA pode assumir um grau fixo de dependência de curta duração.

- Estimador de Abry-Veitch (AV): Os estimadores de parâmetros de dependência de longa duração, como o parâmetro *H*, normalmente assumem processos estacionários e são sucessíveis a desvios quando isso não é verdade. Entretanto, o estimador AV tem propriedades muito melhores quando a série não é estacionária, fornecendo uma estimativa mais precisa para o parâmetro *H* [RouVei99].

4.1.2. Experimentos com o tempo entre misses

Nesta seção, verifica-se se os tempos entre misses podem ser representados usando a noção de auto-semelhança. Foi usado o estimador de Abry-Veitch (AV), descrito em [RouVei99]. Este estimador pode ser usado para determinar o valor do parâmetro de Hurst (*H*) para uma série de dados.

O estimador AV foi aplicado a uma série de tempos entre misses coletados para as políticas RTIME e SIZE. Foram usadas as primeiras 500 mil entradas dos arquivos dos tempos entre misses por motivos de eficiência.

Os resultados dos testes para essas duas políticas são apresentados nas Tabelas 4.1 e 4.2, respectivamente. Foram escolhidos os tamanhos de cache 10% e 30% da cache máxima por estes apresentarem valores bastante diferentes para os parâmetros avaliados nos experimentos iniciais. Para a cache de Berkeley2, foram testados também os resultados para 50% e 70% da cache, e os resultados para este trace são mostrados na Figura 4.1. A figura mostra que os valores calculados para *H* são maiores para a política RTIME do que para a política SIZE, embora essa diferença seja considerada muito pequena.

Trace	Valor de H	Intervalo de confidência
Berkeley1 – 10% da cache máxima	0,555	0,553 – 0,557
Berkeley1 – 30% da cache máxima	0,560	0,558 – 0,562
Berkeley2 – 10% da cache máxima	0,577	0,557 – 0,579
Berkeley2 – 30% da cache máxima	0,572	0,570 – 0,574
Berkeley2 – 50% da cache máxima	0,569	0,567 – 0,571
Berkeley2 – 70% da cache máxima	0,569	0,567 – 0,571
Digital1 – 10% da cache máxima	0,569	0,567 – 0,571
Digital 1 – 30% da cache máxima	0,586	0,584 – 0,588
Digital2 – 10% da cache máxima	0,548	0,546 – 0,550
Digital2 – 30% da cache máxima	0,577	0,575 – 0,579

Tabela 4.1: Valores estimados de H para a política RTIME, que usa como chave o tempo de recuperação do documento.

Trace	Valor de H	Intervalo de confidência
Berkeley1 – 10% da cache máxima	0,560	0,558 – 0,562
Berkeley1 – 30% da cache máxima	0,560	0,558 – 0,562
Berkeley2 – 10% da cache máxima	0,568	0,566 – 0,570
Berkeley2 – 30% da cache máxima	0,569	0,567 – 0,571
Berkeley2 – 50% da cache máxima	0,569	0,567 – 0,571
Berkeley2 – 70% da cache máxima	0,569	0,567 – 0,571
Digital1 – 10% da cache máxima	0,583	0,581 – 0,585
Digital 1 – 30% da cache máxima	0,582	0,580 – 0,584
Digital2 – 10% da cache máxima	0,572	0,570 – 0,574
Digital2 – 30% da cache máxima	0,574	0,572 – 0,576

Tabela 4.2: Valores estimados de H para a política SIZE, que usa como chave inverso do tamanho do documento.

Através da análise dos dados apresentados, podemos concluir que os tempos entre misses apresentam um grau de auto-semelhança muito baixo, já que o valor do parâmetro *H* varia de 0.5 a 1, e que o valor 1 representa o maior grau de auto-semelhança. Para verificar se estes resultados são válidos para outras políticas estudadas, foram feitos alguns estudos complementares, mostrados na Tabela 4.3. Foram feitos testes para as políticas FIFO, LFU, LRU, a política que usa como chave o número de referências ao objeto (NREF) e a política que usa como chave o tempo de recuperação do objeto e que ainda usa controle de admissão (RTIME-THR1). Os resultados para as essas políticas mostra a mesma tendência observada anteriormente.

Política	Valor de H	Intervalo de confidência
FIFO – 10% da cache máxima	0,574	0,572 – 0,576
FIFO – 30% da cache máxima	0,572	0,570 – 0,574
LFU – 10% da cache máxima	0,574	0,572 – 0,576
LFU – 30% da cache máxima	0,572	0,570 – 0,574
LRU – 10% da cache máxima	0,574	0,572 – 0,576
LRU – 30% da cache máxima	0,572	0,570 – 0,574
NREF – 10% da cache máxima	0,574	0,572 – 0,576
NREF – 30% da cache máxima	0,572	0,570 – 0,574
RTIME-THR1 – 10% da cache máxima	0,556	0,554 – 0,559
RTIME-THR1 – 30% da cache máxima	0,553	0,551 – 0,555

Tabela 4.3: Valores estimados de H para as políticas FIFO, LFU, LRU, NREF e RTIME-THR1. Trace de Berkeley2.

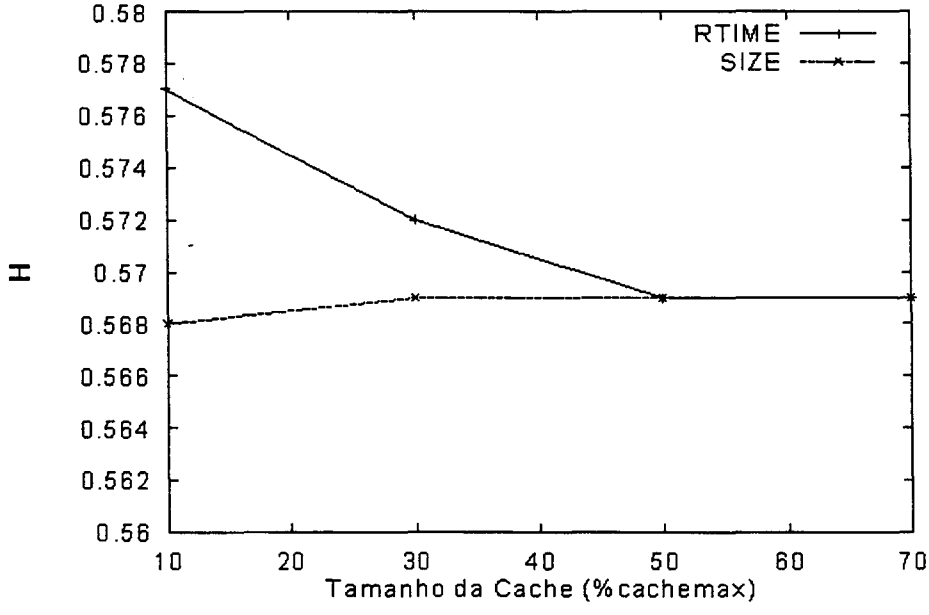

Figura 4.1: Valor estimado do parâmetro H para as políticas RTIME, que usam como chave o tempo de recuperação do documento, e SIZE, que usam como chave o inverso do tamanho do documento. O trace usado foi Berkeley2.

4.2. A distribuição do tempo entre misses

Descrevem-se, nessa seção, experimentos que tentam determinar se os tempos entre misses podem ser modelados segundo alguma distribuição conhecida. O objetivo é tentar usar essa modelagem para fazer previsões dos tempos entre dois misses consecutivos. Essas previsões poderiam ser usadas, por exemplo, em algoritmos de *prefetching*.

Existe um conjunto de tempos entre misses diferente para cada política e para cada tamanho de cache estudado. A Tabela 4.4 mostra algumas características dos conjuntos de tempos entre misses estudados, classificadas pelas políticas, traces e tamanhos de cache de cada um. Foram escolhidas para estudo as políticas i) FIFO, ii) LFU, iii) LRU, iv) a política que usa como chave o número de referências ao objeto (NREF), v) a política que usa como chave o tempo de recuperação do objeto (RTIME), vi) a política que usa como chave o inverso do tamanho do objeto (SIZE) e vii) a política que usa como chave o tempo de recuperação do objeto e que ainda usa controle de admissão (RTIME-THR1). Na tabela, são mostrados o número de misses ocorridos (N), a média do tempo entre misses e o desvio padrão. Uma média do tempo entre misses igual a 1, por exemplo, indica que, na média, houve um hit entre cada dois misses.

Política	Cache	N	Média	Desv.Padrão
FIFO	Ber1-10	1681957	0,458132	0,910849
	Ber1-50	1508093	0,626236	1,143162
	Ber2-10	1216883	0,315791	0,730419
	Ber2-50	1015078	0,57738	1,073092
	Dig1-10	1680326	0,354636	0,861989
	Dig1-50	1279142	0,779498	1,484633
	Dig2-10	1485751	0,327158	0,792664
	Dig2-50	1138117	0,732524	1,348049
LFU	Ber1-10	1679828	0,45998	0,913163
	Ber1-50	1512248	0,621768	1,13765
	Ber2-10	1216804	0,315877	0,73022
	Ber2-50	1015163	0,577248	1,072891
	Dig1-10	1681778	0,353466	0,859843
	Dig1-50	1280125	0,778131	1,482642
	Dig2-10	1487663	0,325452	0,789001
	Dig2-50	1138819	0,731456	0,346346

Tabela 4.4: Características dos conjuntos de tempos entre misses estudados. Valores do número de misses ocorridos (N), a média do tempo entre misses e o desvio padrão.

Política	Cache	N	Média	Desv.Padrão
LRU	Ber1-10	1679832	0,459976	0,913006
	Ber1-50	1510813	0,623308	1,139228
	Ber2-10	1216804	0,315877	0,73022
	Ber2-50	1015163	0,577248	1,072891
	Dig1-10	1681004	0,35409	0,860901
	Dig1-50	1280037	0,778253	1,482916
	Dig2-10	1487663	0,325452	0,789001
	Dig2-50	1138819	0,731456	1,346346
NREF	Ber1-10	1669102	0,469362	0,923514
	Ber1-50	1509848	0,624346	1,141143
	Ber2-10	1199048	0,335363	0,754949
	Ber2-50	1011174	0,58347	1,081458
	Dig1-10	1636217	0,391154	0,917733
	Dig1-50	1253574	0,815792	1,537434
	Dig2-10	1451330	0,358634	0,832723
	Dig2-50	1130321	0,744475	1,361169
RTIME	Ber1-10	2250355	0,089835	0,422987
	Ber1-50	2250355	0,089835	0,422987
	Ber2-10	1506023	0,063174	0,361664
	Ber2-50	1204142	0,329714	0,81205
	Dig1-10	2098110	0,084895	0,409085
	Dig1-50	1445342	0,574873	1,233758
	Dig2-10	1235632	0,595804	1,200088
	Dig2-50	1235632	0,595804	1,200088
SIZE	Ber1-10	1516460	0,617263	1,106589
	Ber1-50	1369943	0,790231	1,343578
	Ber2-10	1011836	0,582434	1,078836
	Ber2-50	945154	0,694077	1,236651
	Dig1-10	1260597	0,805676	1,502609
	Dig1-50	1124831	1,02362	1,903807
	Dig2-10	1132643	0,740905	1,323817
	Dig2-50	1013911	0,94476	0,94476
RTIME–THR1	Ber1-10	2386100	0,027834	0,200651
	Ber1-50	1964145	0,248643	0,630996
	Ber2-10	1522161	0,051902	0,281998
	Ber2-50	1264333	0,26641	0,662136
	Dig1-10	2132019	0,067641	0,335036
	Dig1-50	1560986	0,4582	0,992661
	Dig2-10	1886309	0,045336	0,266518
	Dig2-50	1378033	0,430899	0,906596

Tabela 4.4 (continuação): Características dos conjuntos de tempos entre misses estudados. Valores do número de misses ocorridos (N), a média do tempo entre misses e o desvio padrão.

4.2.1. Ambiente do experimento

Foram feitas análises dos tempos entre misses obtidos, com o objetivo de verificar se os dados seguem alguma distribuição conhecida.

Para realizar estes testes, foi utilizado o software estatístico SAS, versão 6.12. Este software permite a realização do teste de *goodness-of-fit*, que tem por objetivo calcular se um modelo estatístico é apropriado ou não para um conjunto de dados [Wadswo89]. Além do conjunto de dados a ser testado, o teste de goodness-of-fit requer os seguintes parâmetros:

1. Distribuições a serem testadas: por default, são testadas as distribuições Exponencial, Lognormal, Normal, Weibull e Kernel. Podem ser testadas também as distribuições Beta e Gamma, mas são necessários parâmetros adicionais. Caso o conjunto de dados seja aceito para uma determinada distribuição, seus parâmetros são estimados.

2. Threshold: valor usado como parâmetro para algumas das distribuições testadas (Beta, Exponencial, Gamma, Lognormal e Weibull). Este valor deve ser menor que o menor dos valores nos dados. Como não existe tempo entre misses menor que zero, usamos o valor –1 para a maioria dos experimentos.

3. Critério: pede o *nível de probabilidade* usado no teste das distribuições. Este nível de probabilidade indica o nível mínimo para que uma curva possa ser considerada como seguindo uma determinada distribuição. Este valor pode variar de 0,01 a 0,25, sendo que um valor de 0,01 é o menos restritivo de todos.

4. Tipo de teste estatístico a ser usado: existem as opções de Anderson-Darling, Cramer-von Mises e Kolmogorov-Smirnov. Foi usada a primeira opção.

4.2.2. Experimentos iniciais

Os experimentos iniciais foram feitos para as políticas FIFO, LFU, LRU, a política que usa como chave o número de referências ao objeto (NREF), a política que usa como chave o tempo de recuperação do objeto (RTIME), a política que usa como chave o inverso do tamanho do objeto (SIZE) e a política que usa como chave o tempo de recuperação do objeto e que ainda usa controle de admissão (RTIME-THR1), e apenas para os tamanhos de cache correspondendo a 10% e 50% da cache máxima. Foi usado o teste estatístico de Anderson-Darling, com threshold igual a –1 e nível de probabilidade (critério) igual a 0,01.

O resultado de todos os testes foi o mesmo: os dados foram aceitos para a distribuição de Weibull, com alguma diferença nos parâmetros estimados para essa distribuição (shape, scale), mostrados na Tabela 4.5.

Política	Cache	HR (%)	TSM (s)	Threshold	Shape	Scale
FIFO	Ber1-10	19	21,72	-1	1,8	1,7
	Ber1-50	24,131	22,28		1,6	1,8
	Ber2-10	17	20,85		1,9	1,5
	Ber2-50	21,908	21,27		1,7	1,8
	Dig1-10	32,953	8,49		1,8	1,5
	Dig1-50	39,313	8,84		1,4	2
	Dig2-10	30,491	9,31		1,8	1,5
	Dig2-50	37,112	9,71		1,5	1,9
LFU	Ber1-10	18,626	21,55	-1	1,8	1,7
	Ber1-50	22,418	21,95		1,6	1,8
	Ber2-10	14,429	20,58		1,9	1,5
	Ber2-50	20,015	20,99		1,7	1,8
	Dig1-10	25,537	8,16		1,8	1,5
	Dig1-50	36,255	8,59		1,4	2
	Dig2-10	23,253	9,15		1,9	1,5
	Dig2-50	34,668	9,63		1,5	1,9

Tabela 4.5: Resultado dos testes de goodness-of-fit e valores estimados para os parâmetros da distribuição Weibull (shape e scale).

Política	Cache	HR (%)	TSM (s)	Threshold	Shape	Scale
LRU	Ber1-10	18,558	21,53	-1	1,8	1,7
	Ber1-50	22,37	21,94		1,6	1,8
	Ber2-10	14,419	20,58		1,9	1,5
	Ber2-50	20,015	20,98		1,7	1,8
	Dig1-10	25,615	8,16		1,8	1,5
	Dig1-50	36,201	8,58		1,4	2
	Dig2-10	23,513	9,15		1,9	1,5
	Dig2-50	34,677	9,63		1,5	1,9
NREF	Ber1-10	18,532	21,52	-1	1,8	1,7
	Ber1-50	22,405	21,95		1,6	1,8
	Ber2-10	14,387	20,57		1,9	1,5
	Ber2-50	20,015	20,99		1,7	1,8
	Dig1-10	23,685	8,03		1,7	1,6
	Dig1-50	35,858	8,55		1,4	2
	Dig2-10	21,465	9,04		1,8	1,5
	Dig2-50	34,446	9,61		1,5	2
RTIME	Ber1-10	5,025	17,83	-1	2,3	1,2
	Ber1-50	16,799	20,69		2,3	1,2
	Ber2-10	3,698	18,80		2,4	1,2
	Ber2-50	14,079	19,65		1,8	1,5
	Dig1-10	6,347	6,72		2,3	1,2
	Dig1-50	28,408	7,62		1,5	1,8
	Dig2-10	4,95	7,78		2,5	1,2
	Dig2-50	29,019	8,78		1,6	1,8
SIZE	Ber1-10	22,582	20,47	-1	1,7	1,8
	Ber1-50	25,555	22,57		1,5	2
	Ber2-10	21,035	21,78		1,7	1,8
	Ber2-50	23,171	21,45		1,6	1,9
	Dig1-10	36,116	8,98		1,4	2
	Dig1-50	41,156	9,11		1,3	2,2
	Dig2-10	33,754	9,96		1,5	2
	Dig2-50	38,879	9,84		1,4	2,2
RTIME–THR1	Ber1-10	0,889	17,29	-1	3,3	1,1
	Ber1-50	5,031	16,73		2,1	1,4
	Ber2-10	1,692	18,46		2,8	1,1
	Ber2-50	5,029	18,18		2	1,4
	Dig1-10	2,67	6,62		2,6	1,2
	Dig1-50	6,704	6,57		1,7	1,7
	Dig2-10	1,332	7,57		2,9	1,1
	Dig2-50	5,168	7,26		1,8	1,6

Tabela 4.5 (continuação): Resultado dos testes de goodness-of-fit e valores estimados para os parâmetros da distribuição Weibull (shape e scale).

Os resultados dos experimentos atenderam às expectativas, já que a distribuição de Weibull é usada principalmente para modelar tempos entre falhas [Wadswo89]. Os parâmetros dessa distribuição podem ser usados para representar os três tipos de falhas existentes: crescente, quando a probabilidade de falha no próximo instante cresce com o tempo, decrescente, quando a probabilidade de falha no próximo instante decresce com o tempo, ou constante, quando a probabilidade de falha no próximo instante não muda com o tempo.

O parâmetro α (shape) da distribuição Weibull determina o formato da distribuição. Quando α é maior que 1, a taxa de falha aumenta com o tempo; quando α é menor que 1, essa taxa decresce com o tempo; e quando α é igual a 1, a taxa de falha é constante. O parâmetro β (scale) representa a escala da distribuição, e determina o quanto os valores ficam espalhados.

Teoricamente, Weibull é uma boa distribuição para modelar tempos entre misses, visto que os misses são falhas que ocorrem no tempo (no caso dos experimentos, número de pedidos). Entretanto, o nível de probabilidade usado foi o menos restritivo possível, já que quanto menor o nível de probabilidade, menos o conjunto de dados se aproxima da distribuição testada.

O resultado dos experimentos, até então, indica que os tempos entre misses aproximam-se mais da distribuição Weibull do que das outras testadas, mas ainda não significa que os dados sigam a distribuição Weibull. Para poder afirmar isso, são necessários experimentos adicionais com níveis de probabilidade maiores.

4.2.3. Experimentos adicionais

Os experimentos adicionais têm como objetivo confirmar os resultados de que os tempos entre misses realmente seguem a distribuição Weibull. Para confirmar esse resultado, foram feitos experimentos com níveis de probabilidade.

Os resultados desses experimentos adicionais mostraram que os tempos entre misses não podem ser modelados segundo nenhuma das distribuições testadas, para níveis de probabilidade maiores que 0.01. Foram feitos alguns testes com o nível de probabilidade 0.1, e estes rejeitaram todas as distribuições. Mesmo nos testes feitos com um nível de probabilidade de 0.02 (valor quase igual ao do primeiro teste) a distribuição Weibull foi rejeitada. Isso indica que embora os tempos entre misses se aproximem mais de Weibull do que das outras distribuições testadas, sua relação com a distribuição é muito pequena.

Para complementar o estudo, notou-se que o valor de threshold usado nos testes influenciava o valor dos parâmetros de Weibull, quando aceito (nível de probabilidade de 0,01). Ao invés de usar um threshold igual a –1, foram feitos testes com o threshold estimado pelo programa (-0.033), e os resultados não alteraram a aceitação da distribuição de Weibull. Entretanto, foram obtidos parâmetros muito diferentes, como mostrado na Tabela 4.6, para o workload de Berkeley2. Esta tabela mostra os resultados dos testes com os níveis de probabilidade 0.01 e 0.1.

Política	Cache	HR (%)	TSM (s)	Critério	Threshold	Shape	Scale
RTIME	Ber2-10	3,698	18,80	0,01	-1	2,4	1,2
				0,01	-0,033	0,72	0,06
				0,10	-1	não	não
	Ber2-50	14,079	19,65	0,01	-1	1,8	1,5
				0,01	-0,033	0,54	0,17
				0,10	-1	não	não
SIZE	Ber2-10	21,035	21,78	0,01	-1	1,7	1,8
				0,01	-0,033	0,53	0,32
				0,10	-1	não	não
	Ber2-50	23,171	21,45	0,01	-1	1,6	1,9
				0,01	-0,033	0,53	0,38
				0,10	-1	não	não
RTIME–THR1	Ber2-10	1,692	18,46	0,01	-1	2,8	1,1
				0,01	-0,033	0,75	0,06
				0,10	-1	não	não
	Ber2-50	5,029	18,18	0,01	-1	2	1,4
				0,01	-0,033	0,56	0,15
				0,10	-1	não	não

Tabela 4.6: Resultado dos testes de goodness-of-fit adicionais, com diferentes valores para o nível de probabilidade (critério) e valores estimados para os parâmetros da distribuição Weibull (shape e scale).

Deve-se notar que é possível que haja interferência nos resultados dos testes das distribuições, devido ao fato de que os algoritmos de simulação começam com a cache vazia, e por isso geram uma grande quantidade inicial de misses (que aparecem como valores 0, no arquivo de tempos entre misses). Então, foram feitos testes retirando os zeros iniciais de cada conjunto de dados, até o primeiro tempo entre misses diferente de zero, e não houve alteração nos resultados. Isso pode ser explicado pela pequena proporção que essa seqüência de misses inicial representa, levando-se em conta o arquivo de tempos entre misses inteiro. Enquanto esta seqüência de misses que poderia ser desprezada corresponde a algumas centenas de pedidos (100,200,...), o número total de misses corresponde a alguns milhões de pedidos (de 1 a 2 milhões).

4.3. Resumo dos principais resultados

Através dos experimentos realizados, chegou-se à conclusão de que os tempos entre misses de uma Web cache, usando as políticas estudadas no capítulo 3, possuem um baixo grau de auto-semelhança. Para ser considerado auto-semelhante, um conjunto de dados deve apresentar um parâmetro H entre 0,5 e 1. Todos os valores obtidos nos experimentos para o parâmetro de Hurst variam entre 0,546 e 0,588, aproximando-se do valor mínimo para caracterização de auto-semelhança.

A tentativa de modelar os tempos entre misses segundo alguma distribuição estatística conhecida não gerou os resultados esperados. Embora os experimentos tenham mostrado que o comportamento dos tempos entre misses assemelham-se mais à distribuição Weibull do que às outras estudadas (Exponencial, Lognormal, Normal e Kernel), viu-se que essa relação é muito fraca e essa distribuição não pode ser usada para modelá-los.

Capítulo 5

Conclusões

A Internet vem tendo um aumento considerável no seu tráfego, resultando em maiores tempos de resposta a pedidos dos usuários. O tempo de resposta é dominado pelo tempo de recuperação dos objetos. Diversos são os trabalhos que estudam técnicas para diminuição deste tempo de recuperação. Uma das possíveis técnicas para diminuição do tempo de recuperação de documentos da Web é o uso de caches para armazenamento de objetos.

Este trabalho investigou as políticas de expulsão e de controle de admissão em Web caches. Tais políticas têm papel fundamental no desempenho das caches, que potencialmente podem diminuir bastante o tempo de recuperação de documentos e o tráfego de informações na Web. Mais especificamente, este trabalho investigou um parâmetro pouco estudado até então: o tempo de recuperação de um documento. Foram estudadas diversas políticas que levam em consideração este parâmetro, com o objetivo principal de deixar na cache documentos que demorariam muito para serem trazidos dos seus servidores de origem.

Ainda, este trabalho procurou caracterizar o tempo entre misses consecutivos em caches que fazem uso de algumas das políticas estudadas. Essa caracterização pode ser usada, entre outras coisas, para tentar melhorar o desempenho de algoritmos de *prefetching*, já que uma previsão do tempo entre misses poderia ser usada para decidir o tempo que o algoritmo teria para pegar objetos Web não pedidos pelo usuário.

5.1. Principais contribuições

As principais contribuições desse trabalho são:

- A chave tempo de recuperação fornece a menor média e variância do tempo de recuperação em todas as políticas analisadas, o que mostra que o uso desta medida é bastante recomendável quando se quer melhorar parâmetros relacionados à qualidade de serviço da Web, como o tempo de resposta dos pedidos e a variância dos tempos de resposta.

- Evidenciou-se que é vantajoso expulsar documentos antigos da cache, porém os resultados são altamente sensíveis à escolha do valor limite para a decisão de expulsão. Se este valor de limite for muito grande, o limite não influenciará muito a política, já que poucos documentos alcançarão o limite. Entretanto, se esse limite for muito pequeno, podem ser expulsos documentos que serão referenciados novamente, o que diminui o desempenho da política.

- Investigou-se o uso de políticas que dividem a cache em grupos, segundo algum critério, e aplicam uma política de substituição a cada grupo. Usou-se o tempo de recuperação e o inverso do tamanho do documento para divisão em grupos e como política de substituição. Os resultados mostraram que, dependendo da divisão dos grupos, pode-se obter uma política que tenha um desempenho intermediário para os parâmetros tempo médio de recuperação por miss e taxa de acerto na cache. Isso é um bom resultado, já que chegou-se à conclusão de que esses dois parâmetros são antagônicos, e que quando procuramos otimizar um deles, estamos diminuindo o desempenho no outro.

- O uso de políticas de controle de admissão baseadas em valor de corte para o tempo de recuperação mostrou-se bastante eficaz na diminuição da média e da variância do tempo de recuperação. Entretanto, a diminuição da taxa de acerto na cache foi muito grande, o que exige que este controle seja usado com cuidado.

- Através dos experimentos realizados com os tempos entre misses, chegou-se à conclusão de que esses tempos possuem um baixo grau de auto-semelhança, com valores para o parâmetro de Hurst variando entre 0,546 e 0,588.

- Outros experimentos sobre os tempos entre misses mostraram que estes não podem ser modelados segundo as distribuições usadas nos testes, que são Exponencial, Lognormal, Normal, Weibull e Kernel. Observou-se uma tendência de comportamento para a distribuição Weibull.

- Em linhas gerais, no estudo de políticas de gerenciamento de Web caches, concluiu-se que as políticas que levam em conta o tempo de recuperação dos objetos têm melhor desempenho quando existe uma grande quantidade de transferências de objetos Web grandes. Nesses casos, percebe-se uma maior economia de tempo e conseqüente diminuição do tempo médio de resposta. Esse tipo de tráfego começa a ser observado atualmente, com o aumento da velocidade de acesso e o surgimento de tipos de mídia que usam arquivos maiores. Para o tráfego onde a maior parte dos pedidos são para objetos Web muito pequenos, as políticas que levam em consideração o tamanho dos objetos pedidos têm melhor desempenho pois aumentam a taxa de acertos na cache e acabam diminuindo o tempo de resposta para o usuário.

5.2. Trabalhos futuros

Os resultados dessa dissertação podem ser estendidos de diversas formas. Algumas delas são:

- Os traces usados possuem uma grande quantidade de pontos, e representam tipos diferentes de acesso à Web. Entretanto, o período em que os traces foram coletados varia de 29 de agosto de 1996 a 10 de novembro de 1996. Não se sabe quais seriam os resultados dos estudos para traces mais recentes, já que o padrão de acesso à Internet mudou bastante. Existe atualmente uma variedade muito maior de objetos disponíveis na Internet, como arquivos de som, filmes, músicas em formato MP3, entre outros. Embora alguns destes tipos de arquivo existisse na época da coleta dos traces, a sua transferência pela Web não era tão popular devido às baixas velocidades de transferência. Os tamanhos destes objetos são normalmente muito maiores do que os documentos HTML e imagens presentes nos traces usados. Visto que os estudos levam em consideração medidas como o tamanho dos objetos e o tempo de recuperação dos mesmos, seria interessante repetir os experimentos para traces mais recentes, com estes novos tipos de objetos Web.

- As políticas que expulsam documentos pouco referenciados usam um valor de limite para tentar determinar quando um documento é considerado pouco referenciado. Foram explorados apenas três valores de limites, escolhidos de acordo com os traces, mas não se sabe se outros valores poderiam levar a resultados melhores. Um possível extensão seria fazer um estudo sobre qual valor de limite poderia ser usado para determinar que um documento foi pouco referenciado, ou estimar que este provavelmente não será mais referenciado. Preferencialmente, poderiam ser usados traces diferentes.

- As políticas que fazem a classificação dos objetos segundo alguma medida, e aplicam uma política para cada grupo, foram pouco exploradas. Poderiam ser estudados outros intervalos para classificação dos grupos das políticas. Também, poderiam ter sido usados outros parâmetros para classificação dos objetos e para a política de expulsão de cada grupo.

- Os estudos dos tempos entre misses na cache tinham por objetivo caracterizar esses tempos para uso em algoritmos de *prefetching*. Concluiu-se que existe uma fraca relação entre esses tempos e a distribuição Weibull. Embora os tempos entre misses não possam ser modelados com fidelidade por essa distribuição, uma extensão ao trabalho seria fazer experimentos usando algum algoritmo de prefetching que use a distribuição para prever o número de pedidos entre dois misses consecutivos.

Referências Bibliográficas

[AbStAb95+] Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Stephen Williams, Edward A Fox, "Caching Proxies: Limitations and Potentials", 4th International World-Wide Web Conference, paginas 119-133, Boston, Dezembro 1995

[AbFoxAb97+] Ghaleb Abdulla, Edward A Fox, Marc Abrams, Stephen Williams, "WWW Proxy Traffic Characterization with Application to Caching", Technical Report TR-97-03, Computer Science Department, Virginia Tech, Virginia, Março 1997

[AggaYu96] Charu C. Aggarwal, Philip S. Yu, "On Disk *Caching* of Web Objects in Proxy Servers", *Research Report RC 20636*, Divisão de Pesquisas da IBM, T.J.Watson Research Center, Yorktown Heights, NY, Outubro 1996

[AgWoYu97] Charu Aggarwal, Joel Wolf, Philip Yu, Marina Epelman, "On *Caching* Policies for Web Objects", *Research Report RC 20619*, Divisão de Pesquisas da IBM, T.J.Watson Research Center, Yorktown Heights, NY, Março 1997

[AlBeCrOl96] Virgílio Almeida, Azer Bestavros, Mark Crovella, Adriana de Oliveira, "Characterizing Reference Locality in the WWW", *Proceedings of PDIS'96: The IEEE Conference on Parallel and Distributed Information Systems*, Miami Beach, FL, Dezembro 1996

[ArlWil96] Martin F. Arlitt, Carey L. Williamson, "Web Server Workload Characterization: The Search for Invariants", *Proceedings of the 1996 ACM SIGMETRICS Conference*, Philadelphia, PA, Maio 1996.

[ArlWil97] Martin F. Arlitt, Carey L. Williamson, "Internet Web Servers: Workload Characterization and Performance Implications", *IEEE/ACM Transactions on Networking*, volume 5, numero 5, Outubro 1997.

[BaMoRo97] Michael Baentsch, L. Baum, Georg Molter, S. Rothkugel, P. Sturm, "World Wide Web *Caching*: The Application-Level View of the Internet", *IEEE Communication Magazine*, Junho 1997

[BerTraces] UC Berkeley Home IP Web Traces, URL: http://ita.ee.lbl.gov/ html/ contrib/ UCB.home-IP-HTTP.html

[BolHos96] Jean-Chrysostome Bolot, Philipp Hoschka, "Performance engineering of the World Wide Web: Application to dimensioning and *cache* design", *Computer Networks and ISDN Systems*, Proceedings of the Fifth International WWW Conference, Maio 96

[ChaSch96+] Anawat Chankhunthod, Michael F Schwartz, Peter B Danzing, Kurt J Worrell, Chuck Neerdaels, "A Hierarchical Internet Object Cache", USENIX 1996 Annual Technical Conference, San Diego, CA, Janeiro 1996

[CroBes96] Mark E. Crovella, Azer Bestavros, "Self-Similarity in WWW - Evidence and Possible Causes", *Proceedings of the 1996 ACM SIGMETRICS Conference*, Philadelphia, PA, Maio 1996.

[CuBeCr95] Carlos R. Cunha, Azer Bestavros, Mark E. Crovella, "Characteristics of WWW Client-based *Traces*", *Technical Report BU-CS-95-010*, Departamento de Ciências da Computação da Universidade de Boston, 1995.

[DanSit96] Asit Dan, Dinkar Sitaram, "Multimedia *Caching* Strategies for Heterogeneous Application and Server Environments", *Research Report RC 20670*, Divisão de Pesquisas da IBM, T.J.Watson Research Center, Yorktown Heights, NY, Dezembro 1996

[Deng96] Shuang Deng, "Empirical Model of WWW Document Arrivals at Access *Link*", *IEEE International Conference on Communication*, vol.3, 1996.

[DigTraces] Digital's Web Proxy Traces, URL: ftp://ftp.digital.com/ pub/ DEC/ traces/ proxy/ webtraces.html

[FonOli99] Nelson L.S. da Fonseca, Rodrigo M. Oliveira, "Políticas de Gerenciamento de Web Caches Baseadas no Tempo de Recuperação de Documentos", XXVI Semish - Seminário Integrado de Software e Hardware, PUC - RJ, Rio de Janeiro, Julho de 1999

[HeMiYa97] Abdelsalam Heddaya, Sulaiman Mirdad, David Yates, "Diffusion-based *Caching* along Routing Paths", http:// ircache.nlanr.net/ Cache/ Workshop97/ agenda.html, Maio 1997

[HTTPTraces] Sources for HTTP trace/log files, URL: http://www.ircache.net /Cache/ traces.html

[JacCao98] Quinn Jacobson, Pei Cao, "Potential and Limits of Web Prefetching Between Low-Bandwidth Clients and Proxies", *3rd International WWW Caching Workshop*, Junho 1998

[JiaKle98] Zhimei Jiang, Leonard Kleinrock, "An Adaptive Network Prefetch Scheme", IEEE Journal on Selected Areas In Communications, vol.16, no 3, Abril 1998

[JefDas97] Clinton L. Jeffery, Samir R. Das, "Design of a Proxy-Sharing Proxy Server", http://ircache.nlanr.net/ Cache/ Workshop97/ Papers/ Jeffery/ jeffery.html, Abril 1997

[LelTaq94+] Will E. Leland, Murad S. Taqqu, Walter Willinger, Daniel V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)", IEEE/ACM Transactions on Networking, vol.2, no.1, Fevereiro 1994

[Markat96] Evangelos P. Markatos, "Main memory *caching* of Web documents", *Computer Networks and ISDN Systems*, Proceedings of the Fifth International WWW Conference, Maio 96

[MiNgRo98+] Scott Michel, Khoi Nguyen, Adam Rosenstein, Lixia Zhang, "Adaptive Web Caching: Towards a New Global Caching Architecture", *3rd International WWW Caching Workshop*, Junho 1998

[MuAlMe98] Cristina Duarte Murta, Virgilio A. F. Almeida, Wagner Meira Jr., "Analyzing Performance of Partitioned Caches for the WWW", ", *3rd International WWW Caching Workshop*, Junho 1998

[Neal96] Donald Neal, "The Harvest Object Cache in New Zealand", *Computer Networks and ISDN Systems*, Proceedings of the Fifth International WWW Conference, Maio 96

[RoSoTa97] Alex Rousskov, Valery Soloviev, Igor Tatarinov, "Static *Caching*", http://ircache.nlanr.net/ Cache/ Workshop97/ Papers/ Rousskov/ rousskov.html, Abril 1997

[RouVei99] Matthew Roughan, Darryl Veitch, "Measuring Long-Range Dependence under Changing Traffic Conditions", Proceeding Infocom '99 Manhattan Abril 1999

[Squid] Squid Web Proxy Cache, URL: http://www.squid-cache.org/

[Tanen97] Andrew S. Tanenbaum, "Redes de Computadores", Terceira Edição, Editora Campus, Rio de Janeiro - RJ, 1997

[UserSur98] 9th WWW User Survey" conduzido pelo Graphic, Visualization & Usability Center (Georgia Institute of Technology, URL: http://www.cc.gatech.edu/ gvu/ user_surveys/

[Wadswo89] Harrison M. Wadsworth, "Handbook of Statistical Methods for Engineers and Scientists", McGraw Hill, 1989

[WebCon] Web Consortium Protocol Recommendation. URL: http://www.w3.org/ PICS

[WesCla97] Wessels D., Claffy K., "Application Of Internet Cache Protocol (ICP), version 2", Network Working Group - Internet Draft, Julho 1997

[WesCla98] Wessels D., Claffy K., "ICP and the Squid Web Cache", IEEE Journal on Selected Areas In Communications, vol.16, no 3, Abril 1998

[WiAbSt96] Stephen Williams, Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, Edward A. Fox, "Removal Policies in Network *Caches* for World-Wide Web Documents", *Proceedings of the 1996 ACM SIGCOMM Conference*, California, CA, Agosto 1996.

[WilTaq97+] Walter Willinger, Murad S. Taqqu, Robert Sherman, Daniel V. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level", IEEE/ACM Transactions on Networking, vol.5, no.1, Fevereiro 1997

[WooAbr97] R. Wooster and M Abrams, "Proxy caching that estimates page load delays", 6th International World Wide Web Conference, Abril 1997

[YuEdMa97] Philip S. Yu, Edward A. MacNair, "Performance Study of a Collaborative Method for Hierarchical Caching in Proxy Servers", *Research Report RC 21026*, Divisão de Pesquisas da IBM, T.J.Watson Research Center, Yorktown Heights, NY, Novembro 1997.

[ZhFlJa97] Lixia Zhang, Sally Floyd, Van Jacobson, "Adaptive Web *Caching*", http://ircache.nlanr.net/ Cache/ Workshop97/ agenda.html, Abril 1997